

PanaXSeries

The One to Watch for Constant Innovation-Making the Future Come Alive

MICROCOMPUTER

MN102H

MN102H75K/F75K/85K/F85K

LSI User's Manual

Pub.No.22385-011E

Panasonic

PanaXSeries is a trademark of Matsushita Electric Industrial Co., Ltd.

The other corporation names, logotype and product names written in this book are trademarks or registered trademarks of their corresponding corporations.

Request for your special attention and precautions in using the technical information and semiconductors described in this book

- (1) An export permit needs to be obtained from the competent authorities of the Japanese Government if any of the products or technologies described in this book and controlled under the "Foreign Exchange and Foreign Trade Law" is to be exported or taken out of Japan.
- (2) The contents of this book are subject to change without notice in matters of improved function. When finalizing your design, therefore, ask for the most up-to-date version in advance in order to check for any changes.
- (3) We are not liable for any damage arising out of the use of the contents of this book, or for any infringement of patents or any other rights owned by a third party.
- (4) No part of this book may be reprinted or reproduced by any means without written permission from our company.
- (5) This book deals with standard specification. Ask for the latest individual Product Standards or Specifications in advance for more detailed information required for your design, purchasing and applications.

If you have any inquiries or questions about this book or our semiconductors, please contact one of our sales offices listed at the back of this book.

Contents

About This Manual	16
Using This Manual	16
Text Conventions	16
Register Conventions	16
Related Documents	17
Questions and Comments	17
1 General Description	18
1.1 MN102H Series Overview	18
1.2 MN102H Series Features	18
1.3 MN102H Series Description	21
1.4 General Specifications	26
1.5 Block Diagram	28
1.6 Pin Descriptions	30
1.6.1 MN102H85K Pin Description	30
1.6.2 MN102H75K Pin Description	31
1.7 Bus Interface	35
1.7.1 Description	35
1.7.2 Bus Interface Control Registers	36
2 Interrupts	37
2.1 Description	37
2.2 Interrupt Setup Examples	40
2.2.1 Setting Up an External Pin Interrupt	40
2.2.2 Setting Up a Watchdog Timer Interrupt	42
2.3 Interrupt Control Registers	44
3 Low-Power Modes	72
3.1 CPU Modes	72
3.1.1 Description	72
3.1.2 Exiting from SLOW Mode to NORMAL Mode	73
3.1.3 Notes on Invoking and Exiting STOP and HALT Modes	74
3.2 Turning Individual Functions On and Off	75
3.3 CPU Control Register	76
4 Timers	77
4.1 8-Bit Timer Description	77
4.2 8-Bit Timer Features	78
4.3 8-Bit Timer Block Diagrams	79
4.4 8-Bit Timer Timing	81
4.5 8-Bit Timer Setup Examples	82

4.5.1	Setting Up an Event Counter Using Timer 0	82
4.5.2	Setting Up an Interval Timer Using Timers 1 and 2	84
4.6	8-Bit Timer Control Registers	87
4.7	16-Bit Timer Description	88
4.8	16-Bit Timer Features	89
4.9	16-Bit Timer Block Diagrams	90
4.10	16-Bit Timer Timing	90
4.11	16-Bit Timer Setup Examples	94
4.11.1	Setting Up an Event Counter Using Timer 4	94
4.11.2	Setting Up a Single-Phase PWM Output Signal Using Timer 4	96
4.11.3	Setting Up a Two-Phase PWM Output Signal Using Timer 4	101
4.11.4	Setting Up a Single-Phase Capture Input Using Timer 4	106
4.11.5	Setting Up a Two-Phase Capture Input Using Timer 4	108
4.11.6	Setting Up a 4x Two-Phase Encoder Input Using Timer 5	111
4.11.7	Setting Up a 1x Two-Phase Encoder Input Using Timer 5	114
4.11.8	Setting Up a One-Shot Pulse Output Using Timer 5	117
4.11.9	Setting Up an External Count Direction Controller Using Timer 5	120
4.11.10	Setting Up External Reset Control Using Timer 5	123
4.12	16-Bit Timer Control Registers	125
5	Serial Interfaces	127
5.1	Description	127
5.2	Features	127
5.3	Connecting the Serial Interfaces	128
5.3.1	Synchronous Serial Mode Connections	128
5.3.2	UART Mode Connections	128
5.3.3	I ² C Mode Connection	128
5.4	UART Mode Baud Rates	129
5.5	Serial Interface Timing	129
5.5.1	Synchronous Serial Mode Timing	129
5.5.2	UART Mode Timing	130
5.6	Serial Interface Setup Examples	131
5.6.1	Setting Up UART Transmission Using Serial Interface 0	131
5.6.2	Setting Up Synchronous Serial Reception Using Serial Interface 0	134
5.6.3	Setting Up the Serial Interface Clock	135
5.6.4	Setting Up I ² C Transmission Using Serial Interface 0	137
5.6.5	Setting Up I ² C Reception Using Serial Interface 0	139
5.7	Serial Interface Control Registers	140
6	Analog-to-Digital Converter	143
6.1	Description	143
6.2	Features	143
6.3	Block Diagram	144
6.4	A/D Conversion Timing	144
6.4.1	Selecting the ADC Clock Source	144

6.4.2	Single Channel/Single Conversion Timing	145
6.4.3	Multiple Channel/Single Conversion Timing	145
6.4.4	Single Channel/Continuous Conversion Timing	146
6.4.5	Multiple Channel/Continuous Conversion Timing	146
6.5	ADC Setup Examples	147
6.5.1	Setting Up Software-Controlled Single-Channel A/D Conversion	147
6.5.2	Setting Up Hardware-Controlled Intermittent Three-Channel A/D Conversion	148
6.6	ADC Control Registers	150
6.7	Caution about Analog-to-Digital Converter	152
7	On-Screen Display	153
7.1	Description	153
7.2	Features	153
7.3	Block Diagram	154
7.4	Power-Saving Considerations in the OSD Block	155
7.5	OSD Operation	156
7.5.1	OSD Clock	156
7.5.2	External Input Sync Signals	156
7.5.3	Multi-Layer Format	156
7.5.4	Output Pin Setup	157
7.5.5	Microcontroller Interface	157
7.5.6	VRAM	157
7.5.7	Conditions for VRAM Writes	158
7.6	Standard and Extended Display Modes	159
7.6.1	Cursor Layer Display Modes	159
7.6.2	Graphics Layer Display Modes	160
7.7	Display Setup Examples	161
7.7.1	Setting Up the Graphics Layer	161
7.7.2	Setting Up the Text Layer	163
7.8	VRAM	165
7.8.1	VRAM Operation	165
7.8.2	VRAM Organization	169
7.8.3	Cautions about the number of display code set to VRAM	171
7.9	ROM	172
7.9.1	ROM Organization	172
7.9.2	Graphics ROM Organization in Different Color Modes	173
7.10	Setting Up the OSD	178
7.10.1	Setting Up the OSD Display Colors	178
7.10.2	Text Layer Functions	184
7.10.3	Display Sizes	187
7.10.4	Setting Up the OSD Display Position	189
7.11	DMA and Interrupt Timing	191
7.12	Selecting the OSD Dot Clock	193
7.13	Controlling the Shuttering Effect	194
7.13.1	Controlling the Shuttered Area	194
7.13.2	Controlling Shutter Movement	196

7.13.3	Controlling Shuttering Effects	198
7.13.4	Controlling Line Shuttering	200
7.14	Field Detection Circuit	201
7.14.1	Block Diagram	201
7.14.2	Description	201
7.14.3	Considerations for Interlaced Displays	202
7.15	OSD Registers	203
8	IR Remote Signal Receiver	216
8.1	Description	216
8.2	Block Diagram	217
8.3	IR Remote Signal Receiver Operation	218
8.3.1	Operating Modes	218
8.3.2	Noise Filter	218
8.3.3	8-Bit Data Reception	219
8.3.4	Identifying the Data Format	220
8.3.5	Generating Interrupts	221
8.3.5.1	Leader Detection	221
8.3.5.2	Trailer Detection	221
8.3.5.3	8-Bit Data Reception Detection	221
8.3.5.4	Pin Edge Detection	221
8.3.6	Controlling the SLOW Mode	222
8.4	IR Remote Signal Receiver Control Registers	223
9	Closed-Caption Decoder	227
9.1	Description	227
9.2	Block Diagram	227
9.3	Functional Description	228
9.3.1	Analog-to-Digital Converter	228
9.3.2	Clamping Circuit	229
9.3.3	Sync Separator Circuit	230
9.3.3.1	HSYNC Separator	232
9.3.3.2	VSYNC Separator	233
9.3.3.3	Field Detection Circuit	233
9.3.4	Data Slicer	233
9.3.5	Controller and Sampling Circuit	234
9.3.5.1	CRI Detection for Sampling Clock Generation	235
9.3.5.2	Data Capture Control	235
9.4	Closed-Caption Decoder Registers	236
10	Pulse Width Modulator	249
10.1	Description	249
10.2	Block Diagram	250
10.3	PWM Data Registers	250

11	I/O Ports	251
11.1	Description	251
11.2	I/O Port Circuit Diagrams	252
11.3	I/O Port Control Registers	277
12	ROM Correction	288
12.1	Description	288
12.2	Block Diagram	289
12.3	Programming Considerations	289
12.4	ROM Correction Control Registers	290
13	I²C Bus Controller	293
13.1	Description	293
13.2	Block Diagram	296
13.3	Functional Description	296
13.4	Setting Up the I ² C Bus Connection	298
13.5	SDA and SCL Waveform Characteristics	299
13.6	I ² C Interface Setup Examples	300
13.6.1	Setting Up a Transition from Master Transmitter to Master Receiver	300
13.6.1.1	Pre-configuring	300
13.6.1.2	Setting Up the First Interrupt	300
13.6.1.3	Setting Up the Second Interrupt	301
13.6.1.4	Setting Up the Third Interrupt	301
13.6.2	Setting Up a Transition from Slave Receiver to Slave Transmitter	302
13.6.2.1	Pre-configuring	302
13.6.2.2	Setting Up the First Interrupt	302
13.6.2.3	Setting Up the Second Interrupt	303
13.6.2.4	Setting Up the Third Interrupt	303
13.7	I ² C Bus Interface Registers	304
14	H Counter	307
14.1	Description	307
14.2	Block Diagram	307
14.3	H Counter Operation	307
14.4	H Counter Control Registers	310
Appendix A Register Map		312
Appendix B MN102HF75K Flash EEPROM Version		316
B.1	Description	316
B.2	Benefits	317
B.3	Using the PROM Writer Mode	317
B.4	Using the Onboard Serial Programming Mode	319
B.4.1	Configuring the System for Onboard Serial Programming	320

B.4.2	Circuit Requirements for the Target Board	321
B.4.3	Microcontroller Hardware Used in Onboard Serial Programming	322
B.4.3.1	Serial Writer Interface Description	322
B.4.3.2	Serial Writer Interface Block Diagram	322
B.4.4	Microcontroller Memory Map Used During Onboard Serial Programming	323
B.4.4.1	Flash ROM Address Space	323
B.4.4.2	RAM Address Space	324
B.4.5	Microcontroller Clock on the Target Board	324
B.4.6	Setting Up the Onboard Serial Programming Mode	325
B.4.7	Branching to the User Program	327
B.4.7.1	Branching to the Reset Start Routine	327
B.4.7.2	Branching to the Interrupt Start Routine	327
B.5	Reprogramming Flow	328
B.6	Programming Times	328

List of Tables

1-1	General Specifications	26
1-2	Block Diagram Explanation	29
1-3	Pin Functions	32
1-4	Wait Count Settings	36
2-1	Comparison of MN102H75K and MN102L35G Interrupt Features	37
2-2	Handler Preprocessing	39
2-3	Handler Postprocessing	39
2-4	Interrupt Control Registers	46
3-1	Peripheral Function On/Off Switches	75
3-2	CPU Mode Bit Settings	76
4-1	8-Bit Timer Functions and Features	78
4-2	8-Bit Timer Control Registers	87
4-3	16-Bit Timer Functions and Features	89
4-4	Count Direction for 4x Two-Phase Encoder Timing Example	113
4-5	Count Direction for 1x Two-Phase Encoder Timing Example	116
4-6	16-Bit Timer Control Registers	125
5-1	Serial Interface Functions and Features	127
5-2	Example Baud Rate Settings for the UART Mode	129
5-3	Serial Interface Control Registers	140
6-1	ADC Functions and Features	143
6-2	ADC Control Registers	150
7-1	OSD Functions and Features	153
7-2	Power-Saving Control Bits for the OSD	155
7-3	OSDPOFF and OSDREGE Settings	155
7-4	Associated Tiles for Cursor Tile Code Registers	160
7-5	Example Graphics VRAM Settings	161
7-6	Example Text VRAM Settings	163
7-7	VRAM Bit Allocation in Internal RAM	165
7-8	Color Palette Registers	178
7-9	RGB, YM, and YS Output Control Settings	181
7-10	OSD Dot Clock Source Settings	193
7-11	OSD Dot Clock Division Settings	193
7-12	Bit Settings for Controlling the Shuttered Area	194
7-13	Bit Settings for Controlling Shutter Movement	196
7-14	Bit Settings for Controlling Shuttering Effects	198
7-15	EOMON Output Criteria	202
7-16	Cursor Vertical Size Settings	205
7-17	Graphics Vertical Size Settings	206
7-18	Text Vertical Size Settings	206
8-1	Logic Level Conditions for Data Formats	220
8-2	Long and Short Data Identification	220
8-3	Leader Detection Conditions	221
8-4	Differences between SLOW and NORMAL Modes	222

8-5	IR Remote Signal Receiver Registers	223
8-6	HEAMA and 5-/6-Bit Data Pulse Widths	223
9-1	Pins Used for CCD0 and CCD1	227
9-2	Caption decoder register setting	229
9-3	Clamping Reference and Compare Levels	229
9-4	Current Level Control	230
9-5	Control Registers for Clamping Circuit.	230
9-6	Control Registers for Sync Separator Circuit	232
9-7	Control Registers for Data Slicer.	234
9-8	Control Registers for Controller and Sampling Circuit.	234
9-9	Closed-Caption Decoder Register	236
10-1	Register Settings for Internal PWM Pullup.	249
11-1	I/O Port Pins	251
12-1	ROM Correction Address Match and Data Registers	290
13-1	I ² C Bus Terminology.	293
13-2	Operating Modes for Devices on an I ² C Bus	294
13-3	Control Registers for Clamping Circuit.	296
13-4	Registers Settings for SDA0/SCL0 or SDA1/SCL1 Ports.	298
13-5	SDA and SCL Waveform Characteristics	299
13-6	STA and STO Settings.	304
14-1	H Counter Pins.	308
A-1	Register Map: x'007E00' to x'007FFF'	312
A-2	Register Map: x'00FC00' to x'00FDFF'	313
A-3	Register Map: x'00FE00' to x'00FFFF'	314
B-1	Programmable Areas in Each Programming Mode.	316
B-2	PROM Writer Hardware	318
B-3	Pin Descriptions for Target Board–Serial Writer Connection	321
B-4	Flash ROM Address Space in Serial Programming Mode	323
B-5	RAM Address Space in Serial Programming Mode	324
B-6	Microcontroller Clock Frequencies during Serial Programming	324
B-7	Programming Times for PROM and Serial Writers.	328

List of Figures

1-1	Conventional vs. MN102H Series Code Assignments	19
1-2	Three-Stage Pipeline	19
1-3	MN102H Series Interrupt Servicing	20
1-4	Internal Registers, Memory, and Special Function Registers	23
1-5	Address Space	24
1-6	Interrupt Controller Configuration	25
1-7	Interrupt Servicing Sequence.	25
1-8	Functional Block Diagram.	28
1-9	MN102H85K Pin Configuration in Single-Chip Mode.	30
1-10	MN102H75K Pin Configuration in Single-Chip Mode.	31
1-11	Power Supply Wiring.	34
1-12	OSC1and OSC2 Connection Examples.	34
1-13	Reset Pin Connection Example 1	34
1-14	OSDXI and OSDXO Connection Examples	34
1-15	Memory Space in External Extension Mode.	35
2-1	Interrupt Controller Block Diagram	37
2-2	Interrupt Vector Group and Class Assignments.	38
2-3	Interrupt Servicing Time	39
2-4	Block Diagram of External Pin Interrupt.	40
2-5	Timing for External Pin Interrupt Setup (Example)	41
2-6	Block Diagram of Watchdog Timer Interrupt	42
2-7	Timing for Watchdog Timer Interrupt Setup (Example)	43
3-1	CPU State Changes	72
3-2	CPU Clock Switch (NORMAL/SLOW Modes)	73
4-1	Timer Configuration Examples	77
4-2	Block Diagram of 8-Bit Timers.	77
4-3	Timer 0 Block Diagram.	79
4-4	Timer 1 Block Diagram.	79
4-5	Timer 2 Block Diagram.	80
4-6	Timer 3 Block Diagram.	80
4-7	Event Timer Input Timing (8-Bit Timers)	81
4-8	Clock Output and Interval Timer Timing (8-Bit Timers)	81
4-9	Block Diagram of Event Counter Using Timer 0	82
4-10	Event Counter Timing (Timer 0)	83
4-11	Configuration Example of Interval Timer Using Timers 1 and 2	84
4-12	Block Diagram of Interval Timer Using Timers 1 and 2.	84
4-13	Interval Timer Timing (Timers 1 and 2)	86
4-14	Block Diagram of 16-Bit Timers.	88
4-15	Timer 4 Block Diagram.	90
4-16	Timer 5 Block Diagram.	90
4-17	Single-Phase PWM Output Timing (16-Bit Timers).	90
4-18	Single-Phase PWM Output Timing with Data Change (16-Bit Timers).	91
4-19	Two-Phase PWM Output Timing (16-Bit Timers)	91

4-20	One-Shot Pulse Output Timing (16-Bit Timers)	91
4-21	External Count Direction Control Timing (16-Bit Timers)	92
4-22	Event Timer Input Timing (16-Bit Timers)	92
4-23	Single-Phase Capture Input Timing (16-Bit Timers)	92
4-24	Two-Phase Capture Input Timing (16-Bit Timers)	93
4-25	Two-Phase 4x Encoder Timing (16-Bit Timers)	93
4-26	Two-Phase 1x Encoder Timing (16-Bit Timers)	93
4-27	Block Diagram of Event Counter Using Timer 4	94
4-28	Event Counter Timing (Timer 4)	95
4-29	Block Diagram of Single-Phase PWM Output Using Timer 4	96
4-30	Single-Phase PWM Output Timing (Timer 4)	99
4-31	Single-Phase PWM Output Timing with Dynamic Duty Changes (Timer 4)	100
4-32	Block Diagram of Two-Phase PWM Output Using Timer 4	101
4-33	Two-Phase PWM Output Timing (Timer 4)	104
4-34	Two-Phase PWM Output Timing with Dynamic Duty Changes (Timer 4)	105
4-35	Block Diagram of Single-Phase Capture Input Using Timer 4	106
4-36	Single-Phase Capture Input Timing (Timer 4)	107
4-37	Block Diagram of Two-Phase Capture Input Using Timer 4	108
4-38	Two-Phase Capture Input Timing (Timer 4)	110
4-39	Block Diagram of 4x Two-Phase Capture Input Using Timer 5	111
4-40	Configuration Example 1 of 4x Two-Phase Capture Input Using Timer 5	111
4-41	Configuration Example 2 of 4x Two-Phase Capture Input Using Timer 5	111
4-42	4x Two-Phase Encoder Input Timing (Timer 5)	113
4-43	Block Diagram of 1x Two-Phase Capture Input Using Timer 5	114
4-44	Configuration Example 1 of 1x Two-Phase Capture Input Using Timer 5	114
4-45	Configuration Example 2 of 1x Two-Phase Capture Input Using Timer 5	114
4-46	1x Two-Phase Encoder Input Timing (Timer 5)	116
4-47	Block Diagram of One-Shot Pulse Output Using Timer 5	117
4-48	One-Shot Pulse Output Timing (Timer 5)	119
4-49	Block Diagram of External Count Direction Control Using Timer 5	120
4-50	Configuration Example of External Count Direction Control Using Timer 5	120
4-51	External Count Direction Control Timing (Timer 5)	122
4-52	Block Diagram of External Reset Control Using Timer 5	123
4-53	External Reset Control Timing (Timer 5)	124
5-1	Serial Interface Configuration Example	127
5-2	Synchronous Serial Mode Connections	128
5-3	UART Mode Connections	128
5-4	I ² C Mode Connection	128
5-5	Synchronous Serial Transmission Timing	129
5-6	Synchronous Serial Reception Timing	130
5-7	UART Transmission Timing	130
5-8	UART Reception Timing	130
5-9	Block Diagram of UART Transmission Using Serial Interface 0	131
5-10	UART Transmission Timing (Serial Interface 0)	133
5-11	Block Diagram of Serial Interface Clock	135

List of Figures

5-12	Serial Interface Clock Timing	136
5-13	Master Transmitter Timing in I ² C Mode (with ACK)	138
5-14	Master Receiver Timing in I ² C Mode (with ACK)	139
6-1	ADC Architecture	143
6-2	ADC Block Diagram	144
6-3	ADC Timing	144
6-4	Single Channel/Single Conversion Timing	145
6-5	Multiple Channel/Single Conversion Timing	145
6-6	Single Channel/Continuous Conversion Timing	146
6-7	Multiple Channel/Continuous Conversion Timing	146
6-8	Single-Channel A/D Conversion	147
6-9	Timing of Software-Controlled Single-Channel A/D Conversion	148
6-10	Multiple-Channel A/D Conversion	148
6-11	Timing of Hardware-Controlled Intermittent Three-Channel A/D Conversion	150
6-12	Cautions on Analog-to-Digital Converter	152
7-1	OSD Block Diagram	154
7-2	Cursor Tiles in Standard and Extended Modes	159
7-3	Graphic Tiles in Standard and Extended Modes	160
7-4	Graphics Display Example	162
7-5	Text Display Example	164
7-6	VRAM Organization (When GEXTE = 0)	169
7-7	Graphics VRAM Organization for Two Modes	170
7-8	Timing for OSD data	171
7-9	ROM Organization	172
7-10	Graphics ROM Setup Example for a Single Line	173
7-11	Graphics ROM in the Four Color Modes (16W x 16H Tiles)	174
7-12	Graphics ROM in the Four Color Modes (16W x 18H Tiles)	175
7-13	Graphics ROM Organization in 16-Color Mode (16W x 16H Tiles)	176
7-14	Graphics ROM Organization in 8-Color Mode (16W x 16H Tiles)	176
7-15	Graphics ROM Organization in 4-Color Mode (16W x 16H Tiles)	176
7-16	Graphics ROM Organization in 2-Color Mode (16W x 16H Tiles)	176
7-17	Graphics ROM Organization in 16-Color Mode (16W x 18H Tiles)	177
7-18	Graphics ROM Organization in 8-Color Mode (16W x 18H Tiles)	177
7-19	Graphics ROM Organization in 4-Color Mode (16W x 18H Tiles)	177
7-20	Graphics ROM Organization in 2-Color Mode (16W x 18H Tiles)	177
7-21	OSD Signal Waveform	182
7-22	OSD Signal Output Switches	183
7-23	Character Outlining Example	184
7-24	Character Shadowing Example	184
7-25	Box Shadowing Example	185
7-26	Italicizing and Underlining Example	186
7-27	Graphic Tile Size Combinations	187
7-28	Character Size Combinations	188
7-29	HP _{max} of Horizontal Display Position	189
7-30	DMA and Interrupt Timing for the OSD	192

7-31	Shuttered Area Setup Examples	195
7-32	Shutter Movement Setup Examples	197
7-33	Text-Layer Shuttering Setup Examples	199
7-34	Shutter Blanking Setup Examples	200
7-35	Line Shuttering Setup Example	200
7-36	Field Detection Circuit Block Diagram	201
7-37	Field Detection Timing	201
8-1	IR Remote Signal Receiver Block Diagram	217
8-2	IR Remote Signal Noise Filtering	218
8-3	Reception of 8-Bit Data with No Leader	219
8-4	Reception of 8-Bit Data with Leader	219
8-5	Conditions for Detecting Data Formats	220
8-6	Pin Edge Detection	221
9-1	Closed-Caption Decoder Block Diagram	227
9-2	Recommended ADC Configuration	228
9-3	External Connection with Both CCD0 and CCD1 Unused	228
9-4	External Connection with Only CCD1 Unused	228
9-5	Clamping Circuit	229
9-6	Sync Separator Circuit Block Diagram	231
9-7	HSYNC Securement and Interpolation	232
9-8	VSYNC Masking	233
9-9	Data Slice Level Calculation	233
9-10	Sampling Clock Timing Determination	235
9-11	Caption Data Capture Timing	235
9-12	SLSF and SLHD Multiplexing	238
9-13	Backporch Position Setting	244
9-14	Sync Separator Level	244
9-15	BSP and PSP Multiplexing	245
10-1	PWM Output Waveform	249
10-2	PWM Block Diagram	250
11-1	P00/RMIN/IRQ0 (Port 0)	252
11-2	P03/ADIN0 to P07/ADIN4 (Port 0)	253
11-3	P10/ADIN5/IRQ1, P11/ADIN6/IRQ2, and P12/ADIN7/IRQ3 (Port 1)	254
11-4	P13/ADIN8/WDOU and P14/ADIN9/STOP (Port 1)	255
11-5	P15/ADIN10/PWM0 and P16/ADIN11/PWM1 (Port 1)	256
11-6	/PWM2 (Port 1), P20/PWM3, P21/PWM4, P22/PWM5, and P23/PWM6 (Port 2)	257
11-7	P24/TM4IC/SBT1 (Port 2)	258
11-8	P27/TM0IO (Port 2)	259
11-9	P35/DAROUT/R, P36/DAGOUT/G, P37/DABOUT/B (Port 3), and P40/DAYMOUT/YM (Port 4)	260
11-10	P25/TM4IOB/SBI1/SBD1 and P26/TM4IOA/SBO1 (Port 2)	261
11-11	P55 and P56 (Port 5)	262
11-12	P57/SBT0 (Port 5)	263
11-13	P02/SCL1 (Port 0) and P61/SCL0 (Port 6)	264
11-14	P01/SDA1 (Port 1) and P60/SDA0 (Port 6)	265
11-15	P31/CVBS0 and P32/CVBS1 (Port 3)	266

List of Figures

11-16	P30/CLH and P33/CLL (Port 3)	267
11-17	P34/VREF (Port 3)	268
11-18	P41/TM1IO, P42/TM5IOA, and P43/TM5IOB/HI0 (Port 4)	269
11-19	P44/TM5IC/HI1 (Port 4)	269
11-20	P45/OSDXO and P46/OSDXI (Port 4)	270
11-21	P47/ $\overline{\text{HSYNC}}$ (Port 4)	271
11-22	P50/SYSCLK (Port 5)	272
11-23	P51/YS (Port 5)	273
11-24	P52/IRQ4/VI0 (Port 5)	274
11-25	P53/ $\overline{\text{RST}}$ (Port 5)	275
11-26	P54/IRQ5/ $\overline{\text{VSYNC}}$ (Port 5)	276
12-1	ROM Area Schematic Diagram	288
12-2	ROM Correction Flow	288
12-3	ROM Correction Block Diagram	289
13-1	Example of I ² C Bus Application	293
13-2	Connection of Two Microcontrollers to the I ² C Bus	294
13-3	I ² C Bus Interface Operation	295
13-4	I ² C Bus Controller Block Diagram	296
13-5	Pin Control Circuit for the I ² C Bus Controller	298
13-6	SDA and SCL Waveforms	299
13-7	Waveform for Master Transmitter Transitioning to Master Receiver	301
13-8	Waveform for Slave Receiver Transitioning to Slave Transmitter	303
14-1	H Counter Block Diagram	307
14-2	H Counter Operation Example	307
14-3	H Counter Input Signal Timing	308
B-1	Memory Map for Onboard Serial Programming Mode	316
B-2	PROM Writer Hardware Setup	317
B-3	Pin Configuration for Socket Adaptor	317
B-4	Serial Writer Programming Configuration	319
B-5	Serial Writer Hardware Setup	320
B-6	Target Board–Serial Writer Connection	321
B-7	Serial Writer Interface Block Diagram	322
B-8	Timing for the Serial Writer Interface	325
B-9	Load Program Start Flow	326
B-10	Flow of Branch to Reset Start Routine	327
B-11	Flow of Branch to Interrupt Start Routine	327
B-12	EEPROM Programming Flow	328

About This Manual

This manual is intended for assembly-language programming engineers. It describes the internal configuration and hardware functions of the MN102H75K and MN102H85K microcontrollers. Except when discussing differing specifications, this manual refers to the two microcontrollers as a single device : MN102H75K/85K.

Using This Manual

The chapters in this manual deal with the internal blocks of the MN102H75K/85K. Chapters 1 to 5 provide an overview of the MN102H75K/85K's general specifications, interrupts, power modes, timers, and serial connections. Chapters 6 to 10 describe the on-screen display and other specialized functions available with the MN102H75K/85K. Chapter 11 provides the I/O port specifications, chapter 12 describes the ROM correction feature, chapter 13 describes the I²C interface, and chapter 14 describes the H scan line counter. Appendix A provides a register map, and Appendix B describes the flash EEPROM version.

Text Conventions

Where applicable, this manual provides special notes and warnings. Helpful or supplementary comments appear in the sidebar. In addition, the following symbols indicate key information and warnings:



Key information

These notes summarize key points relating to an operation.



Warning

Please read and follow these instructions to prevent damage or reduced performance.

Register Conventions

This manual presents 8- and 16-bit registers in the following format:

REGISTER: Register Name **x'000000'**

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	—	Bit Name	Bit Name	Bit Name	Bit Name	Bit Name	Bit Name	Bit Name	Bit Name	Bit Name	Bit Name	Bit Name	Bit Name	Bit Name
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

The hexadecimal value (x'000000') indicates the register address. The top row of the register diagram holds the bit numbers. Bit 15 is the most significant bit (MSB). The second row holds the bit or field names. A dash (—) indicates a reserved bit. The third row shows the reset values, and the fourth row shows the accessibility. (R = read only, W = write only, and R/W = readable/writable.)

Related Documents

- *MN102H Series LSI User Manual*
(Describes the core hardware.)

- *MN102H Series Instruction Manual*
(Describes the instruction set.)

- *MN102H Series C Compiler User Manual: Usage Guide*
(Describes the installation, commands, and options for the C compiler.)

- *MN102H Series C Compiler User Manual: Language Description*
(Describes the syntax for the C compiler.)

- *MN102H Series C Compiler User Manual: Library Reference*
(Describes the standard libraries for the C compiler.)

- *MN102H Series Cross-Assembler User Manual*
(Describes the assembler syntax and notation.)

- *MN102H Series C Source Code Debugger User Manual*
(Describes the use of the C source code debugger.)

- *MN102H Series Installation Manual*
(Describes the installation of the C compiler, cross-assembler, and C source code debugger and the procedures for using the in-circuit emulator.)

1 General Description

1.1 MN102H Series Overview

The 16-bit MN102H series is the high-speed linear addressing version of the MN10200 series. The new architecture in this series is designed for C-language programming and is based on a detailed analysis of the requirements for embedded applications. From miniaturization to power savings, it provides for a wide range of needs in user systems, surpassing all previous architectures in speed and functionality.

This series uses a load/store architecture for computing within the registers rather than the accumulator system for computing within the memory space, which Panasonic has used in most of its previous major series. The basic instructions are one byte/one machine cycle, drastically shrinking code size and improving compiler efficiency. The circuit is designed for submicron technology, providing optimized hardware and low system power consumption.

The devices in this series contain up to 16 megabytes of linear address space and enable highly efficient program development. In addition, the optimized hardware structure allows for low system-wide power consumption even in large systems.

1.2 MN102H Series Features

Designed for embedded applications, the MN102H series contains a flexible and optimized hardware architecture as well as a simple and efficient instruction set. It provides both economy and speed. This section provides the features of the MN102H series CPU.

■ High-speed signal processing

An internal multiplier multiplies two 16-bit registers for a 32-bit product in a single cycle. In addition, the hardware contains a saturation calculator to ensure that no signal processing is missed and to increase signal processing speed.

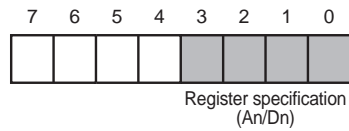
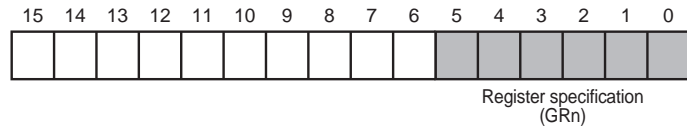
■ Linear addressing for large systems

The MN102H series provides up to 16 megabytes of linear address space. With linear addressing, the CPU does not detect any borders between memory banks, which provides an effective development environment. The hardware architecture is also optimized for large-scale designs. The memory is not divided into instruction and data areas, so operations can share instructions.

■ **Single-byte basic instruction length**

The MN102H series has replaced general registers with eight internal CPU registers divided functionally into four address registers (A0 - A3) and four data registers (D0 - D3). The program can address a register pair in four or less bits, and basic instructions such as register-to-register operations and load/store operations occupy only one byte.

Conventional code assignment for general register instructions



New Panasonic code assignments

Figure 1-1 Conventional vs. MN102H Series Code Assignments

■ **High-speed pipeline throughput**

The MN102H series executes instructions in a high-speed three-stage pipeline: fetch, decode, execute. With this architecture, the MN102H series can execute single-byte instructions in only one machine cycle (50 ns at 40 MHz).

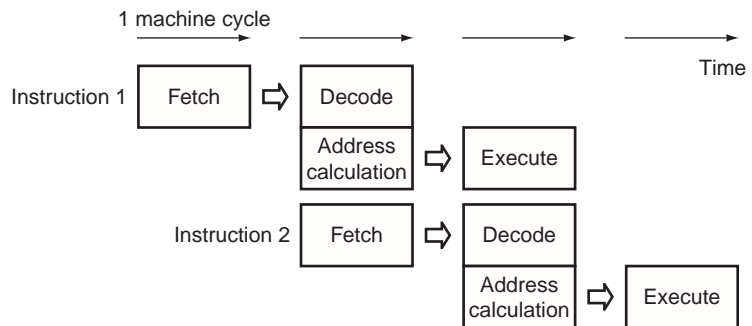


Figure 1-2 Three-Stage Pipeline

■ **Simple instruction set**

The MN102H series uses a streamlined set of 41 instructions, designed specifically for the programming model for embedded applications. To shrink code size, instructions have a variable length of one to seven bytes, and the most frequently used basic instructions are single-byte.

■ **Fast interrupt response**

MN102H series devices can stop executing instructions, even those with long execution cycles, to service interrupts immediately. After an interrupt occurs, the program branches to the interrupt service routine within six cycles or less. The architecture also includes a programmable interrupt handler, which allows you to adjust interrupt servicing speed within the software when necessary, improving real-time control performance.

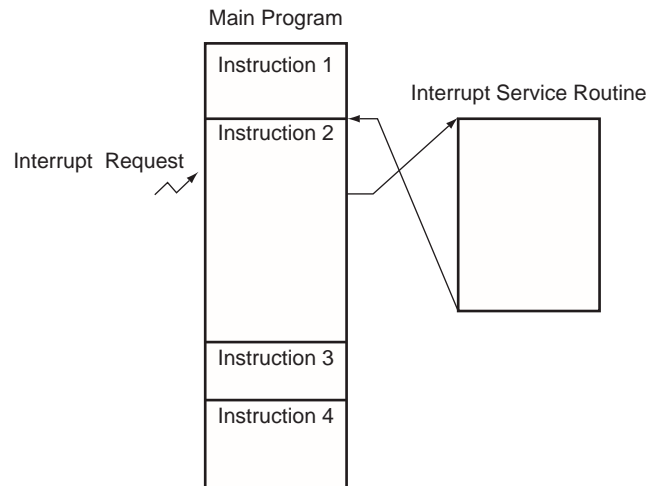


Figure 1-3 MN102H Series Interrupt Servicing

■ **Flexible interrupt control structure**

The interrupt controller supports a maximum of 64 interrupt vectors. (Vectors 0 to 3 are nonmaskable interrupts.) Groups of up to four vectors are assigned to classes, and each class can be set to one of seven priority levels. This gives the software designer great flexibility and fine control. The core is also backwards compatible with software from previous Panasonic peripheral modules.

■ **High-speed, high-functionality external interface**

The MN102H series provides DMA, handshaking, bus arbitration, and other functions that ensure a fast, efficient interface with other devices.

■ **Optimal C-Language development environment**

The MN102H series combines hardware optimized for C language programming with a highly efficient C compiler, resulting in assembly codes the same size as that produced directly in assembly language. This gives designers the advantage of short development time in a C language environment without the trade-off in code size expansion. The PanaXSeries development tools support MN102H series devices.

■ Outstanding power savings

The MN102H series contains separate buses for instructions, data, and peripheral functions, which distributes and reduces load capacitance, dramatically reducing overall power consumption. The series also supports three HALT and STOP modes for even greater power savings.

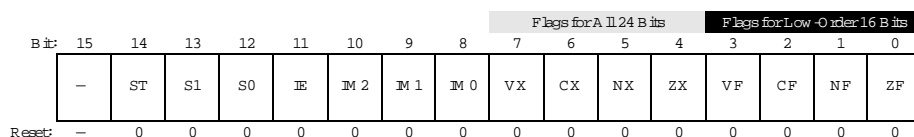
The MN102H series is the flagship product for Panasonic's new high-performance architecture. Panasonic will expand the series as it strives to improve the CPU core's performance and speed, and as it develops devices incorporating ASSPs, ASICs, internal EPROM, and other products to meet the needs of a wide array of embedded designs.

1.3 MN102H Series Description

This section describes the basic architecture and functions of MN102H series devices.

■ Processor status word (PSW)

The PSW contains the operation status flags and interrupt mask levels flags. Note that the PSW for the MN102H series contains flags for both 16- and 24-bit operation results.



ST: Saturation

This bit controls whether or not the CPU calculates a saturation limit for an operation. When it is set to 1, the CPU executes a saturate operation, and when it is 0, the CPU executes a normal operation. The PXST instruction can reverse the meaning of this bit for the next (and only the next) instruction.

S[1:0]: Software control

These bits are the control field for OS software. It is reserved for the OS.

IE: Interrupt enable

If set, this flag enables maskable interrupts; if reset, it disables them.

IM[2:0]: Interrupt mask level

This field indicates the mask level (from 0 to 7) of interrupts that the CPU will accept from its seven interrupt input pins. The CPU will not accept any interrupt from a pin at a higher level than that indicated here.

VX: Extension overflow

If the operation causes the sign bit to change in a 24-bit signed number, this flag is set; otherwise it is reset.

CX: Extension carry flag

If the operation resulted in a carry into (from addition) or a borrow out of (from subtraction or a comparison) the most significant bit, this flag is set; otherwise it is reset.

NX: Extension negative flag

If the most significant bit of the result of an operation has the value 1, this flag is set; if that bit is 0, this flag is reset.

ZX: Extension zero flag

If all bits of the result of an operation have the value 0, this flag is set; otherwise it is reset.

VF: Overflow flag

If the operation causes the sign bit to change in a 16-bit signed number, this flag is set; otherwise it is reset.

CF: Carry flag

If the operation resulted in a carry into (from addition) or a borrow out of (from subtraction or a comparison) bit 15, this flag is set; otherwise it is reset.

NF: Negative flag

If bit 15 of the result of an operation has the value 1, this flag is set; if that bit is 0, this flag is reset.

ZF: Zero flag

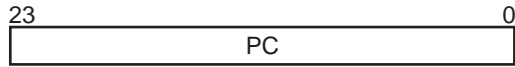
If the least significant 16 bits of the result of an operation have the value 0, this flag is set; otherwise it is reset.

General Description

MN102H Series Description

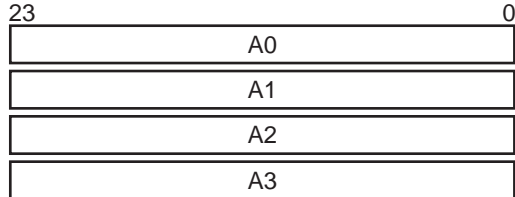
■ Internal registers, memory, and special function registers

Program Counter



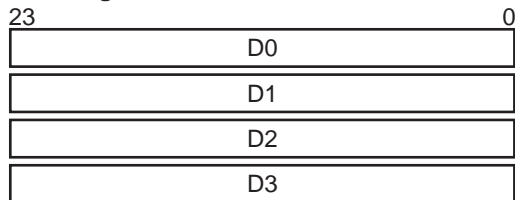
The program counter specifies the 24-bit address of the program instruction being executed.

Address Registers



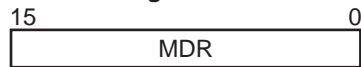
The four address registers specify the location of the data in the memory. A3 is assigned as the stack pointer.

Data Registers



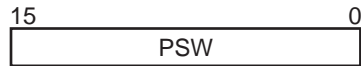
The four data registers handle all arithmetic and logic operations. When byte-length (8-bit) or word-length (16-bit) data is to be transferred to memory or to another register, an instruction adds a zero or sign extension.

Multiplication/Division Register



The dedicated multiplication/division register stores the high-order 16 bits of the 32-bit product of multiplication operations. In division operations, before execution it stores the high-order 16 bits of the 32-bit dividend, and after execution it stores the 16-bit remainder of the quotient.

Processor Status Word



Memory, SFRs, and I/O Ports

ROM	
RAM	
CPUM, EFCR, IAGR	Internal control registers ¹
NMICR, xxICR	Interrupt control registers ¹
SCCTRn, TRXBUFn, SCSTRn	Serial interface registers ¹
ANCTR, ANnBUF	A/D converter registers ¹
TMn, BCn, BRn, ...	Timer/counter registers ¹
MEMMD	Memory control registers ¹
PnOUT, PnIN, PnDIR	I/O port registers ¹

Memory (ROM and RAM), special function registers for controlling peripheral functions, and I/O ports can all be assigned to the same address space.

Note: 1. This allocation is a representative example. Actual memory, peripheral, SFR, and I/O port configuration depends on the product.

Figure 1-4 Internal Registers, Memory, and Special Function Registers

■ **Address space**

The memory in the MN102H series is configured as linear address space. The instruction and data areas are not separated, so the basic segments are internal ROM, internal RAM, and special function registers.

Figure 1-5 shows the address space for the MN102H75K/85K. The internal ROM contains the instructions and the font data for the on-screen display (OSD), in any location. The internal RAM contains the MCU data and the VRAM for the OSD, in any location.

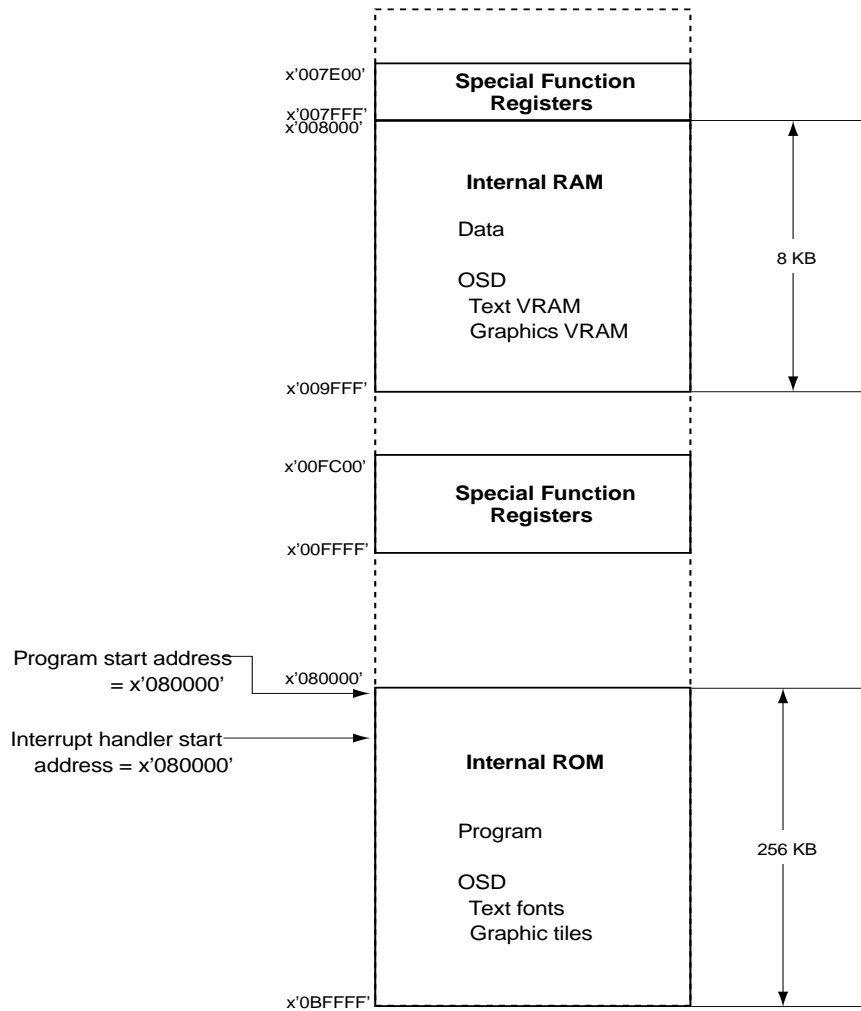


Figure 1-5 Address Space

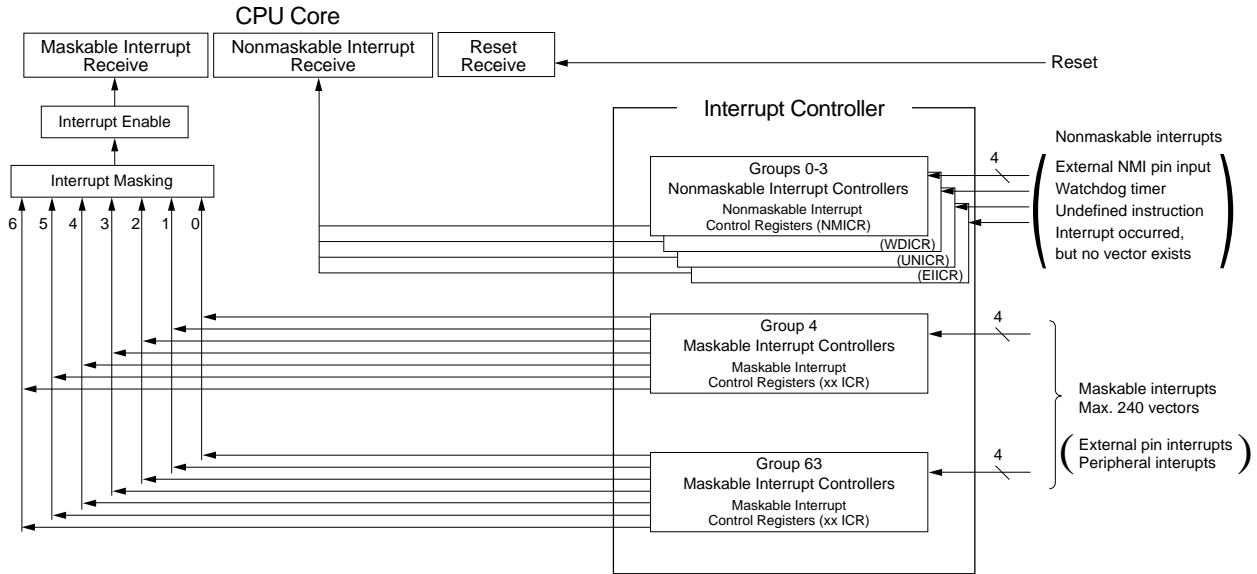
Note: In writing, do not use MOV_B instruction to access Special Function Registers (`x'00FC00'` - `x'00FFFF'`), access by word. In reading, access by byte is possible.

General Description

MN102H Series Description

■ Interrupt controller

An interrupt controller external to the core controls all nonmaskable and maskable interrupts except reset. There are a maximum of sixteen interrupt classes (class 0 to 15). Each class can have up to four interrupt factors and any of seven priority levels.



Note: Interrupt control hardware configuration varies between products.

Figure 1-6 Interrupt Controller Configuration

The CPU checks the processor status word to determine whether or not to accept an interrupt request. If it accepts the request, automatic hardware servicing begins and the contents of the program counter and other necessary registers are pushed to the stack. The program then looks up and branches to the entry address of the interrupt service routine for the interrupt that occurred.

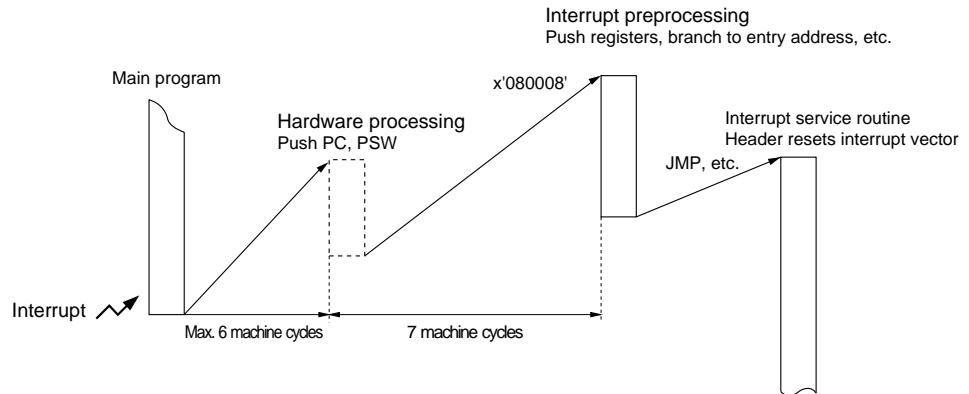


Figure 1-7 Interrupt Servicing Sequence

1.4 General Specifications

Table 1-1 General Specifications

Parameter	Specification
Structure	Internal multiplier (16-bit × 16-bit = 32-bit) and saturate calculator
	Load/store architecture
	Eight registers: <ul style="list-style-type: none"> ◆ Four 24-bit data registers ◆ Four 24-bit address registers
	Other: <ul style="list-style-type: none"> ◆ 24-bit program counter ◆ 16-bit processor status word ◆ 16-bit multiply/divide register
Instruction set	<ul style="list-style-type: none"> ◆ 41 instructions ◆ 6 addressing modes ◆ 1-byte basic instruction length ◆ Code assignment: 1 byte (basic) + 0 to 6 bytes (extension)
Performance	12-MHz internal operating frequency (with a 4-MHz external oscillator)
	Instruction execution clock cycles: <ul style="list-style-type: none"> ◆ Minimum 1 clock cycle (83.3 ns) for register-to-register operations ◆ Minimum 1 clock cycle (83.3 ns) for load/store operations ◆ Minimum 2 clock cycles (167 ns) for branch operations
Pipeline	3-stage: fetch, decode, execute
Address space	<ul style="list-style-type: none"> ◆ Linear address space ◆ Shared instruction/data space
Interrupts	<ul style="list-style-type: none"> ◆ 6 external ◆ 30 internal ◆ 7 priority level settings
Low-power modes	<ul style="list-style-type: none"> ◆ STOP ◆ HALT ◆ SLOW
Oscillation frequency	4 MHz (48 MHz with internal PLL)

Table 1-1 General Specifications

Parameter	Specification
Timer/counters	Four 8-bit timers: <ul style="list-style-type: none"> ◆ Cascading function (forming 16- or 32-bit timers) ◆ Timer output ◆ Selectable clock source (internal or external) ◆ Serial interface clock generation ◆ Start timing generation for analog-to-digital converter
	Two 16-bit timers: <ul style="list-style-type: none"> ◆ Compare/capture registers ◆ Selectable clock source (internal or external) ◆ PWM and one-shot pulse output ◆ Two-phase encoder input (4x or 1x formats)
	16-bit watchdog timer
ROM correction	16 bytes (8-bit × 16)
SYSCLK output	SYSCLK or SYSCLK/2 ¹⁴ (732.42 Hz)
Serial interfaces	<ul style="list-style-type: none"> ◆ Two UART/synchronous serial/I²C (master only) interfaces ◆ One I²C interface (multimaster; 2-channel with 1 internal circuit)
Analog-to-digital converter	<ul style="list-style-type: none"> ◆ 8-bit with 12 channels ◆ Automatic scanning
IR remote signal receiver	<ul style="list-style-type: none"> ◆ Automatic HEAMA / 5-/6-bit detection ◆ 1-bit interrupt
PWM	8-bit with 7 channels (3.3-volt tolerance)
Closed-caption decoder	<ul style="list-style-type: none"> ◆ 2 channels ◆ Internal sync separator
On-screen display	Three-layer format <ul style="list-style-type: none"> ◆ Text layer: 16 × 18 pixels (16 × 26 in closed caption mode), blinking, outlining, shadowing (foreground and background), shutter effect, italics (CC mode), underlining (CC mode) ◆ Graphics layer: 16 × 16 / 16 × 18 pixels ◆ Cursor layer: 16 × 16 / 32 × 32 pixels (1 cursor, displaying one graphic tile)
	Color depth: One 16-color palette out of 4096 colors
	Dot clock <ul style="list-style-type: none"> ◆ Internal PLL frequencies: 12, 16, 24, 32 and 48 MHz ◆ External clock: 16–48 MHz ◆ LC blocking oscillator: 16–48 MHz
I/O ports	66(MN102H75K/F75K) / 50(MN102H85K/F85K)
Package	84-pin-QFP(MN102H75K/F75K) / 64-pin-SDIL(MN102H85K/F85K)

1.5 Block Diagram

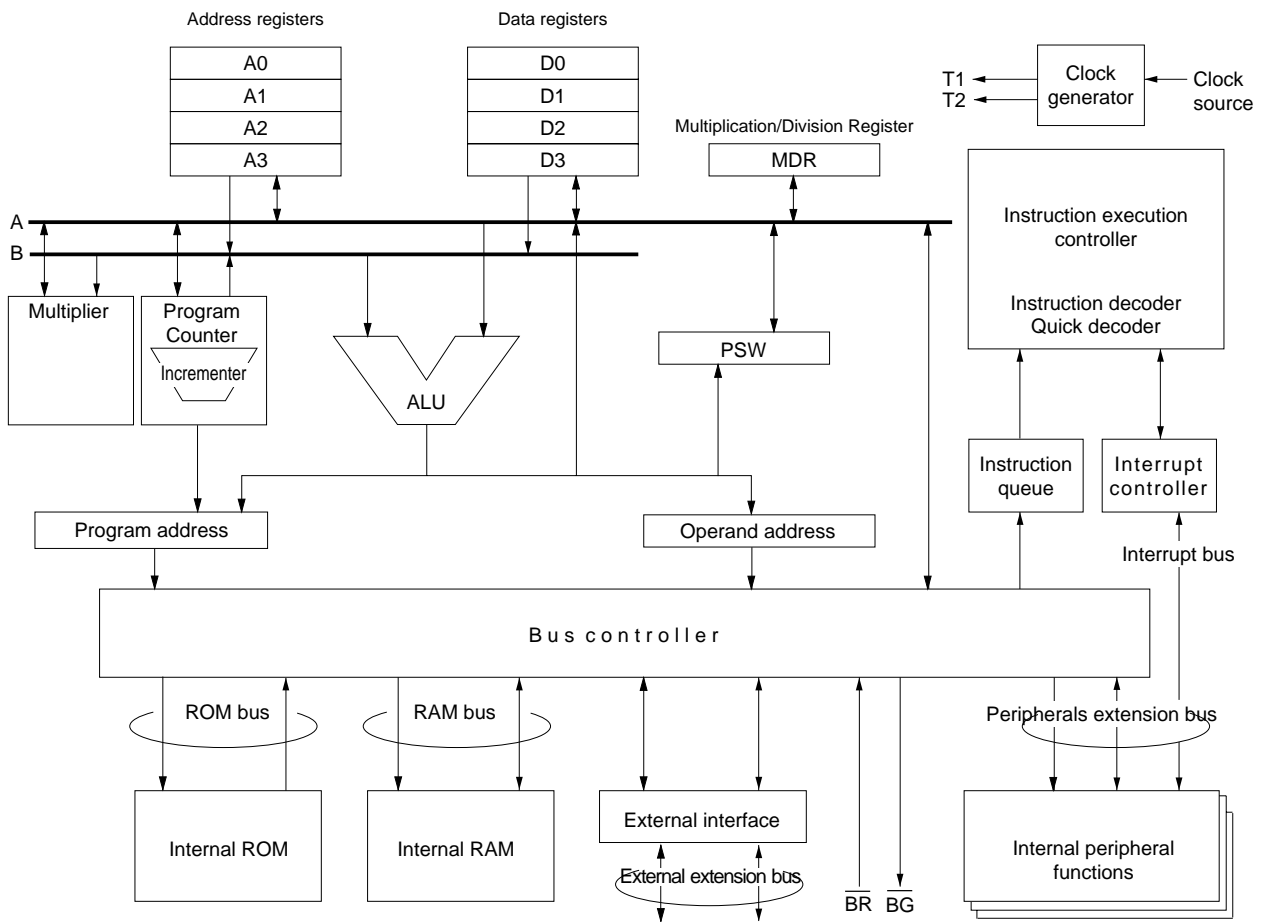


Figure 1-8 Functional Block Diagram

General Description

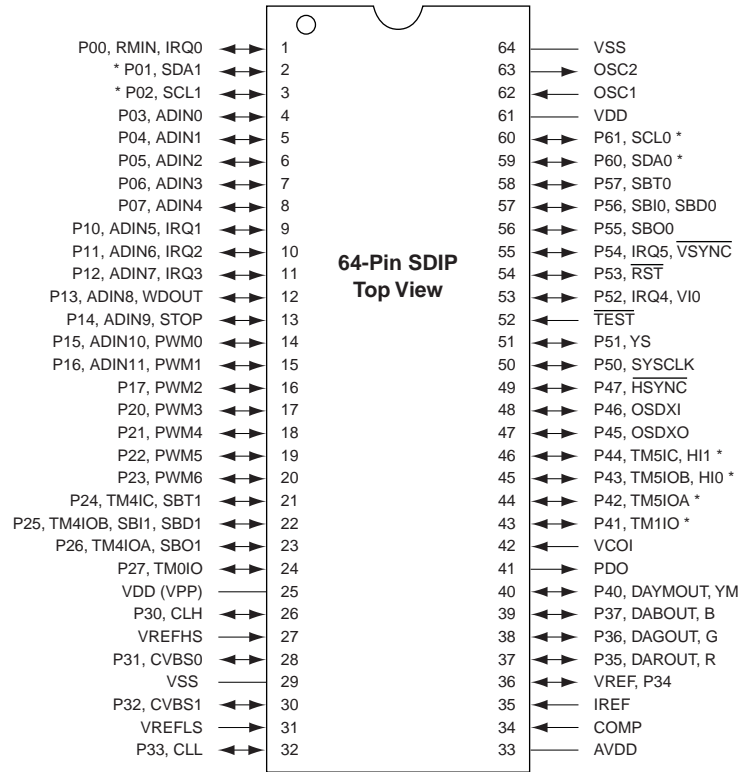
Block Diagram

Table 1-2 Block Diagram Explanation

Block	Description
Clock generator	An oscillation circuit connected to an external crystal supplies the clock to all blocks within the CPU.
Program counter	The program counter generates addresses for queued instructions. Normally it increments based on the sequencer indications, but for branch instructions it is set as the branch head address, and for interrupt servicing, it is set as the result of the ALU operation.
Instruction queue	This block contains up to four bytes of prefetched instructions.
Instruction decoder	The instruction decoder decodes the contents of the instruction queue, generates, in the proper sequence, the control signals necessary for executing the instruction, and controls every block in the chip to execute the instruction.
Quick decoder	This block decodes instructions that are 2 bytes or larger in at a much faster rate than previously possible.
Instruction execution controller	This block controls the operation of every block within the CPU using the results from the instruction decoder and interrupt requests.
ALU	Arithmetic and logic unit. This block calculates the operand addresses for arithmetic operations, logic operations, shift operations, relative indirect register addressing, indexed addressing, and indirect register addressing.
Multiplier	This block multiplies 16 bits \times 16 bits = 32 bits.
Internal ROM and RAM	These memory blocks contain the program, data, and stack areas.
Address registers (An)	The address registers store the addresses in memory to be accessed in data transfers. In relative indirect, indexed, and indirect addressing modes, they store the base address.
Operation registers (Dn, MDR)	The data registers store data to be transferred to memory and results of operations. In indexed and indirect addressing modes, they store the offset address. The multiplication/division register stores data for multiplication and division operations.
PSW	The processor status word contains flags that indicate the status of the CPU interrupt controller and provide information about operation results.
Interrupt controller	This block detects interrupt requests from peripheral function blocks and requests the CPU to service the interrupt.
Bus controller	This block controls the connection between the CPU's internal and external buses. It also contains a bus arbitration function.
Internal peripheral functions	MN102H series devices contain a wide range of internal peripheral devices, such as timers, serial interfaces, ADCs, and DACs.

1.6 Pin Descriptions

1.6.1 MN102H85K Pin Description



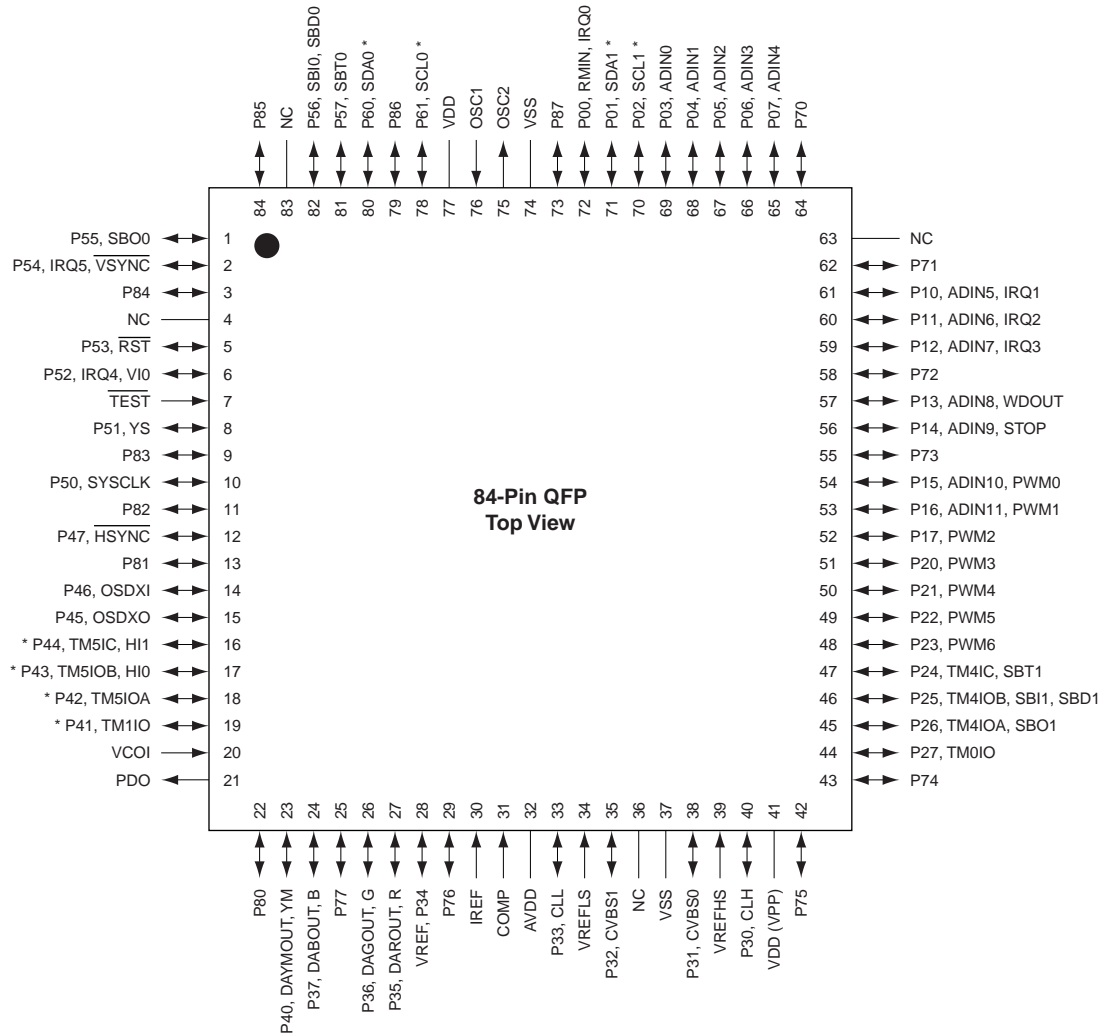
- Notes:
1. Pins marked with an asterisk (*) are N-channel, open-drain pins.
 2. Pin 25 is V_{DD} in the MN102H85K and V_{PP} in the MN102HF85K.

Figure 1-9 MN102H85K Pin Configuration in Single-Chip Mode

General Description

Pin Descriptions

1.6.2 MN102H75K Pin Description



- Notes:
1. Pins marked with an asterisk (*) are N-channel, open-drain pins.
 2. Pin 41 is V_{DD} in the MN102H75K and V_{PP} in the MN102HF75K.

Figure 1-10 MN102H75K Pin Configuration in Single-Chip Mode

Table 1-3 Pin Functions

Block		Pin Name	I/O	Pin Count	Description
Power		V _{DD}	I	1	Voltage supply
		V _{SS}	I	2	Ground reference
		AV _{DD}	I	1	Analog voltage supply
		V _{DD} /V _{PP}	I	1	Voltage supply: V _{DD} in mask ROM version and V _{PP} in EEPROM version
Clocks		SYSCLK	O	1	System clock output
		OSC1	I	1	Oscillator input connection (with internal PLL)
		OSC2	O	1	Oscillator output connection (with internal PLL)
		OSDXI	I	1	OSD oscillator input connection (alt. function: P46)
		OSDXO	O	1	OSD oscillator output connection (alt. function: P45)
Reset		R _{ST}	I/O	1	Reset (alt. function: P53)
Interrupts (external)		IRQ ₀ –IRQ ₅	I	6	External interrupt request to microcontroller (alt. functions: P00, P10, P11, P12, P52, P54)
OSD		HSYNC	I	1	Horizontal sync signal input
		VS _{YNC}	I	1	Vertical sync signal input
		YS	O	1	Video signal cut
Timers	16-bit (2)	TMnIOA (n=4,5)	I/O	2	Input capture/output compare A
		TMnIOB (n=4,5)	I/O	2	Input capture/output compare B
		TMnIC (n=4,5)	I	2	Timer/counter clear signal
	8-bit (4)	TMnIO (n=0,1)	I/O	2	Timer clock input/timer output
Serial interfaces (2)		SBI0/SBI1	I	2	Serial data input
		SBD0/SBD1	I/O	2	Serial data input
		SBO0/SBO1	I/O	2	Serial data output
		SBT0/SBT1	I/O	2	Serial clock signal
I ² C interfaces (2)		SDA0/SDA1	I/O	2	I ² C data
		SCL0/SCL1	I/O	2	I ² C clock
IR remote signal receiver		RMIN	I	1	Remote signal input
PWM (8-bit, 7-channel)		PWM0–PWM6	O	7	Pulse width modulator output

General Description

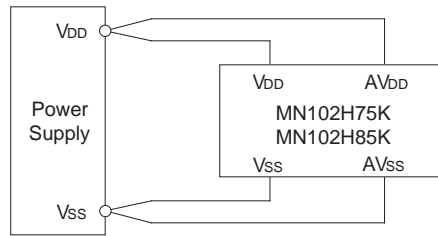
Pin Descriptions

Table 1-3 Pin Functions (Continued)

Block	Pin Name	I/O	Pin Count	Description
I/O ports MN102H75K/HF75K: total 66 pins MN102H85K/HF85K: total 50 pins	P00–P07	I/O	8	General-purpose port 0 I/O
	P10–P17	I/O	8	General-purpose port 1 I/O
	P20–P27	I/O	8	General-purpose port 2 I/O
	P30–P37	I/O	8	General-purpose port 3 I/O
	P40–P47	I/O	8	General-purpose port 4 I/O
	P50–P57	I/O	8	General-purpose port 5 I/O
	P60–P61	I/O	2	General-purpose port 6 I/O
I/O ports only in MN102H75K/F75K	P70–P77	I/O	8	General-purpose port 7 I/O
	P80–P87	I/O	8	General-purpose port 8 I/O
Closed-caption decoders (2)	CVBS0/CBVS1	I	2	Composite video signal input
	CLH	I	1	Clamp level high input
	CLL	I	1	Clamp level low input
	VREFHS	I	1	CCD reference voltage input
	VREFLS	I	1	CCD reference voltage input
A/D converter (12-channel)	ADIN0–ADIN11	I	12	Analog signal input
D/A converter (4-bit, 4-channel)	DAROUT ⁽¹⁾	O	1	DAC output (red)
	DAGOUT ⁽¹⁾	O	1	DAC output (green)
	DABOUT ⁽¹⁾	O	1	DAC output (blue)
	DAYMOUT ⁽¹⁾	O	1	DAC output (YM)
	IREF	I	1	Resistance connection for DAC bias current setting
	VREF	I	1	DAC reference voltage connection
	COMP	I	1	DAC phase compensator connection
PLL	VCOI	I	1	Internal VCO input (external LPF input)
	PDO	O	1	Internal phase compare output (external LPF output)
Mode	STOP	O	1	STOP mode status signal
	WDOUT	O	1	Watchdog timer overflow signal
Test	$\overline{\text{TEST}}$	I	1	Test pin (Connect to ground.)

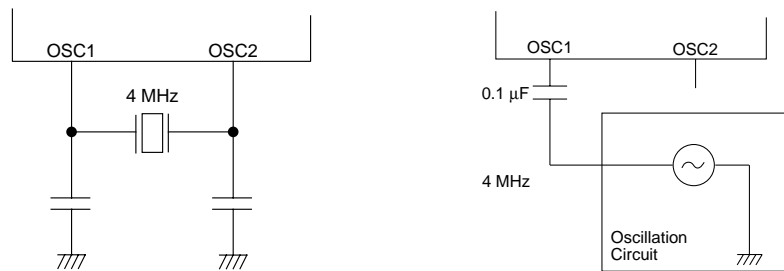
Notes: 1. When DAROUT, DAGOUT, DABOUT, and DAYMOUT are used for digital output, their names are R, G, B, and YM, respectively.

■ Considerations for power supply, clock, and reset pins



Note: If the circuit uses the same power supply for digital and analog supplies, connect the pins in the location closest to the power supply.

Figure 1-11 Power Supply Wiring



Note: The capacitance values vary depending on the oscillator.

Figure 1-12 OSC1 and OSC2 Connection Examples

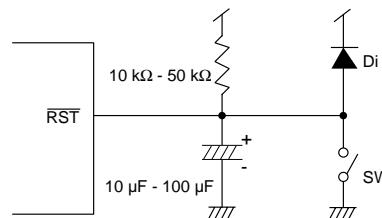


Figure 1-13 Reset Pin Connection Example 1



Note: The capacitance values vary depending on the oscillator.

Figure 1-14 OSDXI and OSDXO Connection Examples

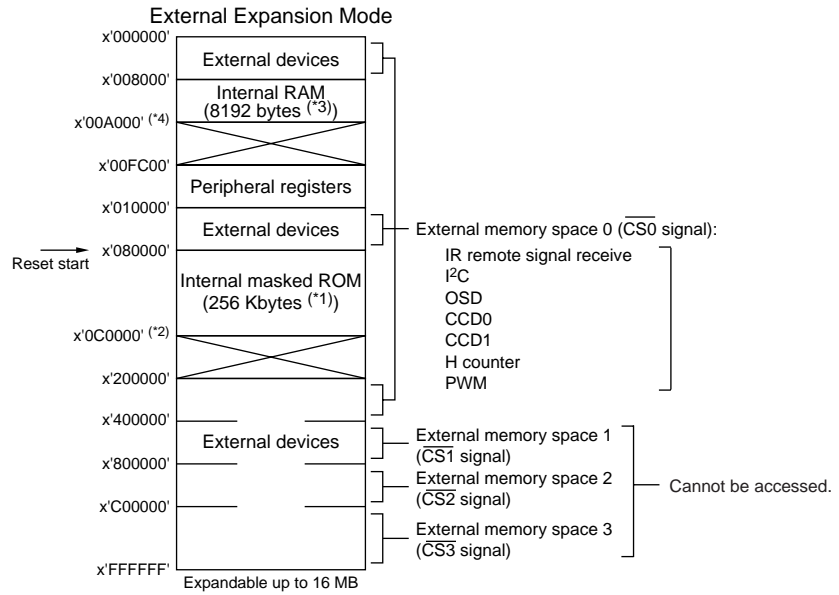
■ Connection the PLL circuit

The MN102H75K/85K contains an internal PLL circuit. To use this circuit, you must connect it to an external (lag-lead) filter.

1.7 Bus Interface

1.7.1 Description

The bus interface operates in external extension mode. Figure 1-15 provides the memory space for the MCU in this mode.



	*1	*2	*3	*4
MN102H75K	256 Kbytes	x'0C0000'	8192 bytes	x'00A000'

Figure 1-15 Memory Space in External Extension Mode

1.7.2 Bus Interface Control Registers

The external memory wait register (EXWMD) and memory mode register 1 (MEMMD1) control the bus interface.

EXWMD: External Memory Wait Register

x'00FF80'

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	EW 33	EW 32	EW 31	EW 30	EW 23	EW 22	EW 21	EW 20	EW 13	EW 12	EW 11	EW 10	EW 03	EW 02	EW 01	EW 00
Reset:	1	1	1	0	1	1	1	0	1	1	1	0	1	1	1	0
R/W :	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

EW[33:30], EW[23:20], EW[13:10], EW[03:00]

These fields contain the wait settings for external memory spaces 3, 2, 1, and 0, respectively. One wait corresponds to one instruction cycle. When the external oscillator is 4 MHz, one wait is 83 ns.

The OSD, VBI0, VBI1, I2C, IR remote signal receiver, and H counter blocks apply to external memory space 0.

Table 1-4 Wait Count Settings

EW[n3:n0] Setting	Wait Count	Cycles
0000	0.0	1.0
0001	Reserved	
0010	1.0	2.0
0011	Reserved	
0100	2.0	3.0
0101	Reserved	
0110	3.0	4.0
0111	Reserved	
1000	4.0	5.0
1001	Reserved	
1010	5.0	6.0
1011	Reserved	
1100	6.0	7.0
1101	Reserved	
1110	7.0	8.0
1111	Reserved	

MEMMD1: Memory Mode Register 1

x'00FF82'

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	EB 31	EB 32	EB 21	EB 20	EB 11	EB 10	EB 01	EB 00	BRS1	BRS0	BRC3	BRC2	BRC1	BRC0	IOW 1	IOW 0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
R/W :	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Write 0s to bits 15 to 2.

IOW[1:0]: Wait setting for internal I/O space

- 00: 1 wait
- 01: Reserved
- 10: 2 waits
- 11: 3 waits

2 Interrupts

2.1 Description

The most important factor in real-time control is an MCU's speed in servicing interrupts. The MN102H75K/85K has an extremely fast interrupt response time due to its ability to abort instructions, such as multiply or divide, that require multiple clock cycles. The MN102H75K/85K re-executes an aborted instruction after returning from the interrupt service routine.

This section describes the interrupt system in the MN102H75K/85K. The MN102H75K/85K contains 36 interrupt group controllers. Each controls a single interrupt group. Because each group contains only one interrupt vector, the MN102H75K/85K can handle interrupts much quicker than previously possible. Each interrupt group belongs to one of twelve classes, which defines its interrupt priority level.

With the exception of reset interrupts, all interrupts from timers, other peripheral circuits, and external pins must be registered in an interrupt group controller. Once they are registered, interrupt requests are sent to the CPU in accordance with the interrupt mask level (0 to 6) set in the interrupt group controller. Groups 1 to 3 are dedicated to system interrupts. Table 2-1 compares the interrupt parameters of the MN102H75K/85K to those of the MN102L35G, the comparable MCU in the previous generation of the 16-bit series.

Table 2-1 Comparison of MN102H75K/85K and MN102L35G Interrupt Features

Parameter	MN102L35G	MN102H75K/85K
Interrupt groups (IAGR group numbers)	4 vectors per group (Separated by interrupt service routine)	1 vector per group (Group number generated for each interrupt)
Interrupt response time	Good	Excellent
Interrupt level settings	4 vectors per level	4 vectors per level
Software compatibility	—	Easily modified

The MN102H75K/85K has six external interrupt pins. Set the interrupt condition (positive edge, negative edge, either edge, or active low) in the EXTMD register.

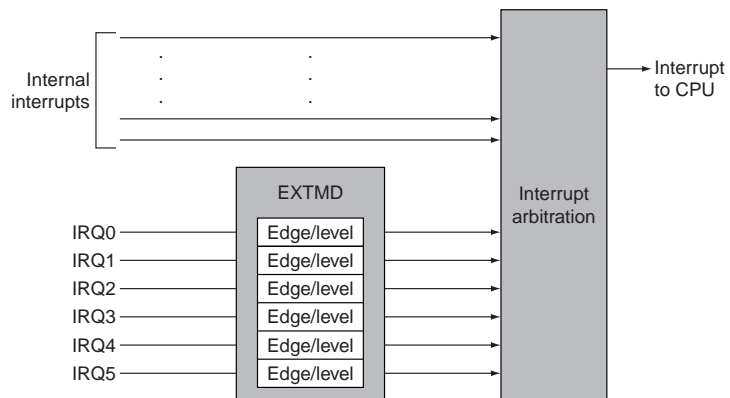


Figure 2-1 Interrupt Controller Block Diagram

Interrupts

Description

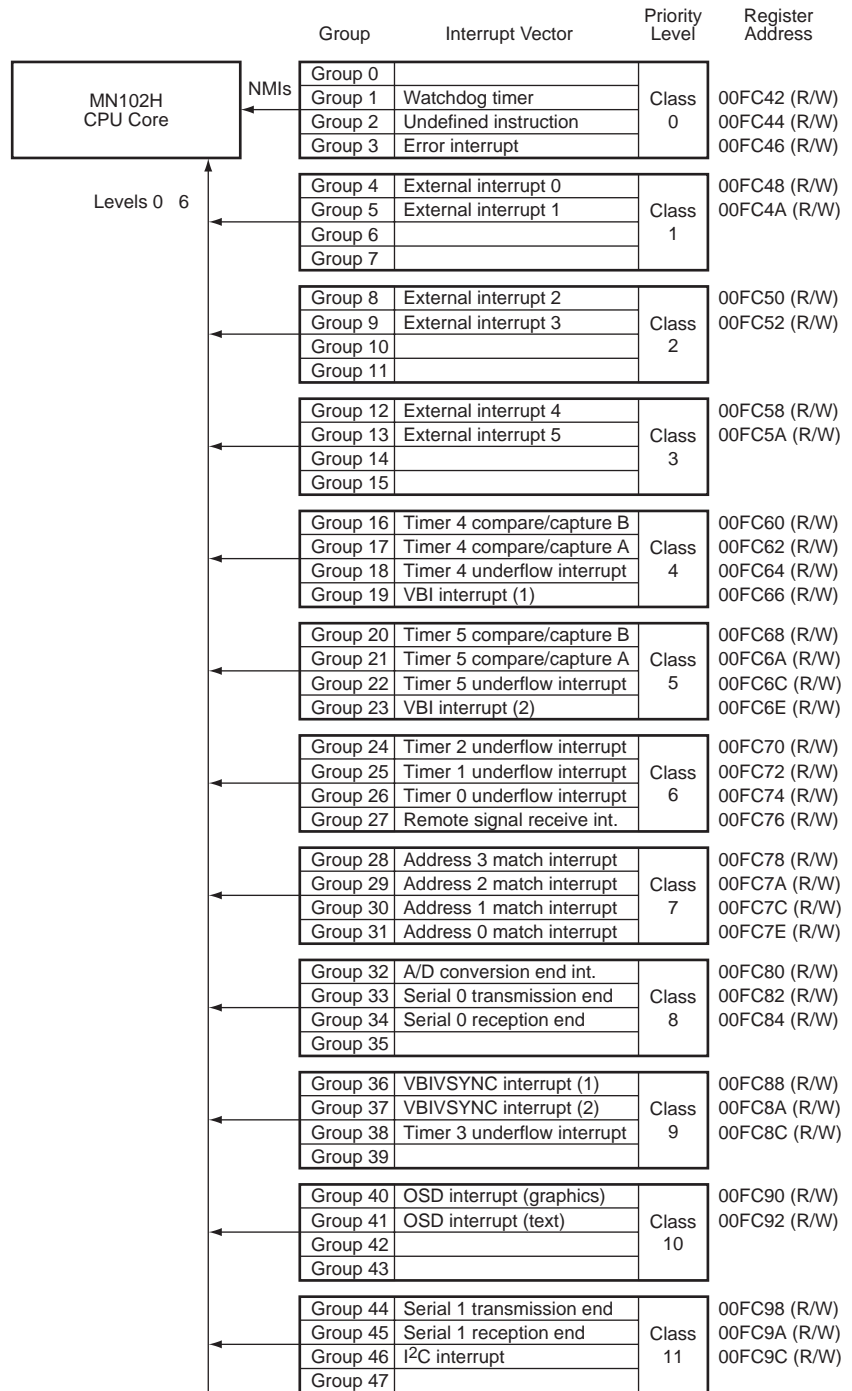


Figure 2-2 Interrupt Vector Group and Class Assignments

Interrupts

Description

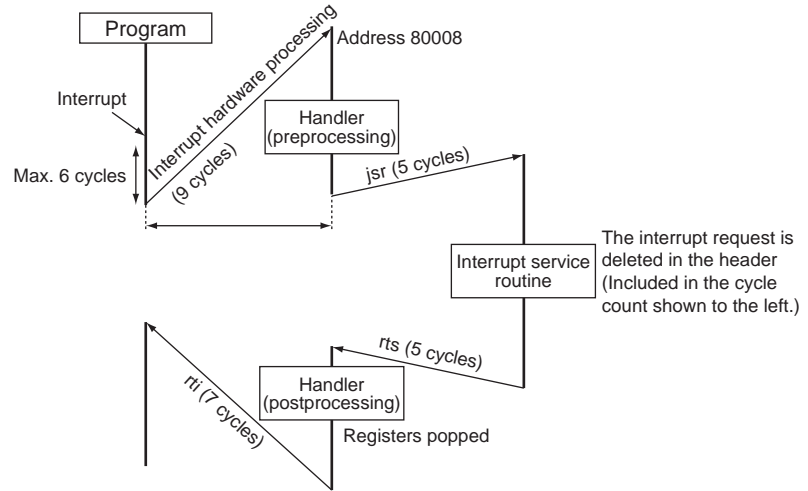


Figure 2-3 Interrupt Servicing Time

Table 2-2 Handler Preprocessing

Sequence	Assembler	Bytes	Cycles
Push registers	add -8, A3	2	1
	mov A0, (A3)	2	2
	movx D0, (4, A3)	3	3
Interrupt ACK	mov (FC0E), D0	3	1
Generate header address for interrupt service routine	mov BASE, A0	3	1
	mov (D0, A0), A0	2	2
Branch	jsr (A0)	2	5
Total		17	15

Table 2-3 Handler Postprocessing

Sequence	Assembler	Bytes	Cycles
Pop registers	mov (A3), A0	2	2
	movx (4, A3), D0	3	3
	add 8, A3	2	1
Total		7	6

2.2 Interrupt Setup Examples

2.2.1 Setting Up an External Pin Interrupt

In this example, an interrupt occurs on a falling-edge signal from the IRQ0 (P00) external interrupt pin, and the interrupt priority level is 5.

On reset, the external edge setting in the EXTMD register is low (b'00' = active-low interrupt), and the IQ0IR bit of the IQ0ICL register is 0.

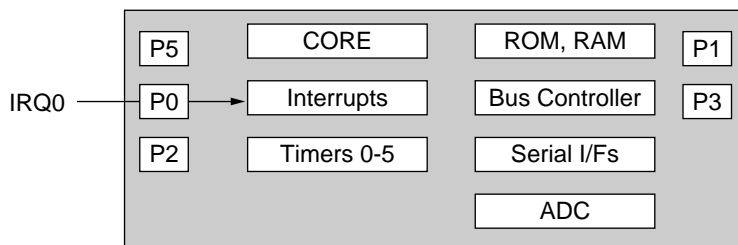


Figure 2-4 Block Diagram of External Pin Interrupt

■ Enabling external interrupt 0

1. Set the interrupt conditions for the IRQ0 (P00) pin. For this example, set the IQ0TG[1:0] bits of EXTMD to b'10' (negative-edge-triggered interrupt).

EXTMD (example) **x'00FCF8'**

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	0	0	0	0	IQ5TG 1	IQ5TG 0	IQ4TG 1	IQ4TG 0	IQ3TG 1	IQ3TG 0	IQ2TG 1	IQ2TG 0	IQ1TG 1	IQ1TG 0	IQ0TG 1	IQ0TG 0
Setting:	—	—	—	—	0	0	0	0	0	0	0	0	0	0	1	0

2. Cancel any existing interrupt requests and enable IRQ0 interrupts. To do this, set the IQ0IR bit of IQ0ICL to 0, set the IQ0LV[2:0] bits of IQ0ICH to b'101' (priority level 5), and set the IQ0IE bit to 1.

IQ0ICL (example) **x'00FC48'**

Bit:	7	6	5	4	3	2	1	0
	—	—	—	IQ0IR	—	—	—	IQ0ID
Setting:	0	0	0	0	0	0	0	0

IQ0ICH (example) **x'00FC49'**

Bit:	7	6	5	4	3	2	1	0
	—	IQ0LV2	IQ0LV1	IQ0LV0	—	—	—	IQ0IE
Setting:	0	1	0	1	0	0	0	1

Interrupts

Interrupt Setup Examples

- Enable interrupts by writing a 1 to the interrupt enable flag (IE) in the PSW and setting the interrupt masking level (IM[2:0]) to 7 (b'111').

Now if a falling edge occurs on IRQ0 (P00), an interrupt will occur. If the CPU accepts the interrupt, the program branches to address x'080008'.

■ Servicing the interrupt

The main program normally generates and branches to the interrupt start address.

During the interrupt service routine, prevent the CPU from accepting any other maskable interrupts by setting the IM[2:0] and IE bits of the PSW to 0.

To accept the same interrupt during the interrupt service routine, clear IR flag at the beginning of it.

- During interrupt preprocessing, read the accepted interrupt group number register (IAGR) to determine the interrupt group (group 4, in this case).
- Branch to the interrupt service routine.
- At the beginning of the interrupt service routine, clear the IQ0IR bit in IQ0ICL to 0.
- After the service routine ends, return to the main program with the RTI instruction.

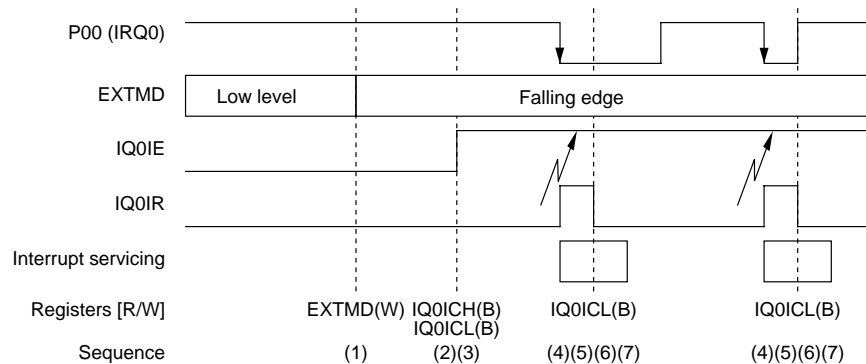


Figure 2-5 Timing for External Pin Interrupt Setup (Example)



The watchdog timer interrupt is provided for detecting and handling racing. Normal operation is not guaranteed if the program returns after a watchdog interrupt. For actions requiring returns, use a timer interrupt.

2.2.2 Setting Up a Watchdog Timer Interrupt

In this example, a watchdog timer reset occurs. The watchdog timer starts running after a reset, when the NWDEN flag in the CPU mode register (CPUM) is enabled (set to 0). When the watchdog timer overflows, a nonmaskable interrupt occurs. This means that the watchdog timer must be cleared in the main program.

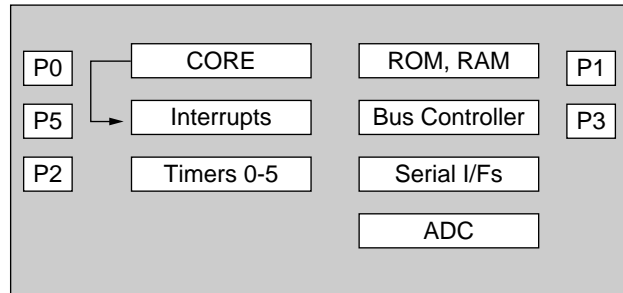


Figure 2-6 Block Diagram of Watchdog Timer Interrupt

■ Enabling watchdog timer interrupts

1. Enable interrupts by writing a 1 to the interrupt enable flag (IE) in the PSW and setting the interrupt masking level (IM[2:0]) to 7 (b'111').
2. Activate the watchdog timer by clearing the NWDEN bit of the CPUM register. Set the time limit for the racing detection function in the WDM[1:0] field.

CPUM (example)

x'00FC00'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	NW DEN	WDM 1	WDM 0	—	—	—	—	—	—	—	—	OSC ID	STOP	HALT	OSCI	OSCO
Setting:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

If WDM[1:0] = 00, a watchdog interrupt occurs when the watchdog timer counts 2^{16} cycles (5.4613 ms at 4-MHz f_{OSC} /12-MHz f_{SYSCLK}). The WDM settings have the following meanings:

- 00: 2^{16} (5.46 ms)
- 01: 2^4 (1.33 μ s)
- 10: 2^{12} (0.34 ms)
- 11: 2^{14} (1.37 ms)

The main program normally clears the watchdog timer prior to a watchdog interrupt.

■ Clearing the watchdog timer

3. Set the NWDEN bit in CPUM to 1, then immediately reset it to 0. The watchdog timer clears to 0 when NWDEN is 1.

Interrupts

Interrupt Setup Examples

The main program normally generates and branches to the interrupt start address.

If the CPU accepts an interrupt, the program branches to address x'080008'.

The oscillator delay timer shares the counter for the watchdog timer. The oscillator delay timer is activated when the circuit exits the STOP mode, so the program must clear the WDID flag to 0 prior to entering the STOP mode. It must also re-clear WDID after returning to NORMAL mode. For further details, see section 2-6, "Standby Function," in the *MN10200 Series Linear Addressing Version LSI User Manual*.

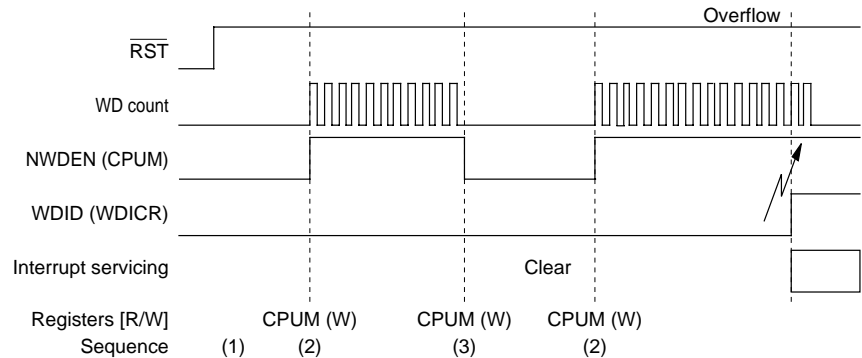


Figure 2-7 Timing for Watchdog Timer Interrupt Setup (Example)

2.3 Interrupt Control Registers

A control register is assigned to each interrupt vector group. Except for the class 0 registers (WDICR, PIICR, and EIICR), the control registers allow you to enable and set the priority level for interrupt groups.

Below is the general format of the registers in class 0 and classes 1 to 11.

Class 0 (X):

WD (watchdog overflow interrupts)

PI (undefined instruction interrupts)

EI (interrupt error interrupts)

XICR (System Interrupt)

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	ID

ID: Interrupt detect flag

0: Interrupt undetected

1: Interrupt detected

Classes 1–11 (X):

IQ (external interrupts)

TM (timer interrupts)

SC (serial interrupts)

I2C (I2C interrupts)

OSD (OSD interrupts)

AN (A/D conversion end interrupts)

RMC (remote signal receive interrupts)

VBI (VBI interrupts)

ADM (address match interrupts)

XnICH (System Interrupt)

Bit:	7	6	5	4	3	2	1	0
	—	LV2	LV1	LV0	—	—	—	IE

LV[2:0]: Interrupt priority level

Sets the priority from 0 to 6 (000 = 0, 001 = 1, etc.). When LV = 7, interrupts are not serviced.

Note that some registers do not contain the LV field. In this case, these bits always read 0.

IE: Interrupt enable flag

0: Disable

1: Enable

Interrupts

Interrupt Control Registers

XnICL (System Interrupt)

Bit:	7	6	5	4	3	2	1	0
	—	—	—	IR	—	—	—	ID

IR: Interrupt request flag

0: No interrupt requested

1: Interrupt requested

ID: Interrupt detect flag

0: Interrupt undetected

1: Interrupt detected

The following is an example program setting an interrupt group's priority level (LV field) and enabling the interrupt group (IE) in the interrupt control register (XnICH). Note that interrupts must be disabled during this routine.

Example 2-1 Setting the Interrupt Priority Level

```
... ;
and 0xf7ff,psw ;Clear the IE bit of the PSW.
nop ; Inserted to ensure that IE clears
nop ; completely, so XnICH is accessible.
mov d0,(XnICH) ;Write to LV/IE
mov (XnICH),d0 ;Synchronize with the store buffer.
or 0x0800,psw ;Set the IE bit of the PSW.
... ;
```

The program does not need to clear the IE bit of the PSW to disable interrupts during interrupt servicing, since the interrupt service routine has already cleared it.

You can replace the NOP instructions in the example above with any instruction except for those that modify the PSW IE bit or the LV or IE bits of an XnICH register. Inserting any of these instructions would cause interrupt error to occur.

The example includes two NOP instructions to ensure that the minimum number of cycles required for a write to IE have passed. However, you can also insert more than two NOPs.

Table 2-4 provides a list of the interrupt control registers, and a description of the fields in each register follows.

Table 2-4 Interrupt Control Registers

Register	Address	R/W	Description
IAGR	x'00FC0E'	R	Accepted interrupt group number register
WDICR	x'00FC42'	R/W	Watchdog interrupt control register
PIICR	x'00FC44'	R/W	Undefined instruction interrupt control register
EIICR	x'00FC46'	R	Interrupt error interrupt control register
EXTMD	x'00FCF8'	R/W	External interrupt mode register
IQ0ICL	x'00FC48'	R/W	External interrupt 0 interrupt control register (low)
IQ0ICH	x'00FC49'	R/W	External interrupt 0 interrupt control register (high)
IQ1ICL	x'00FC4A'	R/W	External interrupt 1 interrupt control register (low)
IQ1ICH	x'00FC4B'	R/W	External interrupt 1 interrupt control register (high)
IQ2ICL	x'00FC50'	R/W	External interrupt 2 interrupt control register (low)
IQ2ICH	x'00FC51'	R/W	External interrupt 2 interrupt control register (high)
IQ3ICL	x'00FC52'	R/W	External interrupt 3 interrupt control register (low)
IQ3ICH	x'00FC53'	R/W	External interrupt 3 interrupt control register (high)
IQ4ICL	x'00FC58'	R/W	External interrupt 4 interrupt control register (low)
IQ4ICH	x'00FC59'	R/W	External interrupt 4 interrupt control register (high)
IQ5ICL	x'00FC5A'	R/W	External interrupt 5 interrupt control register (low)
IQ5ICH	x'00FC5B'	R/W	External interrupt 5 interrupt control register (high)
TM4CBICL	x'00FC60'	R/W	Timer 4 compare/capture B interrupt control register (low)
TM4CBICH	x'00FC61'	R/W	Timer 4 compare/capture B interrupt control register (high)
TM4CAICL	x'00FC62'	R/W	Timer 4 compare/capture A interrupt control register (low)
TM4CAICH	x'00FC63'	R/W	Timer 4 compare/capture A interrupt control register (high)
TM4UDICL	x'00FC64'	R/W	Timer 4 underflow interrupt control register (low)
TM4UDICH	x'00FC65'	R/W	Timer 4 underflow interrupt control register (high)
VBIICL	x'00FC66'	R/W	VBI (1) interrupt control register (low)
VBIICH	x'00FC67'	R/W	VBI (1) interrupt control register (high)
TM5CBICL	x'00FC68'	R/W	Timer 5 compare/capture B interrupt control register (low)
TM5CBICH	x'00FC69'	R/W	Timer 5 compare/capture B interrupt control register (high)
TM5CAICL	x'00FC6A'	R/W	Timer 5 compare/capture A interrupt control register (low)
TM5CAICH	x'00FC6B'	R/W	Timer 5 compare/capture A interrupt control register (high)
TM5UDICL	x'00FC6C'	R/W	Timer 5 underflow interrupt control register (low)
TM5UDICH	x'00FC6D'	R/W	Timer 5 underflow interrupt control register (high)
VBIWICL	x'00FC6E'	R/W	VBI (2) interrupt control register (low)
VBIWICH	x'00FC6F'	R/W	VBI (2) interrupt control register (high)
TM2UDICL	x'00FC70'	R/W	Timer 2 underflow interrupt control register (low)
TM2UDICH	x'00FC71'	R/W	Timer 2 underflow interrupt control register (high)
TM1UDICL	x'00FC72'	R/W	Timer 1 underflow interrupt control register (low)
TM1UDICH	x'00FC73'	R/W	Timer 1 underflow interrupt control register (high)
TM0UDICL	x'00FC74'	R/W	Timer 0 underflow interrupt control register (low)
TM0UDICH	x'00FC75'	R/W	Timer 0 underflow interrupt control register (high)
RMCICL	x'00FC76'	R/W	Remote signal receive interrupt control register (low)
RMCICH	x'00FC77'	R/W	Remote signal receive interrupt control register (high)

Interrupts

Interrupt Control Registers

Table 2-4 Interrupt Control Registers

Register	Address	R/W	Description
ADM3ICL	x'00FC78'	R/W	Address 3 match interrupt control register (low)
ADM3ICH	x'00FC79'	R/W	Address 3 match interrupt control register (high)
ADM2ICL	x'00FC7A'	R/W	Address 2 match interrupt control register (low)
ADM2ICH	x'00FC7B'	R/W	Address 2 match interrupt control register (high)
ADM1ICL	x'00FC7C'	R/W	Address 1 match interrupt control register (low)
ADM1ICH	x'00FC7D'	R/W	Address 1 match interrupt control register (high)
ADM0ICL	x'00FC7E'	R/W	Address 0 match interrupt control register (low)
ADM0ICH	x'00FC7F'	R/W	Address 0 match interrupt control register (high)
ANICL	x'00FC80'	R/W	A/D conversion end interrupt control register (low)
ANICH	x'00FC81'	R/W	A/D conversion end interrupt control register (high)
SCT0ICL	x'00FC82'	R/W	Serial 0 transmission end interrupt control register (low)
SCT0ICH	x'00FC83'	R/W	Serial 0 transmission end interrupt control register (high)
SCR0ICL	x'00FC84'	R/W	Serial 0 reception end interrupt control register (low)
SCR0ICH	x'00FC85'	R/W	Serial 0 reception end interrupt control register (high)
VBIVICL	x'00FC88'	R/W	VBIVSYNC (1) interrupt control register (low)
VBIVICH	x'00FC89'	R/W	VBIVSYNC (1) interrupt control register (high)
VBIVWICL	x'00FC8A'	R/W	VBIVSYNC (2) interrupt control register (low)
VBIVWICH	x'00FC8B'	R/W	VBIVSYNC (2) interrupt control register (high)
TM3UDICL	x'00FC8C'	R/W	Timer 3 underflow interrupt control register (low)
TM3UDICH	x'00FC8D'	R/W	Timer 3 underflow interrupt control register (high)
OSDGICL	x'00FC90'	R/W	OSD (graphics) interrupt control register (low)
OSDGICH	x'00FC91'	R/W	OSD (graphics) interrupt control register (high)
OSDCICL	x'00FC92'	R/W	OSD (text) interrupt control register (low)
OSDCICH	x'00FC93'	R/W	OSD (text) interrupt control register (high)
SCT1ICL	x'00FC98'	R/W	Serial 1 transmission end interrupt control register (low)
SCT1ICH	x'00FC99'	R/W	Serial 1 transmission end interrupt control register (high)
SCR1ICL	x'00FC9A'	R/W	Serial 1 reception end interrupt control register (low)
SCR1ICH	x'00FC9B'	R/W	Serial 1 reception end interrupt control register (high)
I2CICL	x'00FC9C'	R/W	I ² C interrupt control register (low)
I2CICH	x'00FC9D'	R/W	I ² C interrupt control register (high)

Note: The interrupt error interrupt control register does not exist in the hardware, but if no matching interrupt vector is found for an interrupt that occurs, the CPU writes a C to IAGR to indicate that it detected an abnormality.

IAGR: Accepted Interrupt Group Number Register

x'00FC0E'

	Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		—	—	—	—	—	—	—	—	GN5	GN4	GN3	GN2	GN1	GN0	—	—
Reset:		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:		R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

IAGR returns the group number of an accepted interrupt, indicated in the 6-bit GN field. When the interrupt handler has to calculate the header address for the interrupt service routine, it merely needs to add the contents of IAGR to the header address for the table in which are registered the vector addresses for servicing all interrupts. IAGR is a 16-bit access register.

GN[5:0]: Group Number

Contains the group number multiplied by four.

EXTMD: External Interrupt Mode Register

x'00FCF8'

	Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		—	—	—	—	IQ5TG 1	IQ5TG 0	IQ4TG 1	IQ4TG 0	IQ3TG 1	IQ3TG 0	IQ2TG 1	IQ2TG 0	IQ1TG 1	IQ1TG 0	IQ0TG 1	IQ0TG 0
Reset:		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:		R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

EXTMD sets the trigger conditions for external interrupts. IQnTG[1:0] sets the interrupt mode on the associated IRQ pin. Each IRQ pin can have any polarity or edge setting. EXTMD is a 16-bit access register.

00: Active-low interrupt

01: Either-edge-triggered interrupt (positive or negative)

10: Negative-edge-triggered interrupt

11: Positive-edge-triggered interrupt

WDICR: Watchdog Interrupt Control Register

x'00FC42'

	Bit:	7	6	5	4	3	2	1	0
		—	—	—	—	—	—	—	WDID
Reset:		0	0	0	0	0	0	0	0
R/W:		R	R	R	R	R	R	R	R/W

WDICR is an 8-bit access register.

WDID: Watchdog interrupt detect flag

0: Interrupt undetected

1: Interrupt detected



The watchdog timer interrupt is provided for detecting and handling racing. Normal operation is not guaranteed if the program returns after a watchdog interrupt. For actions requiring returns, use a timer interrupt.

Interrupts

Interrupt Control Registers

PIICR: Undefined Instruction Interrupt Control Register

x'00FC44'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	PIID
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R/W

PIICR is an 8-bit access register.

PIID: Undefined instruction interrupt detect flag

0: Interrupt undetected

1: Interrupt detected

EIICR: Interrupt error Interrupt Control Register

x'00FC46'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	—
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R

EIICR does not exist in the hardware, but if the CPU finds no matching interrupt vector for an interrupt that occurs, it writes a C to IAGR to indicate that it detected an abnormality. EIICR is an 8-bit access register.

IQ0ICL: External Interrupt 0 Interrupt Control Register (Low)

x'00FC48'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	IQ0IR	—	—	—	IQ0ID
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R/W	R	R	R	R

IQ0ICL requests and verifies interrupt requests for external interrupt 0. It is an 8-bit access register. Use the MOV_B instruction to access it.

IQ0IR: External interrupt 0 interrupt request flag

0: No interrupt requested

1: Interrupt requested

IQ0ID: External interrupt 0 interrupt detect flag

0: Interrupt undetected

1: Interrupt detected

IQ0ICH: External Interrupt 0 Interrupt Control Register (High)

x'00FC49'

Bit:	7	6	5	4	3	2	1	0
	—	IQ0LV2	IQ0LV1	IQ0LV0	—	—	—	IQ0IE
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R/W	R/W	R/W	R	R	R	R/W

IQ0ICH sets the priority level for and enables external interrupt 0. It is an 8-bit access register. Use the MOV_B instruction to access it.

IQ0LV[2:0]: External interrupt 0 interrupt priority level
Sets the priority from 0 to 6.

IQ0IE: External interrupt 0 interrupt enable flag
0: Disable
1: Enable

IQ1ICL: External Interrupt 1 Interrupt Control Register (Low)

x'00FC4A'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	IQ1IR	—	—	—	IQ1ID
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R/W	R	R	R	R

IQ1ICL requests and verifies interrupt requests for external interrupt 1. It is an 8-bit access register. Use the MOV_B instruction to access it.

IQ1IR: External interrupt 1 interrupt request flag
0: No interrupt requested
1: Interrupt requested

IQ1ID: External interrupt 1 interrupt detect flag
0: Interrupt undetected
1: Interrupt detected

IQ1ICH: External Interrupt 1 Interrupt Control Register (High)

x'00FC4B'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	IQ1IE
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R/W

IQ1ICH enables external interrupt 1. It is an 8-bit access register. Use the MOV_B instruction to access it.

The priority level for external interrupt 1 is written to the IQ0LV[2:0] field of the IQ0ICH register.

IQ1IE: External interrupt 1 interrupt enable flag
0: Disable
1: Enable

Interrupts

Interrupt Control Registers

IQ2ICL: External Interrupt 2 Interrupt Control Register (Low)

x'00FC50'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	IQ2IR	—	—	—	IQ2ID
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R/W	R	R	R	R

IQ2ICL requests and verifies interrupt requests for external interrupt 2. It is an 8-bit access register. Use the MOV_B instruction to access it.

IQ2IR: External interrupt 2 interrupt request flag

- 0: No interrupt requested
- 1: Interrupt requested

IQ2ID: External interrupt 2 interrupt detect flag

- 0: Interrupt undetected
- 1: Interrupt detected

IQ2ICH: External Interrupt 2 Interrupt Control Register (High)

x'00FC51'

Bit:	7	6	5	4	3	2	1	0
	—	IQ2LV2	IQ2LV1	IQ2LV0	—	—	—	IQ2IE
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R/W	R/W	R/W	R	R	R	R/W

IQ2ICH sets the priority level for and enables external interrupt 2. It is an 8-bit access register. Use the MOV_B instruction to access it.

IQ2LV[2:0]: External interrupt 2 interrupt priority level
Sets the priority from 0 to 6.

IQ2IE: External interrupt 2 interrupt enable flag

- 0: Disable
- 1: Enable

IQ3ICL: External Interrupt 3 Interrupt Control Register (Low)

x'00FC52'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	IQ3IR	—	—	—	IQ3ID
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R/W	R	R	R	R

IQ3ICL requests and verifies interrupt requests for external interrupt 3. It is an 8-bit access register. Use the MOV_B instruction to access it.

IQ3IR: External interrupt 3 interrupt request flag

- 0: No interrupt requested
- 1: Interrupt requested

IQ3ID: External interrupt 3 interrupt detect flag

- 0: Interrupt undetected
- 1: Interrupt detected

IQ3ICH: External Interrupt 3 Interrupt Control Register (High)

x'00FC53'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	IQ3IE
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R/W

IQ3ICH enables external interrupt 3. It is an 8-bit access register. Use the MOV_B instruction to access it.

The priority level for external interrupt 3 is written to the IQ2LV[2:0] field of the IQ2ICH register.

IQ3IE: External interrupt 3 interrupt enable flag

- 0: Disable
- 1: Enable

IQ4ICL: External Interrupt 4 Interrupt Control Register (Low)

x'00FC58'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	IQ4IR	—	—	—	IQ4ID
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R/W	R	R	R	R

IQ4ICL requests and verifies interrupt requests for external interrupt 4. It is an 8-bit access register. Use the MOV_B instruction to access it.

IQ4IR: External interrupt 4 interrupt request flag

- 0: No interrupt requested
- 1: Interrupt requested

IQ4ID: External interrupt 4 interrupt detect flag

- 0: Interrupt undetected
- 1: Interrupt detected

IQ4ICH: External Interrupt 4 Interrupt Control Register (High)

x'00FC59'

Bit:	7	6	5	4	3	2	1	0
	—	IQ4LV2	IQ4LV1	IQ4LV0	—	—	—	IQ4IE
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R/W	R/W	R/W	R	R	R	R/W

IQ4ICH sets the priority level for and enables external interrupt 4. It is an 8-bit access register. Use the MOV_B instruction to access it.

IQ4LV[2:0]: External interrupt 4 interrupt priority level

Sets the priority from 0 to 6.

IQ4IE: External interrupt 4 interrupt enable flag

- 0: Disable
- 1: Enable

Interrupts

Interrupt Control Registers

IQ5ICL: External Interrupt 5 Interrupt Control Register (Low)

x'00FC5A'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	IQ5IR	—	—	—	IQ5ID
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R/W	R	R	R	R

IQ5ICL requests and verifies interrupt requests for external interrupt 5. It is an 8-bit access register. Use the MOV_B instruction to access it.

IQ5IR: External interrupt 5 interrupt request flag

0: No interrupt requested

1: Interrupt requested

IQ5ID: External interrupt 5 interrupt detect flag

0: Interrupt undetected

1: Interrupt detected

IQ5ICH: External Interrupt 5 Interrupt Control Register (High)

x'00FC5B'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	IQ5IE
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R/W

IQ5ICH enables external interrupt 5. It is an 8-bit access register. Use the MOV_B instruction to access it.

The priority level for external interrupt 5 is written to the IQ4LV[2:0] field of the IQ4ICH register.

IQ5IE: External interrupt 5 interrupt enable flag

0: Disable

1: Enable

TM4CBICL: Timer 4 Compare/Capture B Interrupt Control Register (Low)

x'00FC60'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	TM4CBIR	—	—	—	TM4CBID
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R/W	R	R	R	R

TM4CBICL detects and requests timer 4 compare/capture B interrupts. It is an 8-bit access register. Use the MOV_B instruction to access it.

TM4CBIR: Timer 4 compare/capture B interrupt request flag

0: No interrupt requested

1: Interrupt requested

TM4CBID: Timer 4 compare/capture B interrupt detect flag

0: Interrupt undetected

1: Interrupt detected

TM4CBICH: Timer 4 Compare/Capture B Interrupt Control Register (High) x'00FC61'

Bit:	7	6	5	4	3	2	1	0
	—	TM4CB LV2	TM4CB LV1	TM4CB LV0	—	—	—	TM4CB IE
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R/W	R/W	R/W	R	R	R	R/W

TM4CBICH sets the priority level for and enables timer 4 compare/capture B interrupts. It is an 8-bit access register. Use the MOV_B instruction to access it.

TM4CBLV[2:0]: Timer 4 compare/capture B interrupt priority level

Sets the priority from 0 to 6.

TM4CBIE: Timer 4 compare/capture B interrupt enable flag

0: Disable

1: Enable

TM4CAICL: Timer 4 Compare/Capture A Interrupt Control Register (Low) x'00FC62'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	TM4CA IR	—	—	—	TM4CA ID
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R/W	R	R	R	R

TM4CAICL detects and requests timer 4 compare/capture interrupts. It is an 8-bit access register. Use the MOV_B instruction to access it.

TM4CAIR: Timer 4 compare/capture A interrupt request flag

0: No interrupt requested

1: Interrupt requested

TM4CAID: Timer 4 compare/capture A interrupt detect flag

0: Interrupt undetected

1: Interrupt detected

TM4CAICH: Timer 4 Compare/Capture A Interrupt Control Register (High) x'00FC63'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	TM4CA IE
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R/W

TM4CAICH enables timer 4 compare/capture interrupts. It is an 8-bit access register. Use the MOV_B instruction to access it.

The priority level for timer 4 compare/capture interrupts is written to the TM4CBLV[2:0] field of the TM4CBICH register.

TM4CAIE: Timer 4 compare/capture A interrupt enable flag

0: Disable

1: Enable

Interrupts

Interrupt Control Registers

TM4UDICL: Timer 4 Underflow Interrupt Control Register (Low)

x'00FC64'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	TM4UD IR	—	—	—	TM4UD ID
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R/W	R	R	R	R

TM4UDICL detects and requests timer 4 underflow interrupts. It is an 8-bit access register. Use the MOV_B instruction to access it.

TM4UDIR: Timer 4 underflow interrupt request flag

0: No interrupt requested

1: Interrupt requested

TM4UDID: Timer 4 underflow interrupt detect flag

0: Interrupt undetected

1: Interrupt detected

TM4UDICH: Timer 4 Underflow Interrupt Control Register (High)

x'00FC65'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	TM4UD IE
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R/W

TM4UDICH enables timer 4 underflow interrupts. It is an 8-bit access register. Use the MOV_B instruction to access it.

The priority level for timer 4 underflow interrupts is written to the TM4CBLV[2:0] field of the TM4CBICH register.

TM4UDIE: Timer 4 underflow interrupt enable flag

0: Disable

1: Enable

VBIICL: VBI (1) Interrupt Control Register (Low)

x'00FC66'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	VBI IR	—	—	—	VBI ID
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R/W	R	R	R	R

VBIICL detects and requests VBI (1) interrupts. It is an 8-bit access register. Use the MOV_B instruction to access it.

VBIIR: VBI (1) interrupt request flag

0: No interrupt requested

1: Interrupt requested

VBIID: VBI (1) interrupt detect flag

0: Interrupt undetected

1: Interrupt detected

VBIICH: VBI (1) Interrupt Control Register (High)

x'00FC67'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	VBI IE
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R/W

VBIICH enables VBI (1) interrupts. It is an 8-bit access register. Use the MOV_B instruction to access it.

The priority level for VBI (1) interrupts is written to the TM4CBLV[2:0] field of the TM4CBICH register.

VBIE: VBI (1) interrupt enable flag

- 0: Disable
- 1: Enable

TM5CBICL: Timer 5 Compare/Capture B Interrupt Control Register (Low) x'00FC68'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	TM5CB IR	—	—	—	TM5CB ID
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R/W	R	R	R	R

TM5CBICL detects and requests timer 5 compare/capture B interrupts. It is an 8-bit access register. Use the MOV_B instruction to access it.

TM5CBIR: Timer 5 compare/capture B interrupt request flag

- 0: No interrupt requested
- 1: Interrupt requested

TM5CBID: Timer 5 compare/capture B interrupt detect flag

- 0: Interrupt undetected
- 1: Interrupt detected

TM5CBICH: Timer 5 Compare/Capture B Interrupt Control Register (High) x'00FC69'

Bit:	7	6	5	4	3	2	1	0
	—	TM5CB LV2	TM5CB LV1	TM5CB LV0	—	—	—	TM5CB IE
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R/W	R/W	R/W	R	R	R	R/W

TM5CBICH sets the priority level for and enables timer 5 compare/capture B interrupts. It is an 8-bit access register. Use the MOV_B instruction to access it.

TM5CBLV[2:0]: Timer 5 compare/capture B interrupt priority level

Sets the priority from 0 to 6.

TM5CBIE: Timer 5 compare/capture B interrupt enable flag

- 0: Disable
- 1: Enable

Interrupts

Interrupt Control Registers

TM5CAICL: Timer 5 Compare/Capture A Interrupt Control Register (Low) x'00FC6A'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	TM5CA IR	—	—	—	TM5CA ID
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R/W	R	R	R	R

TM5CAICL detects and requests timer 5 compare/capture interrupts. It is an 8-bit access register. Use the MOV_B instruction to access it.

TM5CAIR: Timer 5 compare/capture A interrupt request flag

- 0: No interrupt requested
- 1: Interrupt requested

TM5CAID: Timer 5 compare/capture A interrupt detect flag

- 0: Interrupt undetected
- 1: Interrupt detected

TM5CAICH: Timer 5 Compare/Capture A Interrupt Control Register (High) x'00FC6B'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	TM5CA IE
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R/W

TM5CAICH enables timer 5 compare/capture interrupts. It is an 8-bit access register. Use the MOV_B instruction to access it.

The priority level for timer 5 compare/capture interrupts is written to the TM5CBLV[2:0] field of the TM5CBICH register.

TM5CAIE: Timer 5 compare/capture A interrupt enable flag

- 0: Disable
- 1: Enable

TM5UDICL: Timer 5 Underflow Interrupt Control Register (Low) x'00FC6C'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	TM5UD IR	—	—	—	TM5UD ID
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R/W	R	R	R	R

TM5UDICL detects and requests timer 5 underflow interrupts. It is an 8-bit access register. Use the MOV_B instruction to access it.

TM5UDIR: Timer 5 underflow interrupt request flag

- 0: No interrupt requested
- 1: Interrupt requested

TM5UDID: Timer 5 underflow interrupt detect flag

- 0: Interrupt undetected
- 1: Interrupt detected

TM5UDICH: Timer 5 Underflow Interrupt Control Register (High) **x'00FC6D'**

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	TMSUD IE
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R/W

TM5UDICH enables timer 5 underflow interrupts. It is an 8-bit access register. Use the MOV_B instruction to access it.

The priority level for timer 5 underflow interrupts is written to the TM5CBLV[2:0] field of the TM5CBICH register.

TM5UDIE: Timer 5 underflow interrupt enable flag

- 0: Disable
- 1: Enable

VBIWICL: VBI (2) Interrupt Control Register (Low) **x'00FC6E'**

Bit:	7	6	5	4	3	2	1	0
	—	—	—	VBIW IR	—	—	—	VBIW ID
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R/W	R	R	R	R

VBIWICL detects and requests VBI (2) interrupts. It is an 8-bit access register. Use the MOV_B instruction to access it.

VBIWIR: VBI (2) interrupt request flag

- 0: No interrupt requested
- 1: Interrupt requested

VBIWID: VBI (2) interrupt detect flag

- 0: Interrupt undetected
- 1: Interrupt detected

VBIWICH: VBI (2) Interrupt Control Register (High) **x'00FC6F'**

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	VBIW IE
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R/W

VBIWICH register enables VBI (2) interrupts. It is an 8-bit access register. Use the MOV_B instruction to access it.

The priority level for VBI (2) interrupts is written to the TM5CBLV[2:0] field of the TM5CBICH register.

VBIWIE: VBI (2) interrupt enable flag

- 0: Disable
- 1: Enable

Interrupts

Interrupt Control Registers

TM2UDICL: Timer 2 Underflow Interrupt Control Register (Low)

x'00FC70'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	TM2UD IR	—	—	—	TM2UD ID
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R/W	R	R	R	R

TM2UDICL register detects and requests timer 2 underflow interrupts. It is an 8-bit access register. Use the MOV B instruction to access it.

TM2UDIR: Timer 2 underflow interrupt request flag

- 0: No interrupt requested
- 1: Interrupt requested

TM2UDID: Timer 2 underflow interrupt detect flag

- 0: Interrupt undetected
- 1: Interrupt detected

TM2UDICH: Timer 2 Underflow Interrupt Control Register (High)

x'00FC71'

Bit:	7	6	5	4	3	2	1	0
	—	TM2UD LV2	TM2UD LV1	TM2UD LV0	—	—	—	TM2UD IE
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R/W	R/W	R/W	R	R	R	R/W

TM2UDICH sets the priority level for and enables timer 2 underflow interrupts. It is an 8-bit access register. Use the MOV B instruction to access it.

TM2UDLV[2:0]: Timer 2 underflow interrupt priority level

Sets the priority from 0 to 6.

TM2UDIE: Timer 2 underflow interrupt enable flag

- 0: Disable
- 1: Enable

TM1UDICL: Timer 1 Underflow Interrupt Control Register (Low)

x'00FC72'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	TM1UD IR	—	—	—	TM1UD ID
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R/W	R	R	R	R

TM1UDICL detects and requests timer 1 underflow interrupts. It is an 8-bit access register. Use the MOV B instruction to access it.

TM1UDIR: Timer 1 underflow interrupt request flag

- 0: No interrupt requested
- 1: Interrupt requested

TM1UDID: Timer 1 underflow interrupt detect flag

- 0: Interrupt undetected
- 1: Interrupt detected

TM1UDICH: Timer 1 Underflow Interrupt Control Register (High) x'00FC73'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	TM1UD IE
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R/W

TM1UDICH enables timer 1 underflow interrupts. It is an 8-bit access register. Use the MOV_B instruction to access it.

The priority level for timer 1 underflow interrupts is written to the TM2UDLV[2:0] field of the TM2UDICH register.

TM1UDIE: Timer 1 underflow interrupt enable flag

- 0: Disable
- 1: Enable

TM0UDICL: Timer 0 Underflow Interrupt Control Register (Low) x'00FC74'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	TM0UD IR	—	—	—	TM0UD ID
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R/W	R	R	R	R

TM0UDICL register detects and requests timer 0 underflow interrupts. It is an 8-bit access register. Use the MOV_B instruction to access it.

TM0UDIR: Timer 0 underflow interrupt request flag

- 0: No interrupt requested
- 1: Interrupt requested

TM0UDID: Timer 0 underflow interrupt detect flag

- 0: Interrupt undetected
- 1: Interrupt detected

TM0UDICH: Timer 0 Underflow Interrupt Control Register (High) x'00FC75'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	TM0UD IE
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R/W

TM0UDICH enables timer 0 underflow interrupts. It is an 8-bit access register. Use the MOV_B instruction to access it.

The priority level for timer 0 underflow is written to the TM2UDLV[2:0] field of the TM2UDICH register.

TM0UDIE: Timer 0 underflow interrupt enable flag

- 0: Disable
- 1: Enable

Interrupts

Interrupt Control Registers

RMCICL: Remote Signal Receive Interrupt Control Register (Low) x'00FC76'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	RMC IR	—	—	—	RMC ID
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R/W	R	R	R	R

RMCICL detects and requests remote signal receive interrupts. It is an 8-bit access register. Use the MOV_B instruction to access it.

RMCIR: Remote signal receive interrupt request flag

0: No interrupt requested

1: Interrupt requested

RMCID: Remote signal receive interrupt detect flag

0: Interrupt undetected

1: Interrupt detected

RMCICH: Remote Signal Receive Interrupt Control Register (High) x'00FC77'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	RMC IE
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R/W

RMCICH enables remote signal receive interrupts. It is an 8-bit access register. Use the MOV_B instruction to access it.

The priority level for remote signal receive interrupts is written to the TM2UDLV[2:0] field of the TM2UDICH register.

RMCIE: Remote signal receive interrupt enable flag

0: Disable

1: Enable

ADM3ICL: Address 3 Match Interrupt Control Register (Low) x'00FC78'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	ADM3 IR	—	—	—	ADM3 ID
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R/W	R	R	R	R

ADM3ICL detects and requests address match 3 interrupts. It is an 8-bit access register. Use the MOV_B instruction to access it.

ADM3IR: Address match 3 interrupt request flag

0: No interrupt requested

1: Interrupt requested

ADM3ID: Address match 3 interrupt detect flag

0: Interrupt undetected

1: Interrupt detected

ADM3ICH: Address 3 Match Interrupt Control Register (High)

x'00FC79'

Bit:	7	6	5	4	3	2	1	0
	—	ADM3 LV2	ADM3 LV1	ADM3 LV0	—	—	—	ADM3 IE
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R/W	R/W	R/W	R	R	R	R/W

ADM3ICH sets the priority level for and enables address match 3 interrupts. It is an 8-bit access register. Use the MOV_B instruction to access it.

ADM3LV[2:0]: Address match 3 interrupt priority level

Sets the priority from 0 to 6.

ADM3IE: Address match 3 interrupt enable flag

0: Disable

1: Enable

ADM2ICL: Address 2 Match Interrupt Control Register (Low)

x'00FC7A'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	ADM2 IR	—	—	—	ADM2 ID
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R/W	R	R	R	R

ADM2ICL detects and requests address match 2 interrupts. It is an 8-bit access register. Use the MOV_B instruction to access it.

ADM2IR: Address match 2 interrupt request flag

0: No interrupt requested

1: Interrupt requested

ADM2ID: Address match 2 interrupt detect flag

0: Interrupt undetected

1: Interrupt detected

ADM2ICH: Address 2 Match Interrupt Control Register (High)

x'00FC7B'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	ADM2 IE
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R/W

ADM2ICH enables address match 2 interrupts. It is an 8-bit access register. Use the MOV_B instruction to access it.

The priority level for address match 2 interrupts is written to the ADM3LV[2:0] field of the ADM3ICH register.

ADM2IE: Address match 2 interrupt enable flag

0: Disable

1: Enable

Interrupts

Interrupt Control Registers

ADM1ICL: Address 1 Match Interrupt Control Register (Low)

x'00FC7C'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	ADM1 IR	—	—	—	ADM1 ID
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R/W	R	R	R	R

ADM1ICL detects and requests address match 1 interrupts. It is an 8-bit access register. Use the MOV_B instruction to access it.

ADM1IR: Address match 1 interrupt request flag

0: No interrupt requested

1: Interrupt requested

ADM1ID: Address match 1 interrupt detect flag

0: Interrupt undetected

1: Interrupt detected

ADM1ICH: Address 1 Match Interrupt Control Register (High)

x'00FC7D'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	ADM1 IE
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R/W

ADM1ICH enables address match 1 interrupts. It is an 8-bit access register. Use the MOV_B instruction to access it.

The priority level for address match 1 interrupts is written to the ADM3LV[2:0] field of the ADM3ICH register.

ADM1IE: Address match 1 interrupt enable flag

0: Disable

1: Enable

ADM0ICL: Address 0 Match Interrupt Control Register (Low)

x'00FC7E'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	ADM0 IR	—	—	—	ADM0 ID
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R/W	R	R	R	R

ADM0ICL detects and requests address match 0 interrupts. It is an 8-bit access register. Use the MOV_B instruction to access it.

ADM0IR: Address match 0 interrupt request flag

0: No interrupt requested

1: Interrupt requested

ADM0ID: Address match 0 interrupt detect flag

0: Interrupt undetected

1: Interrupt detected

ADM0ICH: Address 0 Match Interrupt Control Register (High)

x'00FC7F'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	ADM0 IE
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R/W

ADM0ICH enables address match 0 interrupts. It is an 8-bit access register. Use the MOV_B instruction to access it.

The priority level for address match 0 interrupts is written to the ADM3LV[2:0] field of the ADM3ICH register.

ADM0IE: Address match 0 interrupt enable flag

- 0: Disable
- 1: Enable

ANICL: A/D Conversion End Interrupt Control Register (Low)

x'00FC80'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	ANIR	—	—	—	ANID
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R/W	R	R	R	R

ANICL detects and requests A/D conversion end interrupts. It is an 8-bit access register. Use the MOV_B instruction to access it.

ANIR: A/D conversion end interrupt request flag

- 0: No interrupt requested
- 1: Interrupt requested

ANID: A/D conversion end interrupt detect flag

- 0: Interrupt undetected
- 1: Interrupt detected

ANICH: A/D Conversion End Interrupt Control Register (High)

x'00FC81'

Bit:	7	6	5	4	3	2	1	0
	—	ANLV2	ANLV1	ANLV0	—	—	—	ANIE
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R/W	R/W	R/W	R	R	R	R/W

ANICH sets the priority level for and enables A/D conversion end interrupts. It is an 8-bit access register. Use the MOV_B instruction to access it.

ANLV[2:0]: A/D conversion end interrupt priority level

Sets the priority from 0 to 6.

ANIE: A/D conversion end interrupt enable flag

- 0: Disable
- 1: Enable

Interrupts

Interrupt Control Registers

SCT0ICL: Serial 0 Transmission End Interrupt Control Register (Low) x'00FC82'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	SCT0 IR	—	—	—	SCT0 ID
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R/W	R	R	R	R

SCT0ICL detects and requests serial 0 transmission end interrupts. It is an 8-bit access register. Use the MOV_B instruction to access it.

SCT0IR: Serial 0 transmission end interrupt request flag

0: No interrupt requested

1: Interrupt requested

SCT0ID: Serial 0 transmission end interrupt detect flag

0: Interrupt undetected

1: Interrupt detected

SCT0ICH: Serial 0 Transmission End Interrupt Control Register (High) x'00FC83'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	SCT0 IE
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R/W

SCT0ICH enables serial 0 transmission end interrupts. It is an 8-bit access register. Use the MOV_B instruction to access it.

The priority level for serial 0 transmission end interrupts is written to the ANLV[2:0] field of the ANICH register.

SCT0IE: Serial 0 transmission end interrupt enable flag

0: Disable

1: Enable

SCR0ICL: Serial 0 Reception End Interrupt Control Register (Low) x'00FC84'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	SCR0 IR	—	—	—	SCR0 ID
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R/W	R	R	R	R

SCR0ICL detects and requests serial 0 reception end interrupts. It is an 8-bit access register. Use the MOV_B instruction to access it.

SCR0IR: Serial 0 reception end interrupt request flag

0: No interrupt requested

1: Interrupt requested

SCR0ID: Serial 0 reception end interrupt detect flag

0: Interrupt undetected

1: Interrupt detected

SCR0ICH: Serial 0 Reception End Interrupt Control Register (High) x'00FC85'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	SCR0 IE
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R/W

SCR0ICH enables serial 0 reception end interrupts. It is an 8-bit access register. Use the MOV_B instruction to access it.

The priority level for serial 0 reception end interrupts is written to the ANLV[2:0] field of the ANICH register.

SCR0IE: Serial 0 reception end interrupt enable flag

- 0: Disable
- 1: Enable

VBIVICL: VBIVSYNC (1) Interrupt Control Register (Low) x'00FC88'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	VBIVIR	—	—	—	VBIVID
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R/W	R	R	R	R

VBIVICL detects and requests VBIVSYNC (1) interrupts. It is an 8-bit access register. Use the MOV_B instruction to access it.

VBIVIR: VBIVSYNC (1) interrupt request flag

- 0: No interrupt requested
- 1: Interrupt requested

VBIVID: VBIVSYNC (1) interrupt detect flag

- 0: Interrupt undetected
- 1: Interrupt detected

VBIVICH: VBIVSYNC (1) Interrupt Control Register (High) x'00FC89'

Bit:	7	6	5	4	3	2	1	0
	—	VBIV LV2	VBIV LV1	VBIV LV0	—	—	—	VBIV IE
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R/W	R/W	R/W	R	R	R	R/W

VBIVICH sets the priority level for and enables VBIVSYNC (1) interrupts. It is an 8-bit access register. Use the MOV_B instruction to access it.

VBIVLV[2:0]: VBIVSYNC (1) interrupt priority level

Sets the priority from 0 to 6.

VBIVIE: VBIVSYNC (1) interrupt enable flag

- 0: Disable
- 1: Enable

Interrupts

Interrupt Control Registers

VBIVWICL: VBIVSYNC (2) Interrupt Control Register (Low)

x'00FC8A'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	VBIVW IR	—	—	—	VBIVW ID
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R/W	R	R	R	R

VBIVWICL detects and requests VBIVSYNC (2) interrupts. It is an 8-bit access register. Use the MOV_B instruction to access it.

VBIVWIR: VBIVSYNC (2) interrupt request flag

0: No interrupt requested

1: Interrupt requested

VBIVWID: VBIVSYNC (2) interrupt detect flag

0: Interrupt undetected

1: Interrupt detected

VBIVWICH: VBIVSYNC (2) Interrupt Control Register (High)

x'00FC8B'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	VBIVW IE
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R/W

VBIVWICH enables VBIVSYNC (2) interrupts. It is an 8-bit access register. Use the MOV_B instruction to access it.

The priority level for VBIVSYNC (2) interrupts is written to the VBIVLV[2:0] field of the VBIVICH register.

VBIVWIE: VBIVSYNC (2) interrupt enable flag

0: Disable

1: Enable

TM3UDICL: Timer 3 Underflow Interrupt Control Register (Low)

x'00FC8C'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	TM3UD IR	—	—	—	TM3UD ID
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R/W	R	R	R	R

TM3UDICL detects and requests timer 3 underflow interrupts. It is an 8-bit access register. Use the MOV_B instruction to access it.

TM3UDIR: Timer 3 underflow interrupt request flag

0: No interrupt requested

1: Interrupt requested

TM3UDID: Timer 3 underflow interrupt detect flag

0: Interrupt undetected

1: Interrupt detected

TM3UDICH: Timer 3 Underflow Interrupt Control Register (High) **x'00FC8D'**

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	TM3UD IE
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R/W

TM3UDICH enables timer 3 underflow interrupts. It is an 8-bit access register. Use the MOV_B instruction to access it.

The priority level for timer 3 underflow interrupts is written to the VBIVLV[2:0] field of the VBIVICH register.

TM3UDIE: Timer 3 underflow interrupt enable flag

- 0: Disable
- 1: Enable

OSDGICL: OSD (Graphics) Interrupt Control Register (Low) **x'00FC90'**

Bit:	7	6	5	4	3	2	1	0
	—	—	—	OSDG IR	—	—	—	OSDG ID
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R/W	R	R	R	R

OSDGICL detects and requests OSD (graphics) interrupts. It is an 8-bit access register. Use the MOV_B instruction to access it.

OSDGIR: OSD (graphics) interrupt request flag

- 0: No interrupt requested
- 1: Interrupt requested

OSDGID: OSD (graphics) interrupt detect flag

- 0: Interrupt undetected
- 1: Interrupt detected

OSDGICH: OSD (Graphics) Interrupt Control Register (High) **x'00FC91'**

Bit:	7	6	5	4	3	2	1	0
	—	OSDG LV2	OSDG LV1	OSDG LV0	—	—	—	OSDG IE
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R/W	R/W	R/W	R	R	R	R/W

OSDGICH sets the priority level for and enables OSD (graphics) interrupts. It is an 8-bit access register. Use the MOV_B instruction to access it.

OSDGLV[2:0]: OSD (graphics) interrupt priority level

Sets the priority from 0 to 6.

OSDGIE: OSD (graphics) interrupt enable flag

- 0: Disable
- 1: Enable

Interrupts

Interrupt Control Registers

OSDCICL: OSD (Text) Interrupt Control Register (Low)

x'00FC92'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	OSDC IR	—	—	—	OSDC ID
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R/W	R	R	R	R

OSDCICL detects and requests OSD (text) interrupts. It is an 8-bit access register. Use the MOV_B instruction to access it.

OSDCIR: OSD (text) interrupt request flag

0: No interrupt requested

1: Interrupt requested

OSDCID: OSD (text) interrupt detect flag

0: Interrupt undetected

1: Interrupt detected

OSDCICH: OSD (Text) Interrupt Control Register (High)

x'00FC93'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	OSDC IE
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R/W

OSDCICH enables timer OSD (text) interrupts. It is an 8-bit access register. Use the MOV_B instruction to access it.

The priority level for OSD (text) interrupts is written to the OSDGLV[2:0] field of the OSDGICH register.

OSDCIE: OSD (text) interrupt enable flag

0: Disable

1: Enable

SCT1ICL: Serial 1 Transmission End Interrupt Control Register (Low) x'00FC98'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	SCT1 IR	—	—	—	SCT1 ID
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R/W	R	R	R	R

SCT1ICL detects and requests serial 1 transmission end interrupts. It is an 8-bit access register. Use the MOV_B instruction to access it.

SCT1IR: Serial 1 transmission end interrupt request flag

0: No interrupt requested

1: Interrupt requested

SCT1ID: Serial 1 transmission end interrupt detect flag

0: Interrupt undetected

1: Interrupt detected

SCT1ICH: Serial 1 Transmission End Interrupt Control Register (High) x'00FC99'

Bit:	7	6	5	4	3	2	1	0
	—	SCT1 LV2	SCT1 LV1	SCT1 LV0	—	—	—	SCT1 IE
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R/W	R/W	R/W	R	R	R	R/W

SCT1ICH sets the priority level for and enables serial 1 transmission end interrupts. It is an 8-bit access register. Use the MOV_B instruction to access it.

SCT1LV[2:0]: Serial 1 transmission end interrupt priority level

Sets the priority from 0 to 6.

SCT1IE: Serial 1 transmission end interrupt enable flag

0: Disable

1: Enable

SCR1ICL: Serial 1 Reception End Interrupt Control Register (Low) x'00FC9A'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	SCR1 IR	—	—	—	SCR1 ID
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R/W	R	R	R	R

SCR1ICL detects and requests serial 1 reception end interrupts. It is an 8-bit access register. Use the MOV_B instruction to access it.

SCT1IR: Serial 1 reception end interrupt request flag

0: No interrupt requested

1: Interrupt requested

SCT1ID: Serial 1 reception end interrupt detect flag

0: Interrupt undetected

1: Interrupt detected

SCR1ICH: Serial 1 Reception End Interrupt Control Register (High) x'00FC9B'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	SCR1 IE
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R/W

SCR1ICH enables serial 1 reception end interrupts. It is an 8-bit access register. Use the MOV_B instruction to access it.

The priority level for serial 1 reception end interrupts is written to the SCT1LV[2:0] field of the SCT1ICH register.

SCR1IE: Serial 1 reception end interrupt enable flag

0: Disable

1: Enable

Interrupts

Interrupt Control Registers

I2CICL: I²C Interrupt Control Register (Low)

x'00FC9C'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	I2C IR	—	—	—	I2C ID
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R/W	R	R	R	R

I2CICL detects and requests I²C interrupts. It is an 8-bit access register. Use the MOV_B instruction to access it.

I2CIR: I²C interrupt request flag

0: No interrupt requested

1: Interrupt requested

I2CID: I²C interrupt detect flag

0: Interrupt undetected

1: Interrupt detected

I2CICH: I²C Interrupt Control Register (High)

x'00FC9D'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	I2C IE
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R/W

I2CICH enables I²C interrupts. It is an 8-bit access register. Use the MOV_B instruction to access it.

The priority level for I²C interrupts is written to the SCT1LV[2:0] field of the SCT1ICH register.

I2CIE: I²C interrupt enable flag

0: Disable

1: Enable

3 Low-Power Modes

The MN102H75K/85K provides two ways to reduce power consumption, controlling CPU operating and standby modes to cut overall consumption and shutting down unused functions by stopping the system clock supplied to them.

3.1 CPU Modes

3.1.1 Description

The MN102H75K/85K has two CPU operating modes, NORMAL and SLOW, and two CPU standby modes, HALT and STOP. Effective use of these modes can significantly reduce power consumption. Figure 3-1 shows the CPU states in the different modes.

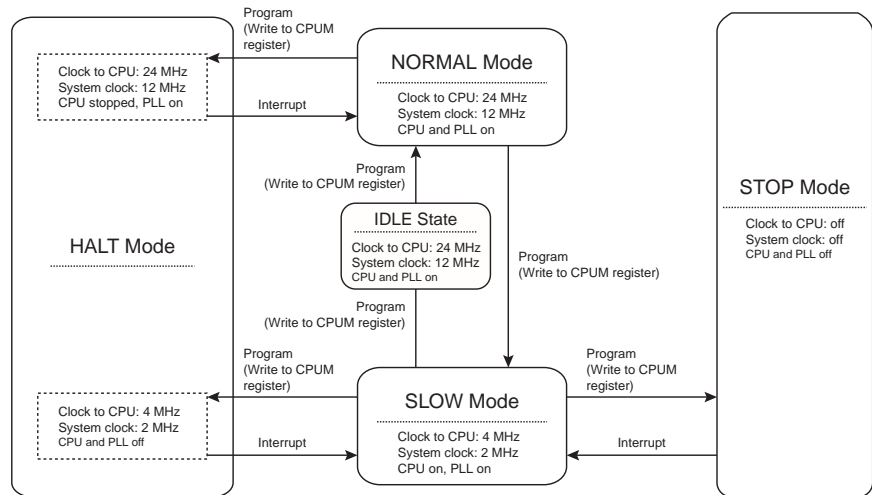


Figure 3-1 CPU State Changes



You cannot enter STOP mode from NORMAL mode.

The CPU mode control register (CPUM) controls transitions between NORMAL and SLOW modes and from NORMAL and SLOW modes to the standby modes. A normal reset or an interrupt wakes the MCU from a standby mode.

Note that you cannot invoke the STOP mode from NORMAL mode. You can only enter STOP from the SLOW mode.



The MN102H75K/85K recovers from power up and reset in SLOW mode. For normal operation, the program must switch the MCU from SLOW to NORMAL mode.

3.1.2 Exiting from SLOW Mode to NORMAL Mode

The MN102H75K/85K contains a PLL circuit that, in NORMAL mode, multiplies the clock input through the OSC1 and OSC2 pins by 12, divides the signal by 2, then sends the resulting clock to the CPU. (See figure 3-2.) The MCU starts in SLOW mode on power up and on recovery from a reset. In SLOW mode (system clock = 2 MHz), the clock from the OSC pins feeds directly to the CPU, without going through the PLL circuit. This means that the program must switch the CPU from SLOW to NORMAL mode (system clock = 12 MHz).

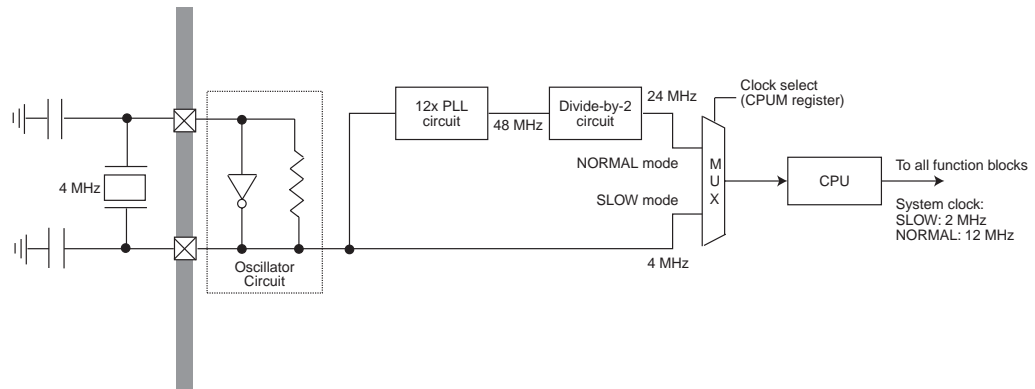


Figure 3-2 CPU Clock Switch (NORMAL/SLOW Modes)

Below is an example routine for exiting SLOW mode. You should run this routine immediately after power up.

Example 3-1 Exiting SLOW Mode

```

MOV x'FC00',A1
MOV (A1),D0           ;Read CPUM register
AND x'FFFD',D0       ;Invoke IDLE mode
MOV (D0),A1
MOV (A1),D0           ;Read CPUM register
AND x'FFF0',D0       ;Invoke NORMAL mode
MOV (D0),A1
    
```



The OSD cannot display in SLOW mode.

Because the system clock in SLOW mode is 2 MHz, the OSD does not function. The specifications also differ for the PWM function and functions such as the IR remote signal receiver and the H counter that use the PWM waveforms.

For information on invoking SLOW mode from NORMAL mode, see *MN102H Series LSI User Manual*.

3.1.3 Notes on Invoking and Exiting STOP and HALT Modes

■ When invoking STOP and HALT modes...

To reduce power consumption before invoking the STOP or HALT mode, stop current flow from output pins and stabilize the input level of input pins. For output pins, either match the output level to the external level or set the pin to input. For input pins, ensure that the external level is fixed. To further reduce power consumption, shut down unnecessary functions through the control registers. (See section 3.2, “Turning Individual Functions On and Off,” on page 75.) Before entering the STOP mode, set all of the bits shown in table 3-1 to disable all of these functions. Disable all functions in the NORMAL mode except the PLL circuit, which you can only shut down once you have entered the SLOW mode.



Using OSDX clock (both an LC blocking oscillator and external source), OSDXI and OSDXO must be set to port (P46, P45) and output 'H' before invoking STOP mode.

To allow the MCU to exit the STOP or HALT mode on reset or interrupt, you must set the interrupt registers before you invoke the standby mode. To specify a particular interrupt vector as the signal for waking up, enable that vector in the interrupt registers. (For more information on controlling interrupts, see “section 2, “Interrupts,”” on page 37.)

■ When exiting STOP and HALT modes...

The MCU exits STOP and HALT modes on reset or interrupt. For information on exiting on interrupt, see Figure 3-1, “CPU State Changes,” on page 72. When the MCU exits on reset, it always exits to SLOW mode.

Low-Power Modes

Turning Individual Functions On and Off



You cannot set the PLL function control bit during NORMAL mode. You must set it from the SLOW mode.



To turn off the OSD block to save power:

1. Write a 0 to OSD (OSD1, bit10).
2. Wait for the next VSYNC input.
3. Write a 0 to OSDPOFF(PCNT0, bit 7), turning the clock off.

If you turn the clock off before the VSYNC input, power usage may not drop or the microcontroller may halt.

3.2 Turning Individual Functions On and Off

The MN102H75K/85K allows you to turn each peripheral function on or off through writing to the registers. You can significantly reduce power consumption by turning off unused functions. Table 3-1 shows the register bits controlling on and off for each function block. The ADC used for the OSD and CCD functions is turned off on reset. Write a 1 to the function to enable it, when necessary.

You cannot read from or write to the registers associated with a function that is disabled. Turning on the function enables register reads and writes.

See the sections on each of these peripheral functions for more information.

Table 3-1 Peripheral Function On/Off Switches

Block Name	Description	Bit Name	Address	Operation	Reset Value
OSD	OSD block control	OSDPOFF	PCNT0, x'00FF90', bit 7	0: OSD block off 1: OSD block enabled	0
	OSD function control	OSD	OSD1, x'007F06', bit 10	0: OSD function off 1: OSD function on	0
	OSD register R/W control	OSDREG E	PCNT2, x'00FF92', bit 0	0: OSD register R/W off 1: OSD register R/W enabled	0
CCD	ADC control for CCD1	ADC1ON	PCNT0, x'00FF90', bit 5	0: ADC for CCD1 off 1: ADC for CCD1 enabled	0
	ADC control for CCD0	ADC0ON	PCNT0, x'00FF90', bit 4	0: ADC for CCD0 off 1: ADC for CCD0 enabled	0
	CCD1 function control	VBI1OFF	PCNT0, x'00FF90', bit 1	0: CCD1 block off 1: CCD1 block enabled	0
	CCD0 function control	VBI0OFF	PCNT0, x'00FF90', bit 0	0: CCD0 block off 1: CCD0 block enabled	0
PLL	PLL function control	PLLPOFF	PCNT0, x'00FF90', bit 6	0: PLL block enabled 1: PLL block off	0
H counter	H counter function control	HCNTOFF	PCNT0, x'00FF90', bit 3	0: H counter block enabled 1: H counter block off	0
IR remote signal receiver	IR remote signal receiver function control	RMCOFF	PCNT0, x'00FF90', bit 2	0: IR remote signal receiver block off 1: IR remote signal receiver block enabled	0
I ² C	I ² C function control	I2COFF	PCNT2, x'00FF92', bit 2	0: I ² C block enabled 1: I ² C block off	0
PWM	PWM function control	PWMOFF	PCNT2, x'00FF92', bit 1	0: PLL block enabled 1: PLL block off	0

3.3 CPU Control Register

CPUM: CPU Mode Control Register

x'00FC00'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	NW DEN	—	—	—	—	—	—	—	—	—	OSC ID	STOP	HALT	OSC1	OSC0	
Reset:	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R/W	R	R	R	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W

This register controls the invoking of all of the CPU modes.

NWDEN: Watchdog timer reset

0: Enable watchdog timer

1: Disable and clear watchdog timer

Setting the watchdog timer to 1, then setting it to 0 clears and restarts the watchdog timer.

OSCID: Oscillator select

System clock monitor

0: Fast

1: Slow

STOP: STOP mode request

CPU operating state control. See table 3-2.

HALT: HALT mode request

CPU operating state control. See table 3-2.

OSC[1:0]: Oscillator control

See table 3-2.

Table 3-2 CPU Mode Bit Settings

STOP	HALT	OSC1	OSC0	CPU Mode	Clock to CPU	System Clock	PLL	CPU
0	0	0	0	NORMAL	24 MHz	12 MHz	On	On
0	0	1	1	SLOW	4 MHz	2 MHz	Off	On
0	1	0	0	HALT0 (Invoked from NORMAL)	24 MHz	12 MHz	On	Off
0	1	1	1	HALT1 (Invoked from SLOW)	4 MHz	2 MHz	Off	Off
1	0	x	x	STOP	Off	Off	Off	Off

Note: All unindicated bit settings are reserved.

4 Timers

4.1 8-Bit Timer Description

The MN102H75K/85K contains four 8-bit timers that can serve as interval timers, event timer/counters, clock generators (divide-by-2 output of the underflow), reference clocks for the serial interfaces, or start timers for A/D conversions. The clock source can be the internal clock (oscillator frequency divided by 2) or the external clock (1/4 or less the oscillator frequency input). A timer interrupt is generated by a timer underflow.

All passages below assume a clock B_{OSC} of 24 MHz.

The 8-bit timers are cascadable into true 16-bit timers. For instance, if you cascade timers 0 and 1, timer 0 sends cascaded output to timer 1. The result is true 16-bit division, rather than two successive 8-bit divisions.

Cascading Connections

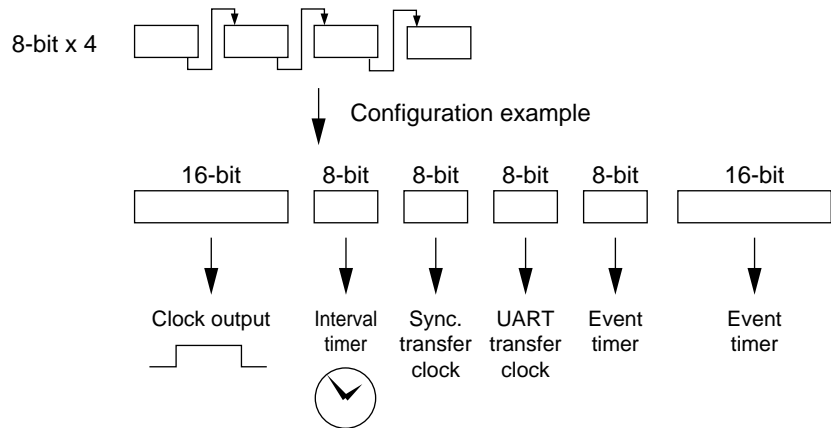
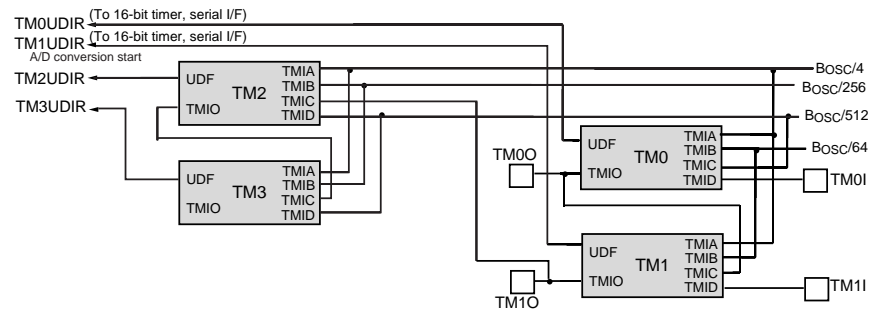


Figure 4-1 Timer Configuration Examples



Note: $B_{OSC} = 24 \text{ MHz}$

Figure 4-2 Block Diagram of 8-Bit Timers

4.2 8-Bit Timer Features

Table 4-1 8-Bit Timer Functions and Features

Function/Feature	Timer 0	Timer 1	Timer 2	Timer 3
Interrupt request flag(s)	TM0UDICL register (TM0UDIR bit)	TM1UDICL register (TM1UDIR bit)	TM2UDICL register (TM2UDIR bit)	TM3UDICL register (TM3UDIR bit)
Interrupt source(s)	Timer 0 underflow	Timer 1 underflow	Timer 2 underflow	Timer 3 underflow
Interval timer function	—	✓	✓	✓
Event counter function	✓	✓	—	—
Clock source for 16-bit timer	✓	✓	—	—
Timer output function	✓ (TM0O signal)	✓ (TM1O signal)	—	—
Serial interface transfer clock source	✓	✓	—	—
A/D conversion trigger function	—	✓	—	—
Clock sources	0	$B_{OSC}/4$	$B_{OSC}/4$	$B_{OSC}/4$
	1	$B_{OSC}/64$	$B_{OSC}/64$	$B_{OSC}/256$
	2	$B_{OSC}/512$	Cascade connection	Cascade connection
	3	TM0I signal	TM1I signal	$B_{OSC}/512$
Cascade connection		✓	✓	✓

Note: When $B_{OSC} = 24$ MHz:

$B_{OSC}/4 = 6$ MHz

$B_{OSC}/64 = 375$ kHz

$B_{OSC}/256 = 93.75$ kHz

$B_{OSC}/512 = 48.875$ kHz.

4.3 8-Bit Timer Block Diagrams

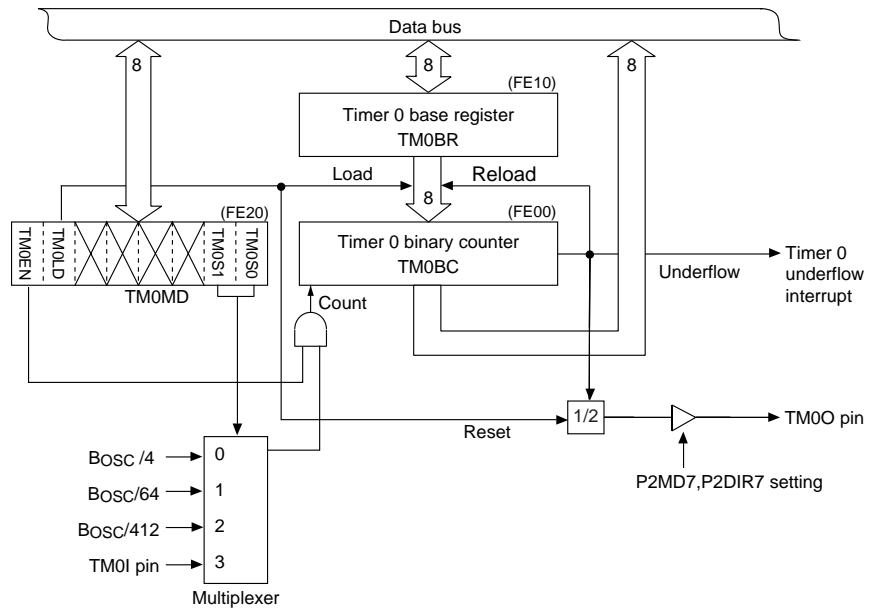


Figure 4-3 Timer 0 Block Diagram

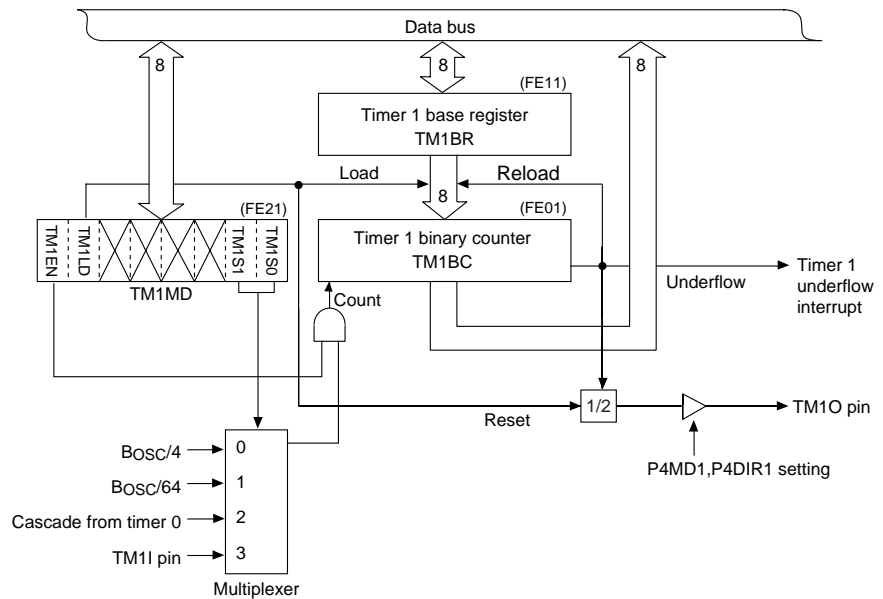


Figure 4-4 Timer 1 Block Diagram

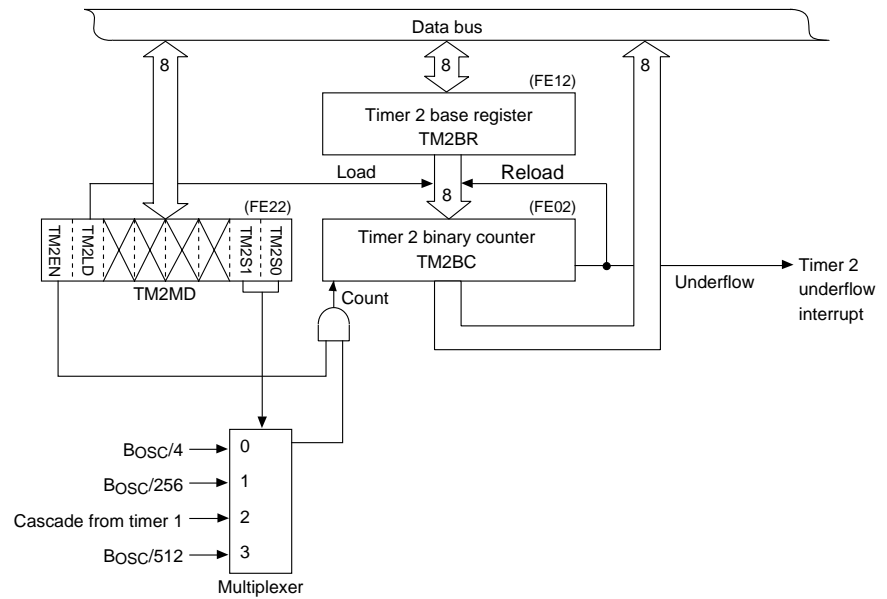


Figure 4-5 Timer 2 Block Diagram

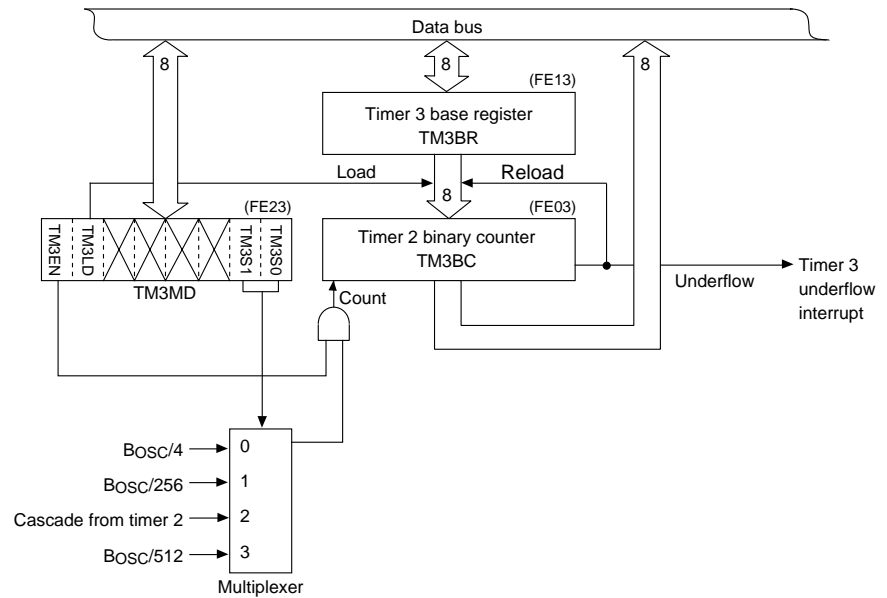


Figure 4-6 Timer 3 Block Diagram

4.4 8-Bit Timer Timing

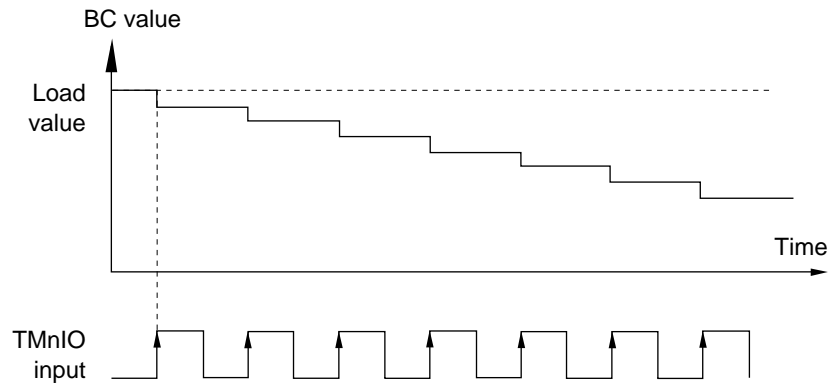


Figure 4-7 Event Timer Input Timing (8-Bit Timers)

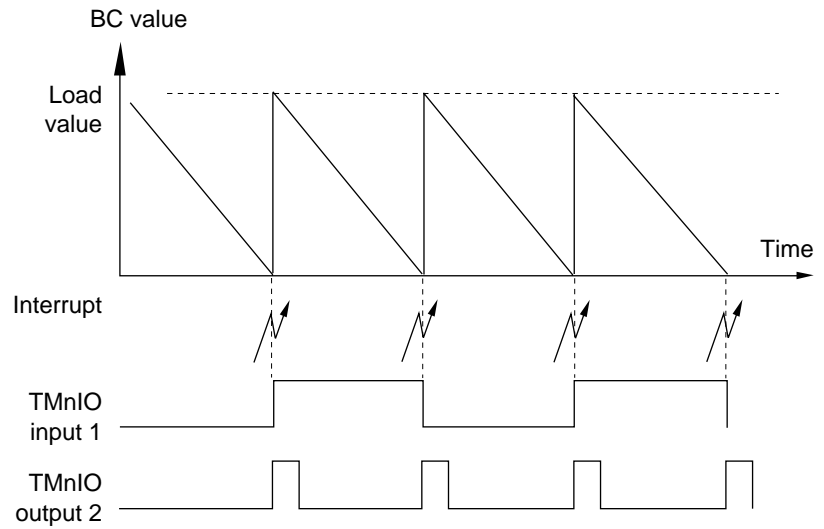


Figure 4-8 Clock Output and Interval Timer Timing (8-Bit Timers)

4.5 8-Bit Timer Setup Examples

4.5.1 Setting Up an Event Counter Using Timer 0

In this example, timer 0 generates an underflow interrupt on the fourth rising edge of the TM0IO signal.

The event counter continues to operate during STOP mode. In all modes but STOP, the TMnIO signal input is synchronized to B_{OSC} . In STOP mode, the timer counts TMnIO signal directly. When an interrupt occurs, the CPU returns to NORMAL mode after the oscillator stabilization wait. The event counter continues to count the TMnIO signal during stabilization wait, and at the same time that the CPU returns to NORMAL mode, the event counter begins counting the signal resulting from the B_{OSC} sampling of the TMnIO signal input.

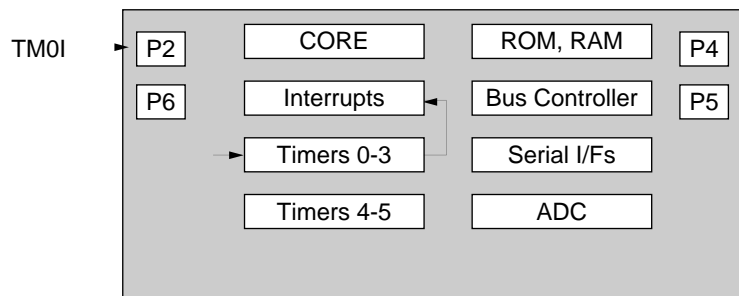


Figure 4-9 Block Diagram of Event Counter Using Timer 0

1. Set the interrupt enable flag (IE) of the processor status word (PSW) to 1.
2. Disable timer 0 counting in the timer 0 mode register (TM0MD). This step is unnecessary immediately after a reset, since TM0MD resets to 0.

TM0MD (example)

x'00FE20'

Bit:	7	6	5	4	3	2	1	0
	TM0 EN	TM0 LD	—	—	—	—	TM0 S1	TM0 S0
Setting:	0	0	0	0	0	0	0	0



TM2UDICH, TM0UDICL, and TM0UDICH are 8-bit access registers. Use the MOV_B instruction to access them.

3. Cancel all existing interrupt requests and enable timer 0 underflow interrupts. To do this, set the TM2UDLV[2:0] bits of TM2UDICH (priority level 4 in this example), set the TM0UDIE bit to 1, and set the TM0UDIR bit of TM0UDICL to 0. (Note that you set the priority level for timer 0 interrupts in the timer 2 interrupt control register.) From this point on, an interrupt request is generated whenever timer 0 underflows.

TM2UDICH (example)

x'00FC71'

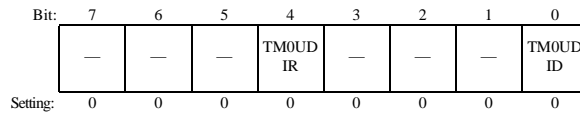
Bit:	7	6	5	4	3	2	1	0
	—	TM2UD LV2	TM2UD LV1	TM2UD LV0	—	—	—	TM2UD IE
Setting:	0	1	0	0	0	0	0	0

Timers

8-Bit Timer Setup Examples

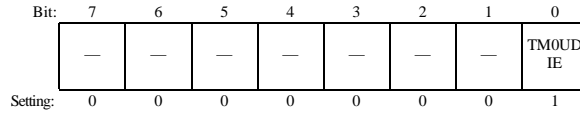
TM0UDICL (example)

x'00FC74'



TM0UDICH (example)

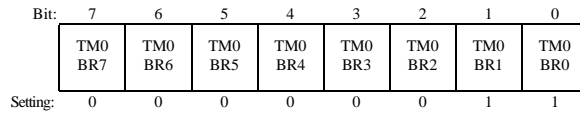
x'00FC75'



- Set the divide-by ratio for timer 0. Since the timer will count 4 TM0IO cycles, write x'03' to the timer 0 base register (TM0BR). (The valid range for TM0BR is 0 to 255.)

TM0BR (example)

x'00FE10'

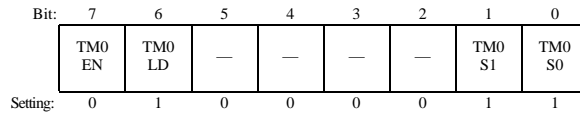


Do not change the clock source once you select it. Selecting the clock source while you set up the count operation control will corrupt the value in the binary counter.

- Set the TM0LD bit of the TM0MD register to 1. This loads the value in the base register to the binary counter. At the same time, select the clock source as the TM0IO signal input by writing b'11' to TM0S[1:0].

TM0MD (example)

x'00FE20'



In the bank and linear addressing versions of the MN102 series, it was necessary to set TM0EN and TM0LD to 0 between steps 5 and 6, to ensure stable operation. This is unnecessary in the high-speed linear addressing version.

- Set TM0LD to 0 and TM0EN to 1. This starts the timer. Counting begins at the start of the next cycle. When the binary counter reaches 0 and loads the value x'03' from the base register, in preparation for the next count, a timer 0 underflow interrupt request is sent to the CPU.

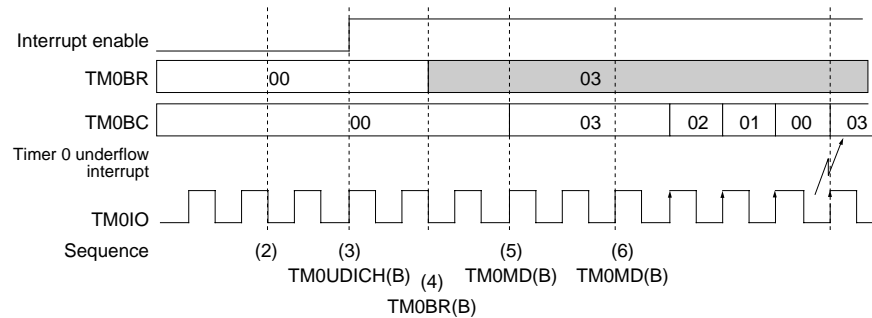


Figure 4-10 Event Counter Timing (Timer 0)

4.5.2 Setting Up an Interval Timer Using Timers 1 and 2

In this example, timers 1 and 2 are cascaded to divide $B_{OSC}/4$ by 60,000 and generate an underflow interrupt.

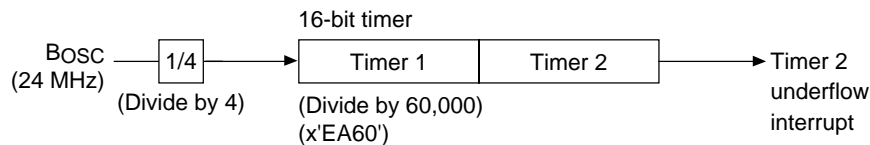


Figure 4-11 Configuration Example of Interval Timer Using Timers 1 and 2

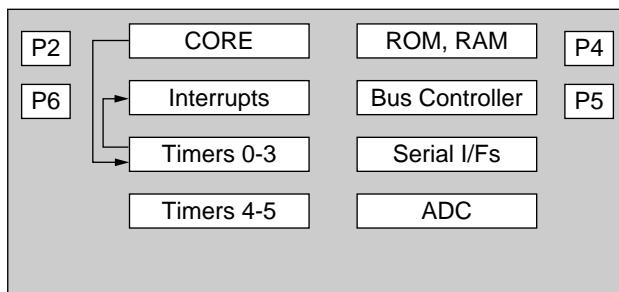


Figure 4-12 Block Diagram of Interval Timer Using Timers 1 and 2

1. Disable timer 1 and 2 counting in the timer 1 and 2 mode registers (TM1MD, TM2MD). This step is unnecessary immediately after a reset, since TM1MD and TM2MD reset to 0.

TM1MD (example)

x'00FE21'

Bit:	7	6	5	4	3	2	1	0
	TM1 EN	TM1 LD	—	—	—	—	TM1 S1	TM1 S0
Setting:	0	0	0	0	0	0	0	0

TM2MD (example)

x'00FE22'

Bit:	7	6	5	4	3	2	1	0
	TM2 EN	TM2 LD	—	—	—	—	TM2 S1	TM2 S0
Setting:	0	0	0	0	0	0	0	0

2. Cancel all existing interrupt requests and enable timer 2 underflow interrupts. To do this, set the TM2UDLV[2:0] bits of TM2UDICH (priority level 4 in this example), set the TM2UDIE bit to 1, set the TM2UDIR bit of TM2UDICL to 0, set the TM1UDIE bit of TM1UDICH to 0, and set the TM1UDIR bit of TM1UDICL to 0. (Note that you set the priority level for both timer 1 and 2 interrupts in the timer 2 interrupt control register.) From this point on, an interrupt request is generated whenever timer 2 underflows. Timer 1 underflows are unused.

Timers

8-Bit Timer Setup Examples

TM2UDICH (example)

x'00FC71'

Bit:	7	6	5	4	3	2	1	0
	—	TM2UD LV2	TM2UD LV1	TM2UD LV0	—	—	—	TM2UD IE
Setting:	0	1	0	0	0	0	0	1

TM2UDICL (example)

x'00FC70'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	TM2UD IR	—	—	—	TM2UD ID
Setting:	0	0	0	0	0	0	0	0

TM1UDICH (example)

x'00FC73'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	TM0UD IE
Setting:	0	0	0	0	0	0	0	0

TM1UDICL (example)

x'00FC72'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	TM1UD IR	—	—	—	TM1UD ID
Setting:	0	0	0	0	0	0	0	0

- Set the divide-by ratio for timer 0. Since the timer will count 60,000 cycles (x'EA60'), write x'5F' to the timer 1 base register (TM1BR) and x'EA' to the timer 2 base register (TM2BR). (The valid range for TMnBR is 0 to 255.)

TM1BR (example)

x'00FE11'

Bit:	7	6	5	4	3	2	1	0
	TM1 BR7	TM1 BR6	TM1 BR5	TM1 BR4	TM1 BR3	TM1 BR2	TM1 BR1	TM1 BR0
Setting:	0	1	0	1	1	1	1	1

TM2BR (example)

x'00FE12'

Bit:	7	6	5	4	3	2	1	0
	TM2 BR7	TM2 BR6	TM2 BR5	TM2 BR4	TM2 BR3	TM2 BR2	TM2 BR1	TM2 BR0
Setting:	1	1	1	0	1	0	1	0



Do not change the clock source once you select it. Selecting the clock source while you set up the count operation control will corrupt the value in the binary counter.

- Set the TM1LD bit of the TM1MD register and the TM2LD bit of the TM2MD register to 1. This loads the value in the base register to the binary counter. At the same time, select the clock source as the BOSC/4 for timer 1 and cascade to timer 1 for timer 2. (Write to TMnS[1:0].)

TM1MD (example)

x'00FE21'

Bit:	7	6	5	4	3	2	1	0
	TM1 EN	TM1 LD	—	—	—	—	TM1 S1	TM1 S0
Setting:	0	1	0	0	0	0	0	0

TM2MD (example)

x'00FE22'

	7	6	5	4	3	2	1	0
Bit:	TM2 EN	TM2 LD	—	—	—	—	TM2 S1	TM2 S0
Setting:	0	1	0	0	0	0	1	0

In the bank and linear addressing versions of the MN102 series, it was necessary to set TM0EN and TM0LD to 0 between steps 4 and 5, to ensure stable operation. This is unnecessary in the high-speed linear addressing version.

5. Set TM2LD to 0 and TM2EN to 1, then set TM1LD to 0 and TM1EN to 1. This starts the timers. Counting begins at the start of the next cycle. When both the timer 1 and 2 binary counters reach 0 and loads the values from the base registers, in preparation for the next count, a timer 2 underflow interrupt request is sent to the CPU. The timer 1 interrupt is unused.



Access TM2MD and TM1MD with a 16-bit write, using the MOV instruction, or set the two registers consecutively, beginning with TM2MD.

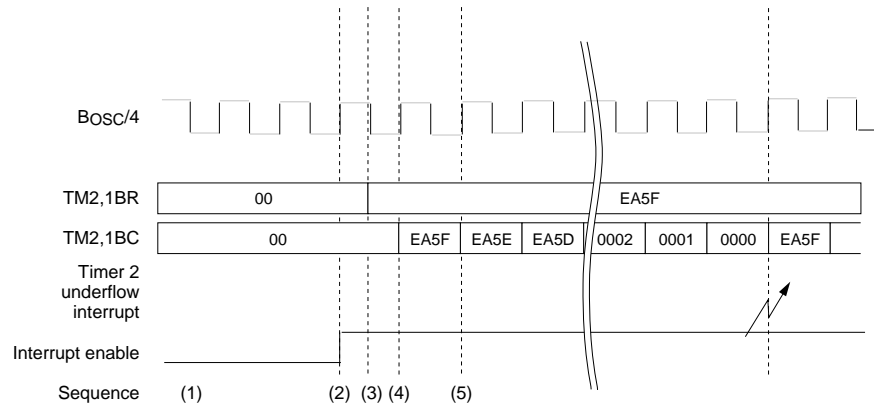


Figure 4-13 Interval Timer Timing (Timers 1 and 2)

4.6 8-Bit Timer Control Registers

Table 4-2 shows the registers used to control the 8-bit timers. A binary counter (TMnBC), a time base counter (TMnBR), and a timer mode register (TMnMD) is associated with each 8-bit timer.

Table 4-2 8-Bit Timer Control Registers

Register		Address	R/W	Description
Timer 0	TM0BC	x'00FE00'	R	Timer 0 binary counter
	TM0BR	x'00FE10'	R/W	Timer 0 base register
	TM0MD	x'00FE20'	R/W	Timer 0 mode register
Timer 1	TM1BC	x'00FE01'	R	Timer 1 binary counter
	TM1BR	x'00FE11'	R/W	Timer 1 base register
	TM1MD	x'00FE21'	R/W	Timer 1 mode register
Timer 2	TM2BC	x'00FE02'	R	Timer 2 binary counter
	TM2BR	x'00FE12'	R/W	Timer 2 base register
	TM2MD	x'00FE22'	R/W	Timer 2 mode register
Timer 3	TM3BC	x'00FE03'	R	Timer 3 binary counter
	TM3BR	x'00FE13'	R/W	Timer 3 base register
	TM3MD	x'00FE23'	R/W	Timer 3 mode register

TM0BC–TM3BC: Timer n Binary Counter

x'00FE00'–x'00FE03'

Bit:	7	6	5	4	3	2	1	0
	TMn BC7	TMn BC6	TMn BC5	TMn BC4	TMn BC3	TMn BC2	TMn BC1	TMn BC0
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R

TM0BR–TM3BR: Timer n Base Register

x'00FE10'–x'00FE13'

Bit:	7	6	5	4	3	2	1	0
	TMn BR7	TMn BR6	TMn BR5	TMn BR4	TMn BR3	TMn BR2	TMn BR1	TMn BR0
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

TM0MD–TM3MD: Timer n Mode Register

x'00FE20'–x'00FE23'

Bit:	7	6	5	4	3	2	1	0
	TMn EN	TMn LD	—	—	—	—	TMn S1	TMn S0
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R	R	R	R	R/W	R/W

TMnEN: TMnBC count enable
0: Disable / 1: Enable

TMnLD: TMnBR value load to TMnBC
0: Do not load value / 1: Load value

TMnS[1:0]: Timer n clock source select
See table 4-1 on page 78 for clock sources. 00 = clock source 0, 01 = clock source 1, 10 = clock source 2, and 11 = clock source 3.

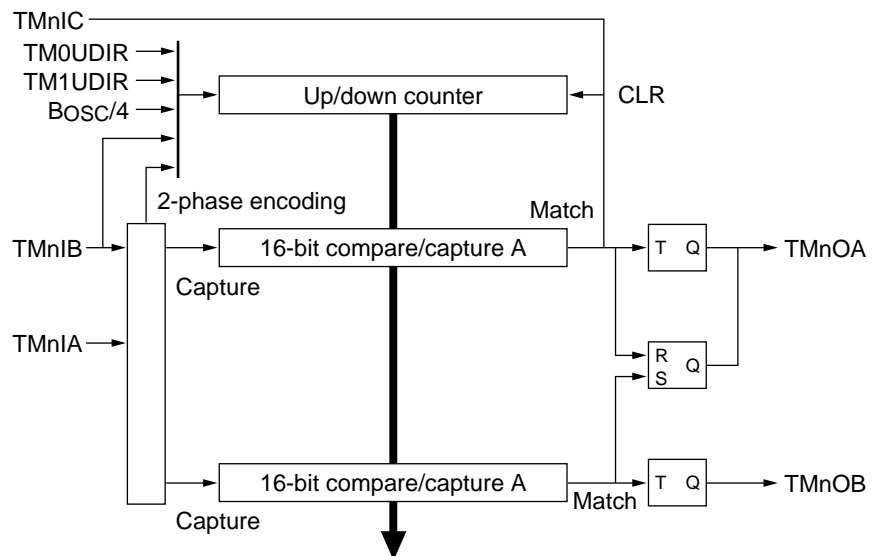
4.7 16-Bit Timer Description

The MN102H75K/85K contains two 16-bit up/down timers, timers 5 and 6. Associated with each timer are two compare/capture registers that can capture and compare the up/down counter values, generate PWM signals, and generate interrupts. The PWM function has a double buffering mode that causes cycle and transition changes to occur at the beginning of the next clock cycle. This prevents PWM signal losses and minimizes waveform distortion during timing changes.



Underflow interrupts can only occur during down counting.

Timers 5 and 6 can serve as interval timers, event counters (in clock oscillation mode), one- or two-phase PWMs, dual capture inputs, dual two-phase encoders, one-shot pulse generators, and external count direction controllers. The clock source can be the internal clock, the external clock, or the TM0UDIR or TM1UDIR signals from the 8-bit timers.



Note: $B_{OSC} = 24 \text{ MHz}$

Figure 4-14 Block Diagram of 16-Bit Timers

4.8 16-Bit Timer Features

Table 4-3 16-Bit Timer Functions and Features

Function/Feature	Timer 4	Timer 5
Interrupt request flag(s)	TM4UDIR bit of TM4UDICL TM4CAICL bit of TM4CAIR TM4CBICL bit of TM4CBIR	TM5UDIR bit of TM5UDICL TM5CAICL bit of TM5CAIR TM5CBICL bit of TM5CBIR
Interrupt sources	Timer 4 underflow Timer 4 compare A match Timer 4 capture A Timer 4 compare B match Timer 4 capture B	Timer 5 underflow Timer 5 compare A match Timer 5 capture A Timer 5 compare B match Timer 5 capture B
Clock sources	Timer 0 underflow Timer 1 underflow TM4IB signal 4x two-phase encoder (TM4IA and TM4IB signals) 1x two-phase encoder (TM4IA and TM4IB signals)	Timer 0 underflow Timer 1 underflow TM5IB signal 4x two-phase encoder (TM5IA and TM5IB signals) 1x two-phase encoder (TM5IA and TM5IB signals)
Count direction	Up/down counter	Up/down counter
Interval timer function	✓	✓
Event counter function	✓	✓
PWM function	✓	✓
One-shot pulse output	✓	✓
Single-phase capture input	✓	✓
Two-phase capture input	✓	✓
Two-phase encoding (4x)	✓	✓
Two-phase encoding (1x)	✓	✓
External count direction control	✓	✓

4.9 16-Bit Timer Block Diagrams

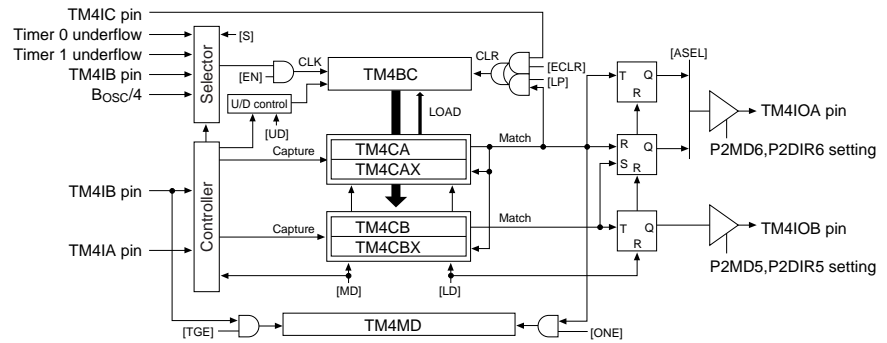


Figure 4-15 Timer 4 Block Diagram

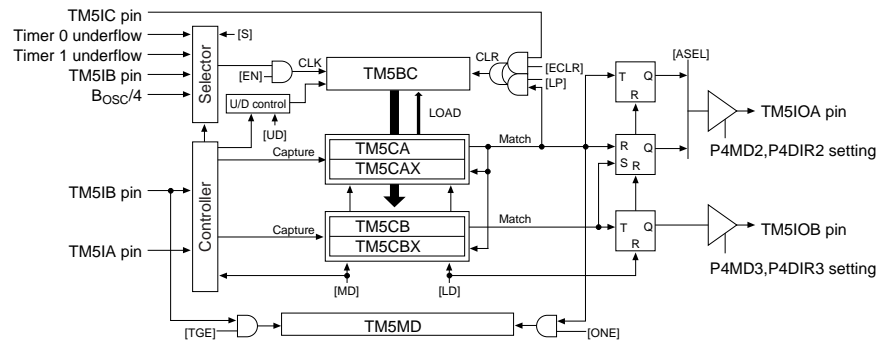


Figure 4-16 Timer 5 Block Diagram

4.10 16-Bit Timer Timing

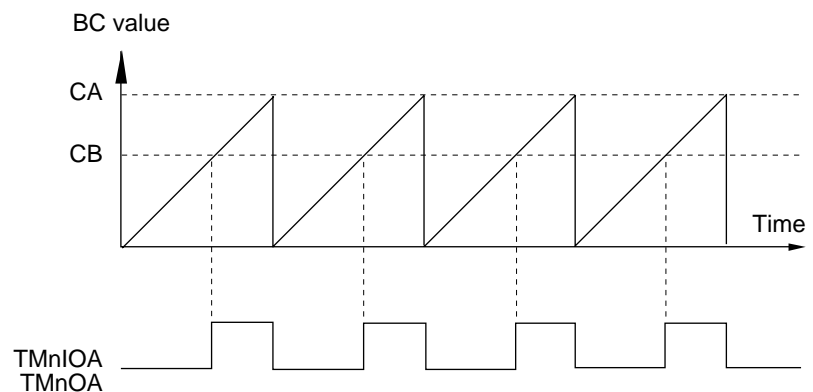


Figure 4-17 Single-Phase PWM Output Timing (16-Bit Timers)

Timers

16-Bit Timer Timing

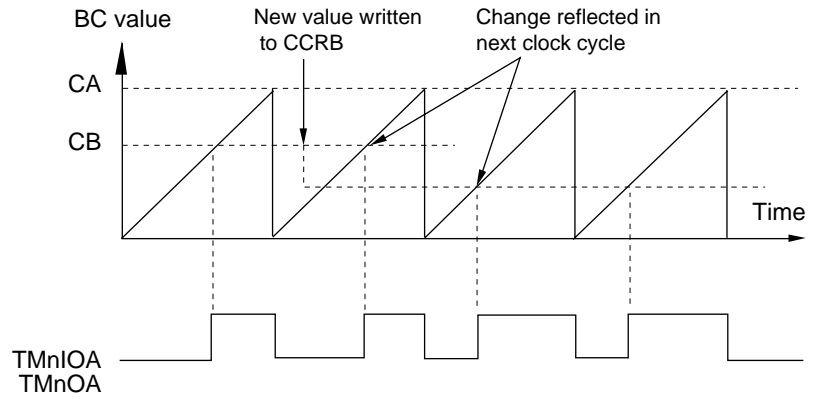


Figure 4-18 Single-Phase PWM Output Timing with Data Change (16-Bit Timers)

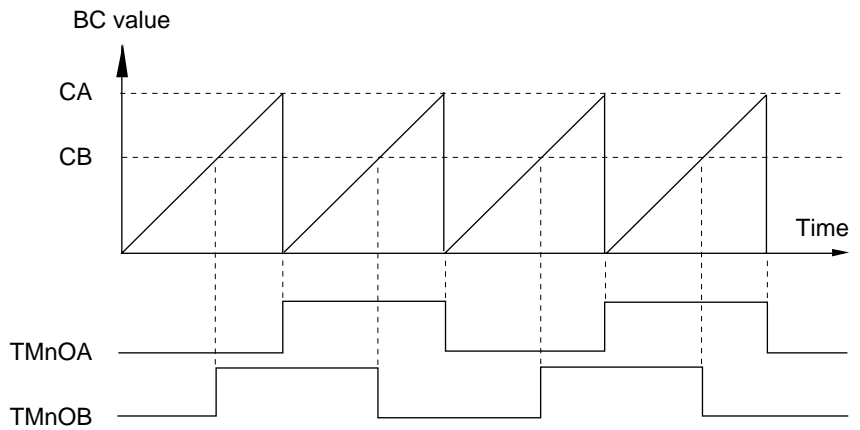


Figure 4-19 Two-Phase PWM Output Timing (16-Bit Timers)

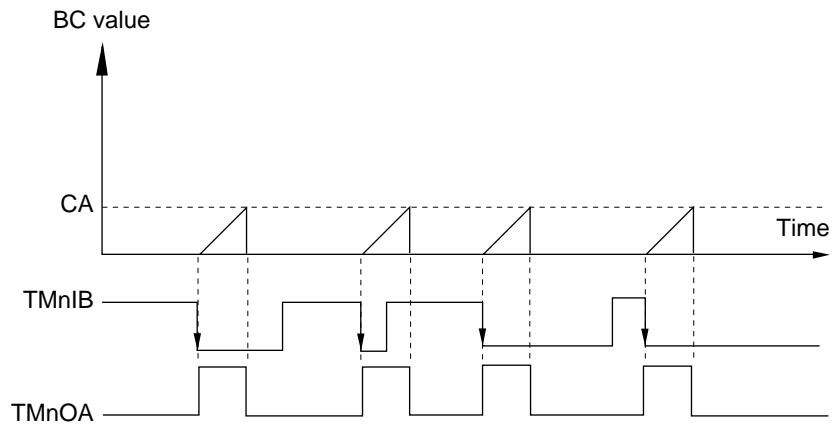


Figure 4-20 One-Shot Pulse Output Timing (16-Bit Timers)

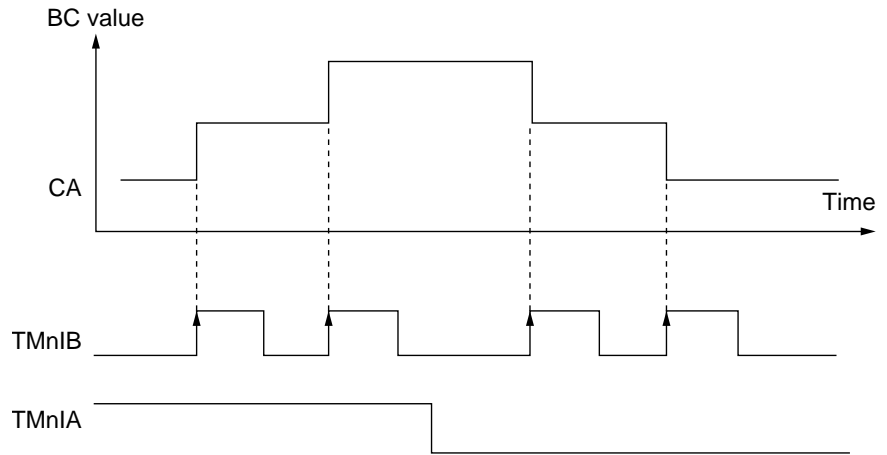


Figure 4-21 External Count Direction Control Timing (16-Bit Timers)

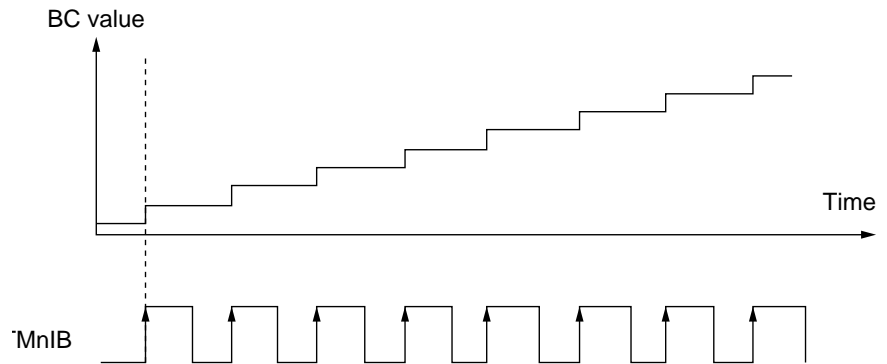


Figure 4-22 Event Timer Input Timing (16-Bit Timers)

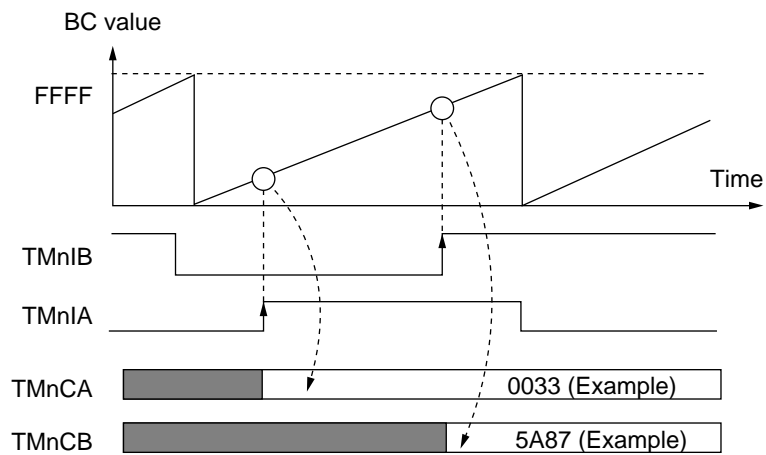


Figure 4-23 Single-Phase Capture Input Timing (16-Bit Timers)

Timers

16-Bit Timer Timing

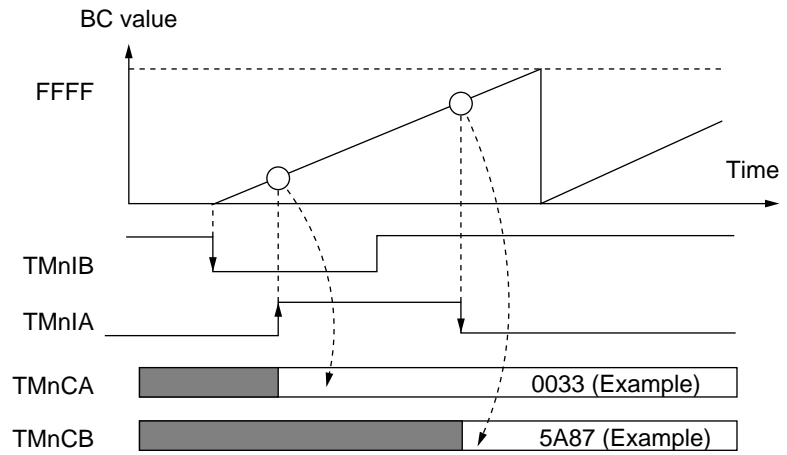


Figure 4-24 Two-Phase Capture Input Timing (16-Bit Timers)

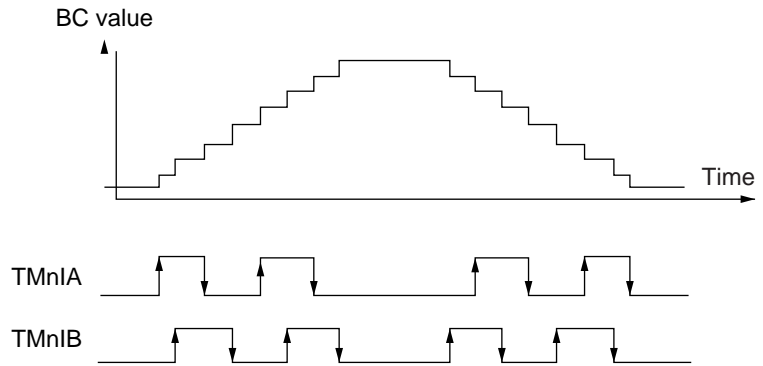


Figure 4-25 Two-Phase 4x Encoder Timing (16-Bit Timers)

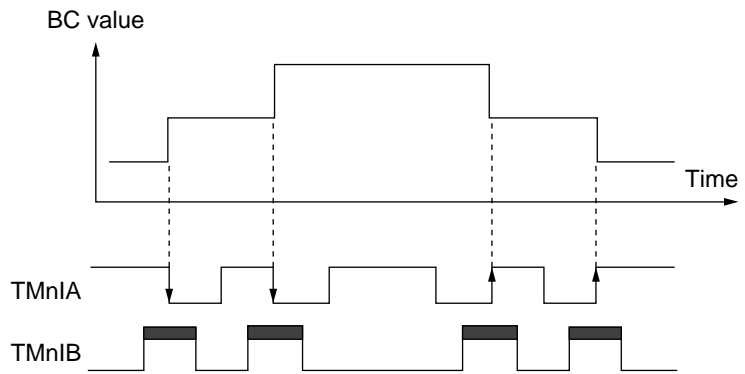
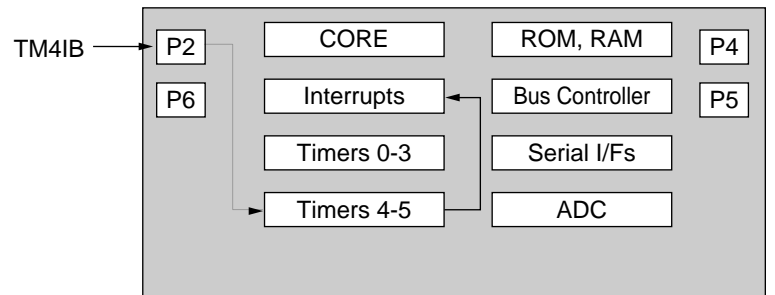


Figure 4-26 Two-Phase 1x Encoder Timing (16-Bit Timers)

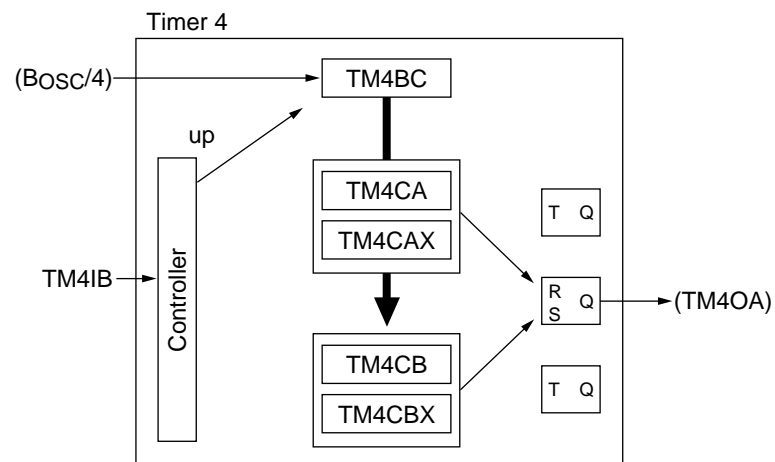
4.11 16-Bit Timer Setup Examples

4.11.1 Setting Up an Event Counter Using Timer 4

In this example, timer 4 counts the TM4IB input signal ($B_{OSC}/4 = 6\text{ MHz}$ or less) and generates an interrupt on the second and fifth cycles.



A. Chip Level



B. Block Level

Figure 4-27 Block Diagram of Event Counter Using Timer 4

■ **To set up timer 4:**

1. Set the operating mode in the timer 4 mode register (TM4MD). Disable timer 4 counting and interrupts. Select up counting. Select TM4IB as the clock source.



Use the MOV instruction for this setup and only use 16-bit write operations.

This step stops the TM4BC count and clears both TM4BC and the S-R flip-flop to 0.

TM4MD (example)

x'00FE80'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TM4 EN	TM4 NLD	—	—	TM4 UD1	TM4 UD0	TM4 TGE	TM4 ONE	TM4 MD1	TM4 MD0	TM4 ECLR	TM4 LP	TM4 ASEL	TM4 S2	TM4 S1	TM4 S0
Setting:	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0

2. Set the divide-by ratio for timer 4. To divide the TM4IB input signal by 5, write x'0004' to timer 4 compare/capture register A (TM4CA). (The valid range for TM4CA is x'0001' to x'FFFE'.)

Timers

16-Bit Timer Setup Examples

TM4CA (example)																x'00FE84'
Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TM4 CA15	TM4 CA14	TM4 CA13	TM4 CA12	TM4 CA11	TM4 CA10	TM4 CA9	TM4 CA8	TM4 CA7	TM4 CA6	TM4 CA5	TM4 CA4	TM4 CA3	TM4 CA2	TM4 CA1	TM4 CA0
Setting:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

- Set the phase difference for timer 4. For a 2-cycle phase difference, write x'0001' to timer 4 compare/capture register B (TM4CB). (The valid range is $-1 \leq \text{TM4CB} < \text{the TM4CA value.}$)

TM4CB (example)																x'00FE88'
Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TM4 CB15	TM4 CB14	TM4 CB13	TM4 CB12	TM4 CB11	TM4 CB10	TM4 CB9	TM4 CB8	TM4 CB7	TM4 CB6	TM4 CB5	TM4 CB4	TM4 CB3	TM4 CB2	TM4 CB1	TM4 CB0
Setting:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

- Set the TM4NLD bit of the TM4MD register to 1 and the TM4EN bit to 0. This enables TM4BC and the S-R flip-flop. This step ensures stable operation. If it is omitted, the binary counter may not count the first cycle. Do not change any other operating modes during this step.
- Set TM4NLD and TM4EN to 1. This starts the timer. Counting begins at the start of the next cycle.

■ To enable timer 4 capture interrupts:

Cancel all existing interrupt requests. Next, set the interrupt priority level in the TM4CBLV[2:0] bits of the TM4CBICH register (levels 0 to 6), set the TM4CBIE bit to 1, set the TM4CBIR bit of TM4CBICL to 0, set the TM4CAIE bit of TM4CAICH to 1, and set the TM4CAIR bit of TM4CAICL to 0. From this point on, an interrupt request is generated whenever a timer 4 capture A or capture B event occurs.

Timer 4 can operate as an event counter, but timer 4 does not operate in STOP mode, when B_{OSC} is off. If you use an external clock, it must be synchronized to B_{OSC}. This means that the frequency of the event counter clock must be 1/4 or less that of the oscillator (6 MHz with a 24-MHz oscillator).

Figure 4-28 shows an example timing chart.

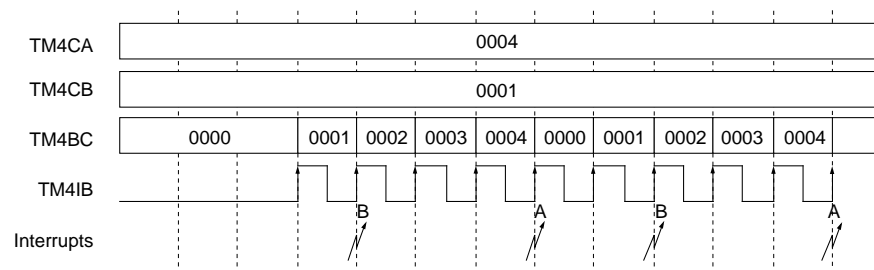
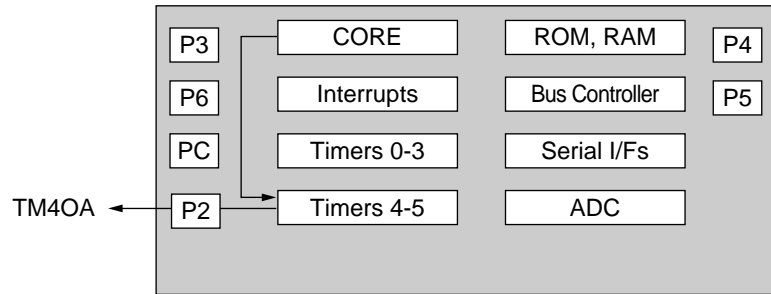


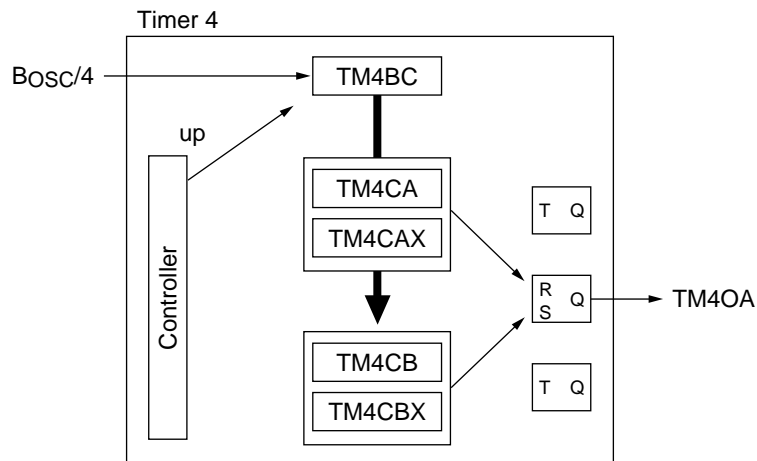
Figure 4-28 Event Counter Timing (Timer 4)

4.11.2 Setting Up a Single-Phase PWM Output Signal Using Timer 4

In this example, timer 4 is used to divide B_{OSC} by 5 and generate a five-cycle, single-phase PWM signal. The duty of this signal is 2:3. To accomplish this, the program must load the divide-by ratio of 5 (actual setting: 4) into compare/capture register A and a cycle count of 2 (actual setting: 1) into compare/capture register B.



A. Chip Level



B. Block Level

Figure 4-29 Block Diagram of Single-Phase PWM Output Using Timer 4

■ **To set up the output port:**

Set the P2MD[13:12] bits of the port 2 output mode register (P2MD) to b'01' (selecting the TM4IOA pin) and set the P2DIR6 bit of the port 2 I/O control register (P2DIR) to 1 (selecting output direction). This step selects the TM4OA pin (P26) as the timer output port.

P2MD (example)

x'00FFF4'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	P2 MD14	P2 MD13	P2 MD12	P2 MD11	P2 MD10	P2 MD9	P2 MD8	P2 MD7	P2 MD6	P2 MD5	P2 MD4	P2 MD3	P2 MD2	P2 MD1	P2 MD0
Setting:	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0

Timers

16-Bit Timer Setup Examples

P2DIR (example)

x'00FFE2'

Bit:	7	6	5	4	3	2	1	0
	P2 DIR7	P2 DIR6	P2 DIR5	P2 DIR4	P2 DIR3	P2 DIR2	P2 DIR1	P2 DIR0
Setting:	0	1	0	0	0	0	0	0

■ To set up timer 4:

1. Set the operating mode in the timer 4 mode register (TM4MD). Disable timer 4 counting and interrupts. Select up counting. Select $B_{OSC}/4$ as the clock source. Select the double-buffer operating mode.



Use the MOV instruction for this setup and only use 16-bit write operations.

This step stops the TM4BC count and clears both TM4BC and the S-R flip-flop to 0.

TM4MD (example)

x'00FE80'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TM4 EN	TM4 NLD	—	—	TM4 UD1	TM4 UD0	TM4 TGE	TM4 ONE	TM4 MD1	TM4 MD0	TM4 ECLR	TM4 LP	TM4 ASEL	TM4 S2	TM4 S1	TM4 S0
Setting:	0	0	0	0	0	0	0	0	0	1	0	1	0	0	1	1

2. Set the divide-by ratio for timer 4. To divide $B_{OSC}/4$ by 5, write x'0004' to timer 4 compare/capture register A (TM4CA). (The valid range for TM4CA is x'0001' to x'FFFE'.)

TM4CA (example)

x'00FE84'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TM4 CA15	TM4 CA14	TM4 CA13	TM4 CA12	TM4 CA11	TM4 CA10	TM4 CA9	TM4 CA8	TM4 CA7	TM4 CA6	TM4 CA5	TM4 CA4	TM4 CA3	TM4 CA2	TM4 CA1	TM4 CA0
Setting:	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

3. Set the timer 4 duty cycle. For a $2/5 B_{OSC}/4$ duty cycle, write x'0001' to timer 4 compare/capture register B (TM4CB). (The valid range is $-1 < TM4CB < \text{the TM4CA value}$.)

TM4CB (example)

x'00FE88'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TM4 CB15	TM4 CB14	TM4 CB13	TM4 CB12	TM4 CB11	TM4 CB10	TM4 CB9	TM4 CB8	TM4 CB7	TM4 CB6	TM4 CB5	TM4 CB4	TM4 CB3	TM4 CB2	TM4 CB1	TM4 CB0
Setting:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

4. Write a dummy data word (of any value) to TM4CAX. In double-buffer mode, TM4CA is compared to TM4CAX. The contents of TM4CA are loaded to TM4CAX when $TM4BC = TM4CAX$. However, since TM4CAX is undefined or x'0000' before this operation starts, this initial dummy write prevents timing errors.
5. Write a dummy data word (of any value) to TM4CBX. In double-buffer mode, TM4CB is compared to TM4CBX. The contents of TM4CB are loaded to TM4CBX when $TM4BC = TM4CBX$. However, since TM4CBX is undefined or x'0000' before this operation starts, this initial dummy write prevents timing errors.

6. Set the TM4NLD bit of the TM4MD register to 1 and the TM4EN bit to 0. This enables TM4BC and the S-R flip-flop. This step ensures stable operation. If it is omitted, the binary counter may not count the first cycle. Do not change any other operating modes during this step.
7. Set TM4NLD and TM4EN to 1. This starts the timer. Counting begins at the start of the next cycle.

Timer 4 can output a single-phase PWM signal at any duty. You must select up counting. Timer 4 does not operate in STOP mode, when B_{OSC} is off. If you use an external clock, it must be synchronized to B_{OSC}.

In this procedure, you set the cycle (x'0001' to x'FFFE') in the TM4CA register and the duty in the TM4CB register. When the contents of TM4BC match those of the TM4CB register, the S-R flip-flop resets at the beginning of the next cycle. Please note the following:

- When $-1 \leq \text{TM4CB} < \text{TM4CA}$, TM4OA output is low during the 0 to TM4CB + 1 cycles of the TM4CA + 1 cycle period and high during the remainder of the cycles.
- When $\text{TM4CA} \leq \text{TM4CB} \leq \text{x'FFFE}$, TM4OA output is always low.
- When $\text{TM4BC} = \text{x'FFFF}$, TM4OA output is always high.

The circuitry is configured so that there are no waveform errors, even when the output is always high or always low. Counting begins after the TM4EN bit is set in the TM4MD register.

Timers

16-Bit Timer Setup Examples

Figure 4-30 below shows the output waveforms for TM40A. Both A and B interrupts can occur, but B interrupts can only occur if the TM4CB setting is from 0 to less than TM4CA. This is because when $TM4CB \leq TM4CA$, TM4BC never matches TM4CB.

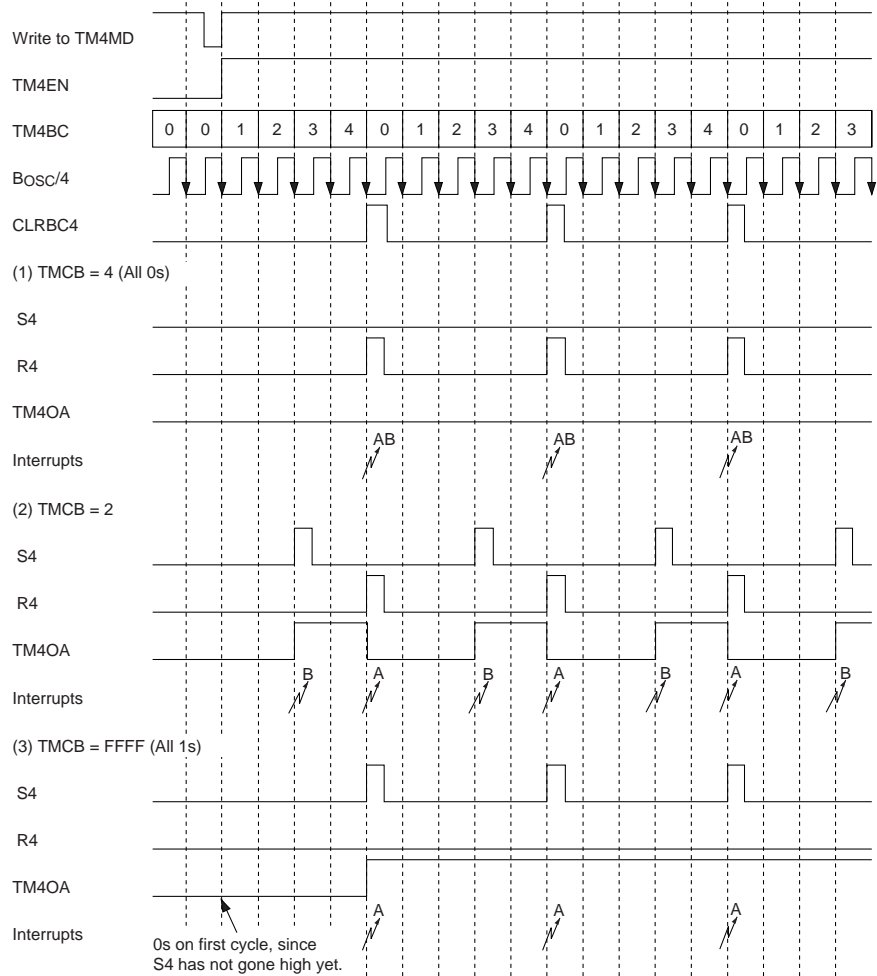


Figure 4-30 Single-Phase PWM Output Timing (Timer 4)

Two potential types of errors are inherent with PWM output. First, because of the circuit configuration, direction errors can occur. The output circuit is configured with T flip-flops, so that even if one transition is missed, the 1s and 0s can reverse direction. Timers 4 and 5 contain an S-R flip-flop to prevent this type of error. Second, if the duty cycle changes dynamically, which often happens in PWM output, the PWM waveform may skip a pulse (see the single buffering section of figure 4-31 below). To prevent these misses, timers 4 and 5 provide a double-buffer mode. In this mode, no matter what the timing of a TMnCB change, the duty change does occur until the beginning of the next cycle, and no signals are lost. Performance is assured even when the output switches from all 1s to all 0s (see the double buffering section of figure 4-31 below).

For this reason, you must always use double-buffer mode for PWM waveform output. Use single-buffer mode only in applications that are unaffected by these issues.

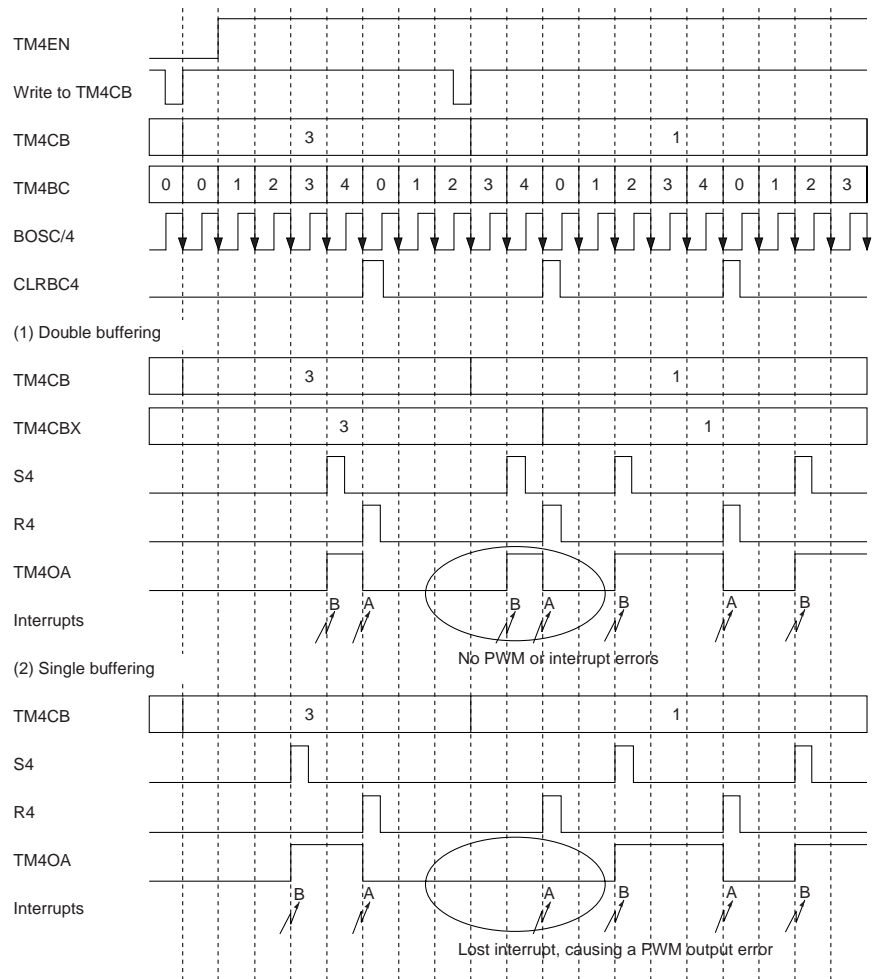


Figure 4-31 Single-Phase PWM Output Timing with Dynamic Duty Changes (Timer 4)

4.11.3 Setting Up a Two-Phase PWM Output Signal Using Timer 4

In this example, timer 4 is used to divide timer 0 underflow by 5 and generate a five-cycle, two-phase PWM signal. The phase difference of this signal is 2 cycles. To accomplish this, the program must load the divide-by ratio of 5 (actual setting: 4) into compare/capture register A and a cycle count of 2 (actual setting: 1) into compare/capture register B.

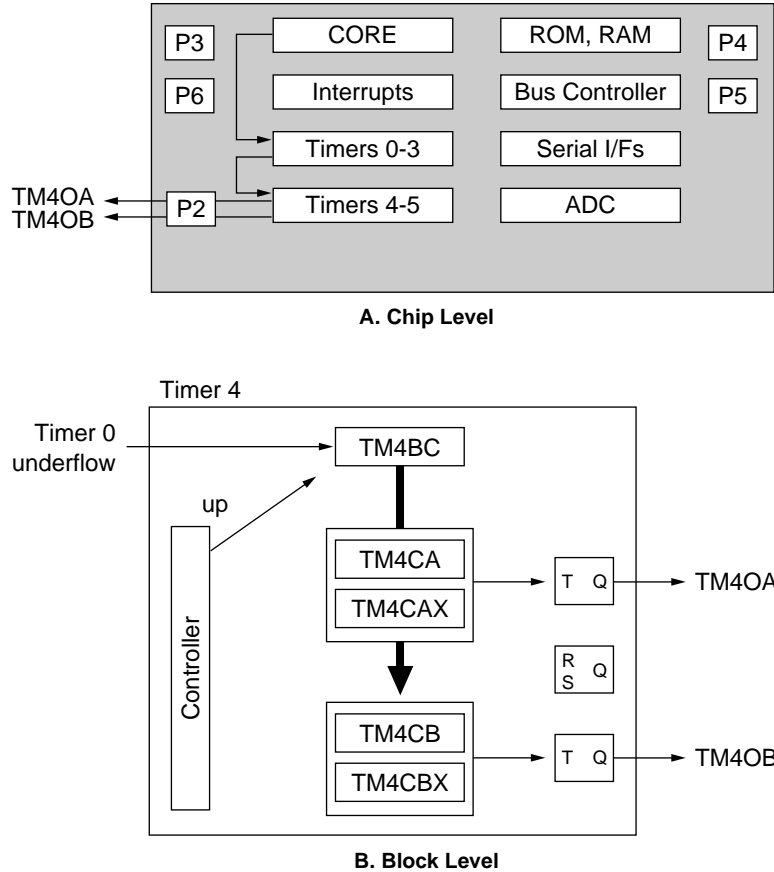


Figure 4-32 Block Diagram of Two-Phase PWM Output Using Timer 4

■ **To set up the output port:**

Set the P2MD[13:12] bits of the port 2 output mode register (P2MD) to b'01' (selecting the TM4IOA pin), set the P2MD[11:10] bits to b'01' (selecting the TM4IOB pin), and set the P2DIR[6:5] bits of the port 2 I/O control register (P2DIR) to b'11' (selecting output direction). This step selects the TM4OA (P26) and TM4OB (P25) pins as the timer output ports.

P2MD (example)

x'00FFF4'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	P2 MD14	P2 MD13	P2 MD12	P2 MD11	P2 MD10	P2 MD9	P2 MD8	P2 MD7	P2 MD6	P2 MD5	P2 MD4	P2 MD3	P2 MD2	P2 MD1	P2 MD0
Setting:	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0

P2DIR (example)

x'00FFE2'

	7	6	5	4	3	2	1	0
Bit:	P2 DIR7	P2 DIR6	P2 DIR5	P2 DIR4	P2 DIR3	P2 DIR2	P2 DIR1	P2 DIR0
Setting:	0	1	1	0	0	0	0	0

■ **To set up timer 0:**

1. Disable timer 0 counting in the timer 0 mode register (TM0MD). This step is unnecessary immediately after a reset, since TM0MD resets to 0.

TM0MD (example)

x'00FE20'

	7	6	5	4	3	2	1	0
Bit:	TM0 EN	TM0 LD	—	—	—	—	TM0 S1	TM0 S0
Setting:	0	0	0	0	0	0	—	—

2. Set the divide-by ratio for timer 0. To divide $B_{OSC}/4$ by two, write x'01' to the timer 0 base register (TM0BR). (The valid range for TM0BR is 0 to 255.)

TM0BR (example)

x'00FE10'

	7	6	5	4	3	2	1	0
Bit:	TM0 BR7	TM0 BR6	TM0 BR5	TM0 BR4	TM0 BR3	TM0 BR2	TM0 BR1	TM0 BR0
Setting:	0	0	0	0	0	0	0	1

3. Set the TM0LD bit of the TM0MD register to 1. This loads the value in the base register to the binary counter. At the same time, select the clock source as $B_{OSC}/4$ by writing b'00' to TM0S[1:0].

TM0MD (example)

x'00FE20'

	7	6	5	4	3	2	1	0
Bit:	TM0 EN	TM0 LD	—	—	—	—	TM0 S1	TM0 S0
Setting:	0	1	0	0	0	0	0	0

4. Set TM0LD to 0 and TM0EN to 1. This starts the timer. Counting begins at the start of the next cycle. When the binary counter reaches 0 and loads the value x'01' from the base register, in preparation for the next count, a timer 0 underflow interrupt request is sent to the CPU.



Do not change the clock source once you select it. Selecting the clock source while you set up the count operation control will corrupt the value in the binary counter.

In the bank and linear addressing versions of the MN102 series, it was necessary to set TM0EN and TM0LD to 0 between steps 3 and 4, to ensure stable operation. This is unnecessary in the high-speed linear addressing version.

Timers

16-Bit Timer Setup Examples



Use the MOV instruction for this setup and only use 16-bit write operations.

This step stops the TM4BC count and clears both TM4BC and the S-R flip-flop to 0.

■ To set up timer 4:

1. Set the operating mode in the timer 4 mode register (TM4MD). Disable timer 4 counting and interrupts. Select up counting. Select timer 0 underflows as the clock source.

TM4MD (example)

x'00FE80'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TM4 EN	TM4 NLD	—	—	TM4 UD1	TM4 UD0	TM4 TGE	TM4 ONE	TM4 MD1	TM4 MD0	TM4 ECLR	TM4 LP	TM4 ASEL	TM4 S2	TM4 S1	TM4 S0
Setting:	0	0	0	0	0	0	0	0	0	1	0	1	1	0	0	0

2. Set the divide-by ratio for timer 4. To divide timer 0 underflow by 5, write x'0004' to timer 4 compare/capture register A (TM4CA). (The valid range for TM4CA is x'0001' to x'FFFE'.)

TM4CA (example)

x'00FE84'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TM4 CA15	TM4 CA14	TM4 CA13	TM4 CA12	TM4 CA11	TM4 CA10	TM4 CA9	TM4 CA8	TM4 CA7	TM4 CA6	TM4 CA5	TM4 CA4	TM4 CA3	TM4 CA2	TM4 CA1	TM4 CA0
Setting:	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

3. Set the phase difference for timer 4. For a phase difference of two timer 0 underflow cycles, write x'0001' to timer 4 compare/capture register B (TM4CB). (The valid range is $-1 < \text{TM4CB} < \text{the TM4CA value.}$)

TM4CB (example)

x'00FE88'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TM4 CB15	TM4 CB14	TM4 CB13	TM4 CB12	TM4 CB11	TM4 CB10	TM4 CB9	TM4 CB8	TM4 CB7	TM4 CB6	TM4 CB5	TM4 CB4	TM4 CB3	TM4 CB2	TM4 CB1	TM4 CB0
Setting:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

4. Write a dummy data word (of any value) to TM4CAX. In double-buffer mode, TM4CA is compared to TM4CAX. The contents of TM4CA are loaded to TM4CAX when $\text{TM4BC} = \text{TM4CAX}$. However, since TM4CAX is undefined or x'0000' before this operation starts, this initial dummy write prevents timing errors.
5. Write a dummy data word (of any value) to TM4CBX. In double-buffer mode, TM4CB is compared to TM4CBX. The contents of TM4CB are loaded to TM4CBX when $\text{TM4BC} = \text{TM4CBX}$. However, since TM4CBX is undefined or x'0000' before this operation starts, this initial dummy write prevents timing errors.

6. Set the TM4NLD bit of the TM4MD register to 1 and the TM4EN bit to 0. This enables TM4BC and the S-R flip-flop. This step ensures stable operation. If it is omitted, the binary counter may not count the first cycle. Do not change any other operating modes during this step.
7. Set TM4NLD and TM4EN to 1. This starts the timer. Counting begins at the start of the next cycle.

Timer 4 can output a two-phase PWM signal with any phase difference. You must select up counting. Timer 4 does not operate in STOP mode, when B_{OSC} is off. If you use an external clock, it must be synchronized to B_{OSC}.

In this procedure, you set the cycle (x'0001' to x'FFFE') in the TM4CA register and the phase difference in the TM4CB register. When the contents of TM4BC match those of the TM4CB register, T flip-flop B reverses at the beginning of the next cycle. When the contents of TM4BC match those of the TM4CA register, T flip-flop A reverses and TM4BC resets at the beginning of the next cycle.

The circuitry is configured so that there are no waveform errors, even when the output is always high or always low. Counting begins after the TM4EN bit is set in the TM4MD register.

Figure 4-30 below shows the output waveforms for TM4OA and TM4OB. Both A and B interrupts occur when the contents of the binary counter matches those of the associated compare/capture register. However, B interrupts can only occur if the TM4CB setting is from 0 to less than TM4CA. This is because when $TM4CB \leq TM4CA$, TM4BC never matches TM4CB.

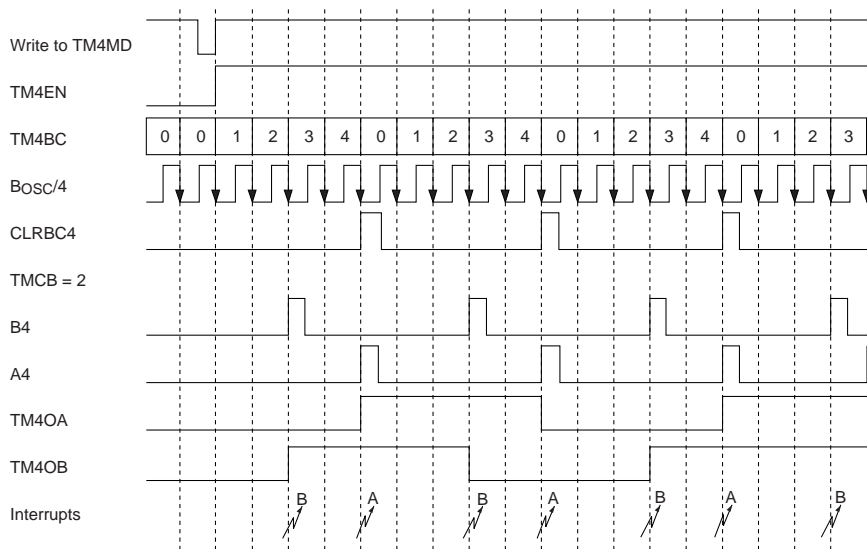


Figure 4-33 Two-Phase PWM Output Timing (Timer 4)

Timers

16-Bit Timer Setup Examples

With PWM output, the duty cycle can change dynamically, which can cause the PWM waveform to skip a pulse (see the single buffering section of figure 4-34 below). To prevent these misses, timers 4 and 5 provide a double-buffer mode. In this mode, no matter what the timing of a TMnCB change, the duty change does not occur until the beginning of the next cycle, and no signals are lost. Performance is assured even when the output switches from all 1s to all 0s (see the double buffering section of figure 4-34 below).

For this reason, you must always use double-buffer mode for PWM waveform output. Use single-buffer mode only in applications that are unaffected by this issues.

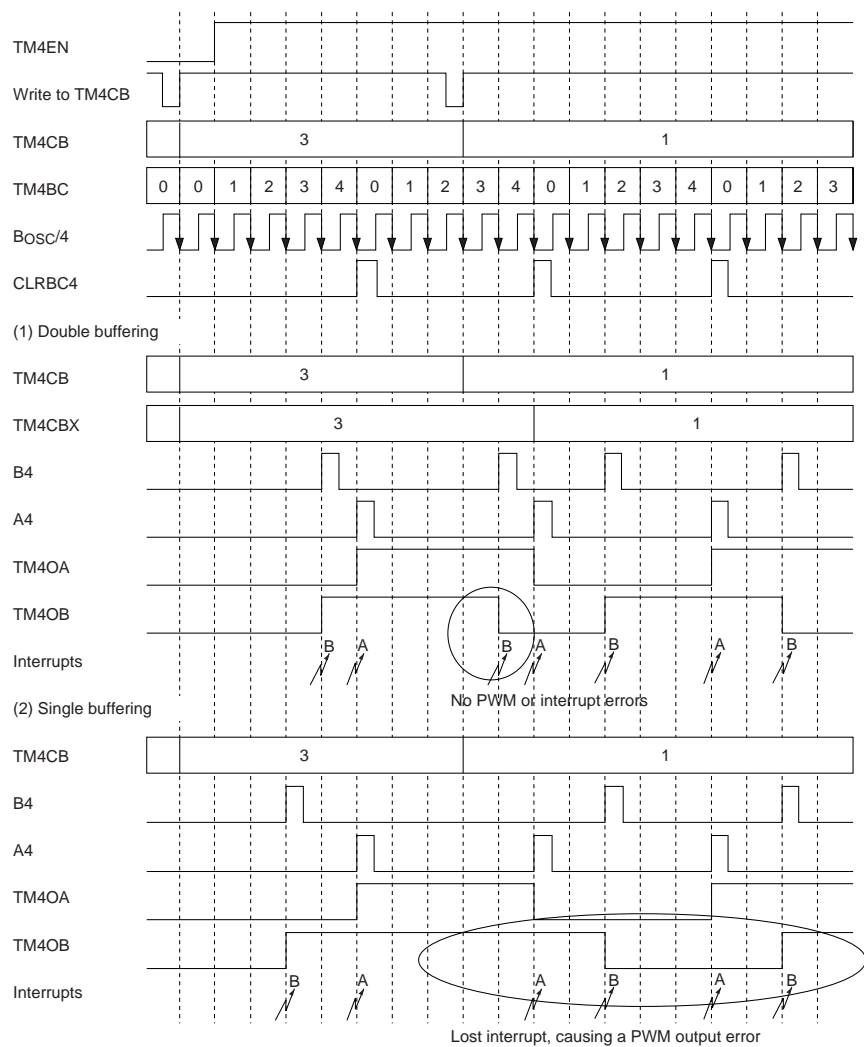
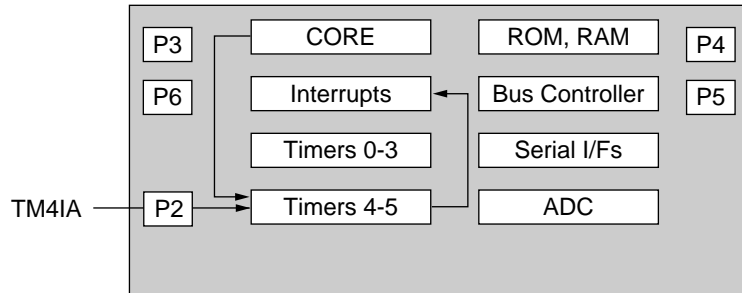


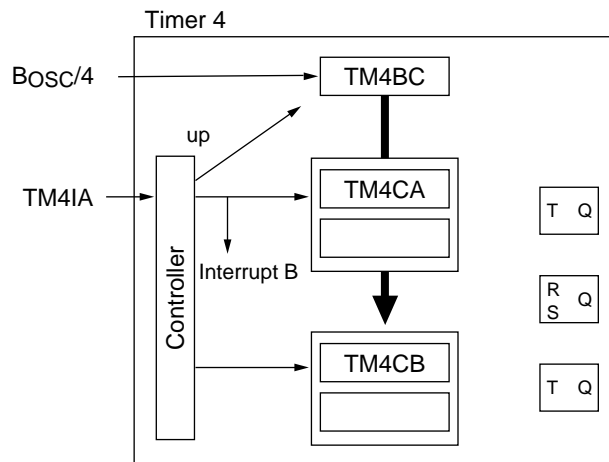
Figure 4-34 Two-Phase PWM Output Timing with Dynamic Duty Changes (Timer 4)

4.11.4 Setting Up a Single-Phase Capture Input Using Timer 4

In this example, timer 4 is used to divide $B_{OSC}/4$ by 65,536 and measure how long the TM4IA input signal stays high. An interrupt occurs on capture B and the software calculates the number of cycles by subtracting the contents of TMnCA from the contents of TMnCB.



A. Chip Level



B. Block Level

Figure 4-35 Block Diagram of Single-Phase Capture Input Using Timer 4

■ To set up timer 4:

1. Set the operating mode in the timer 4 mode register (TM4MD). Disable timer 4 counting and interrupts. Select up counting. Set the TM4NLP bit to 0 to select looped counting from 0 to x'FFFF'. Select $B_{OSC}/4$ as the clock source.



Use the MOV instruction for this setup and only use 16-bit write operations.

This step stops the TM4BC count and clears both TM4BC and the S-R flip-flop to 0.

TM4MD (example)

x'00FE80'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TM4 EN	TM4 NLD	—	—	TM4 UD1	TM4 UD0	TM4 TGE	TM4 ONE	TM4 MD1	TM4 MD0	TM4 ECLR	TM4 LP	TM4 ASEL	TM4 S2	TM4 S1	TM4 S0
Setting:	0	0	0	0	0	0	0	0	1	0	0	0	1 or 0	0	1	1

2. Set the TM4NLD bit of the TM4MD register to 1 and the TM4EN bit to 0. This enables TM4BC and the S-R flip-flop. This step ensures stable operation. If it is omitted, the binary counter may not count the first cycle. Do not

Timers

16-Bit Timer Setup Examples

When $TM4MD[1:0] = b'10'$ (during capture), $TM4CA$ and $TM4CB$ become read-only registers. To write to $TM4CA$ or $TM4CB$, you must first set $TM4MD[1:0] = b'00'$.

Ignore the flags when calculating the signal width, even when $TM3CA$ is the larger value.

change any other operating modes during this step.

3. Set $TM4NLD$ and $TM4EN$ to 1. This starts the timer. Counting begins at the start of the next cycle.

■ **To enable timer 4 capture B interrupts:**

Cancel all existing interrupt requests. Next, set the interrupt priority level in the $TM4CBLV[2:0]$ bits of the $TM4CBICH$ register (levels 0 to 6), set the $TM4BIE$ bit to 1, and set the $TM4BIR$ bit of $TM4CBICL$ to 0. From this point on, an interrupt request is generated whenever a timer 4 capture B event occurs.

■ **To service the interrupts and calculate the signal width:**

1. Run the interrupt service routine. The routine must determine the interrupt group, then clear the interrupt request flag.
2. Calculate the number of cycles the $TM4IA$ signal stays high. Save the contents of $TM4CA$ and $TM4CB$ to the data registers, then subtract the contents of $TM4CA$ from the contents of $TM4CB$. Since $TM4LP$ is set to 0, the difference will be the correct value even if $TM4CA$ is greater than $TM4CB$.

Timer 4 can input a single-phase capture signal. You must select up counting. Timer 4 does not operate in STOP mode, when B_{OSC} is off. If you use an external clock, it must be synchronized to B_{OSC} .

$TM4CA$ captures the count on the rising edge of $TM4IA$, and $TM4CB$ captures the count on the falling edge of $TM4IA$. A timer 4 capture B interrupt occurs when $TM4CB$ captures the count, and the contents of $TM4CA$ and $TM4CB$ are read during the interrupt service routine.

In the example timing chart shown in figure 4-36, $x'000A' - x'0007' = x'0003'$, or 3 cycles. The calculation is correct even when $TM4CA$ is the larger value. The flags are ignored, so for instance, $x'0003' - x'FFFE' = x'0005'$.

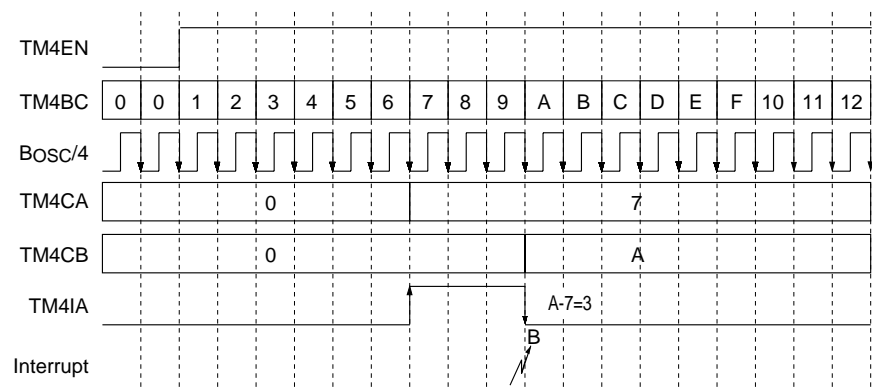
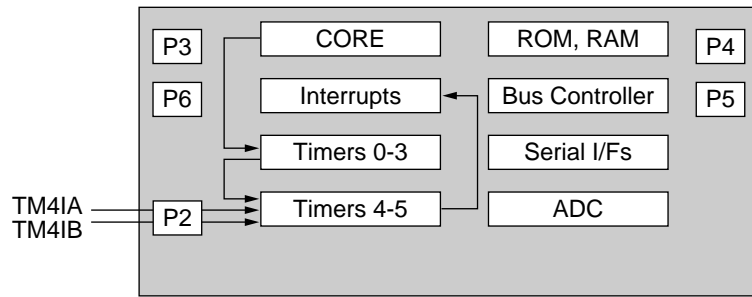


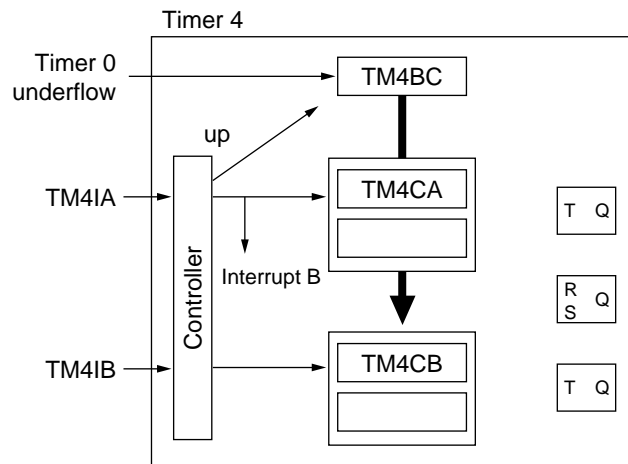
Figure 4-36 Single-Phase Capture Input Timing (Timer 4)

4.11.5 Setting Up a Two-Phase Capture Input Using Timer 4

In this example, timer 4 is used to divide the timer 0 underflow by 65,536 and measure the number of cycles from the rising edge of the TM4IA input signal to the rising edge of the TM4IB input signal. An interrupt occurs on capture B and the software calculates the number of cycles by subtracting the contents of TMnCA from the contents of TMnCB.



A. Chip Level



B. Block Level

Figure 4-37 Block Diagram of Two-Phase Capture Input Using Timer 4

■ **To set up timer 0:**

1. Disable timer 0 counting in the timer 0 mode register (TM0MD). This step is unnecessary immediately after a reset, since TM0MD resets to 0.

TM0MD (example)

x'00FE20'

Bit:	7	6	5	4	3	2	1	0
	TM0 EN	TM0 LD	—	—	—	—	TM0 S1	TM0 S0
Setting:	0	0	0	0	0	0	—	—

2. Set the divide-by ratio for timer 0. To divide $B_{OSC}/4$ by two, write x'01' to the timer 0 base register (TM0BR). (The valid range for TM0BR is 0 to 255.)

Timers

16-Bit Timer Setup Examples



Do not change the clock source once you select it. Selecting the clock source while you set up the count operation control will corrupt the value in the binary counter.

In the bank and linear addressing versions of the MN102 series, it was necessary to set TM0EN and TM0LD to 0 between steps 3 and 4, to ensure stable operation. This is unnecessary in the high-speed linear addressing version.



Use the MOV instruction for this setup and only use 16-bit write operations.

This step stops the TM4BC count and clears both TM4BC and the S-R flip-flop to 0.

When TM4MD[1:0] = b'11' (during capture), TM4CA and TM4CB become read-only registers. To write to TM4CA or TM4CB, you must first set TM4MD[1:0] = b'00'.

TM0BR (example)

x'00FE10'

Bit:	7	6	5	4	3	2	1	0
	TM0 BR7	TM0 BR6	TM0 BR5	TM0 BR4	TM0 BR3	TM0 BR2	TM0 BR1	TM0 BR0
Setting:	0	0	0	0	0	0	0	1

- Set the TM0LD bit of the TM0MD register to 1. This loads the value in the base register to the binary counter. At the same time, select the clock source as $B_{OSC}/4$ by writing b'00' to TM0S[1:0].

TM0MD (example)

x'00FE20'

Bit:	7	6	5	4	3	2	1	0
	TM0 EN	TM0 LD	—	—	—	—	TM0 S1	TM0 S0
Setting:	0	1	0	0	0	0	0	0

- Set TM0LD to 0 and TM0EN to 1. This starts the timer. Counting begins at the start of the next cycle. When the binary counter reaches 0 and loads the value x'01' from the base register, in preparation for the next count, a timer 0 underflow interrupt request is sent to the CPU.

■ To set up timer 4:

- Set the operating mode in the timer 4 mode register (TM4MD). Disable timer 4 counting and interrupts. Select up counting. Set the TM4NLP bit to 0 to select looped counting from 0 to x'FFFF'. Select timer 0 underflow as the clock source.

TM4MD (example)

x'00FE80'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TM4 EN	TM4 NLD	—	—	TM4 UD1	TM4 UD0	TM4 TGE	TM4 ONE	TM4 MD1	TM4 MD0	TM4 ECLR	TM4 LP	TM4 ASEL	TM4 S2	TM4 S1	TM4 S0
Setting:	0	0	0	0	0	0	0	0	1	1	0	0	1 or 0	0	0	0

- Set the TM4NLD bit of the TM4MD register to 1 and the TM4EN bit to 0. This enables TM4BC and the S-R flip-flop. This step ensures stable operation. If it is omitted, the binary counter may not count the first cycle. Do not change any other operating modes during this step.
- Set TM4NLD and TM4EN to 1. This starts the timer. Counting begins at the start of the next cycle.

■ To enable timer 4 capture B interrupts:

Cancel all existing interrupt requests. Next, set the interrupt priority level in the TM4CBLV[2:0] bits of the TM4CBICH register (levels 0 to 6), set the TM4BIE bit to 1, and set the TM4BIR bit of TM4CBICL to 0. From this point on, an interrupt request is generated whenever a timer 4 capture B event occurs.

■ **To service the interrupts and calculate the signal width:**

1. Run the interrupt service routine. The routine must determine the interrupt group, then clear the interrupt request flag.
2. Calculate the number of cycles the TM4IA signal stays high. Save the contents of TM4CA and TM4CB to the data registers, then subtract the contents of TM4CA from the contents of TM4CB. Since TM4LP is set to 0, the difference will be the correct value even if TM4CA is greater than TM4CB.

Ignore the flags when calculating the signal width, even when TM3CA is the larger value.

Timer 4 can input a two-phase capture signal. You must select up counting. Timer 4 does not operate in STOP mode, when B_{OSC} is off. If you use an external clock, it must be synchronized to B_{OSC}.

TM4CA captures the count on the rising edge of TM4IA, and TM4CB captures the count on the rising edge of TM4IB. A timer 4 capture B interrupt occurs when TM4CB captures the count, and the contents of TM4CA and TM4CB are read during the interrupt service routine.

In the example timing chart shown in figure 4-38, $x'000A' - x'0007' = x'0003'$, or 3 cycles. The calculation is correct even when TM4CA is the larger value. The flags are ignored, so for instance, $x'0003' - x'FFFE' = x'0005'$.

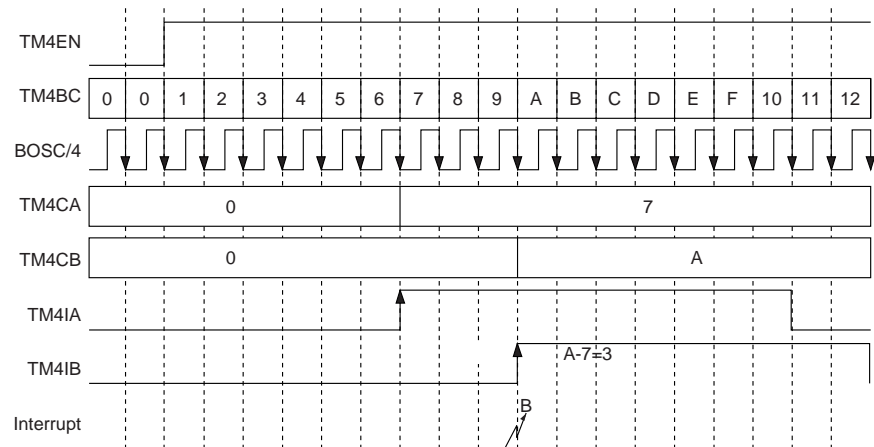


Figure 4-38 Two-Phase Capture Input Timing (Timer 4)

4.11.6 Setting Up a 4x Two-Phase Encoder Input Using Timer 5

In this example, timer 5 inputs a 4x two-phase encoded signal that makes it count up and down. An interrupt occurs when the counter reaches a preset value.

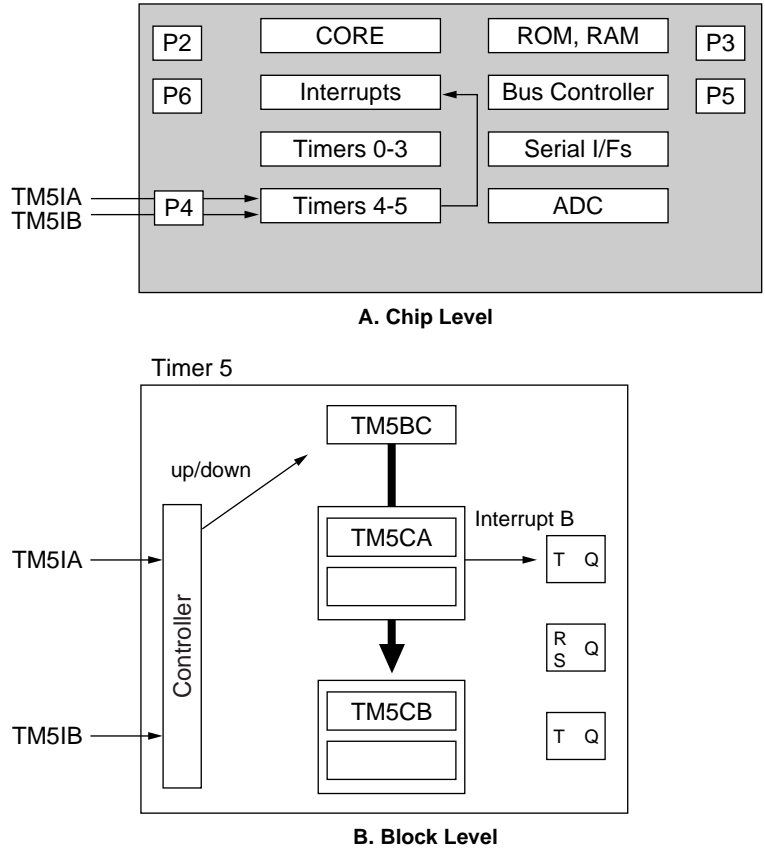


Figure 4-39 Block Diagram of 4x Two-Phase Capture Input Using Timer 5

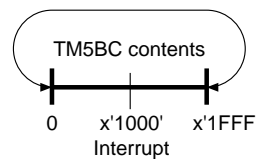


Figure 4-40 Configuration Example 1 of 4x Two-Phase Capture Input Using Timer 5

As figure 4-41 shows, you can set different values for A and B interrupts. (TM5LP must be 0.)

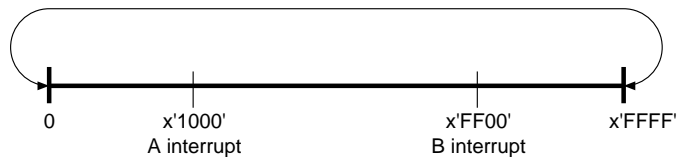


Figure 4-41 Configuration Example 2 of 4x Two-Phase Capture Input Using Timer 5



Use the MOV instruction for this setup and only use 16-bit write operations.

This step stops the TM5BC count and clears both TM5BC and the S-R flip-flop to 0.

■ **To set up timer 5:**

1. Set the operating mode in the timer 5 mode register (TM5MD). Disable timer 5 counting and interrupts. The up/down count bit is ignored in this instance. Set the TM5NLP bit to 1 to select looped counting from 0 to the value in TM5CA. Select the 4x two-phase encoder as the clock source.

TM5MD (example)

x'00FE90'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TM5 EN	TM5 NLD	—	—	TM5 UD1	TM5 UD0	TM5 TGE	TM5 ONE	TM5 MD1	TM5 MD0	TM5 ECLR	TM5 LP	TM5 ASEL	TM5 S2	TM5 S1	TM5 S0
Setting:	0	0	0	0	0	0	0	0	0	0	0	1	1 or 0	1	0	0

2. Write the intended looping value for timer 5 to TM5CA (valid settings: x'0001' to x'FFFF'). For TM5BC to count from x'0000' to x'1FFF', for instance, write x'1FFF' to TM5CA.

TM5CA (example)

x'00FE94'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TM5 CA15	TM5 CA14	TM5 CA13	TM5 CA12	TM5 CA11	TM5 CA10	TM5 CA9	TM5 CA8	TM5 CA7	TM5 CA6	TM5 CA5	TM5 CA4	TM5 CA3	TM5 CA2	TM5 CA1	TM5 CA0
Setting:	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1

3. Write the timer 5 interrupt value (valid settings: x'0000' to the value in TM5CA) to TM5CB. Whenever the binary counter reaches the value in TM5CB, in either up or down counting, a compare/capture B interrupt occurs at the beginning of the next cycle.

TM5CB (example)

x'00FE98'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TM5 CB15	TM5 CB14	TM5 CB13	TM5 CB12	TM5 CB11	TM5 CB10	TM5 CB9	TM5 CB8	TM5 CB7	TM5 CB6	TM5 CB5	TM5 CB4	TM5 CB3	TM5 CB2	TM5 CB1	TM5 CB0
Setting:	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0

4. Set the TM5NLD bit of the TM5MD register to 1 and the TM5EN bit to 0. This enables TM5BC and the S-R flip-flop. This step ensures stable operation. If it is omitted, the binary counter may not count the first cycle. Do not change any other operating modes during this step.
5. Set TM5NLD and TM5EN to 1. This starts the timer. Counting begins at the start of the next cycle.

■ **To enable timer 5 capture B interrupts:**

Cancel all existing interrupt requests. Next, set the interrupt priority level in the TM5CBLV[2:0] bits of the TM5CBICH register (levels 0 to 6), set the TM5BIE bit to 1, and set the TM5BIR bit of TM5CBICL to 0. From this point on, an interrupt request is generated whenever a timer 5 capture B event occurs.

Timers

16-Bit Timer Setup Examples

■ **To service the interrupts:**

Run the interrupt service routine. The routine must determine the interrupt group, then clear the interrupt request flag.

Timer 5 can input a two-phase encoder signal. Timer 5 does not operate in STOP mode, when B_{OSC} is off. If you use an external clock, it must be synchronized to B_{OSC} .

Table 4-4 shows the count direction for the timing diagram in figure 4-42. In down counting, when the binary counter reaches 0, it loops to the value in TM5CA. An interrupt B occurs when the contents of TM5BC match those of TM5CB.

Table 4-4 Count Direction for 4x Two-Phase Encoder Timing Example

	Up Counting				Down Counting			
	↑	1	↓	0	↑	0	↓	1
TM5IA	↑	1	↓	0	↑	0	↓	1
TM5IB	0	↑	1	↓	1	↑	0	↓

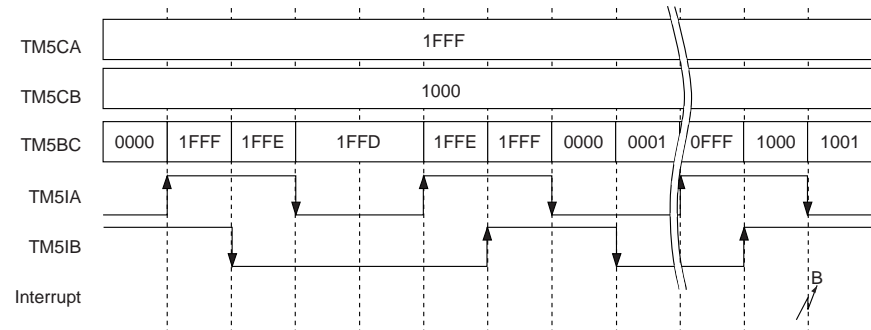
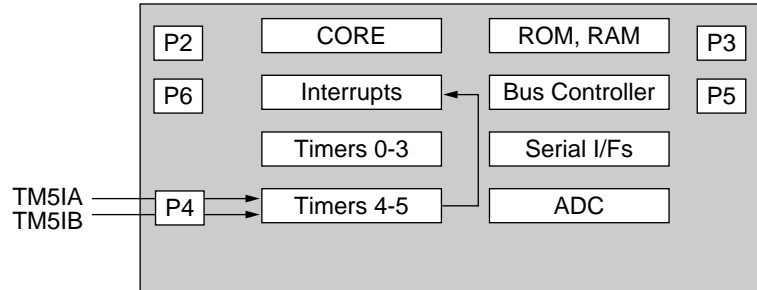


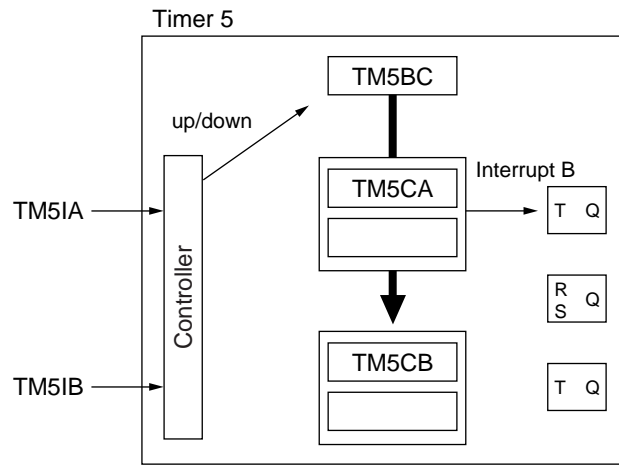
Figure 4-42 4x Two-Phase Encoder Input Timing (Timer 5)

4.11.7 Setting Up a 1x Two-Phase Encoder Input Using Timer 5

In this example, timer 5 inputs a 1x two-phase encoded signal that makes it count up and down. An interrupt occurs when the counter reaches a preset value.



A. Chip Level



B. Block Level

Figure 4-43 Block Diagram of 1x Two-Phase Capture Input Using Timer 5

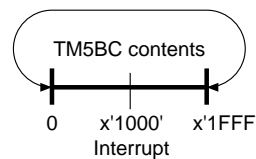


Figure 4-44 Configuration Example 1 of 1x Two-Phase Capture Input Using Timer 5

As figure 4-45 shows, you can set different values for A and B interrupts. (TM5LP must be 0.)

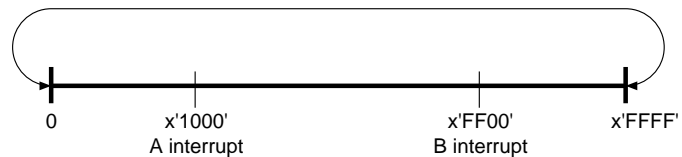


Figure 4-45 Configuration Example 2 of 1x Two-Phase Capture Input Using Timer 5

Timers

16-Bit Timer Setup Examples



Use the MOV instruction for this setup and only use 16-bit write operations.

This step stops the TM5BC count and clears both TM5BC and the S-R flip-flop to 0.

■ To set up timer 5:

1. Set the operating mode in the timer 5 mode register (TM5MD). Disable timer 5 counting and interrupts. The up/down count bit is ignored in this instance. Set the TM5NLP bit to 1 to select looped counting from 0 to the value in TM5CA. Select the 1x two-phase encoder as the clock source.

TM5MD (example)

x'00FE90'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TM5 EN	TM5 NLD	—	—	TM5 UD1	TM5 UD0	TM5 TGE	TM5 ONE	TM5 MD1	TM5 MD0	TM5 ECLR	TM5 LP	TM5 ASEL	TM5 S2	TM5 S1	TM5 S0
Setting:	0	0	0	0	0	0	0	0	0	0	0	1	1 or 0	1	0	1

2. Write the intended looping value for timer 5 to TM5CA (valid settings: x'0001' to x'FFFF'). For TM5BC to count from x'0000' to x'1FFF', for instance, write x'1FFF' to TM5CA.

TM5CA (example)

x'00FE94'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TM5 CA15	TM5 CA14	TM5 CA13	TM5 CA12	TM5 CA11	TM5 CA10	TM5 CA9	TM5 CA8	TM5 CA7	TM5 CA6	TM5 CA5	TM5 CA4	TM5 CA3	TM5 CA2	TM5 CA1	TM5 CA0
Setting:	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1

3. Write the timer 5 interrupt value (valid settings: x'0000' to the value in TM5CA) to TM5CB. Whenever the binary counter reaches the value in TM5CB, in either up or down counting, a compare/capture B interrupt occurs at the beginning of the next cycle.

TM5CB (example)

x'00FE98'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TM5 CB15	TM5 CB14	TM5 CB13	TM5 CB12	TM5 CB11	TM5 CB10	TM5 CB9	TM5 CB8	TM5 CB7	TM5 CB6	TM5 CB5	TM5 CB4	TM5 CB3	TM5 CB2	TM5 CB1	TM5 CB0
Setting:	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0

4. Set the TM5NLD bit of the TM5MD register to 1 and the TM5EN bit to 0. This enables TM5BC and the S-R flip-flop. This step ensures stable operation. If it is omitted, the binary counter may not count the first cycle. Do not change any other operating modes during this step.
5. Set TM5NLD and TM5EN to 1. This starts the timer. Counting begins at the start of the next cycle.

■ To enable timer 5 capture B interrupts:

Cancel all existing interrupt requests. Next, set the interrupt priority level in the TM5CBLV[2:0] bits of the TM5CBICH register (levels 0 to 6), set the TM5BIE bit to 1, and set the TM5BIR bit of TM5CBICL to 0. From this point on, an interrupt request is generated whenever a timer 5 capture B event occurs.

■ **To service the interrupts:**

Run the interrupt service routine. The routine must determine the interrupt group, then clear the interrupt request flag.

Timer 5 can input a two-phase encoder signal. Timer 5 does not operate in STOP mode, when B_{OSC} is off. If you use an external clock, it must be synchronized to B_{OSC}.

Table 4-5 shows the count direction for the timing diagram in figure 4-46. In down counting, when the binary counter reaches 0, it loops to the value in TM5CA. An interrupt B occurs when the contents of TM5BC match those of TM5CB.

Table 4-5 Count Direction for 1x Two-Phase Encoder Timing Example

	Up Counting	Down Counting
TM5IA	↓	↑
TM5IB	1	1

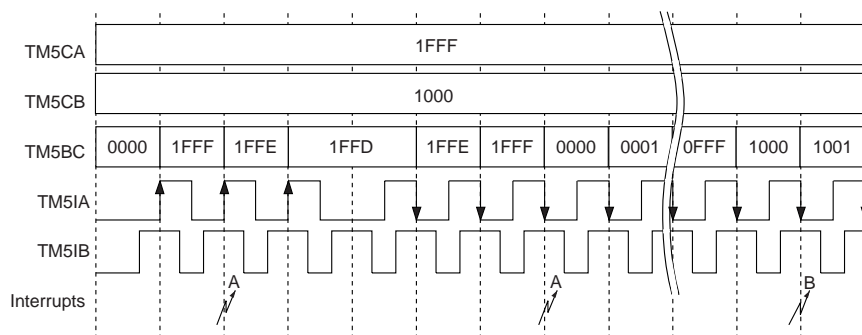
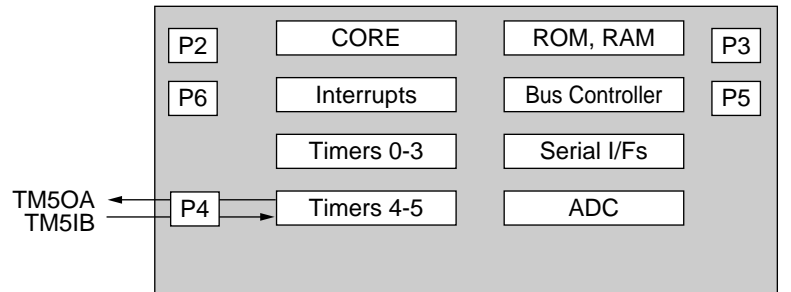


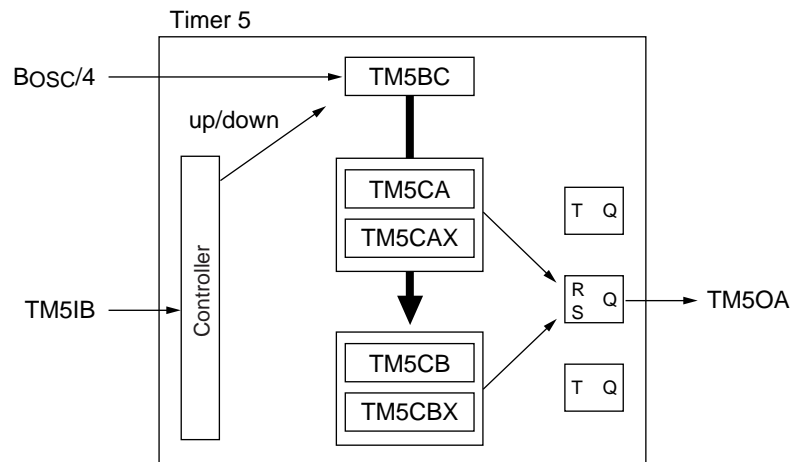
Figure 4-46 1x Two-Phase Encoder Input Timing (Timer 5)

4.11.8 Setting Up a One-Shot Pulse Output Using Timer 5

In this example, timer 5 outputs a one-shot pulse. The pulse width is two clock cycles.



A. Chip Level



B. Block Level

Figure 4-47 Block Diagram of One-Shot Pulse Output Using Timer 5

■ **To set up the output port:**

Set the P4MD2 bit of the port 4 output mode register (P4MD) to 1 (selecting the TM5IOA pin) and set the P4DIR2 bit of the port 4 I/O control register (P4DIR) to 1 (selecting output direction). This step selects the TM5OA pin (P42) as the timer output port.

P4MD (example)

x'00FFF8'

Bit:	7	6	5	4	3	2	1	0
	P4 MD7	P4 MD6	P4 MD5	P4 MD4	P4 MD3	P4 MD2	P4 MD1	P4 MD0
Setting:	0	0	0	0	0	1	0	0

P2DIR (example)

x'00FFE4'

Bit:	7	6	5	4	3	2	1	0
	P4 DIR7	P4 DIR6	P4 DIR5	P4 DIR4	P4 DIR3	P4 DIR2	P4 DIR1	P4 DIR0
Setting:	0	0	0	0	0	1	0	0



Use the MOV instruction for this setup and only use 16-bit write operations.

This step stops the TM5BC count and clears both TM5BC and the S-R flip-flop to 0.

■ **To set up timer 5:**

1. Set the operating mode in the timer 5 mode register (TM5MD). Disable timer 5 counting and interrupts. Select up counting. Select $B_{OSC}/4$ as the clock source.

TM5MD (example)

x'00FE90'

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TM5 EN	TM5 NLD	—	—	TM5 UD1	TM5 UD0	TM5 TGE	TM5 ONE	TM5 MD1	TM5 MD0	TM5 ECLR	TM5 LP	TM5 ASEL	TM5 S2	TM5 S1	TM5 S0
Setting:	0	0	0	0	0	0	1	1	0	0	0	1	1 or 0	0	1	1

2. Set the timer 5 pulse width in TM5CA (valid settings: x'0001' to x'FFFF'). Since the pulse width in this example is two cycles of the $B_{OSC}/4$ clock, write x'0003' to TM5CA. TM5BC counts from 0 to 3, and TM5OA outputs a high signal while the count is 1, 2, and 3. The timer operates essentially the same as it does during two-phase PWM output.

TM5CA (example)

x'00FE94'

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TM5 CA15	TM5 CA14	TM5 CA13	TM5 CA12	TM5 CA11	TM5 CA10	TM5 CA9	TM5 CA8	TM5 CA7	TM5 CA6	TM5 CA5	TM5 CA4	TM5 CA3	TM5 CA2	TM5 CA1	TM5 CA0
Setting:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

3. Write x'0001' to TM5CB.

TM5CB (example)

x'00FE98'

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TM5 CB15	TM5 CB14	TM5 CB13	TM5 CB12	TM5 CB11	TM5 CB10	TM5 CB9	TM5 CB8	TM5 CB7	TM5 CB6	TM5 CB5	TM5 CB4	TM5 CB3	TM5 CB2	TM5 CB1	TM5 CB0
Setting:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

4. Set the TM5NLD bit of the TM5MD register to 1 and the TM5EN bit to 0. This enables TM5BC and the S-R flip-flop. This step ensures stable operation. If it is omitted, the binary counter may not count the first cycle. Do not change any other operating modes during this step.
5. On the falling edge of the TM5IB signal, the hardware sets the TM5EN bit to 1. This means that counting begins at the start of the next cycle after the TM5IB signal falls. The TM5EN bit serves as the busy flag for the one-shot pulse.

Timers

16-Bit Timer Setup Examples

Timer 5 can output a one-shot pulse. Timer 5 does not operate in STOP mode, when B_{OSC} is off. If you use an external clock, it must be synchronized to B_{OSC} .

Figure 4-48 shows an example timing diagram for one-shot pulse output. On the falling edge of TM5IB, the TM5EN flag is set, and counting begins at the start of the next cycle. Before the count starts, TM5BC is 0, the initial TM5OA output value is 0, and the R5 (reset) and S5 (set) signals are not asserted. After the count starts, when it changes from 0 to 1, the S5 signal is asserted. This sets the TM5OA signal high, and it outputs the one-shot pulse. When the count reaches 3, TM5BC resets, changing from 3 to 0, and the R5 signal is asserted, causing the TM5OA signal to go low. Because the TM5ONE bit of the TM5MD register is 1, and the TM5EN bit is reset, the count stops. The circuit state is now the same as it was before the TM5IB signal went low. When the TM5IB signal falls again, the hardware once again sets the TM5EN bit, and the one-shot pulse sequence repeats.

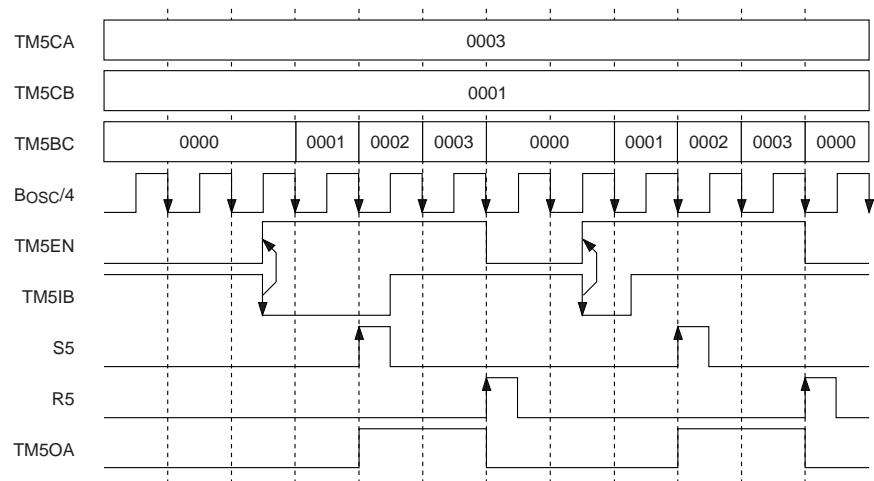
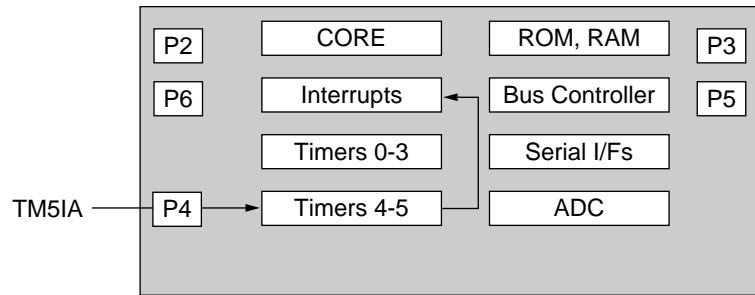


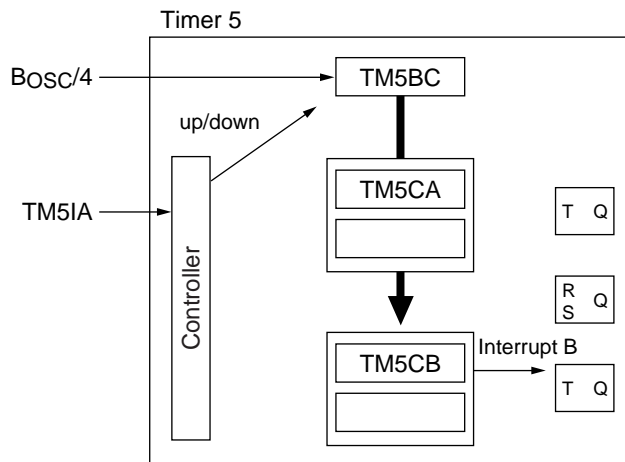
Figure 4-48 One-Shot Pulse Output Timing (Timer 5)

4.11.9 Setting Up an External Count Direction Controller Using Timer 5

In this example, timer 5 counts $B_{OSC}/4$ and the TM5IA pin controls the count direction (up or down). An interrupt occurs when the counter reaches a preset value.



A. Chip Level



B. Block Level

Figure 4-49 Block Diagram of External Count Direction Control Using Timer 5

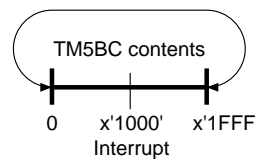


Figure 4-50 Configuration Example of External Count Direction Control Using Timer 5

Timers

16-Bit Timer Setup Examples



Use the MOV instruction for this setup and only use 16-bit write operations.

This step stops the TM5BC count and clears both TM5BC and the S-R flip-flop to 0.

■ To set up timer 5:

1. Set the operating mode in the timer 5 mode register (TM5MD). Disable timer 5 counting and interrupts. Set the TM5UD[1:0] bits to b'10', so that the count direction is up when the TM5IA signal is high and down when the TM5IA signal is low. Select B_{OSC}/4 as the clock source.

TM5MD (example)

x'00FE90'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TM5 EN	TM5 NLD	—	—	TM5 UD1	TM5 UD0	TM5 TGE	TM5 ONE	TM5 MD1	TM5 MD0	TM5 ECLR	TM5 LP	TM5 ASEL	TM5 S2	TM5 S1	TM5 S0
Setting:	0	0	0	0	1	0	0	0	0	0	0	1	1 or 0	0	1	1

2. Write the intended looping value for timer 5 to TM5CA (valid settings: x'0001' to x'FFFF'). For TM5BC to count from x'0000' to x'1FFF', for instance, write x'1FFF' to TM5CA.

TM5CA (example)

x'00FE94'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TM5 CA15	TM5 CA14	TM5 CA13	TM5 CA12	TM5 CA11	TM5 CA10	TM5 CA9	TM5 CA8	TM5 CA7	TM5 CA6	TM5 CA5	TM5 CA4	TM5 CA3	TM5 CA2	TM5 CA1	TM5 CA0
Setting:	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1

3. Write the timer 5 interrupt value (valid settings: x'0000' to the value in TM5CA) to TM5CB. Whenever the binary counter reaches the value in TM5CB, in either up or down counting, a compare/capture B interrupt occurs at the beginning of the next cycle.

TM5CB (example)

x'00FE98'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TM5 CB15	TM5 CB14	TM5 CB13	TM5 CB12	TM5 CB11	TM5 CB10	TM5 CB9	TM5 CB8	TM5 CB7	TM5 CB6	TM5 CB5	TM5 CB4	TM5 CB3	TM5 CB2	TM5 CB1	TM5 CB0
Setting:	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0

4. Set the TM5NLD bit of the TM5MD register to 1 and the TM5EN bit to 0. This enables TM5BC and the S-R flip-flop. This step ensures stable operation. If it is omitted, the binary counter may not count the first cycle. Do not change any other operating modes during this step.
5. Set TM5NLD and TM5EN to 1. This starts the timer. Counting begins at the start of the next cycle.

■ To enable timer 5 capture B interrupts:

Cancel all existing interrupt requests. Next, set the interrupt priority level in the TM5CBLV[2:0] bits of the TM5CBICH register (levels 0 to 6), set the TM5BIE bit to 1, and set the TM5BIR bit of TM5CBICL to 0. From this point on, an interrupt request is generated whenever a timer 5 capture B event occurs.

■ **To service the interrupts:**

Run the interrupt service routine. The routine must determine the interrupt group, then clear the interrupt request flag.

Either the TM5IA or TM5IB signal can control the timer 5 count direction. The count direction is determined at the opposite edge from the count edge (at the source clock transition occurring in the middle of the count cycle.).

Timer 5 does not operate in STOP mode, when B_{OSC} is off. If you use an external clock, it must be synchronized to B_{OSC}.

Figure 4-51 shows an example timing chart. In this example, an interrupt occurs when the timer switches from down to up counting.

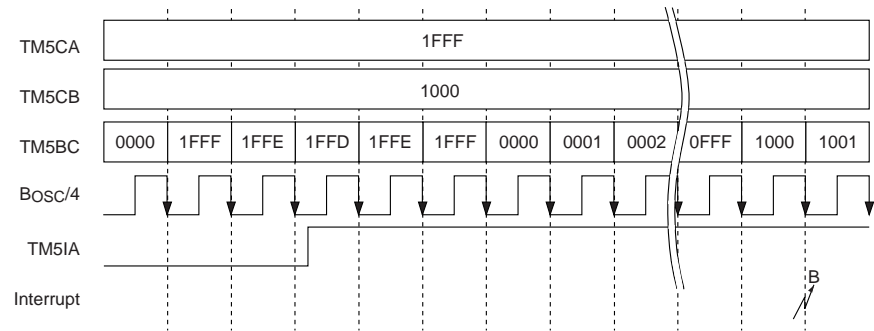


Figure 4-51 External Count Direction Control Timing (Timer 5)

4.11.10 Setting Up External Reset Control Using Timer 5

In this example, timer 5 is reset by an external signal while counting up.

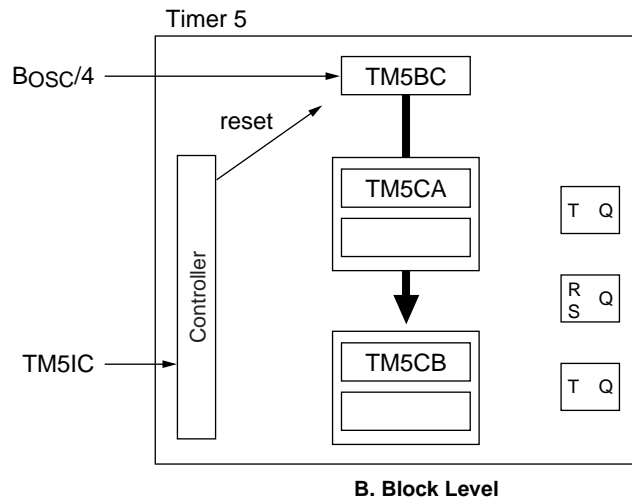
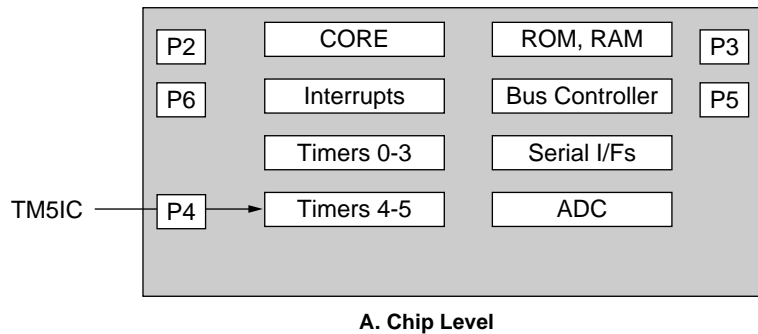


Figure 4-52 Block Diagram of External Reset Control Using Timer 5

■ To set up timer 5:

1. Set the operating mode in the timer 5 mode register (TM5MD). Disable timer 5 counting and interrupts. Select up counting. Since the TM5IC signal will reset the counter asynchronously, set the TM5ECLR bit to 1. Select B_OSC/4 as the clock source.



Use the MOV instruction to set this data and only use 16-bit write operations.

This step stops the TM5BC count and clears both TM5BC and the S-R flip-flop to 0.

TM5MD (example)

x'00FE90'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TM5 EN	TM5 NLD	—	—	TM5 UD1	TM5 UD0	TM5 TGE	TM5 ONE	TM5 MD1	TM5 MD0	TM5 ECLR	TM5 LP	TM5 ASEL	TM5 S2	TM5 S1	TM5 S0
Setting:	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1	1

2. Set the value to which timer 5 will loop (valid settings: x'0001' to x'FFFF'). For TM5BC to count from x'0000' to x'1FFF', for instance, write x'1FFF' to TM5CA.

TM5CA (example)														x'00FE94'		
Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TM5 CA15	TM5 CA14	TM5 CA13	TM5 CA12	TM5 CA11	TM5 CA10	TM5 CA9	TM5 CA8	TM5 CA7	TM5 CA6	TM5 CA5	TM5 CA4	TM5 CA3	TM5 CA2	TM5 CA1	TM5 CA0
Setting:	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1

3. Set the TM5NLD bit of the TM5MD register to 1 and the TM5EN bit to 0. This enables TM5BC and the S-R flip-flop. This step ensures stable operation. If it is omitted, the binary counter may not count the first cycle.
4. Set TM5NLD and TM5EN to 1. This starts the timer. Counting begins at the start of the next cycle.

From this point on, whenever the TM5IC signal is high, timer 5 will be reset asynchronously. This is an easy way to synchronize the microcontroller operation with an external event. You can use it to adjust motor speed or to initialize the timers through the hardware.

Timer 5 does not operate in STOP mode, when B_{OSC} is off. If you use an external clock, it must be synchronized to B_{OSC}.

Figure 4-53 shows an example timing chart.

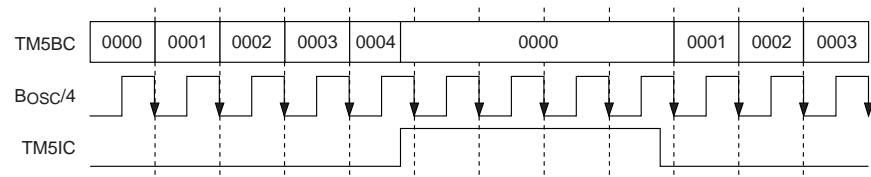


Figure 4-53 External Reset Control Timing (Timer 5)

4.12 16-Bit Timer Control Registers

Table 4-6 shows the registers used to control the 16-bit timers. A binary counter (TMnBC), a compare/capture register A (TMnCA), a compare/capture register B (TMnCB), and a timer mode register (TMnMD) is associated with each 16-bit timer.

Table 4-6 16-Bit Timer Control Registers

Register		Address	R/W	Description
Timer 4	TM4MD	x'00FE80'	R/W	Timer 4 mode register
	TM4BC	x'00FE82'	R	Timer 4 binary counter
	TM4CA	x'00FE84'	R/W	Timer 4 compare/capture register A
	TM4CAX	x'00FE86'	—	Timer 4 compare/capture register set AX
	TM4CB	x'00FE88'	R/W	Timer 4 compare/capture register B
	TM4CBX	x'00FE8A'	—	Timer 4 compare/capture register set BX
Timer 5	TM5MD	x'00FE90'	R/W	Timer 5 mode register
	TM5BC	x'00FE92'	R	Timer 5 binary counter
	TM5CA	x'00FE94'	R/W	Timer 5 compare/capture register A
	TM5CAX	x'00FE96'	—	Timer 5 compare/capture register set AX
	TM5CB	x'00FE98'	R/W	Timer 5 compare/capture register B
	TM5CBX	x'00FE9A'	—	Timer 5 compare/capture register set BX

Note: TM4CAX, TM4CBX, TM5CAX, and TM5CBX are virtual registers used only in double-buffer mode during PWM output. They do not exist in the hardware and are not readable or writable. Depending on the write signal, they load the value in the associated CA or CB register.

TM4BC/TM5BC: Timer n Binary Counter **x'00FE82'/x'00FE92'**

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TMn BC15	TMn BC14	TMn BC13	TMn BC12	TMn BC11	TMn BC10	TMn BC9	TMn BC8	TMn BC7	TMn BC6	TMn BC5	TMn BC4	TMn BC3	TMn BC2	TMn BC1	TMn BC0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

TM4CA/TM5CA: Timer n Compare/Capture Register A **x'00FE84'/x'00FE94'**

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TMn CA15	TMn CA14	TMn CA13	TMn CA12	TMn CA11	TMn CA10	TMn CA9	TMn CA8	TMn CA7	TMn CA6	TMn CA5	TMn CA4	TMn CA3	TMn CA2	TMn CA1	TMn CA0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

TM4CB/TM5CB: Timer n Compare/Capture Register B **x'00FE88'/x'00FE98'**

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TMn CB15	TMn CB14	TMn CB13	TMn CB12	TMn CB11	TMn CB10	TMn CB9	TMn CB8	TMn CB7	TMn CB6	TMn CB5	TMn CB4	TMn CB3	TMn CB2	TMn CB1	TMn CB0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W



TMnCA and TMnCB are 16-bit access registers. Use the MOV instruction to write to them.

TM4MD/TM5MD: Timer n Mode Register**x'00FE80'/x'00FE90'**

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	TMn EN	TMn NLD	—	—	TMn UD1	TMn UD0	TMn TGE	TMn ONE	TMn MD1	TMn MD0	TMn ECLR	TMn LP	TMn ASEL	TMn S2	TMn S1	TMn S0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

TMnEN: TMnBC count

- 0: Disable
- 1: Enable

TMnNLD: TMnBC, T flip-flop, and S-R flip-flop operation select

- 0: Set all to 0 (initialize)
- 1: Operate all

TMnUD[1:0]: Timer n up/down counter mode select

Ignored when two-phase encoding is selected.

- 00: Up counter
- 01: Down counter
- 10: Up when TMnIOA is high; down when low
- 11: Up when TMnIOB is high; down when low

TMnTGE: External trigger enable for start count

- 0: Disable
- 1: Start count at falling edge of TMnIOB

TMnONE: Counter operating mode select

- 0: Repeat (except with PWM output)
- 1: One-shot pulse (counter stops on the next clock after TMnBC = TMnCA)

TMnMD[1:0]: TMnCA and TMnCB operating mode select

- 00: Compare register (single buffer)
- 01: Compare register (double buffer)
- 10: Capture register (TMnIOA high: capture A; TMnIOA low: capture B)
- 11: Capture register (TMnIOA high: capture A; TMnIOB high: capture B)

TMnECLR: Timer n BC external clear

- 0: Don't clear
- 1: Clear TMnBC asynchronously when the TMnIC signal goes high.

TMnLP: Timer n BC loop select

- 0: 0000–FFFF
- 1: 0000–value in TMnCA

TMnASEL: TMnIOA output select

- 0: S-R flip-flop output (single-phase PWM)
- 1: T flip-flop output (two-phase PWM)

TMnS[2:0]: Timer n clock source select

- 000: Timer 0 underflow
- 001: Timer 1 underflow
- 010: TMnIB signal
- 011: B_{OSC}/4
- 100: 4x two-phase encoder
- 101: 1x two-phase encoder
- 110, 111: Reserved

5 Serial Interfaces

5.1 Description

The MN102H75K/85K contains two general-purpose serial interfaces with synchronous serial, UART, and I²C modes. The maximum baud rate in synchronous serial mode is 12 Mbps. In UART mode, the maximum baud rate is 375,000 bps, when B_{OSC} = 24 MHz.

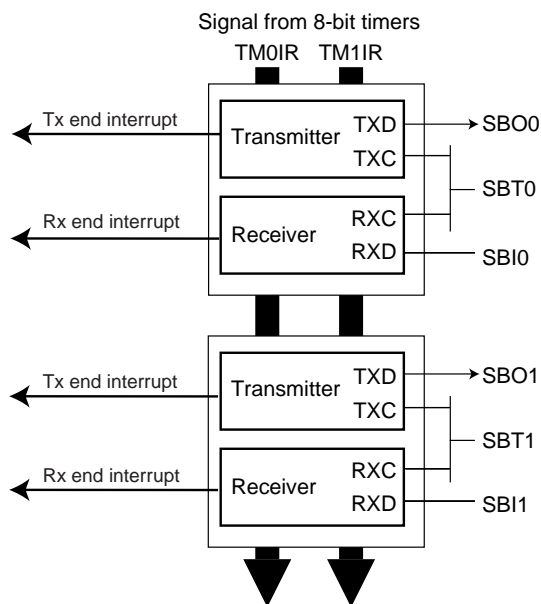


Figure 5-1 Serial Interface Configuration Example

5.2 Features

Table 5-1 Serial Interface Functions and Features

Function/Feature	Synchronous Serial Mode	UART Mode	I ² C Mode
Parity	None, 0, 1, even, or odd		Can be master transmitter or receiver. (No collision detection for start sequence.)
Character length	7-bit or 8-bit		
Transfer bit order	LSB or MSB first (programmable; 8-bit only)	LSB first	
Clock source	<ul style="list-style-type: none"> • 1/2 of timer 0 underflow • 1/8 of timer 0 or 1 underflow • External clock 	<ul style="list-style-type: none"> • 1/8 of timer 0 or 1 underflow 	
Maximum baud rate	12 Mbps (when B _{OSC} = 24 MHz)	375,000 bps (when B _{OSC} = 24 MHz)	
Error detection	<ul style="list-style-type: none"> • Parity error • Overrun error 	<ul style="list-style-type: none"> • Parity error • Overrun error • Framing error 	
Buffers	Independent transmit/receive buffers (single transmit buffer, double receive buffers)		
Interrupts	Transmission or reception complete interrupts		

5.3 Connecting the Serial Interfaces

Figures 5-2, 5-3, and 5-4 illustrate six different methods of connecting the serial interface.

See section 11, "I/O Ports," for details on setting up the SBT port.

5.3.1 Synchronous Serial Mode Connections

Figure 5-2 shows serial port connections for either simplex or full-duplex synchronous serial transfers.

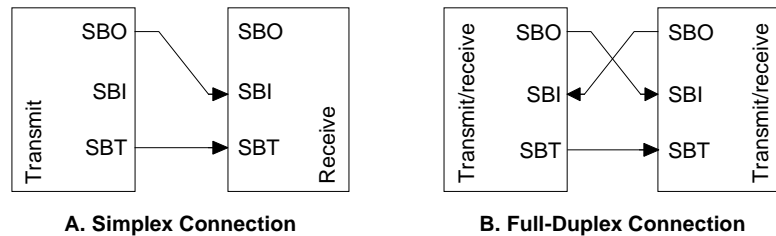


Figure 5-2 Synchronous Serial Mode Connections

5.3.2 UART Mode Connections

Figure 5-3 shows serial port connections for either simplex or full-duplex UART transfers.

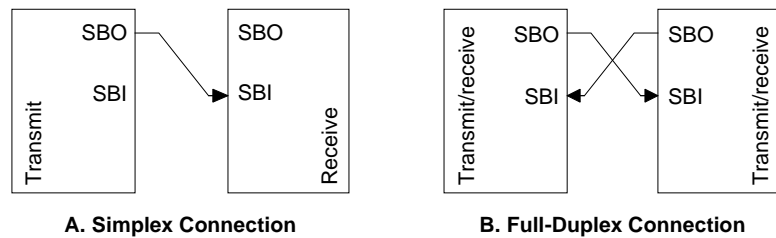


Figure 5-3 UART Mode Connections

5.3.3 I²C Mode Connection

The serial interfaces can connect to I²C slave transmitters or receivers. For this mode, always pull up the SBO and SBT pins to V_{DD}. Either connect a pullup resistor externally or turn on the internal one by setting the PPLU register.

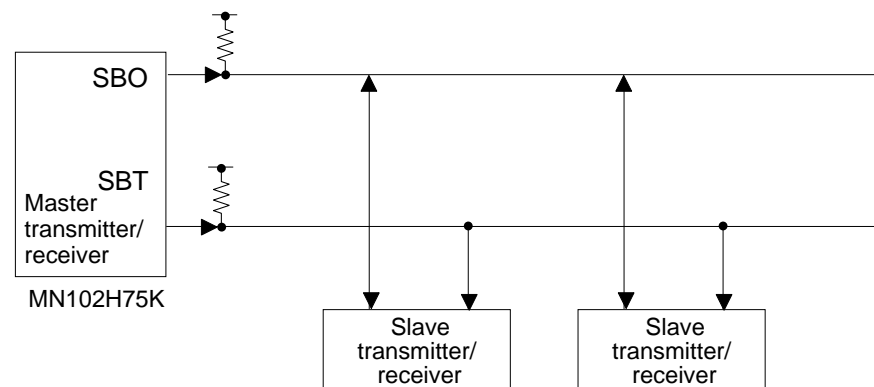


Figure 5-4 I²C Mode Connection

5.4 UART Mode Baud Rates

In UART mode, the serial interface transfer clock is set to 16 times the baud rate clock. The expression below is the formula for calculating the baud rate for the UART mode. Table 5-2 shows the baud rate settings when $B_{OSC} = 24 \text{ MHz}$.

$$\text{baud rate (bps)} = \frac{B_{OSC}}{32 \times (\text{timer divisor})}$$

Table 5-2 Example Baud Rate Settings for the UART Mode

Baud Rate	Timer 0/1 Divide-by Ratio
19200	39
9600	78
4800	156
2400	313
1200	625
600	1250
300	2500

5.5 Serial Interface Timing

5.5.1 Synchronous Serial Mode Timing

In these timing charts, the character length is 8 bits and there is parity.

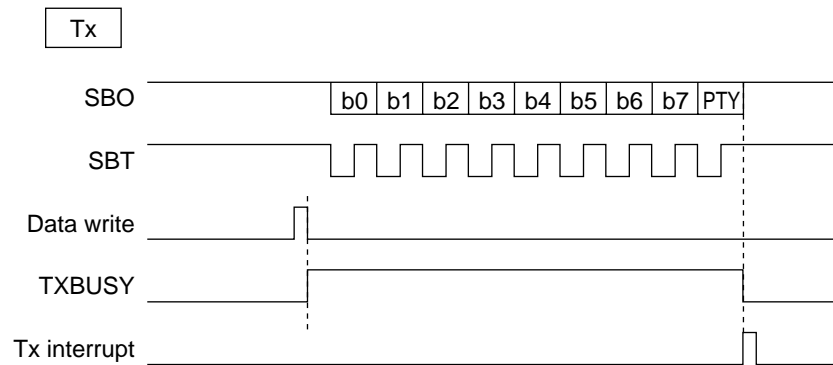


Figure 5-5 Synchronous Serial Transmission Timing

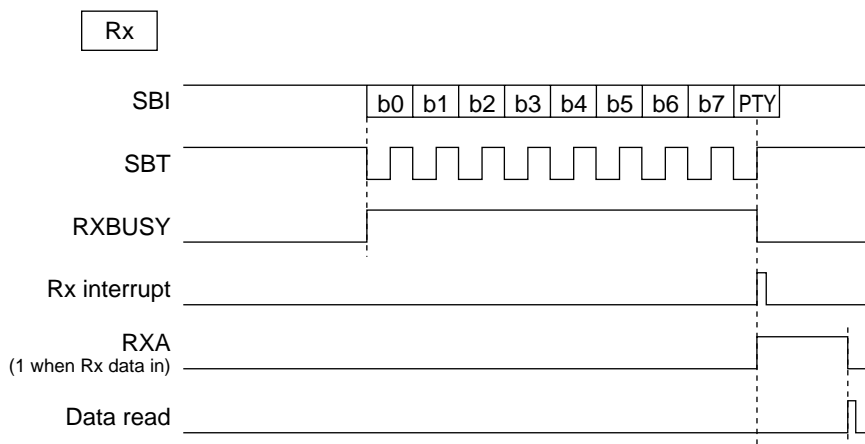


Figure 5-6 Synchronous Serial Reception Timing

5.5.2 UART Mode Timing

In these timing charts, the character length is 8 bits, the parity is none, and the stop bit is 2-bit.

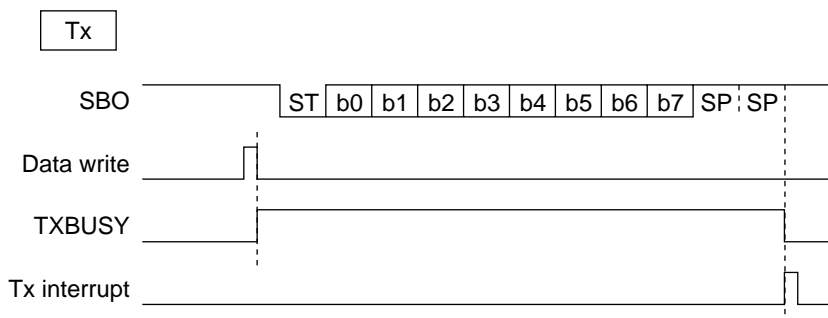


Figure 5-7 UART Transmission Timing

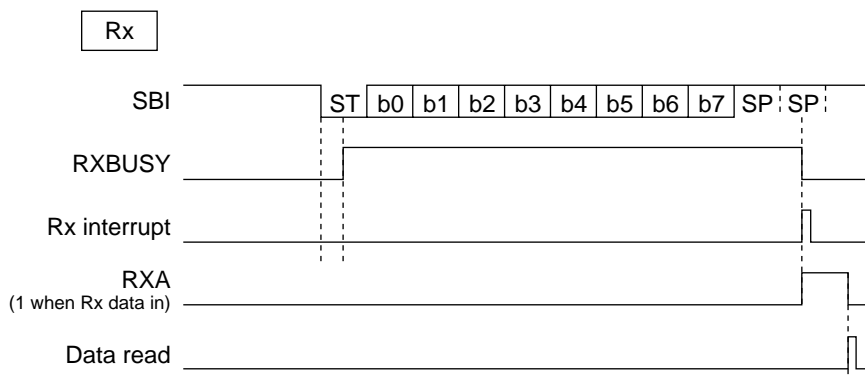


Figure 5-8 UART Reception Timing

5.6 Serial Interface Setup Examples

5.6.1 Setting Up UART Transmission Using Serial Interface 0

This example illustrates serial transmission in the UART mode with the following settings:



You must use an 8-bit timer to set the transfer clock. See section 5.6.3, "Setting Up the Serial Interface Clock," on page 135, for an example setup.

- ◆ B_{OSC} = 24 MHz
- ◆ Baud rate = 9600 bps (transfer clock set up with timer 0)
- ◆ 8-bit character length
- ◆ Two stop bits
- ◆ Odd parity

When a transmission end interrupt occurs, the next data byte is loaded.

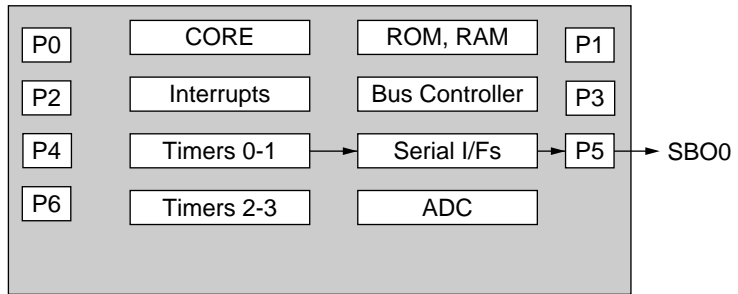


Figure 5-9 Block Diagram of UART Transmission Using Serial Interface 0

Data transmission starts when the CPU writes data to the SC0TRB register. The transmission start is synchronized to timer 0 underflow. An interrupt occurs when transmission ends, and the next data byte is written to SC0TRB. If the application does not use interrupts, it must poll the SC0TBY flag of the SC0STR register. It can write the transmission data when SC0TBY is 0.

■ **To set up the output port:**

Set the P5MD5 bit of the port 5 output mode register (P5MD) to 1. This selects the SBO0 pin as the serial interface output port.

P5MD (example)

x'00FFFA'

Bit:	7	6	5	4	3	2	1	0
	P5 MD7	P5 MD6	P5 MD5	P5 MD4	P5 MD3	P5 MD2	P5 MD1	P5 MD0
Setting:	0	1	0	0	0	0	0	0

■ **To set up serial interface 0:**

1. Configure the transmission settings in the serial port 0 control register (SCOCTR). Since the transfer clock is timer 0 divided by 8, select timer 0 underflow x 1/8 as the serial port 0 clock source. Select UART mode, odd parity, 2-bit stop bit, 8-bit data length, and LSB-first output. Also set the SCOREN and SCOTEN flags to 0, disabling transmission and reception.

SCOCTR (example)

x'00FD80'

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Bit:	SCO TEN	SCO REN	SCO BRE	SCO I2CS	SCO PTL	—	SCO OD	SCO I2CM	SCO LN	SCO PTY2	SCO PTY1	SCO PTY0	SCO SB	—	SCO S1	SCO S0
Setting:	0	0	0	0	0	0	0	0	1	1	1	1	1	0	0	1

2. Set the SCOTEN bit to 1 to enable transmission.

■ **To enable serial 0 transmission end interrupts:**

Cancel all existing interrupt requests. Next, set the interrupt priority level of 5 in the ANLV[2:0] bits of the ANICH register, set the SCT0IE bit of SCT0ICH to 1, and set the SCT0IR bit of SCT0ICL to 0. From this point on, an interrupt request is generated whenever a serial data transmission ends.

ANICH: (example)

x'00FC81'

	7	6	5	4	3	2	1	0
Bit:	—	ANLV2	ANLV1	ANLV0	—	—	—	ANIE
Setting:	0	1	0	1	0	0	0	0

SCT0ICL (example)

x'00FC82'

	7	6	5	4	3	2	1	0
Bit:	—	—	—	SCT0 IR	—	—	—	SCT0 ID
Setting:	0	0	0	0	0	0	0	0

SCT0ICH (example)

x'00FC83'

	7	6	5	4	3	2	1	0
Bit:	—	—	—	—	—	—	—	SCT0 IE
Setting:	0	0	0	0	0	0	0	1

Serial Interfaces

Serial Interface Setup Examples

■ Transmission sequence:

1. Write the first data byte to SC0TRB. Once this data is in the register, transmission begins, synchronized to timer 0.
2. When an interrupt occurs, the program branches to the interrupt service routine. The routine must determine the interrupt group, then clear the interrupt request flag.
3. Write the next data byte to SC0TRB. Once the write is complete, transmission begins in 1 or 2 cycles of the transfer clock (timer 0 underflow).

Figure 5-10 shows an example timing chart.

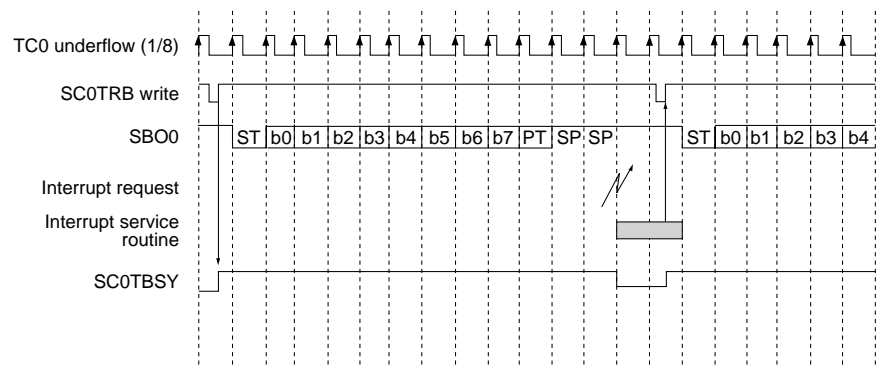


Figure 5-10 UART Transmission Timing (Serial Interface 0)

5.6.2 Setting Up Synchronous Serial Reception Using Serial Interface 0

This example illustrates serial reception in the synchronous serial mode with the following settings:

- ◆ LSB first
- ◆ 8-bit character length
- ◆ Odd parity

When a reception end interrupt occurs, the CPU reads the data byte.

■ **To set up the input port:**

Set the P5DIR7 bit of the port 5 I/O control register (P5DIR) to 0. This sets the SBT0 pin to input.

■ **To set up serial interface 0:**

Configure the reception settings in the serial port 0 control register (SC0CTR). Select timer 0 underflow x 1/8 as the serial port 0 clock source. Select timer 0 underflow x 1/8 as the serial port 0 clock source. Select synchronous serial mode, odd parity, 8-bit data length, and LSB-first output.

SC0CTR (example)

x'00FD80'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SC0 TEN	SC0 REN	SC0 BRE	SC0 I2CS	SC0 PTL	—	SC0 OD	SC0 I2CM	SC0 LN	SC0 PTY2	SC0 PTY1	SC0 PTY0	SC0 SB	—	SC0 S1	SC0 S0
Setting:	1	1	0	0	1	0	0	0	1	1	1	1	0	0	0	1

■ **To enable serial 0 transmission end interrupts:**

Cancel all existing interrupt requests. Next, set the interrupt priority level of 5 in the ANLV[2:0] bits of the ANICH register, set the SCR0IE bit of SCR0ICH to 1, and set the SCR0IR bit of SCR0ICL to 0. From this point on, an interrupt request is generated whenever a serial data reception ends.

ANICH: (example)

x'00FC81'

Bit:	7	6	5	4	3	2	1	0
	—	ANLV2	ANLV1	ANLV0	—	—	—	ANIE
Setting:	0	1	0	1	0	0	0	0

SCR0ICL (example)

x'00FC84'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	SCR0 IR	—	—	—	SCR0 ID
Setting:	0	0	0	0	0	0	0	0

SCR0ICH (example)

x'00FC85'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	SCR0 IE
Setting:	0	0	0	0	0	0	0	1

5.6.3 Setting Up the Serial Interface Clock

This example demonstrates how to set up a 19,200 bps transfer clock for the UART interface by using timer 1 to divide $B_{OSC}/4$ by 39. The example uses the following settings:

- ◆ $B_{OSC} = 24 \text{ MHz}$
- ◆ Clock source = timer 1 underflow x 1/8
- ◆ Transfer clock = baud rate x 8

The serial interface determines the baud rate from the 8-bit underflow. Set up the transfer clock by making the timer 1 underflow either two or eight times the desired baud rate. The serial interface divides the timer underflow by two or eight. (Always select divide-by-eight for UART transactions.) For a baud rate of 19,200, since $B_{OSC}/4 = 6 \text{ MHz}$,

$$6 \text{ MHz}/39/8 = 19230.77 \text{ bps}$$

This means that the timer 1 underflow must be divided by 39.

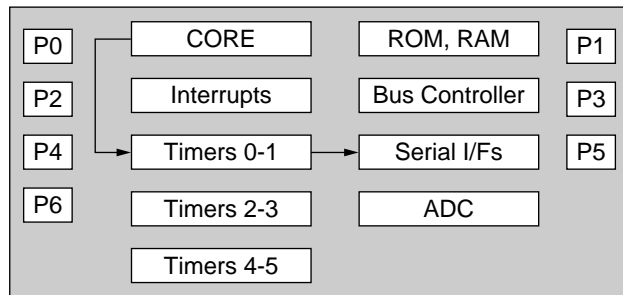


Figure 5-11 Block Diagram of Serial Interface Clock

■ To set timer 1:

1. Disable timer 1 counting in the timer 1 mode register (TM1MD). This step is unnecessary immediately after a reset, since TM1MD resets to 0.

TM1MD (example)

x'00FE21'

Bit:	7	6	5	4	3	2	1	0
	TM1 EN	TM1 LD	—	—	—	—	TM1 S1	TM1 S0
Setting:	0	0	0	0	0	0	0	0

2. Set the divide-by ratio for timer 1. To divide $B_{OSC}/4$ by 39, write x'26' to the timer 1 base register (TM1BR). (The valid range for TM1BR is 0 to 255.)

TM1BR (example)

x'00FE11'

Bit:	7	6	5	4	3	2	1	0
	TM1 BR7	TM1 BR6	TM1 BR5	TM1 BR4	TM1 BR3	TM1 BR2	TM1 BR1	TM1 BR0
Setting:	0	0	1	0	0	1	1	0



Do not change the clock source once you select it. Selecting the clock source while you set up the count operation control will corrupt the value in the binary counter.

In the bank and linear addressing versions of the MN102 series, it was necessary to set TM1EN and TM1LD to 0 between steps 3 and 4, to ensure stable operation. This is unnecessary in the high-speed linear addressing version.

- Set the TM1LD bit of the TM1MD register to 1. This loads the value in the base register to the binary counter. At the same time, select the clock source as $B_{OSC}/4$ by writing b'00' to TM1S[1:0].

TM1MD (example)

x'00FE21'

Bit:	7	6	5	4	3	2	1	0
	TM1 EN	TM1 LD	—	—	—	—	TM1 S1	TM1 S0
Setting:	0	1	0	0	0	0	0	0

- Set TM1LD to 0 and TM1EN to 1. This starts the timer. Counting begins at the start of the next cycle. When the binary counter reaches 0 and loads the value x'26' from the base register, in preparation for the next count, a timer 1 underflow occurs. The serial interface operates in sync with this underflow output.

Figure 5-12 shows an example timing chart.

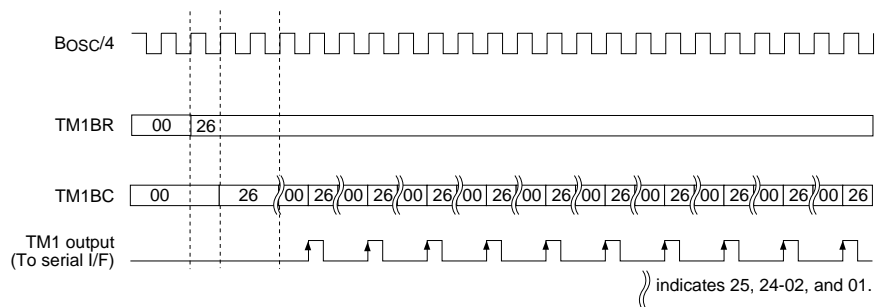


Figure 5-12 Serial Interface Clock Timing

The interrupt process is repeated each time the buffer receives another byte of serial data.

Serial Interfaces

Serial Interface Setup Examples

5.6.4 Setting Up I²C Transmission Using Serial Interface 0

This example illustrates the microcontroller as a master transmitter in the I²C mode, using the SBO0 and SBT0 pins.

I²C mode requires open-drain pins. To set this up, set the ODASCI0 bit of PCNT0 (x'00FF90') to 1. In addition, set the P5PUP7 and P5PUP5 bits of P5PUP (x'00FFB5') to enable pullup control of the SBO0 and STB0.

The parity bits serve as the ACK signal. To output an ACK = 1 signal, select a fixed parity of 1. To output an ACK = 0 signal, select a fixed parity of 0. Select a parity of none if there is no ACK signal.

Reception must be enabled for the circuit to detect a start sequence.

■ To set up the output ports:

Set the P5MD7 and P5MD5 bits of the port 5 output mode register (P5MD) to 1. This selects the SBO0 and SBT0 pins as the output port for the I²C interface.

P5MD (example)

x'00FFFA'

Bit:	7	6	5	4	3	2	1	0
	P5 MD7	P5 MD6	P5 MD5	P5 MD4	P5 MD3	P5 MD2	P5 MD1	P5 MD0
Setting:	1	0	1	0	0	0	0	0

■ To set up the I²C interface:

Set the operating conditions in the serial control register (SC0CTR). Select ACK = 1 output for the transfer clock (SC0PTY[2:0] = b'101'), 8-bit character length (SCLN = 1), I²C protocol (SC0PTL = 1), I²C mode (SC0ICM = 1), and MSB-first bit order (SC0OD = 1). Enable both transmission and reception (SC0TEN and SC0REN = 1) and disable transmission breaks (SC0BRE = 0). Select the timer 0 underflow rate divided by 8 as the clock source.

SC0CTR (example)

x'00FD80'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SC0 TEN	SC0 REN	SC0 BRE	SC0 I2CS	SC0 PTL	—	SC0 OD	SC0 I2CM	SC0 LN	SC0 PTY2	SC0 PTY1	SC0 PTY0	SC0 SB	—	SC0 S1	SC0 S0
Setting:	1	1	0	0	1	0	1	1	1	1	0	1	0	0	0	1

■ To set up the start sequence:

Write a 1 to the SC0IIC bit of the SC0CTR register to signal the start sequence. The SBO0 pin output immediately goes low. Read SC0STR to verify that the start sequence occurred correctly (SC0IST = 1). At this point, even if another start exists on the bus, an arbitration lost will not be detected.

■ To transmit the first data byte:

1. Load the data to the serial port 0 transmit/receive buffer, which initiates data output. The SBO0 pin begins data output to the I²C bus when the SBT0 clock signal goes low, with a 1/8 clock cycle delay.
2. After transmission, both the SBO0 and SBT0 signals stay low.

■ To transmit a second data byte:

Load the data to the serial port 0 transmit/receive buffer. The sequence for the first data byte repeats.

■ To set up the stop sequence:

1. When all the data has been transmitted, set the SC0IIC bit of SC0CTR to 0. Never perform this step during transmission.

Reception must be enabled for the circuit to detect a stop sequence.

2. When you perform step 1, the SBT0 output signal goes high. One cycle later, the SBO0 output signal also goes high, signalling the stop sequence. The SCOISP flag of SC0STR becomes 1. The SCOIST and SC0ISP flags are both cleared by a write to or read from the serial port 0 transmit/receive buffer.

Figure 5-13 shows an example timing chart.

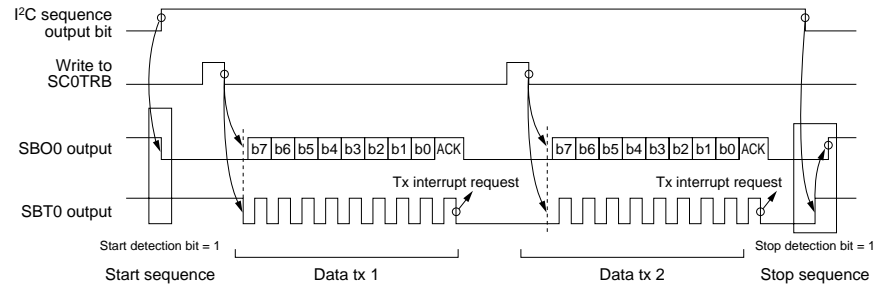


Figure 5-13 Master Transmitter Timing in I²C Mode (with ACK)

5.6.5 Setting Up I²C Reception Using Serial Interface 0

This example illustrates the microcontroller as a master receiver in the I²C mode, using the SBO0 and SBT0 pins.

When initiating master receiver mode, your program must always first transmit a byte of data. The master reception occurs during the interrupt service routine that runs after the data is transmitted. For an example setup of master transmission, see section 5.6.4, “Setting Up I²C Transmission Using Serial Interface 0,” on page 137.

You do not need to enable reception if it is already enabled by the initial settings. You can also omit any settings already in place from the transmission sequence.

The parity bits serve as the ACK signal. To output an ACK = 1 signal, select a fixed parity of 1. To output an ACK = 0 signal, select a fixed parity of 0. Select a parity of none if there is no ACK signal

■ **To set up the I²C interface:**

1. During the interrupt service routine for the serial transmission end, enable reception by setting the SCOREN bit of SC0CTR to 1.
2. Select ACK output of 1.

■ **To set up data reception:**

1. Write a dummy data bit, x'FF', to the serial port 0 transmit/receive buffer. This sets the SBO0 signal high and initiates the master receiver mode.
2. During the service routine for the serial reception interrupt, the CPU reads the transmit/receive buffer to retrieve the reception data. (A transmission interrupt can serve as a reception interrupt.)

■ **To set up the stop sequence:**

1. Set the SC0IIC bit of SC0CTR to 0 to signal the stop sequence.
2. When you signal the stop sequence, the data reception is still in progress. After the stop sequence is output, you must disable reception and reinitialize reception for succeeding bytes.

Figure 5-14 shows an example timing chart.

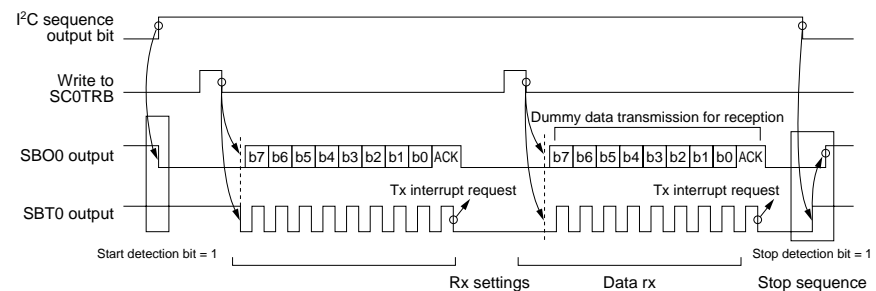


Figure 5-14 Master Receiver Timing in I²C Mode (with ACK)

5.7 Serial Interface Control Registers

Three registers control each of the serial interfaces: the serial port control register (SCnCTR), the serial transmit/receive buffer (SCnTRB), and the serial port status register (SCnSTR).

Table 5-3 Serial Interface Control Registers

Register		Address	R/W	Description
Serial I/F 0	SC0CTR	x'00FD80'	R/W	Serial port 0 control register
	SC0TRB	x'00FD82'	R/W	Serial port 0 transmit/receive buffer
	SC0STR	x'00FD83'	R	Serial port 0 status register
Serial I/F 1	SC1CTR	x'00FD88'	R/W	Serial port 1 control register
	SC1TRB	x'00FD8A'	R/W	Serial port 1 transmit/receive buffer
	SC1STR	x'00FD8B'	R	Serial port 1 status register

SC0CTR/SC1CTR: Serial Port n Control Register **x'00FD80'/x'00FD88'**

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SCn TEN	SCn REN	SCn BRE	SCn IIC	SCn PTL	—	SCn OD	SCn ICM	SCn LN	SCn PTY2	SCn PTY1	SCn PTY0	SCn SB	—	SCn S1	SCn S0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R	R/W	R/W

SC0CTR controls the operating conditions for the serial interface, including the clock source, the parity bit, the protocol, and transmit/receive enabling.

SCnTEN: Serial port n transmit enable
0: Disable
1: Enable

SCnREN: Serial port n receive enable
0: Disable
1: Enable

SCnBRE: Serial port n break transmission
0: Don't break
1: Break (Force SBO_n to 0)

SCnIIC: Serial port n I²C start/stop sequence output
Do not change this bit during transmission or reception.
0: Output stop sequence upon 1-to-0 transition
1: Output start sequence upon 0-to-1 transition

SCnPTL: Serial port n protocol select
0: UART
1: Synchronous serial or I²C

SCnOD: Serial port n bit order
This bit must be set to 0 during 7-bit transmission.
0: LSB first
1: MSB first

Serial Interfaces

Serial Interface Control Registers

SCnICM: Serial port n I²C mode select

- 0: I²C mode off
- 1: I²C mode on

SCnLN: Serial port n character length

- 0: 7-bit
- 1: 8-bit

SCnPTY[2:0]: Serial port n parity bit select

- 000:None
- 001:Reserved
- 010:Reserved
- 011:Reserved
- 100:0 (output low)
- 101:1 (output high)
- 110:Even (1s are even)
- 111:Odd (1s are odd)

SCnSB: Serial port n stop bit select (UART mode only)

- 0: 1-bit
- 1: 2-bit

SCnS[1:0]: Serial port n clock source select

The 00 and 10 settings are reserved in UART and I²C modes.

- 00: SBTn pin
- 01: Timer 0 underflow × 1/8
- 10: Timer 1 underflow × 1/2
- 11: Timer 1 underflow × 1/8

SC0TRB/SC1TRB: Serial Port n Transmit/Receive Buffer **x'00FD82'/x'00FD88'**

Bit:	7	6	5	4	3	2	1	0
	SCn TRB7	SCn TRB6	SCn TRB5	SCn TRB4	SCn TRB3	SCn TRB2	SCn TRB1	SCn TRB0
Reset:	—	—	—	—	—	—	—	—
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Data transmission begins when the CPU writes data to SCnTRB.

The CPU retrieves the data by reading SCnTRB. SCnTRB has two respective buffers, for transmission and for reception. The buffers for reception consist of two buffers and the received data is set to SCnTRB after the reception ends, and held until that of the next data ends

Over-run-error occurs if SCnTRB is not read, before the reception of the next data ends (See 5-5)

When the received data is set to SCnTRB, reception end interrupt occurs and SCnRXA flag of SCnTRB register is set to 1.

During 7-bit transfers, the most significant bit (bit 7) of SCnTRB is always 0.

The reset value of SC0TRB is undefined.

SC0STR/SC1STR: Serial Port n Status Register

x'00FD83'/x'00FD8B'

Bit:	7	6	5	4	3	2	1	0
	SCn TBY	SCn RBY	SCn ISP	SCn RXA	SCn IST	SCn FE	SCn PE	SCn OE
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R

SCnSTR contains the error detection and status flags for the serial interfaces.

SCnTBY: Serial port n transmission busy flag

- 0: OK to transmit
- 1: Transmission in progress

SCnRBY: Serial port n reception busy flag

- 0: OK to receive
- 1: Reception in progress

SCnISP: Serial port n I²C stop sequence detect

A read or write to SCnTRB clears this bit.

- 0: No stop sequence
- 1: Stop sequence detected

SCnRXA: Serial port n received data flag

- 0: Not received
- 1: Received

SCnIST: Serial port n I²C start sequence detect

A read or write to SCnTRB clears this bit.

- 0: No start sequence
- 1: Start sequence detected

SCnFE: Serial port n framing error

A framing error occurs when a 0 is received during stop bit transfer. Framing error data is updated each time the stop bit is received.

- 0: No error
- 1: Framing error occurred

SCnPE: Serial port n parity error

A parity error occurs when the received parity bit is 1 and the parity setting is 0; when the received parity bit is 0 and the parity setting is 1; when the received parity bit is odd and the parity setting is even; or when the received parity bit is even and the parity setting is odd. Parity error data is updated each time the parity bit is received.

- 0: No error
- 1: Parity error occurred

SCnOE: Serial port n overrun error

An overrun error occurs during reception when a data byte reception ends before the CPU has read the previous byte from SCnTRB. Overrun error data is updated each time the last data bit (seventh or eighth bit) is received.

- 0: No error
- 1: Overrun error occurred

6 Analog-to-Digital Converter

6.1 Description

The MN102H75K/85K contains an 8-bit charge redistribution A/D converter (ADC) that can process up to 12 channels. The reference clock is selectable to $B_{OSC} \times 1/8$ or $1/16$. When B_{OSC} is 24 MHz, you must set the reference clock to $B_{OSC}/8$ (conversion rate = 4 μ s) or higher.

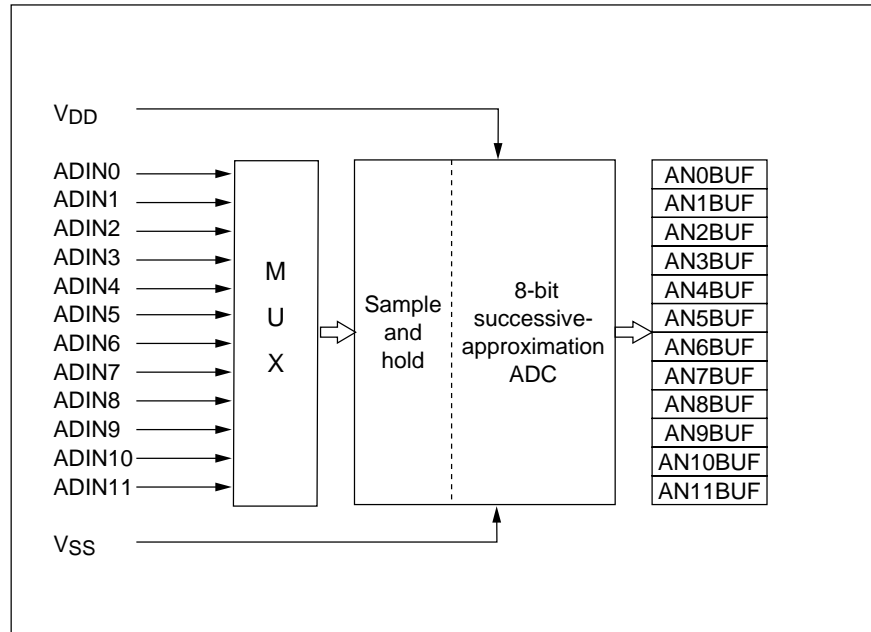


Figure 6-1 ADC Architecture

6.2 Features

Table 6-1 ADC Functions and Features

Function/Feature	Description
Sample and hold	Embedded
Conversion time	4 μ s per channel (when $B_{OSC} = 24$ MHz)
Clock sources	Programmable to B_{OSC} divided by 8 or 16
Operating modes	46 operating modes (four types) ⁽¹⁾ <ul style="list-style-type: none"> • Single conversion of one input (channel 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, or 11) • Single conversion of multiple inputs (channels 0–n, where $n = 1-11$) • Continuous conversion of one input (channel 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, or 11) • Continuous conversion of multiple inputs (channels 0–n, where $n = 1-11$)
Conversion start	Timer 1 underflow or register setting
Interrupts	An interrupt is generated each time a single or continuous conversion sequence ends

Note: 1. Channels correspond to the ADIN pin having the same number. For instance, channel 3 (or ch3) corresponds to ADIN3.

6.3 Block Diagram

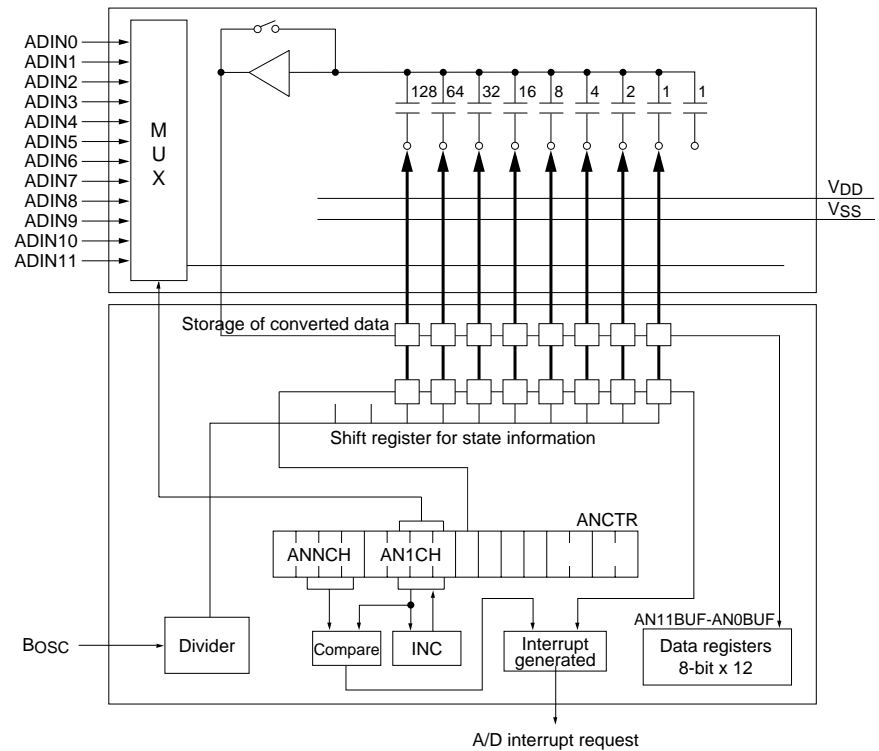


Figure 6-2 ADC Block Diagram

6.4 A/D Conversion Timing

6.4.1 Selecting the ADC Clock Source

Calculate the A/D conversion time as follows:

$$\text{conversion time (s)} = [12 (\text{cycles}) \times (B_{\text{OSC}} \text{ cycle}) (\text{s}) \times \text{divide-by ratio}] / \text{ch}$$

For example, if you set the clock source to $B_{\text{OSC}}/8$, the conversion time is $B_{\text{OSC}} \times 96$ cycles.

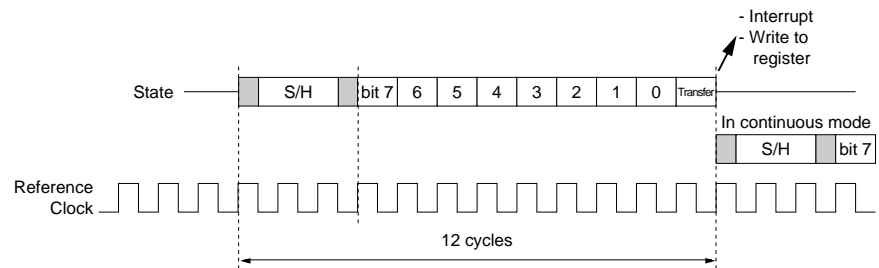


Figure 6-3 ADC Timing

6.4.2 Single Channel/Single Conversion Timing

When ANMD[1:0] = b'00', the ADC converts one ADIN input signal a single time. An interrupt occurs when the conversion ends. Load the number of the channel to be converted to the AN1CH[3:0] field of the ADC control register (ANCTR). (The ANNCH[3:0] field is ignored in this mode.)

When the software starts the conversion, write a 0 to the ANTC bit (disabling conversion start at timer 1 underflow), then write a 1 to ANEN. (If ANTC = 1, ANEN goes high upon a timer 1 underflow.) ANEN remains high during the conversion, then clears to 0 when the conversion ends.

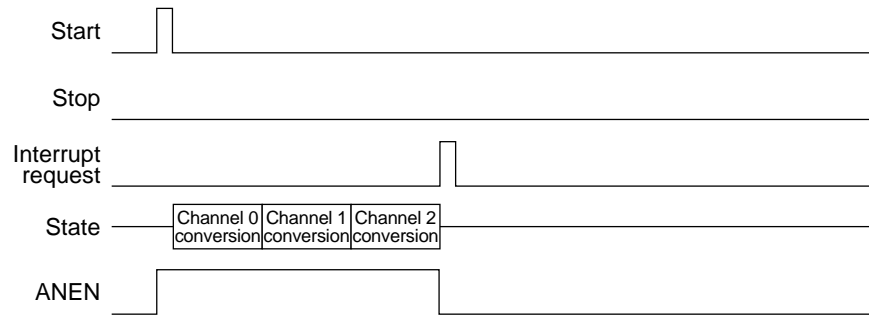


Figure 6-4 Single Channel/Single Conversion Timing

6.4.3 Multiple Channel/Single Conversion Timing

When ANMD[1:0] = b'01', the ADC converts multiple, consecutive ADIN input signals a single time. An interrupt occurs when the conversion sequence ends. Load 0s to the AN1CH[3:0] field of the ADC control register (ANCTR), then load the number of the final channel in the sequence to the ANNCH[3:0] field. The sequence always begins with channel 0.

When the software starts the conversion, write a 0 to the ANTC bit (disabling conversion start at timer 1 underflow), then write a 1 to ANEN. (If ANTC = 1, ANEN goes high upon a timer 1 underflow.) ANEN remains high during the conversion, then clears to 0 when the conversion sequence ends. Note that the AN1CH[3:0] field holds the number of the channel being converted. It clears to 0 when the sequence ends.

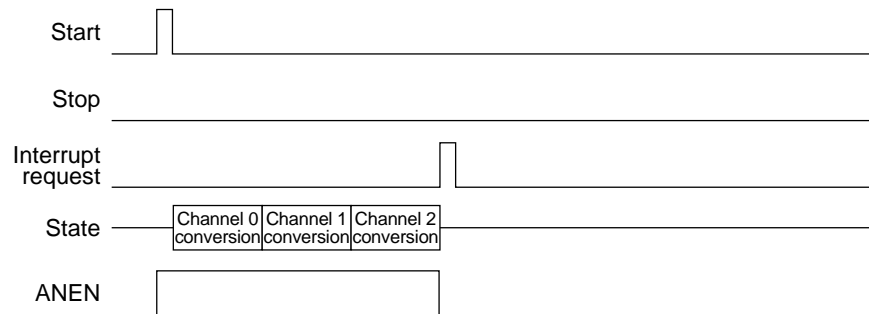


Figure 6-5 Multiple Channel/Single Conversion Timing

6.4.4 Single Channel/Continuous Conversion Timing

When $ANMD[1:0] = b'10'$, the ADC converts one ADIN input signal continuously. An interrupt occurs each time the conversion ends. Load the number of the channel to be converted in the $AN1CH[3:0]$ field of the ADC control register (ANCTR). (The $ANNCH[3:0]$ field is ignored in this mode.)

When the software starts the conversion, write a 0 to the ANTC bit (disabling conversion start at timer 1 underflow), then write a 1 to ANEN. (If $ANTC = 1$, ANEN goes high upon a timer 1 underflow.) ANEN remains high during the conversion. To end the A/D conversion, write a 0 to ANEN.

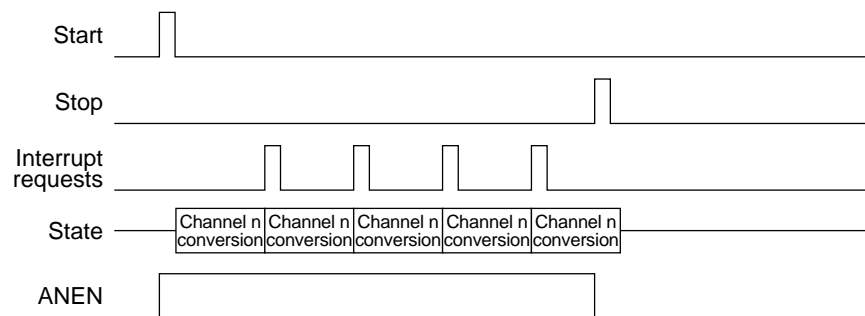


Figure 6-6 Single Channel/Continuous Conversion Timing

6.4.5 Multiple Channel/Continuous Conversion Timing

When $ANMD[1:0] = b'11'$, the ADC converts multiple, consecutive ADIN input signals continuously. An interrupt occurs each time the conversion sequence ends. Load 0s to the $AN1CH[3:0]$ field of the ADC control register (ANCTR), then load the number of the final channel in the sequence to the $ANNCH[3:0]$ field. The sequence always begins with channel 0.

When the software starts the conversion, write a 0 to the ANTC bit (disabling conversion start at timer 1 underflow), then write a 1 to ANEN. (If $ANTC = 1$, ANEN goes high upon a timer 1 underflow.) ANEN remains high during the conversion. To end the A/D conversion, write a 0 to ANEN. Note that the $AN1CH[3:0]$ field holds the number of the channel being converted. It clears to 0 when the sequence ends.

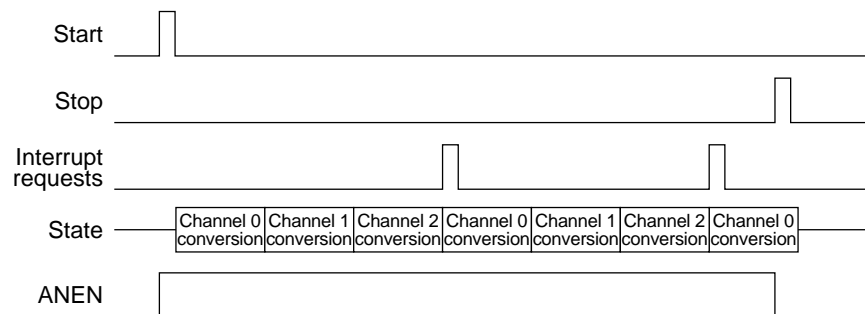


Figure 6-7 Multiple Channel/Continuous Conversion Timing

6.5 ADC Setup Examples

6.5.1 Setting Up Software-Controlled Single-Channel A/D Conversion

This example illustrates single-channel conversion controlled by the software. The ADIN6 pin inputs an analog voltage signal (0.0 V–3.3 V) and the ADC converts it to 8-bit digital values.

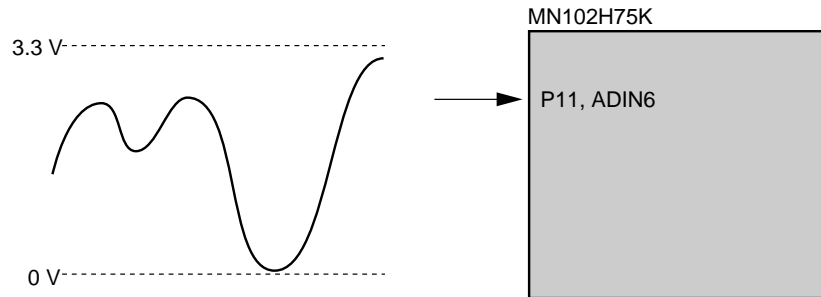


Figure 6-8 Single-Channel A/D Conversion

■ **To set up the input port:**

Set the P1MD2 bit of the port 1 output mode register (P1MD) to 1. This sets the ADIN6 pin (P11) to analog input.

■ **To set up the ADC:**

1. Set the operating conditions in the ADC control register (ANCTR). Select single-channel, single-conversion mode (ANMD[1:0] = b'00'), B_{OSC}/8 as the clock source (ANCK[1:0] = b'10'), and channel 6 as the conversion channel (AN1CH[3:0] = b'011'). Set the conversion start/busy bit, ANEN, to 0.

ANCTR (example)

x'00FF00'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	AN NCH3	AN NCH2	AN NCH1	AN NCH0	AN 1CH3	AN 1CH2	AN 1CH1	AN 1CH0	AN EN	AN TC	—	—	AN CK1	AN CK0	AN MD1	AN MD0
Setting:	0	0	0	0	0	1	1	0	0	0	0	0	1	0	0	0

When the software controls the conversion start, it must set the ANEN flag to 1.

The ADC can also generate an interrupt when the conversion ends, once the data is stored in AN6BUF. In this case, the software does not need to wait for the ANEN flag before reading AN6BUF.

2. Set ANEN to 1 to start conversion. Conversion starts on the first rising edge of the ADC clock after ANEN is set. The conversion time is 12 cycles of the ADC clock. When B_{OSC} = 24 MHz, this is 4.0 μs, or 4.0 μs–4.3 μs after ANEN is set.
3. Wait for the conversion to end. Since ANEN remains high during conversion, then clears to 0, the program must wait until ANEN is 0.
4. Read the ADIN6 conversion data buffer (AN6BUF). The converter divides 0 to 3.3 volts into 256 segments, so the digital result is a value from 0 to 255.

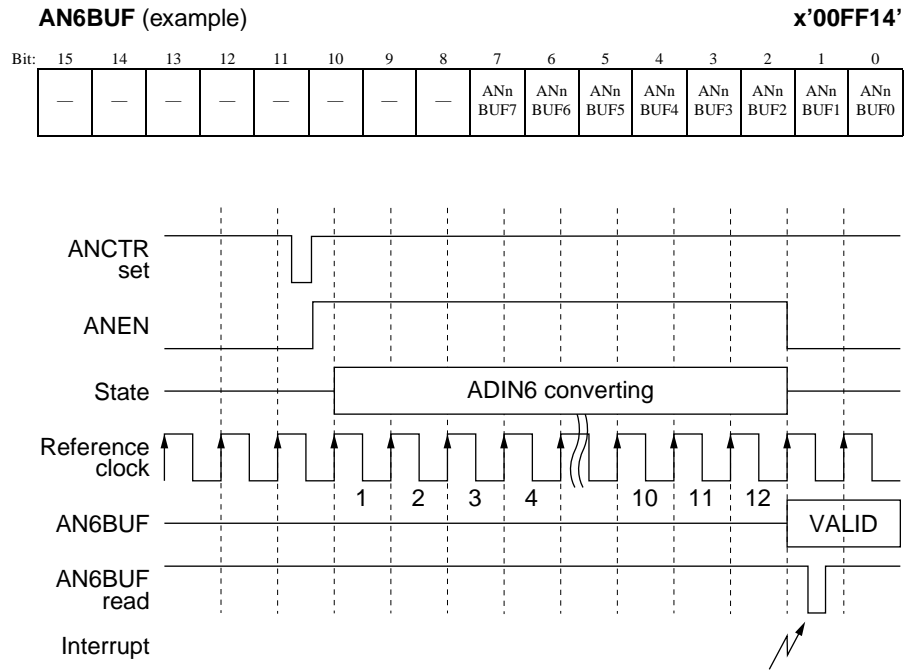


Figure 6-9 Timing of Software-Controlled Single-Channel A/D Conversion

6.5.2 Setting Up Hardware-Controlled Intermittent Three-Channel A/D Conversion

This example illustrates multiple-channel conversion controlled by the hardware. The ADIN2, ADIN1, and ADIN0 pins input analog voltage signals (0.0 V– 3.3 V) and the ADC converts the voltages to 8-bit digital values. It writes the results to the registers periodically, each time timer 1 underflows.

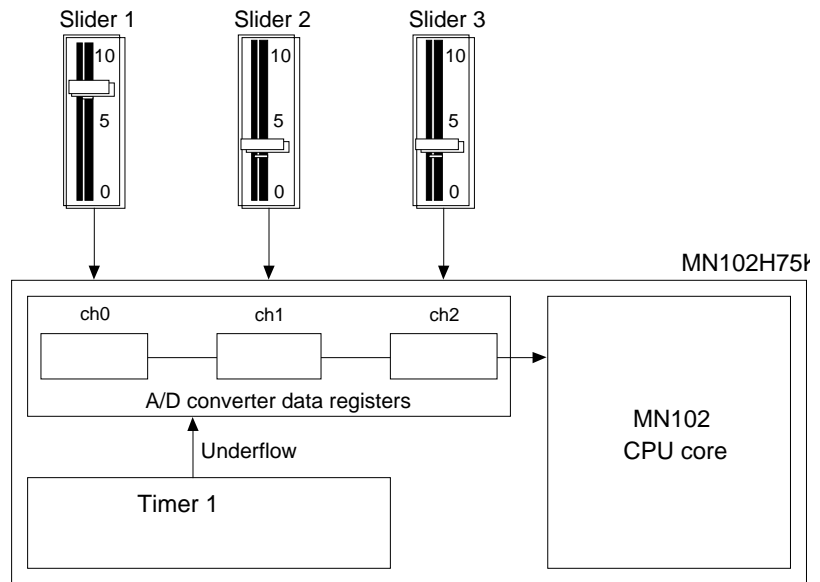


Figure 6-10 Multiple-Channel A/D Conversion

Analog-to-Digital Converter

ADC Setup Examples

■ To set up the input port:

Set the P0DIR[5:3] bits of the port 0 I/O control register (P0DIR) to 0. This sets the ADIN2 (P05), ADIN1 (P04), and ADIN0 (P03) pins (P11) to general-purpose input.

■ To set up the ADC:

Set the operating conditions in the ADC control register (ANCTR). Select multiple-channel, single-conversion mode (ANMD[1:0] = b'01') and $B_{OSC}/8$ as the clock source (ANCK[1:0] = b'10'). Set the conversion start/busy bit, ANEN, to 0. Set ANTC to 1, enabling conversion start at timer 1 underflow. Set the AN1CH[3:0] field to the first channel (channel 0) and set the ANNCH[3:0] to the last channel (channel 2).

ANCTR (example)

x'00FDA0'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	AN NCH3	AN NCH2	AN NCH1	AN NCH0	AN 1CH3	AN 1CH2	AN 1CH1	AN 1CH0	AN EN	AN TC	—	—	AN CK1	AN CK0	AN MD1	AN MD0
Setting:	0	0	1	0	0	0	0	0	0	1	0	0	1	0	0	1

■ To set up the conversion cycle

1. Set the divide-by ratio for timer 1. To divide $B_{OSC}/4$ by 256, write 255 (x'FF') to the timer 1 base register (TM1BR). (The valid range for TM1BR is 1 to 255.)

TM1BR (example)

x'00FE11'

Bit:	7	6	5	4	3	2	1	0
	TM1 BR7	TM1 BR6	TM1 BR5	TM1 BR4	TM1 BR3	TM1 BR2	TM1 BR1	TM1 BR0
Setting:	1	1	1	1	1	1	1	1

2. Set the TM1LD bit of the TM1MD register to 1 and the TM1EN bit to 0. (This loads the value in the base register to the binary counter.)

TM1MD (example)

x'00FE21'

Bit:	7	6	5	4	3	2	1	0
	TM1 EN	TM1 LD	—	—	—	—	TM1 S1	TM1 S0
Setting:	0	1	0	0	0	0	1	0

3. Set TM1LD to 0 and TM1EN to 1. This starts the timer. Counting begins at the start of the next cycle.

When the binary counter (TM1BC) reaches 0, the microcontroller reloads the value in the base register (TM1BR) to TM1BC and simultaneously generates a timer 1 underflow interrupt. After each timer 1 underflow, the ADC converts each of the ADIN[2:0] inputs a single time.



Do not change the clock source once you have selected it. Selecting the clock source while you set up the count operation control will corrupt the value in the binary counter.

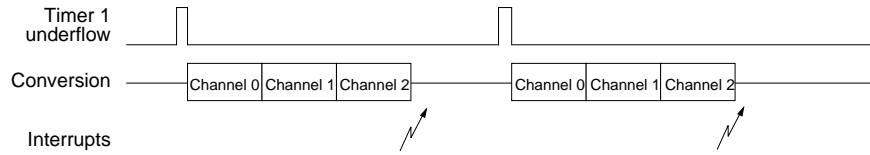


Figure 6-11 Timing of Hardware-Controlled Intermittent Three-Channel A/D Conversion

6.6 ADC Control Registers

The ADC contains thirteen registers—one control register (ANCTR) and twelve data buffers (each associated with one of the ADIN pins). ANCTR controls the operating conditions, and the read-only data buffers hold the results of the A/D conversions.

Table 6-2 ADC Control Registers

Register	Address	R/W	Description
ANCTR	x'00FF00'	R/W	ADC control register
AN0BUF	x'00FF08'	R	ADIN0 conversion data buffer
AN1BUF	x'00FF0A'	R	ADIN1 conversion data buffer
AN2BUF	x'00FF0C'	R	ADIN2 conversion data buffer
AN3BUF	x'00FF0E'	R	ADIN3 conversion data buffer
AN4BUF	x'00FF10'	R	ADIN4 conversion data buffer
AN5BUF	x'00FF12'	R	ADIN5 conversion data buffer
AN6BUF	x'00FF14'	R	ADIN6 conversion data buffer
AN7BUF	x'00FF16'	R	ADIN7 conversion data buffer
AN8BUF	x'00FF18'	R	ADIN8 conversion data buffer
AN9BUF	x'00FF1A'	R	ADIN9 conversion data buffer
AN10BUF	x'00FF1C'	R	ADIN10 conversion data buffer
AN11BUF	x'00FF1C'	R	ADIN11 conversion data buffer

AN0BUF–AN11BUF: ADIN0–ADIN11 Conversion Data Buffers x'00FF00'–x'00FF1C'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	—	ANn BUF7	ANn BUF6	ANn BUF5	ANn BUF4	ANn BUF3	ANn BUF2	ANn BUF1	ANn BUF0
Reset:	0	0	0	0	0	0	0	0	—	—	—	—	—	—	—	—
R/W:	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

These buffers hold the 8-bit A/D conversion data. Their value is unknown after reset.

Analog-to-Digital Converter

ADC Control Registers

ANCTR: ADC Control Register

x'00FF00'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	AN NCH3	AN NCH2	AN NCH1	AN NCH0	AN 1CH3	AN 1CH2	AN 1CH1	AN 1CH0	AN EN	AN TC	0	0	AN CK1	AN CK0	AN MD1	AN MD0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R	R	R/W	R/W	R/W	R/W

ANNCH[3:0]: Channel select for multiple-channel conversion

0000: Convert ADIN0	0111: Convert ADIN0–ADIN7
0001: Convert ADIN0–ADIN1	1000: Convert ADIN0–ADIN8
0010: Convert ADIN0–ADIN2	1001: Convert ADIN0–ADIN9
0011: Convert ADIN0–ADIN3	1010: Convert ADIN0–ADIN10
0100: Convert ADIN0–ADIN4	1011: Convert ADIN0–ADIN11
0101: Convert ADIN0–ADIN5	1100–1111: Reserved
0110: Convert ADIN0–ADIN6	

AN1CH[3:0]: Channel select for single-channel conversion

0000: Convert ADIN0	0111: Convert ADIN7
0001: Convert ADIN1	1000: Convert ADIN8
0010: Convert ADIN2	1001: Convert ADIN9
0011: Convert ADIN3	1010: Convert ADIN10
0100: Convert ADIN4	1011: Convert ADIN11
0101: Convert ADIN5	1100–1111: Reserved
0110: Convert ADIN6	

ANEN: Conversion start/busy flag

- 0: No conversion in progress
- 1: Conversion in progress

ANTC: Conversion start at timer 1 underflow

- 0: Disable
- 1: Enable

ANCTR5

Always set this bit to 0.

ANCK[1:0]: Clock source select

- 00: Reserved
- 01: Reserved
- 10: B_{OSC} /8
- 11: B_{OSC} /16

ANMD[1:0]: Operating mode select

- 00: Single channel, single conversion
- 01: Multiple channels, single conversion
- 10: Single channel, continuous conversion
- 11: Multiple channels, continuous conversion

6.7 Cautions about Analog-to-Digital Converter

The type of this Analog-to-Digital Converter is a sample-and-hold one, and so the current temporarily flows in conversion to charge the condenser of the sample-and-hold circuit. For this reason, the following settings are needed to get the accuracy of conversion:

1. Impedance of analog input terminal must be below $8k\Omega$.
2. When impedance of analog input terminal is over $8k\Omega$, condenser which capacity is above 2000 pF must be connected to suppress the voltage changes of the analog input terminal.
3. During conversion do not change the output level of the terminal (from H to L or from L to H), and not turn on and off the peripheral circuit, cause these actions may influence the power supply voltage.

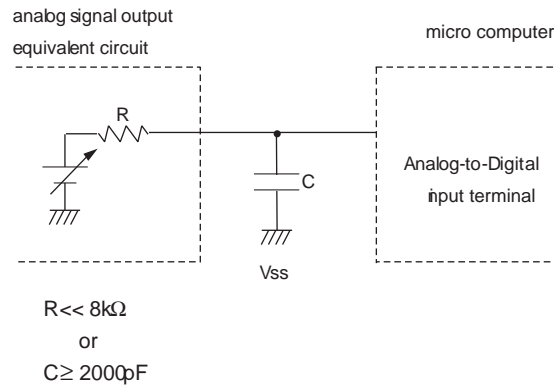


Figure 6-12 Cautions on Analog-to-Digital Converter

7 On-Screen Display



If you use the OSD function, the DMA function executes for both the text and graphics layers, even if your program does not use one of these layers. To prevent error, program data for the unused layer to meet the restrictions outlined here.

7.1 Description

The MN102H75K/85K contains an on-screen display (OSD) function composed of three layers: a text layer, a graphics layer, and a cursor layer. You can control each layer individually, which gives you great freedom in positioning displays. You can also modify the ROM space that contains the text characters and the graphic tiles and the VRAM space that contains the text and graphics programs. This allows you to adjust the memory space to fit your application.

7.2 Features

Table 7-1 OSD Functions and Features

Function/Feature	Text Layer	Graphics Layer
Characters or tiles per line ⁽¹⁾	38 characters per line ⁽²⁾	18 or 28 tiles per line ⁽³⁾
RAM usage	80 bytes per line Line-by-line basis Maximum 64 lines	40 or 64 bytes per line Line-by-line basis Maximum 64 lines
ROM usage	36 bytes per character	16 colors: 128 bytes per tile ⁽⁴⁾ 8 colors: 96 bytes per tile 4 colors: 64 bytes per tile 2 colors: 32 bytes per tile
Max. characters or tiles	1024 characters	512 tiles (in all color modes)
Resolution	16 (wide) x 18 (high) pixels <i>In closed-caption mode:</i> 16 (W) x 26 (H) (underlining is in the hardware)	16 (W) x 16 (H) pixels, or 16 (W) x 18 (H) pixels
Color depth	One 16-color palette out of 4096 colors	Two 16-color palettes out of 4096 colors (Total 32 colors in one display)
Display start position ⁽⁵⁾	H: 1 dot resolution, 1024 steps V: 1 H scan line resolution, 1024 steps	H: 1 dot resolution, 1024 steps V: 1 H scan line resolution, 1024 steps
Character or tile size ⁽⁶⁾	16 character sizes, line-by-line basis H: 1x, 2x, 3x, 4x V: 1x, 2x, 4x, 6x	16 character sizes, line-by-line basis H: 1x, 2x, 3x, 4x V: 1x, 2x, 4x, 6x
Display functions	<ul style="list-style-type: none"> • Shutter effect • Outlining • Shadowing (foreground and background) • Blinking <i>In closed-caption mode:</i> <ul style="list-style-type: none"> • Italics • Underlining 	<ul style="list-style-type: none"> • Repeated tile or blank ⁽⁶⁾
Cursor layer	Selects one tile from the graphic tile area (16 x 16) Display position: H: 1 dot resolution, 1024 steps V: 1 H scan line resolution, 1024 steps	

- Notes:
1. Maximum 60 characters + tiles in one line (when using a 64-byte graphics line). For example, (1) If a graphics line contains 28 tiles, then the corresponding line in the text layer can only contain 32 characters. (2) If a text line contains 38 characters, then the corresponding line in the graphics layer can only contain 22 tiles.
 2. Maximum 38 characters per line (60 characters + tiles) with the default text colors. Each color assignment, including outlining and blinking, decreases this total by one.
 3. The maximum number of tiles per line is programmable in the GEXTE bit of the OSD2 register. 18 tiles requires 40 bytes per line, and 28 tiles requires 64 bytes per line. The setting applies to all lines.
 4. Multiple modes cannot be used simultaneously—the color mode applies to the entire display.
 5. The OSD dot clock frequency controls the horizontal position and size. For details, see section 7.9.4, “Setting Up the OSD Display Position,” on page 180, and section 7.11, “Selecting the OSD Dot Clock,” on page 186.
 6. This function can be used for a wallpapering effect or to insert spaces. One tile code can be repeated up to 16 times. Repeating tiles allows you to use more than 18 (or 28) tiles per line.
 7. For the OSD block to operate correctly, always set bit 7 of the PCNT2 register (x'00FF92') to 0.

7.3 Block Diagram

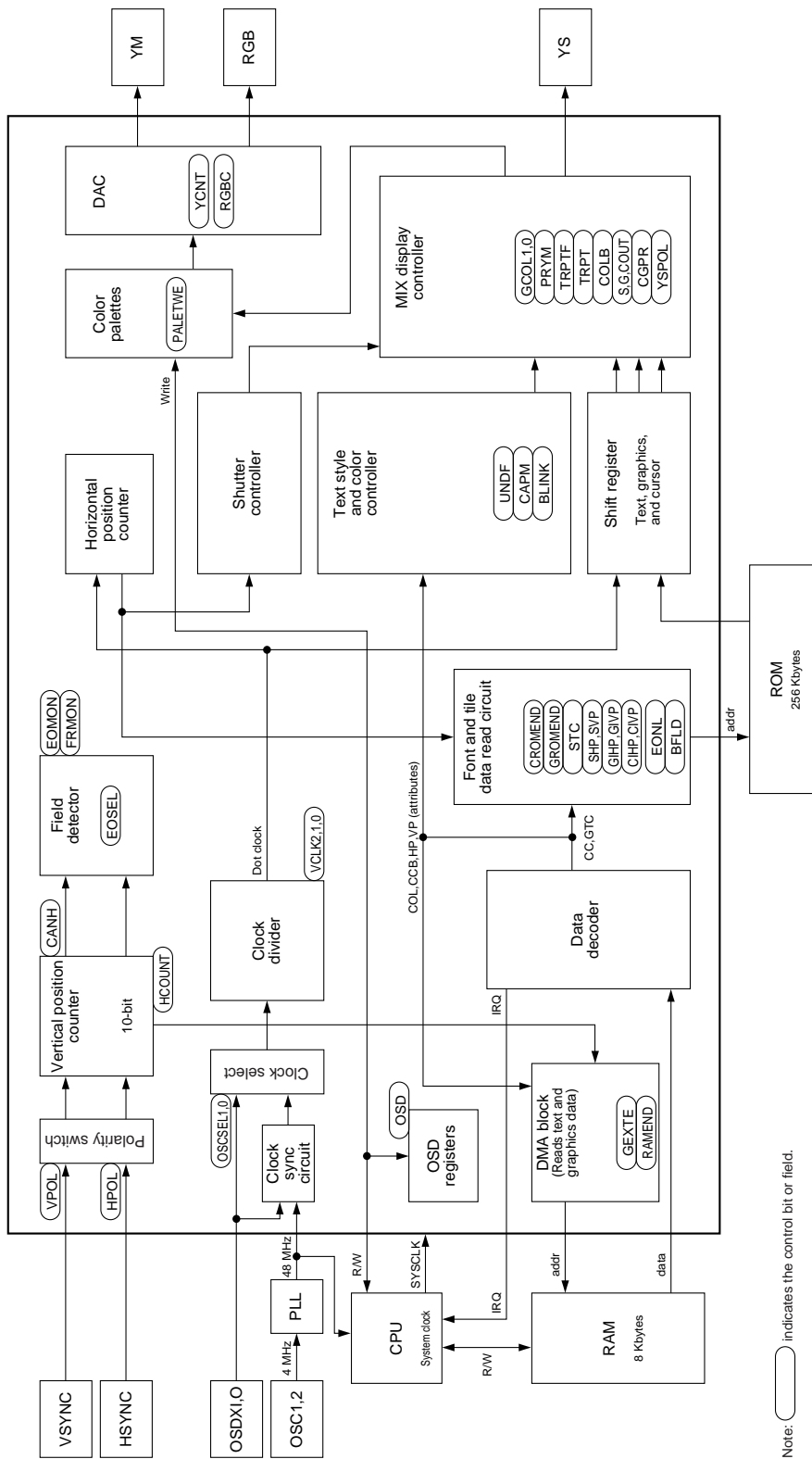


Figure 7-1 OSD Block Diagram

7.4 Power-Saving Considerations in the OSD Block

Table 7-2 shows bits that can decrease the power consumption of the OSD block. This section explains how to use these bits.



OSDPOFF resets to 0. To operate the OSD, you must first set this bit to 1.



To turn off the OSD block to save power:

1. Write a 0 to OSD (OSD1, bit 10).
2. Wait for the next VSYNC input.
3. Write a 0 to OSDPOFF (PCNT0, bit 7), turning the clock off.

If you turn the clock off before the VSYNC input, power usage may not drop or the microcontroller may halt.

Table 7-2 Power-Saving Control Bits for the OSD

Bit Name	Register	Address	Bit	Description	Reset
OSDPOFF	PCNT0	x'00FF90'	7	0: System clock off to OSD 1: System clock on to OSD	0
OSDREG E	PCNT2	x'00FF92'	0	0: R/W disabled for OSD registers 1: R/W enabled for OSD registers	0

■ **Using OSDPOFF to control the system clock supply to the OSD**

The OSDPOFF bit enables or disables the system clock supply to the OSD block. When the OSD is unused, setting this bit to 0 stops the clock supply to the OSD, reducing power dissipation. Setting OSDPOFF to 0 not only disables the OSD display, it disables reads from and writes to the OSD registers. To operate the OSD, set this bit to 1, then set up the OSD registers.

■ **Using OSDREGE to control read/write access to the OSD registers**

The OSDREGE bit enables or disables read/write operations to the OSD registers. Once you have set the OSD registers, you can write a 0 to this bit to disable further reads and writes to them, reducing power dissipation. This bit resets to 0.

Note that when OSDPOFF is 0, you cannot read or write to the OSD registers even if OSDREGE is 1. Table 7-3 shows the combinations of OSDPOFF and OSDREGE. Note also that when OSDREGE is 0, the OSD display runs, but the shuttering motion does not work. If your application requires shutter movement, you must enable OSDREGE.

Table 7-3 OSDPOFF and OSDREGE Settings

OSDPOFF	OSDREGE	OSD	Register R/W	Power Dissipation
0	Don't care	Off	Disabled	Less ↑↓ Greater
1	0	On	Disabled	
1	1	On	Enabled	

7.5 OSD Operation

This section describes the basic operation of the OSD block. The remainder of section 7 provides more detailed specifications.

7.5.1 OSD Clock

The OSD clock source is programmable to either the microcontroller system clock (OSC1, OSC2 pins) or a dedicated OSD clock (OSDXI, OSDXO pins).

■ OSC clock source

An internal phase-locked loop (PLL) multiplies the external 4-MHz frequency to generate an OSD clock that is synchronized to the trailing edge of the horizontal sync signal (HSYNC). This dramatically reduces EMI and character distortion. The output frequency is programmable to 12, 16, 24, 32 or 48 MHz.

■ OSDX clock source

An LC blocking oscillator allows HSYNC to serve as the clock source. You can also input an external clock through the OSDXI pin and synchronize it internally. (Frequency range: 12–48 MHz)

7.5.2 External Input Sync Signals

Input the horizontal sync signal through the HSYNC pin and the vertical sync signal through the VSYNC pin. The pullup resistors and polarity are programmable. An interrupt must occur so that the microcontroller can detect each VSYNC start field. Set the interrupt edge in the IQ1TG[1:0] bits of the EXTMD registers and the OSD input polarity in the VPOL bit of the OSD1 register. Note that you must set these parameters separately.

7.5.3 Multi-Layer Format

Multi-layer technology enables the microcontroller to display text, graphics, and hardware cursor as different display layers with different color depths. You can stack the layers in any order. When the layers contain overlapping images, the cursor always takes highest priority. The priority of the text and graphics layers is programmable in the registers.

■ Text layer

Each character in the text layer contains a foreground (character) color and a background color. Outlining and shadowing are separate options. In closed-caption mode, the OSD can only display the text in the encoded captions. The graphics and cursor layers can be displayed in this mode, but note that if any tiles from either layer overlaps italicized text, the pixels in the graphic will be displaced, distorting the image.

See section 7.11, “Selecting the OSD Dot Clock,” on page 186, for information on setting the OSD clock frequency.



Do not layer any graphic or cursor tiles over italicized characters in closed-caption mode.

■ Graphics layer

The graphics layer contains tiled images. In the 16-color mode, each 4-bit dot on a tile can display one of 16 colors. Each tile can use either of two available color palettes, allowing a total of 32 colors in one display. The graphics layer also supports 2-, 4-, and 8-color modes. All the tiles in a single display screen must be in the same color mode. (For instance, an 8-color-mode tile cannot be displayed at the same time as a 16-color-mode tile.) The size of one tile is 16W x 16H pixels in standard mode and 16W x 18H pixels in extended mode.

■ Cursor layer

The cursor layer displays an icon indicating the position of the next entry. One display screen displays only one cursor (tile). The ROM data and color palettes for the tile are the same as those of the graphics layer. The size of one tile is 16W x 16H pixels in standard mode and 32W x 32H pixels in extended mode.



When you are not using the DAC, set the COMP and IREF pins for the DAC to H.

7.5.4 Output Pin Setup

To set up the output pins, enable the OSD output pins in the I/O registers. Select DAC or digital output for RGB and YM. Set the YS polarity.

7.5.5 Microcontroller Interface

The microcontroller writes display data to be sent to the OSD control registers and the VRAM, which is assigned to internal RAM space. The control registers (CRAMEND and GRAMEND) hold the end address of the data in the VRAM.

7.5.6 VRAM

Display data stored in the VRAM transfers automatically (through) a DMA transfer) from the internal RAM to the OSD as the display approaches its specified position. The microcontroller is suspended while the data is transferred on the bus. See section 7.10, “DMA and Interrupt Timing,” on page 185, for more information on this timing.

The two MSBs of the transferred data contain one or more of the following ID codes:

■ Text layer

- | | |
|---------------------------------------------|-----|
| 1. Character code | CC |
| 2. Color control code (normal mode) | COL |
| 3. Color control code (closed-caption mode) | COL |
| 4. Repeat character/blank code | CCB |
| 5. Character horizontal position code | CHP |
| 6. Character vertical position code | CVP |

■ Graphics layer

- | | |
|--------------------------------------|-----|
| 1. Graphic tile code | GTC |
| 2. Graphics horizontal position code | GHP |
| 3. Graphics vertical position code | GVP |



After a reset clears, the system clock supply to the OSD stops. To operate the OSD, you must first set the OSDPOFF bit of PCNT0 (x'00FF90') to 1.

7.5.7 Conditions for VRAM Writes

■ Text layer



Set CHP, CVP, GHP, and GVP for every line in the VRAM. If you do not, a software processing error may occur.

1. The lead data for each line must be the color control code (COL) or the character code (CC). Never place the horizontal position (CHP), vertical position (CVP), or repeat (CCB) codes at the beginning of a line. (If the lead data is CC, with no COL specification, the character will be text palette color 1, and the background will be color 2.)
2. Place each line's horizontal and vertical position data (CHP and CVP), in that order, at the end of the preceding line.
3. Insert the color control code (COL) before the character code (CC). You do not need to meet condition 3 in the closed-caption mode, since COL can carry over in that mode.
4. A character code (CC) must immediately precede a repeat character/blank code (CCB), and a color control code (COL) or character code (CC) must follow it.
5. To indicate the last line of a display, make the CHP and CVP values for the last line smaller than those in the currently displayed line. In addition, write a 1 to the last line flag of the text layer (CLAST).
6. Two text lines (but no more) can overlap on the screen. The lower line takes priority, appearing to lie on top of the higher line.
7. If the horizontal sync signal is asserted while the microcontroller is accessing CHP and CVP, that line and the next line may not display properly. (For details, see section 7.10.4, "Setting Up the OSD Display Position," on page 189.)

■ Graphics layer

1. Place each line's horizontal and vertical position data (GHP and GVP), in that order, at the end of the preceding line. Do not place GHP and GVP at the start of a line.
2. To indicate the last line of a display, make the GHP and GVP values for the last line smaller than those in the currently displayed line. In addition, write a 1 to the last line flag of the graphics layer (GLAST).
3. Two graphic lines (but no more) can overlap on the screen. The lower line takes priority, appearing to lie on top of the higher line.
4. If the horizontal sync signal is asserted while the microcontroller is accessing GHP and GVP, that line and the next line may not display properly. (For details, see section 7.10.4, "Setting Up the OSD Display Position," on page 189.)

7.6 Standard and Extended Display Modes

Two modes are available for the graphics and cursor layers, standard and extended. In extended mode, the cursor layer can display four grouped graphic tiles rather than one. The graphics layer can display tiles that are two pixels taller than those used in standard mode, giving the graphic tiles the same dimensions as the characters in the text layer.

7.6.1 Cursor Layer Display Modes

The size of the cursor in the cursor layer is programmable to 16W x 16H pixels in standard mode and 32W x 32H pixels in extended mode. Select the mode for this layer in the SPEXT bit of the OSD2 register (x'007F08').

To create a graphic for the cursor layer in extended mode, combine four 16 x 16 tiles. They are ordered as follows: (1) upper left, (2) upper right, (3) lower left, and (4) lower right. Set the pointers to these tiles in cursor tile code registers 0 to 3 (STC0–3), where STC0 corresponds to tile (1). Table 7-4 shows the associated tiles and registers.

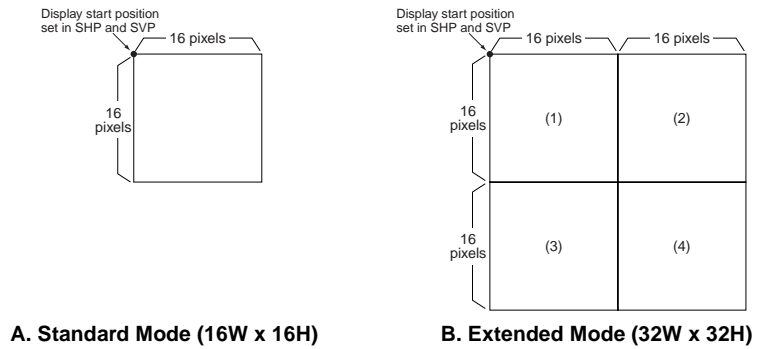


Figure 7-2 Cursor Tiles in Standard and Extended Modes

Table 7-4 Associated Tiles for Cursor Tile Code Registers

Graphic Tile	Register	Register Address
(1) Upper left	STC0	x'007F10'
(2) Upper right	STC1	x'007F2A'
(3) Lower left	STC2	x'007F2C'
(4) Lower right	STC3	x'007F2E'

In standard mode, STC0 is the only cursor tile code register that is enabled. Use the cursor horizontal position register (SHP, x'00F12') and the cursor vertical position register (SVP, x'007F14') in both modes to program the display start position. In extended mode, this position refers to the upper left corner of tile (1). Set the color palette for each tile individually.

7.6.2 Graphics Layer Display Modes

The size of the tiles in the graphics layer is programmable to 16W x 16H pixels in standard mode and 16W x 18H pixels in extended mode. Select the mode for this layer in the GTHT bit of the OSD2 register (x'007F08').

In extended mode, the tiles are the same size as the characters in the text layer. This allows for a cleaner display when text and graphics appear side by side.



You cannot use the tiles created for graphics layer extended mode as cursor tiles.

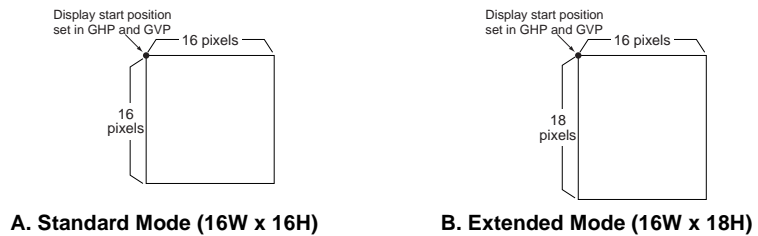


Figure 7-3 Graphic Tiles in Standard and Extended Modes

7.7 Display Setup Examples

7.7.1 Setting Up the Graphics Layer

This section shows how to set up the graphics display data in the VRAM.

■ **Register settings**

RAMEND (x'007F04') = x'80FF' (Graphics RAM end address: x'980F')

GIHP (x'007F16') = x'0822' (GIHP = x'22', GIHSZ = x'1')

GIVP (x'007F18') = x'1803' (GIVP = x'03', GIVSZ = x'3')

OSD2 (x'007F08') = x'0047' (1 line maximum = 18 tiles, 16-color mode, graphics take priority)

Table 7-5 Example Graphics VRAM Settings

Line No.	RAM Addr.	RAM Data	Data Type	Description
1	980E	4000	GTC	Graphic tile, GCB = x'0', GPRT = 0, GTC = x'000'
	980C	4255	GTC	Graphic tile, GCB = x'0', GPRT = 1, GTC = x'055'
	980A	02AA	GTC	Blank tile, GCB = x'0', GPRT = 1, GTC = x'0AA'
	9808	4100	GTC	Graphic tile, GCB = x'0', GPRT = 0, GTC = x'100'
	9806	C004	GHP	GHSZ = x'0', GSHT = 0, GHP = x'04'
	9804	C040	GVP	GLAST = 0, GVSZ = x'0', GINT = 0, GVP = x'40'

2	97E6	4010	GTC	Graphic tile, GCB = x'0', GPRT = 0, GTC = x'010'
	97E4	4011	GTC	Graphic tile, GCB = x'0', GPRT = 0, GTC = x'011'
	97E2	4012	GTC	Graphic tile, GCB = x'0', GPRT = 0, GTC = x'012'
	97E0	4C13	GTC	Graphic tile, GCB = x'3', GPRT = 0, GTC = x'013'
	97DE	4214	GTC	Graphic tile, GCB = x'0', GPRT = 1, GTC = x'014'
	97DC	0815	GTC	Blank tile, GCB = x'2', GPRT = 0, GTC = x'015'
	97DA	4216	GTC	Graphic tile, GCB = x'0', GPRT = 1, GTC = x'016'
	97D8	D810	GHP	GHSZ = x'3', GSHT = 0, GHP = x'10'
	97D6	C858	GVP	GLAST = 0, GVSZ = x'1', GINT = 0, GVP = x'58'
...	
3	97BE	4181	GTC	Graphic tile, GCB = x'0', GPRT = 0, GTC = x'181'
	97BC	4382	GTC	Graphic tile, GCB = x'0', GPRT = 1, GTC = x'182'
	97BA	C044	GHP	GHSZ = x'0', GSHT = 0, GHP = x'44'
	97B8	E020	GVP	GLAST = 1, GVSZ = x'0', GINT = 0, GVP = x'20'

- Notes:
1. Always specify GHP and GVP, in that order, at the end of each line.
 2. Set GINT to 1 in the GVP setting to generate an OSD graphics interrupt.
 3. Set GLAST to 1 in the GVP setting for the last line in the graphics display. Also, set the GVP value to a smaller value than the position of the current line. (In the example in table 7-5, GVP = x'20' is smaller than GVP = v'58'.)

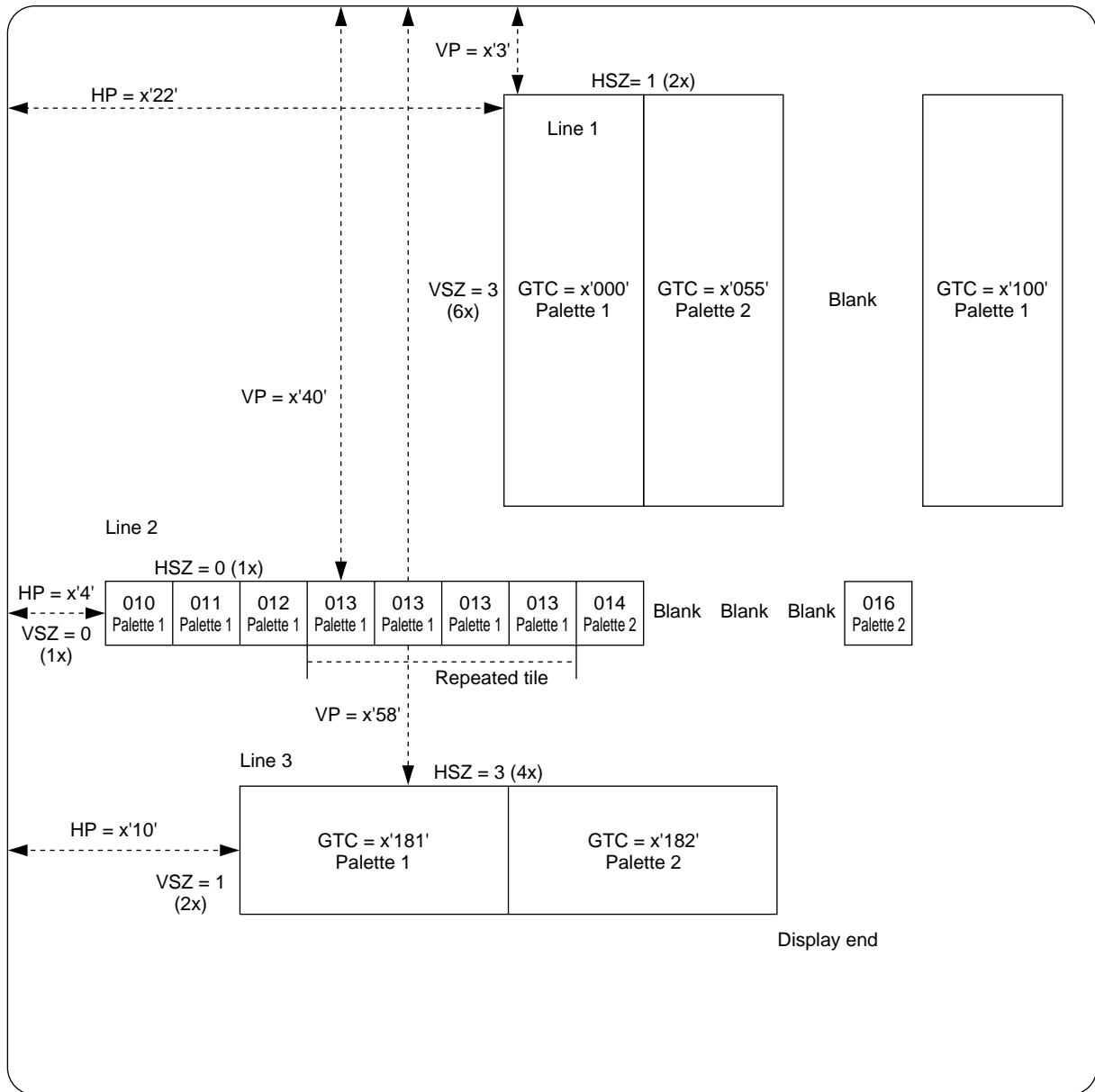


Figure 7-4 Graphics Display Example

7.7.2 Setting Up the Text Layer

This section shows how to set up the text display data in the VRAM.

■ **Register settings**

RAMEND (x'007F04') = x'80FF' (Text RAM end address: x'9FFF')

CIHP (x'007F1A') = x'1020' (CIHP = x'20', CIHSZ = x'2')

CIVP (x'007F1C') = x'1803' (CIVP = x'03', CIVSZ = x'3')

OSD3 (x'007F0A') = x'0000' (CAPM = x'0')

Table 7-6 Example Text VRAM Settings

Line No.	RAM Addr.	RAM Data	Data Type	Description
1	9FFE	0000	CC	Character code = x'000'
	9FFC	825A	COL	BSHAD = 0, CSHAD = 0, FRAME = 1, BCOL = x'5', CCOL = x'A'
	9FFA	0001	CC	Character code = x'001'
	9FF8	0002	CC	Character code = x'002'
	9FF6	C004	CHP	CHSZ = x'0', CSHT = 0, CHP = x'04'
	9FF4	C840	CVP	CLAST = 0, CVSZ = x'1', CINT = 0, CVP = x'40'
...
2	9FAE	9469	COL	BSHAD = 2, CSHAD = 1, FRAME = 0, BCOL = x'6', CCOL = x'9'
	9FAC	0006	CC	Character code = x'006'
	9FAA	0007	CC	Character code = x'007'
	9FA8	4012	CCB	Repeat character code, CCB = x'2'
	9FA6	9A78	COL	BSHAD = 3, CSHAD = 0, FRAME = 1, BCOL = x'7', CCOL = x'8'
	9FA4	0008	CC	Character code = x'008'
	9FA2	4002	CCB	Blank, CCB = x'2'
	9FA0	0010	CC	Character code = x'010'
	9F9E	D810	CHP	CHSZ = x'3', CSHT = 0, CHP = x'10'
	9F9C	C870	CVP	CLAST = 0, CVSZ = x'1', CINT = 0, CVP = x'70'
...
3	9F5E	0300	CC	Character code = x'300'
	9F5C	0310	CC	Character code = x'310'
	9F5A	C044	CHP	CHSZ = x'0', CSHT = 0, CHP = x'44'
	9F58	E030	CVP	CLAST = 1, CVSZ = x'0', CINT = 0, CVP = x'30'

- Notes:
1. Always specify the color code (COL) or character code (CC) at the beginning of each line. If CC is the first code (no COL), the foreground (text) color will be color 1 on the palette and the background will be color 2.
 2. Always specify CHP and CVP, in that order, at the end of each line.
 3. A CC code must immediately follow a COL code.
 4. If you repeat a character with the CCB code, precede the CCB code with a CC code and follow it with a COL or CC code.
 5. Set CINT to 1 in the CVP setting to generate an OSD text interrupt.
 6. Set CLAST to 1 in the CVP setting for the last line in the text display. Also, set the CVP value to a smaller value than the position of the current line. (In the example in table 7-6, CVP = x'30' is smaller than CVP = v'70'.)



The text display starts one dot to the right of the HP setting.

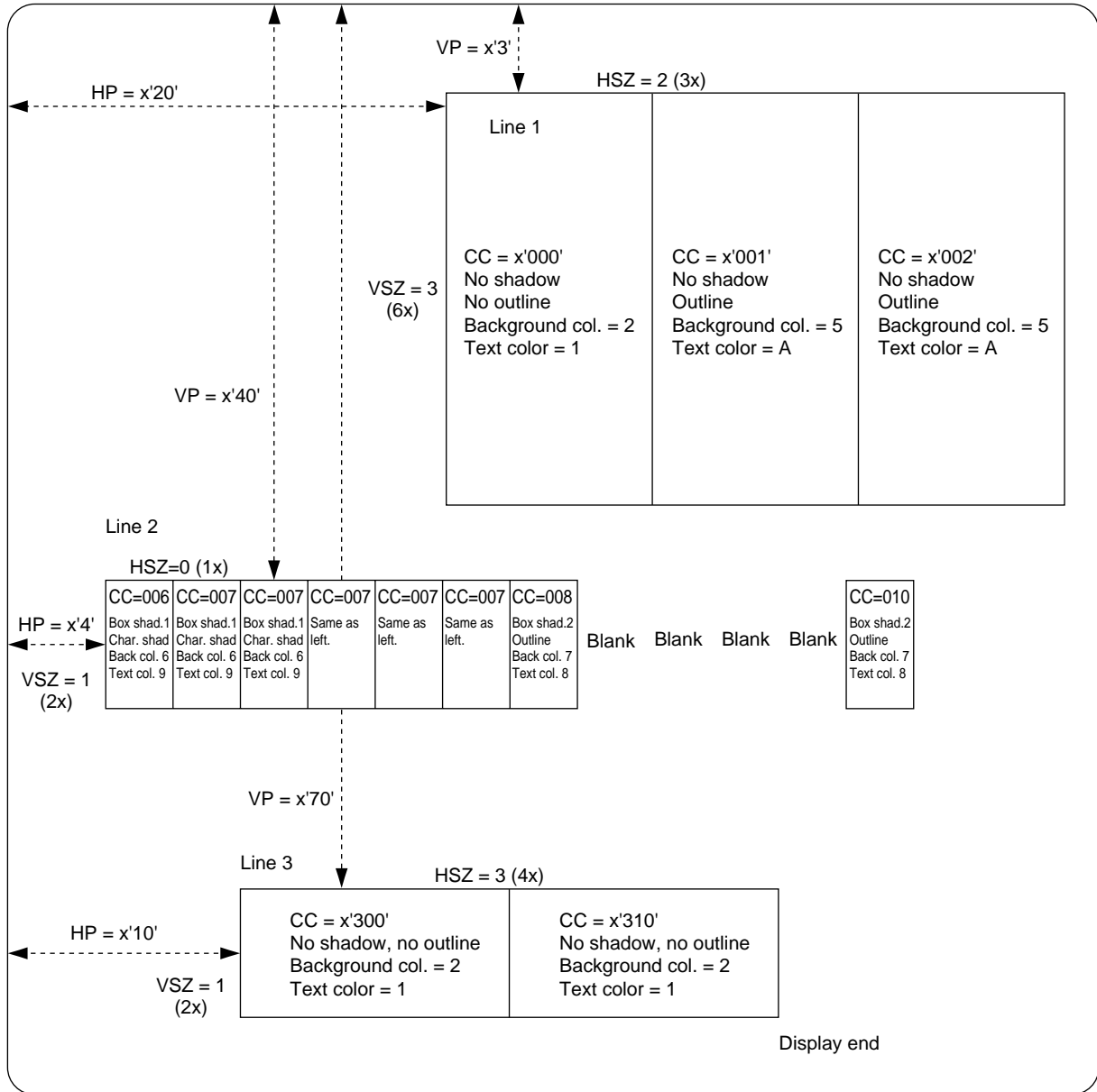


Figure 7-5 Text Display Example

7.8 VRAM

7.8.1 VRAM Operation

Table 7-7 VRAM Bit Allocation in Internal RAM

Text layer	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
CC Character code	0	0	*	*	*	*	CCH9	CCH8	CCH7	CCH6	CCH5	CCH4	CCH3	CCH2	CCH1	CCH0	
	ID code					Character address (1024 characters)											
COL (normal mode) Color control code	1	0	*	BSHAD1	BSHAD0	CSHAD	FRAME	BLINK	BCOL3	BCOL2	BCOL1	BCOL0	CCOL3	CCOL2	CCOL1	CCOL0	
	ID code			Box shadow		Char. shadow	Outline	Blink	Background color (16 colors)				Character color (16 colors)				
COL (closed-caption mode) Color control code	1	0	*	*	CUNDL	ITALIC	FRAME	BLINK	BCOL3	BCOL2	BCOL1	BCOL0	CCOL3	CCOL2	CCOL1	CCOL0	
	ID code				Underline	Italics	Outline	Blink	Background color (16 colors)				Character color (16 colors)				
CCB Repeat character/blank code	0	1	*	*	*	*	*	*	*	*	*	*	CCBF	CCB3	CCB2	CCB1	CCB0
	ID code											Blank/char.	Number of blank/char. repetitions				
CHP Character H position control	1	1	*	CHSZ1	CHSZ0	CSHT	CHP9	CHP8	CHP7	CHP6	CHP5	CHP4	CHP3	CHP2	CHP1	CHP0	
	ID code			H size		Shutter	H display start position (1 dot resolution, 1024 steps)										
CVP Character V position control	1	1	CLAST	CVSZ1	CVSZ0	CINT	CVP9	CVP8	CVP7	CVP6	CVP5	CVP4	CVP3	CVP2	CVP1	CVP0	
	ID code		Last line flag	V size		Interrupt	V display start position (1 H scan line resolution, 1024 steps)										
Graphics layer																	
GTC Graphic tile code	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
	0	GCBF	GCB3	GCB2	GCB1	GCB0	GPRT	GTC8	GTC7	GTC6	GTC5	GTC4	GTC3	GTC2	GTC1	GTC0	
	ID code		Blank/tile	Number of blank/tile repetitions			Palette select	Graphic tile address (512 tiles)									
GHP Graphics H position control	1	1	*	GHSZ1	GHSZ0	GSHT	GHP9	GHP8	GHP7	GHP6	GHP5	GHP4	GHP3	GHP2	GHP1	GHP0	
	ID code			H size		Shutter	H display start position (1 dot resolution, 1024 steps)										
GVP Graphics V position control	1	1	GLAST	GVSZ1	GVSZ0	GINT	GVP9	GVP8	GVP7	GVP6	GVP5	GVP4	GVP3	GVP2	GVP1	GVP0	
	ID code		Last line flag	V size		Interrupt	V display start position (1 H scan line resolution, 1024 steps)										

* Don't-care bits

■ Text Layer

CC: Character Code

ID Code: 00

CCH[9:0]

Specifies the address of one of 1024 characters stored in the ROM.

COL: Color Control Code (Normal Mode)

ID Code: 10

BSHAD[1:0]

Specifies shadowing of the character box for a 3D button effect.

00: Disable

01: Disable

10: Upper left white and lower right black shadows

11: Upper left black and lower right white shadows

CSHAD

Specifies character shadowing for a 3D effect.

0: Disable

1: Enable

FRAME

Specifies character outlining (black).

0: Disable

1: Enable

BLINK

Specifies character blinking.

- 0: Disable
- 1: Enable

BCOL[3:0]

Specifies the background color (1 of 16 colors).

CCOL[3:0]

Specifies the foreground (character) color (1 of 16 colors).

COL: Color Control Code (Closed-Caption Mode)

ID Code: 10

CUNDL

Specifies underlining.

- 0: Disable
- 1: Enable

ITALIC

Specifies italicization.

- 0: Disable
- 1: Enable

FRAME

Specifies character outlining (black).

- 0: Disable
- 1: Enable

BLINK

Specifies character blinking.

- 0: Disable
- 1: Enable

BCOL[3:0]

Specifies the background color (1 of 16 colors).

CCOL[3:0]

Specifies the foreground (character) color (1 of 16 colors).

CCB: Repeat Blank/Character Code

ID Code: 01

CCBF

Repeat blank/repeat character select.

- 0: Repeat blank
- 1: Repeat character

CCB[3:0]

Specifies the number of times (up to 16) a blank space or character is repeated. This function saves RAM space by preventing the VRAM address from incrementing. The program redisplay the preceding character code the specified number of times. This increases the limit beyond 38 characters per line.

CHP: Character Horizontal Position Control Code ID Code: 11

CHSZ[1:0]

Specifies the H size of the characters on the next line.

00: 1 dot = 1 VCLK period

01: 1 dot = 2 VCLK periods

10: 1 dot = 3 VCLK periods

11: 1 dot = 4 VCLK periods

CSHT

Specifies shutter operation for the next line. Setting this bit to 1 disables the shuttering function. You can disable and enable shuttering on a line-by-line basis.

0: Enable

1: Disable

CHP[9:0]

Specifies a VCLK indicating the horizontal start position for the next line. 1024 steps are available.

CVP: Character Vertical Position Control Code

ID Code: 11

CLAST

Specifies the last line in the internal RAM text layer. This resets the line pointer for character reads from the internal RAM to the first line.

0: Disable

1: Enable

CVSZ[1:0]

Specifies the V size of the characters on the next line.

00: 1 dot = 1 H scan line

01: 1 dot = 2 H scan lines

10: 1 dot = 4 H scan lines

11: 1 dot = 6 H scan lines

CINT

Specifies an OSD interrupt.

0: Disable

1: Enable

CVP[9:0]

Specifies an H scan line indicating the vertical start position for the next line. 1024 steps are available.

■ Graphics Layer**GTC: Graphic Tile Code**

ID Code: 00

GCBF

Repeat blank/repeat tile select.

0: Repeat blank

1: Repeat tile



In closed-caption mode, only the b'00', b'01', and b'11' settings are available for CVSZ[1:0]. The b'10' setting is reserved.

GCB[3:0]

Specifies the number of times (up to 16) a blank or graphic tile is repeated.

GPRT

Specifies graphics color palette 1 or 2.

0: Palette 1

1: Palette 2

GTC[8:0]

Specifies the address of one of 512 graphic tiles stored in the ROM.

GHP: Graphics Horizontal Position Control Code

ID Code: 11

GHSZ[1:0]

Specifies the H size of the characters on the next line.

00: 1 dot = 1 VCLK period

01: 1 dot = 2 VCLK periods

10: 1 dot = 3 VCLK periods

11: 1 dot = 4 VCLK periods

GSHT

Specifies shutter operation for the next line. Setting this bit to 1 disables the shuttering function. You can disable and enable shuttering on a line-by-line basis.

0: Enable

1: Disable

GHP[9:0]

Specifies a VCLK indicating the horizontal start position for the next line. 1024 steps are available.

GVP: Graphics Vertical Position Control Code

ID Code: 11

GLAST

Specifies the last line in the internal RAM graphics layer. This resets the line pointer for graphic tile reads from the internal RAM to the first line.

0: Disable

1: Enable

GVSZ[1:0]

Specifies the V size of the tiles on the next line.

00: 1 dot = 1 H scan line

01: 1 dot = 2 H scan lines

10: 1 dot = 4 H scan lines

11: 1 dot = 6 H scan lines

GINT

Specifies an OSD interrupt.

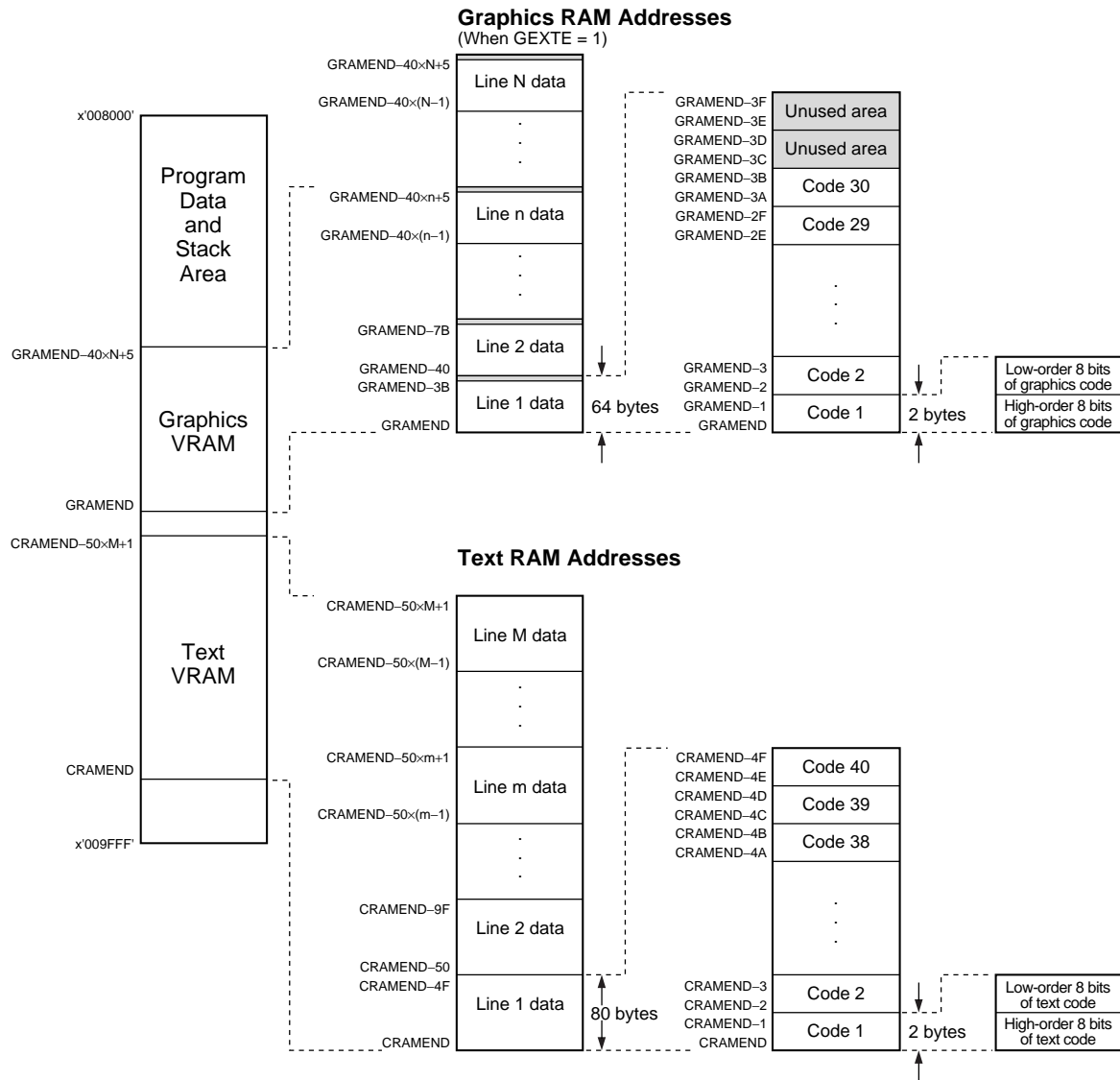
0: Disable

1: Enable

GVP[9:0]

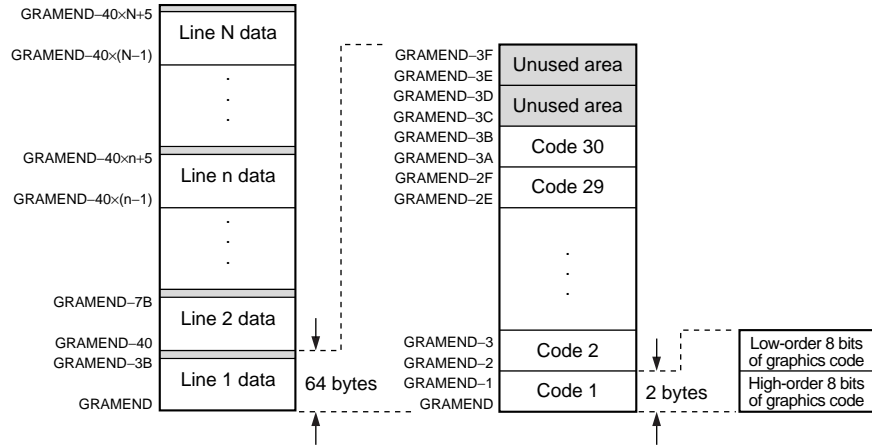
Specifies an H scan line indicating the vertical start position for the next line. 1024 steps are available.

7.8.2 VRAM Organization

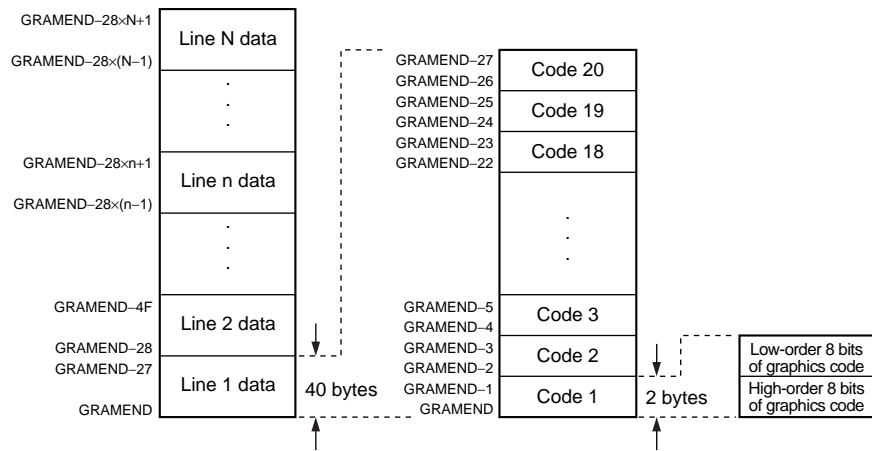


- Notes:
1. All addresses are expressed in hex notation. Other values are decimal.
 2. GRAMEND: Graphics RAM end address (programmable to any address)
 3. CRAMEND: Text RAM end address (programmable to any address)
 4. M: Number of lines in the text layer
 5. m: 1 and up
 6. N: Number of lines in the graphics layer
 7. n: 1 and up

Figure 7-6 VRAM Organization (When GEXTE = 0)



A. GEXTE = 1



B. GEXTE = 0

Figure 7-7 Graphics VRAM Organization for Two Modes

7.8.3 Cautions about the number of display code set to VRAM

When the display lines are adjoined or overlapped, and the number of the above display code is extremely fewer than that of the below one, first line of the display line may not be output correctly.

In OSD circuit, font data to display are read from ROM and stored to the buffer. The data in the buffer are overwritten, after the present display code is output, to the display data of the next line.

Show the example when the display lines are adjoined. The number of the display code in the above line (line M) is referred to n, and that of the below one (line M+1) is referred N.

The buffer holds n display data of the last line of line M at the time of A (see figure below), and the data in the buffer are overwritten to the display data of the first line of line M+1 as time goes by. At the time of B the buffer holds n display data of the first line of line M+1.

In case n is smaller than N, N-n display data are needed to display the first line of line M+1, and to be written to the buffer before it is shown on the display.

Especially when n is extremely smaller than N, a large amount of the display data is needed to be written to the buffer before display. And in case writing is not done in time, it shows wrong data on the screen.

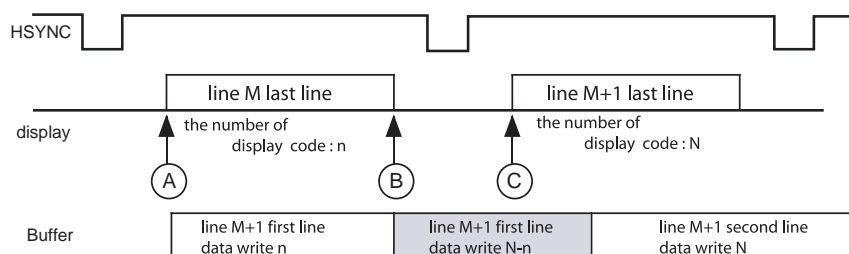


Figure 7-8 Timing for OSD data

To prevent the wrong display written above, the first line of the below line is to be shown later enough after the last line of the above line is displayed.

The time (T_d) needed for the correct display is calculated as below.

$$T_d = 0.8 + (N - n) \times 0.5 \text{ } [\mu\text{s}]$$

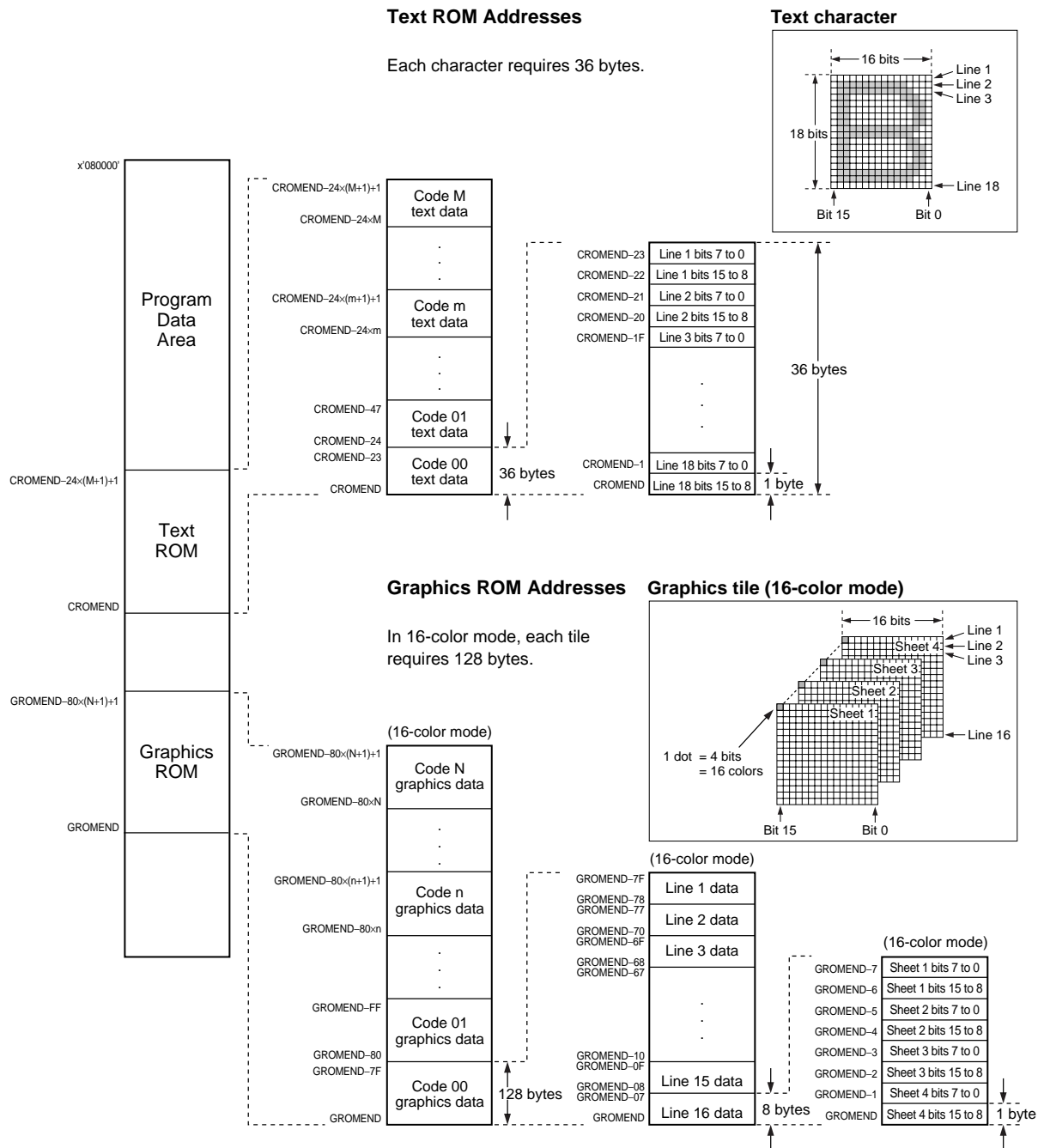
n : the number of display code in the above line. (include HP, VP)

N: the number of display code in the below line. (include HP, VP)

$n < N$

7.9 ROM

7.9.1 ROM Organization



- Notes:
1. All addresses are expressed in hex notation. Other values are decimal.
 2. GROMEND: Graphics ROM end address (programmable to any address)
 3. CROMEND: Text ROM end address (programmable to any address)
 4. M: Number of characters - 1
 5. m: 0 and up
 6. N: Number of graphic tiles - 1 (16-color mode)
 7. n: 0 and up

Figure 7-9 ROM Organization

7.9.2 Graphics ROM Organization in Different Color Modes

The graphics layer supports up to sixteen colors, in the 16-color mode, but also supports 2-, 4-, and 8-color modes. The smaller the number of colors, the less ROM area required per tile. The figures in this section illustrate the ROM organization for each color mode.

The example in figure 7-10 demonstrates the graphics ROM setup for line 16 of the code 00 data when the graphics layer is in 16-color mode. The four bits of data for each pixel, in sheets 1, 2, 3, and 4, determine the color palette used for that pixel.

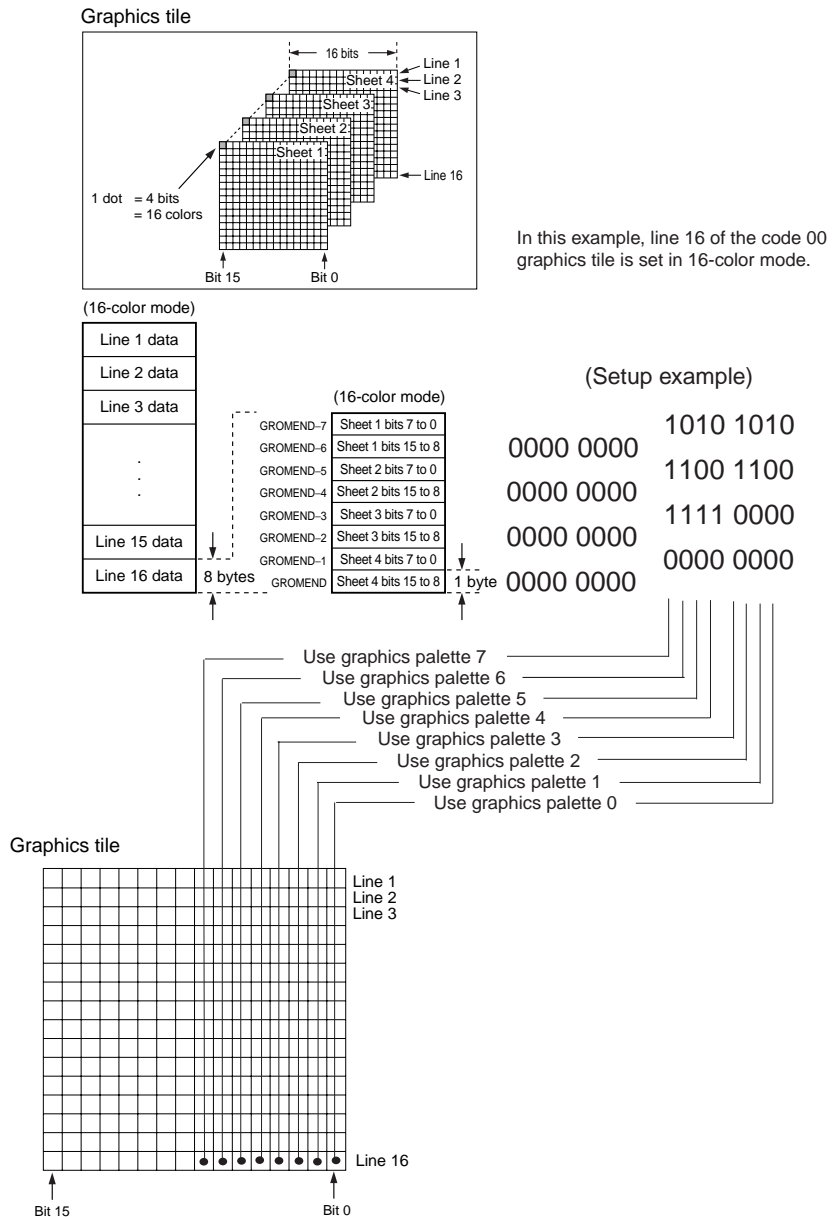


Figure 7-10 Graphics ROM Setup Example for a Single Line

Graphic Tile Codes					
	2 colors	4 colors	8 colors	16 colors	
ROMEND-80xN+1	$N \times 4 - 1$	$N \times 2 - 1$	$N \times 4/3 - 1$	$N - 1$	N is a multiple of 3.
	
	
	
			04	03	
		06			
ROMEND-180	0C				
ROMEND-160	0B				
		05			
ROMEND-140	0A		03		
				02	
ROMEND-120	09				
		04			
ROMEND-100	08				
			02		
ROMEND-E0	07				
		03			
ROMEND-C0	06				
				01	
ROMEND-A0	05				
		02			
ROMEND-80	04		01		
ROMEND-60	03				
		01			
ROMEND-40	02				
				00	
ROMEND-20	01		00		
		00			
ROMEND	00				

128 bytes

Required bytes per tile

32 bytes	64 bytes	96 bytes	128 bytes
↓	↓	↓	↓
See fig.7-15	See fig.7-14	See fig.7-13	See fig.7-12

Figure 7-11 Graphics ROM in the Four Color Modes (16W x 16H Tiles)

Graphic Tile Codes					
	2 colors	4 colors	8 colors	16 colors	
ROMEND-90xN+1	$N \times 4 - 1$	$N \times 2 - 1$	$N \times 4/3 - 1$	$N - 1$	N is a multiple of 3.
	
	
	
			04	03	
		06			
ROMEND-1B0	0C				
ROMEND-18C	0B				
		05			
ROMEND-168	0A		03		
				02	
ROMEND-144	09				
		04			
ROMEND-120	08				
			02		
ROMEND-FC	07				
		03			
ROMEND-D8	06				
				01	
ROMEND-B4	05				
		02			
ROMEND-90	04		01		
ROMEND-6C	03				144 bytes
		01			
ROMEND-48	02				
				00	
ROMEND-24	01		00		
		00			
ROMEND	00				

Required bytes per tile	36 bytes	72 bytes	108 bytes	144 bytes
	↓	↓	↓	↓
	See fig.7-19	See fig.7-18	See fig.7-17	See fig.7-16

Figure 7-12 Graphics ROM in the Four Color Modes (16W x 18H Tiles)

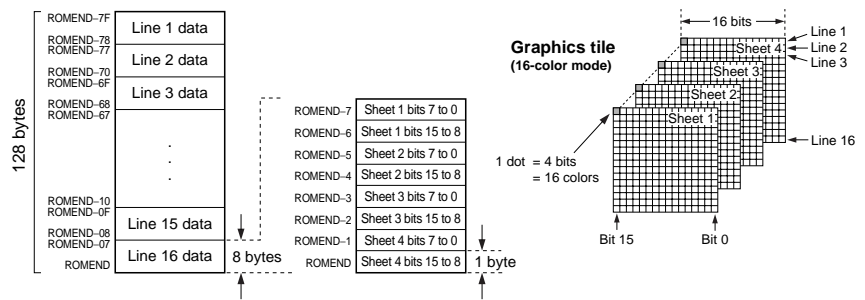


Figure 7-13 Graphics ROM Organization in 16-Color Mode (16W x 16H Tiles)

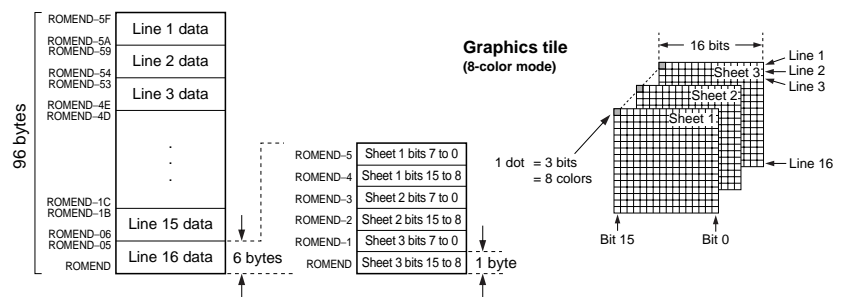


Figure 7-14 Graphics ROM Organization in 8-Color Mode (16W x 16H Tiles)

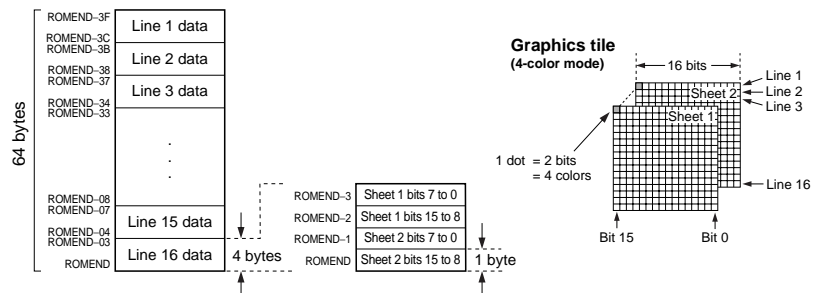


Figure 7-15 Graphics ROM Organization in 4-Color Mode (16W x 16H Tiles)

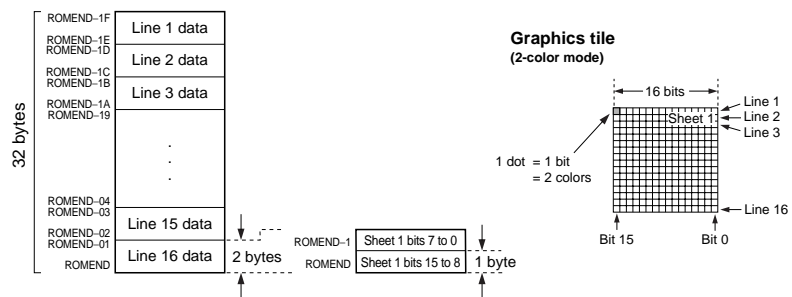


Figure 7-16 Graphics ROM Organization in 2-Color Mode (16W x 16H Tiles)

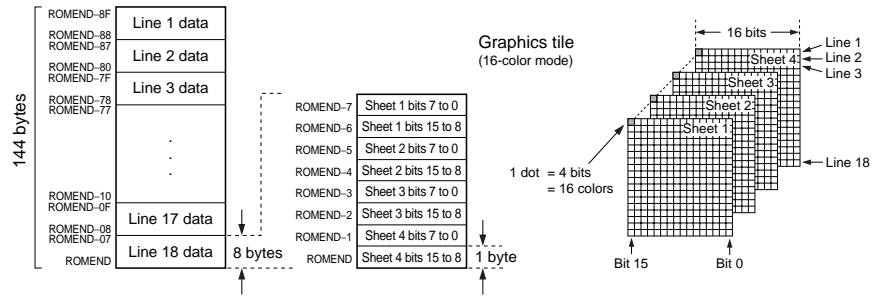


Figure 7-17 Graphics ROM Organization in 16-Color Mode (16W x 18H Tiles)

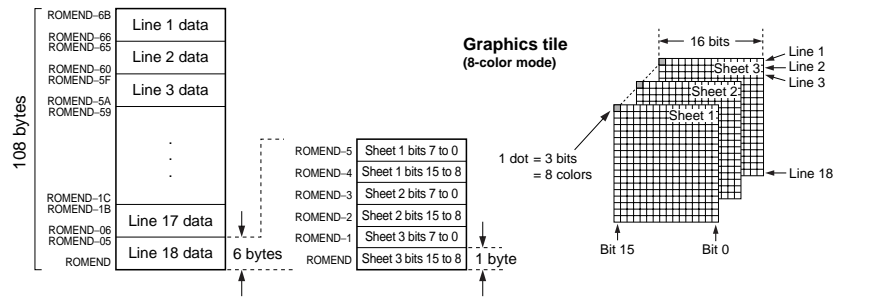


Figure 7-18 Graphics ROM Organization in 8-Color Mode (16W x 18H Tiles)

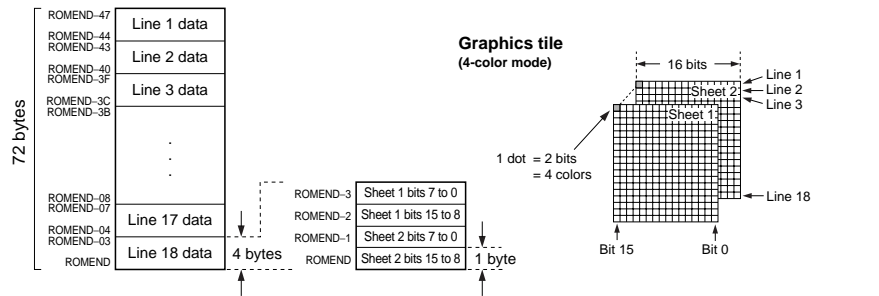


Figure 7-19 Graphics ROM Organization in 4-Color Mode (16W x 18H Tiles)

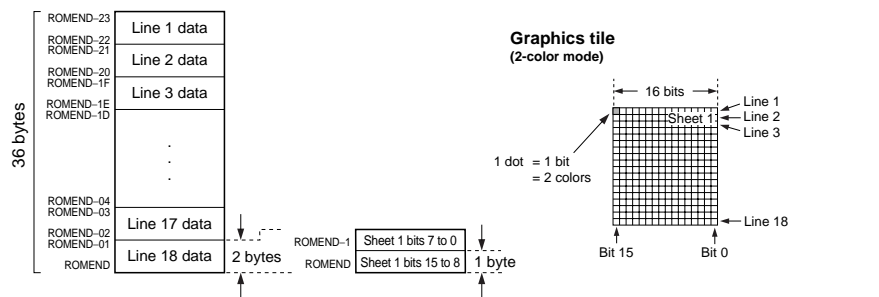


Figure 7-20 Graphics ROM Organization in 2-Color Mode (16W x 18H Tiles)

7.10 Setting Up the OSD

7.10.1 Setting Up the OSD Display Colors

This section describes how to set up the display colors for the OSD.

■ **To set up the color palettes:**

Write your settings to the color palette registers shown in table 7-8.

Table 7-8 Color Palette Registers

Layer	Palette	Address	Applications
Text	CPT0–CPTF	x'007F80'– x'007F9E'	Text foreground and background colors
All layers	COLB	x'007FA0'	Color background
Text	FRAME	x'007FA2'	Outlining and character shadowing colors
	BBSHD	x'007FA4'	Box shadowing color (black)
	WBSHD	x'007FA6'	Box shadowing color (white)
Graphics, cursor	GPT10– GPT1F	x'007FC0'– x'007FDE'	Tile color palette 1
	GPT20– GPT2F	x'007FE0'– x'007FFE'	Tile color palette 2

■ **To set up the cursor display colors:**

Write to the fields described below.

- ◆ **GCOL[1:0]** (x'007F08', bits 9 and 8) sets the number of colors (2, 4, 8, or 16).
- ◆ **STC[8:0]** (x'007F10', bits 8 to 0) specifies the code of the tile to be displayed.
- ◆ **SPRT** (x'007F10', bit 9) selects tile color palette 1 or 2.
- ◆ **GPT1n** (x'007FC0'–x'007FDE) or **GPT2n** (x'007FE0'–x'007FFE) specifies the colors on the palettes corresponding to the tile data stored in the ROM.

■ **To set up the graphics display colors:**

Write to the fields described below.

- ◆ **GCOL[1:0]** (x'007F08', bits 9 and 8) sets the number of colors (2, 4, 8, or 16).
- ◆ **GTC[8:0]** (GTC bits 8 to 0 in the RAM data) specifies the code of the tile to be displayed.
- ◆ **GPRT** (GTC bit 9 in the RAM data) selects tile color palette 1 or 2.
- ◆ **GPT1n** (x'007FC0'–x'007FDE) or **GPT2n** (x'007FE0'–x'007FFE) specifies the colors on the palettes corresponding to the tile data stored in the ROM.

■ To set up the text display colors:

Write to the fields described below.

- ◆ **CCOL[3:0]** (COL bits 3 to 0 in the RAM data) sets the color of the character. This value is in reference to the selected color palette (CPT0–CPTF).
- ◆ **BCOL[3:0]** (COL bits 7 to 4 in the RAM data) sets the background color. As with CCOL, this value is in reference to the selected color palette (CPT0–CPTF).
- ◆ **FRAME** (COL bit 9 in the RAM data) enables character outlining when set to 1. Set the outline color in the FRAME color palette (x'007FA2').
- ◆ **CSHAD** (COL bit 10 in the RAM data) enables character shadowing when set to 1. Set the shadowing color in the FRAME color palette (x'007FA2'). This function is unavailable in the closed-caption mode.
- ◆ **BSHAD[1:0]** (COL bits 12 to 11 in the RAM data) enables character box shadowing. This function is unavailable in the closed-caption mode.
 - 00 and 01: No box shadowing
 - 10: Upper left white and lower right black shadows
 - 11: Upper left black and lower right white shadows
- ◆ **CPT0–CPTF** (x'007F80'–x'007F9E') specifies the colors available for text foreground and background colors.
- ◆ **FRAME** (x'007FA2') specifies the color for outlining or character shadowing.
- ◆ **BBSHD** (x'007FA4') specifies the “black” color for box shadowing.
- ◆ **WBSHD** (x'007FA6') specifies the “white” color for box shadowing.

■ To set up functions applying to all layers:

Write to the fields described below.

Color background function

The color background function allows you to fill the television screen areas that are uncovered by the OSD display (text, graphics, or cursor layers) with any color. Specify the color in the COLB register (x'007FA0').

- ◆ **COLB** (x'007F08', bit 7) enables the color background function when set to 1.
- ◆ **COLB** (x'007FA0') specifies the color of the background.

Transparency

The TRPT bit allows you to make the color 0 transparent in all the palettes (CPT0, GPT10, and GPT20). RGB, YM, and YS are all output at 0 levels for that color. See figure 7-21.

- ◆ **TRPT** (x'007F08', bit 10)
 - 0: Make color 0 on all palettes transparent
 - 1: Output color 0 as specified



Selecting YS palette output, by setting the YSPLT bit of OSD1 (x'007F06') to 1, disables the PRYM bit. With this setting, you must also set the TRPT and TRPTF bits to 1. You can specify transparency for individual color palettes if needed.

Translucency

The TRPTF bit allows you to make the color 15 translucent in all the palettes (CPTF, GPT1F, and GPT2F). RGB and YS are output at low levels for that color, and YM is output at the level specified for the palette. This dims the color on the display. See figure 7-21.

- ◆ **TRPTF** (x'007F08', bit 10)
 - 0: Make color 15 on all palettes translucent
 - 1: Output color 15 as specified

The PRYM bit allows you to make specific regions of the OSD display translucent. This bit is disabled when the YSPLT bit is 1.

- ◆ **PRYM** (x'007F08', bit 12)
 - 0: Output YS high on all OSD display areas
 - 1: Output YS low on OSD display areas that do not have low YM output

YS palette output

The YSPLT bit allows you to control the YS output. See figure 7-22.

- ◆ **YSPLT** (x'007F06', bit 3)
 - 0: Output YS to entire display area (except transparent and translucent areas)
 - 1: Output the MSB (bit 15) of all the color palettes from YS

Analog/digital output

The YCNT and RGBC bits allow you to select 16-bit gradient, analog or digital output through the four OSD output pins, R, G, B, and YM.

- ◆ **YCNT** (x'007F06', bit 1)
 - 0: Analog YM output
 - 1: Digital YM output (outputs bit 12 of color palette)
- ◆ **RGBC** (x'007F06', bit 0)
 - 0: Analog R, G, B output
 - 1: Digital R, G, B output (outputs bits 0 (R), 4 (G), and 8 (B) of color palette)

Table 7-9 summarizes the controls for RGM, YM, and YS output.

On-Screen Display

Setting Up the OSD

Table 7-9 RGB, YM, and YS Output Control Settings

YSPLT	PRYM	TPRT	TRPTF	RGB	YM	YS	Waveform in figure 7-21
0	0	0	0	Color palettes 0 and F output low	Color palettes 0 and F output low	Color palettes 0 and F output low	①
0	0	0	1	Color palette 0 output low	Color palette 0 output low	Color palette 0 output low	②
0	0	1	0	Color palette F output low	Color palette F output low	Color palette F output low	③
0	0	1	1	—	—	—	④
0	1	0	0	Color palettes 0 and F output low	Color palettes 0 and F output low	Color palettes 0 and F output low when YM ≠ low	⑤
0	1	0	1	Color palette 0 output low	Color palette 0 output low	Color palettes 0 output low when YM ≠ low	⑥
0	1	1	0	Color palette F output low	Color palette F output low	Color palettes F output low when YM ≠ low	⑦
0	1	1	1	—	—	Output low when YM ≠ low	⑧
1	0	0	0	Reserved	—	—	
1	0	0	1	Reserved	—	—	
1	0	1	0	Reserved	—	YM3 output	
1	0	1	1	—	—		⑨
1	1	0	0	Reserved	—	—	
1	1	0	1	Reserved	—	—	
1	1	1	0	Reserved	—	—	
1	1	1	1	—	—	YM3 output	⑨

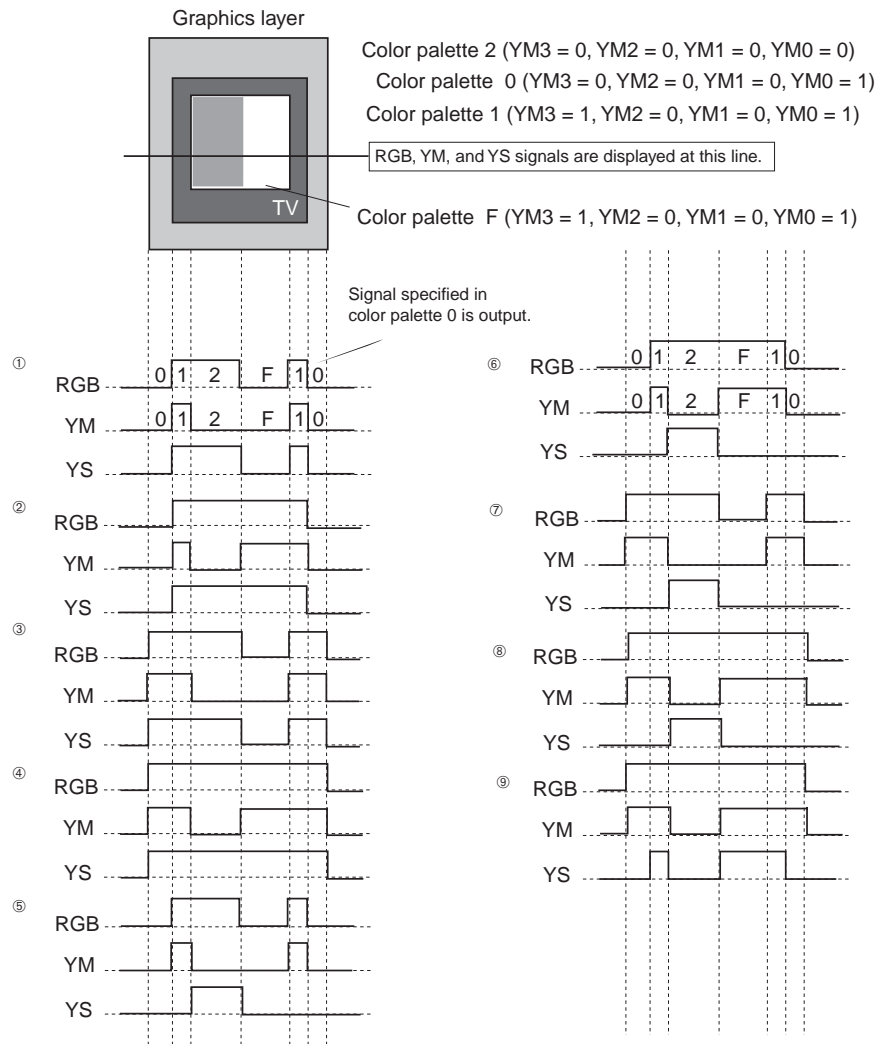


Figure 7-21 OSD Signal Waveform

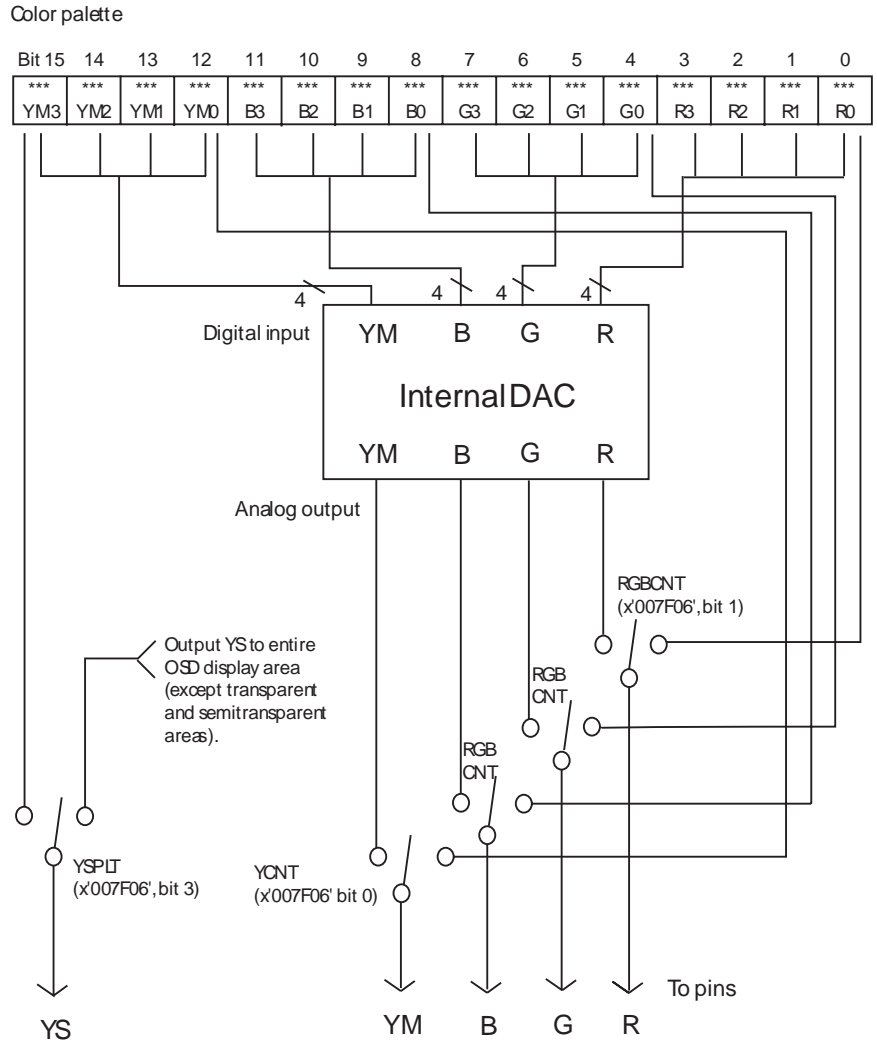


Figure 7-22 OSD Signal Output Switches

7.10.2 Text Layer Functions

This section describes the character enhancement functions available in the text layer.

■ Outlining

In both normal and closed-caption modes, writing a 1 to bit 9 (FRAME) of the COL setting in the VRAM causes an outline to appear around all characters following that COL. You can specify the color of the outline in the FRAME register (x'007FA2'). Figure 7-23 shows an example of character outlining. As shown in the figure, if a character contains dots in the left or right borders of its field, the outlining for those dots appear in the adjacent character field.

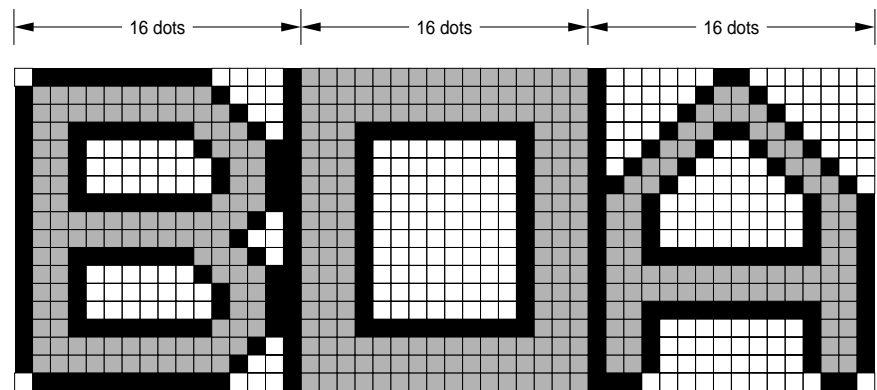


Figure 7-23 Character Outlining Example

■ Character shadowing

In normal mode, writing a 1 to bit 10 (CSHAD) of the COL setting in the VRAM causes a drop shadow to appear behind all characters following that COL. You can specify the color of the shadow in the FRAME register (x'007FA2'). Figure 7-24 shows an example of character shadowing. As shown in the figure, if a character contains dots in right border of its field, the shadowing for those dots appear in the character field to the right.

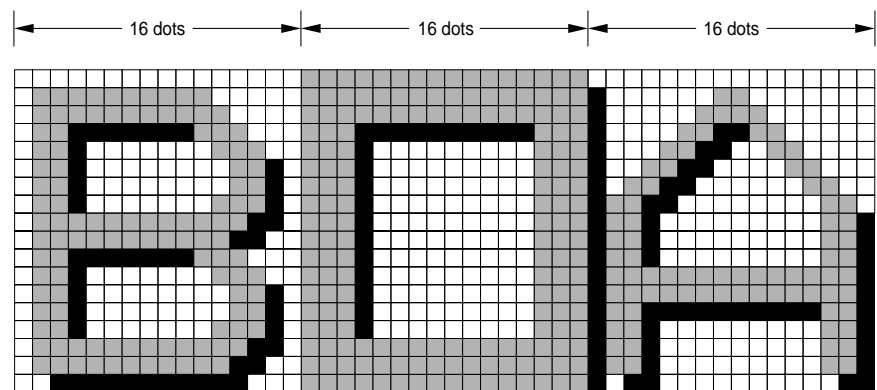


Figure 7-24 Character Shadowing Example

■ Box shadowing

In normal mode, writing a 1 to bit 12 (BSHAD1) of the COL setting in the VRAM causes a box shadow to appear around all characters following that COL. If COL bit 11 (BSHAD0) is 0, the color specified in the WBSHD register (x'007FA6') appears on the top and left sides of the box and the color specified in the BBSHD register (x'007FA4') appears on the bottom and right sides of the box. These positions are reversed if BSHAD0 is 1. Figure 7-25 shows an example of box shadowing. As shown in the figure, the right-hand border of the shadow box appears in the character field to the right of the shadowed text.



To output a box shadow to the right, output a space to the right of the text.

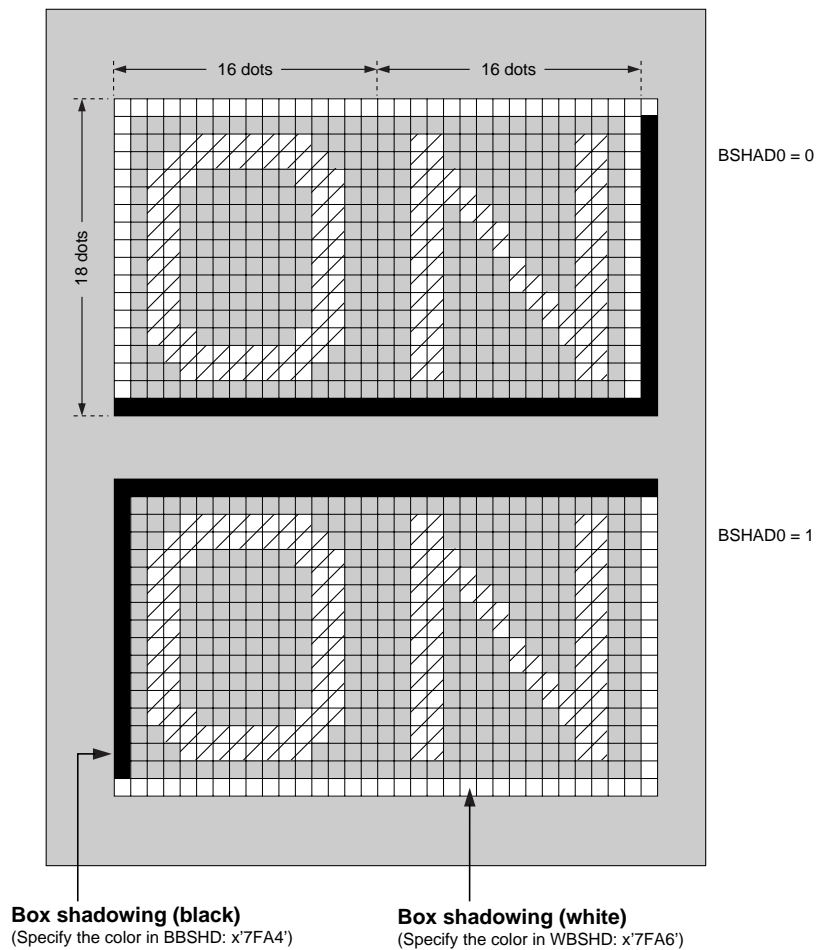


Figure 7-25 Box Shadowing Example

■ **Italicizing**

In closed-caption mode, writing a 1 to bit 10 (ITALIC) of the COL setting in the VRAM italicizes all characters following that COL. Figure 7-26 shows an example of an italicized character.

■ **Underlining**

In closed-caption mode, writing a 1 to bit 11 (CUNDL) of the COL setting in the VRAM underlines all characters following that COL. Figure 7-26 shows an example of an underlined character.

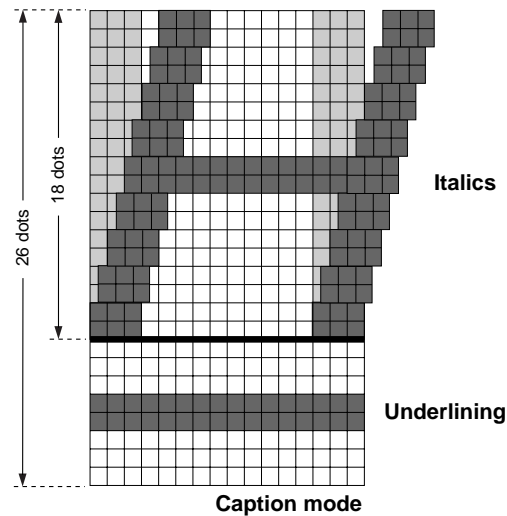


Figure 7-26 Italicizing and Underlining Example

■ **Blinking**

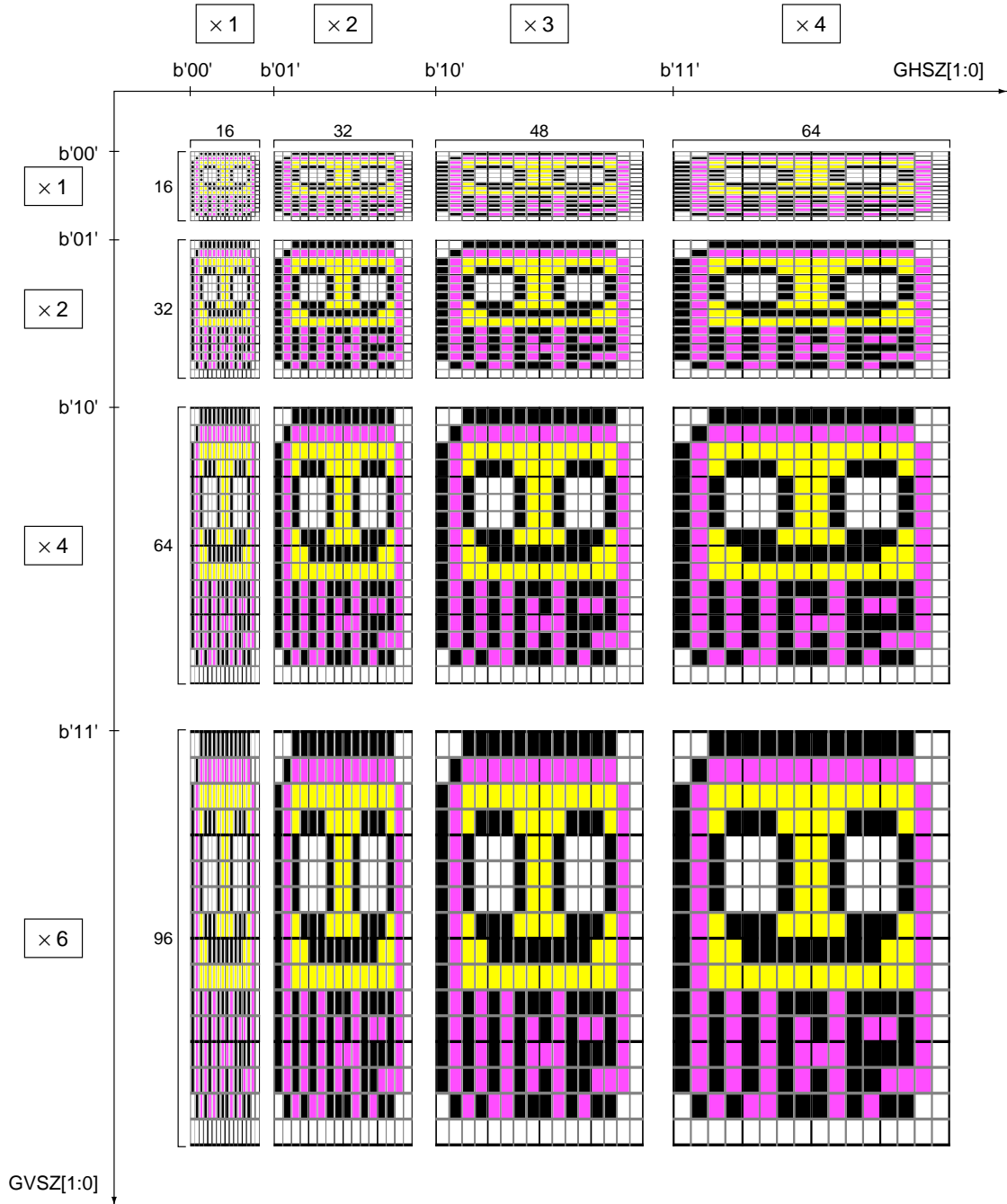
In both normal and closed-caption modes, writing a 1 to bit 8 (BLINK) of the COL setting in the VRAM causes all characters following that COL to blink. To use this function, you must enable blinking by writing a 1 to bit 5 (BLINK) of the OSD3 register (x'007F0A').

In closed-caption mode, you can specify whether or not the underlines blink on underlined, blinking characters. Set bit 0 (UNDF) of the OSD3 register to 0 to disable underline blinking, and set it to 1 to enable underline blinking.

The blink cycle lasts for 128 VSYNC pulses (about 2 seconds). The characters display for 96 VSYNCs (about 1.5 seconds) and turn off for 32 VSYNCs (about 0.5 seconds).

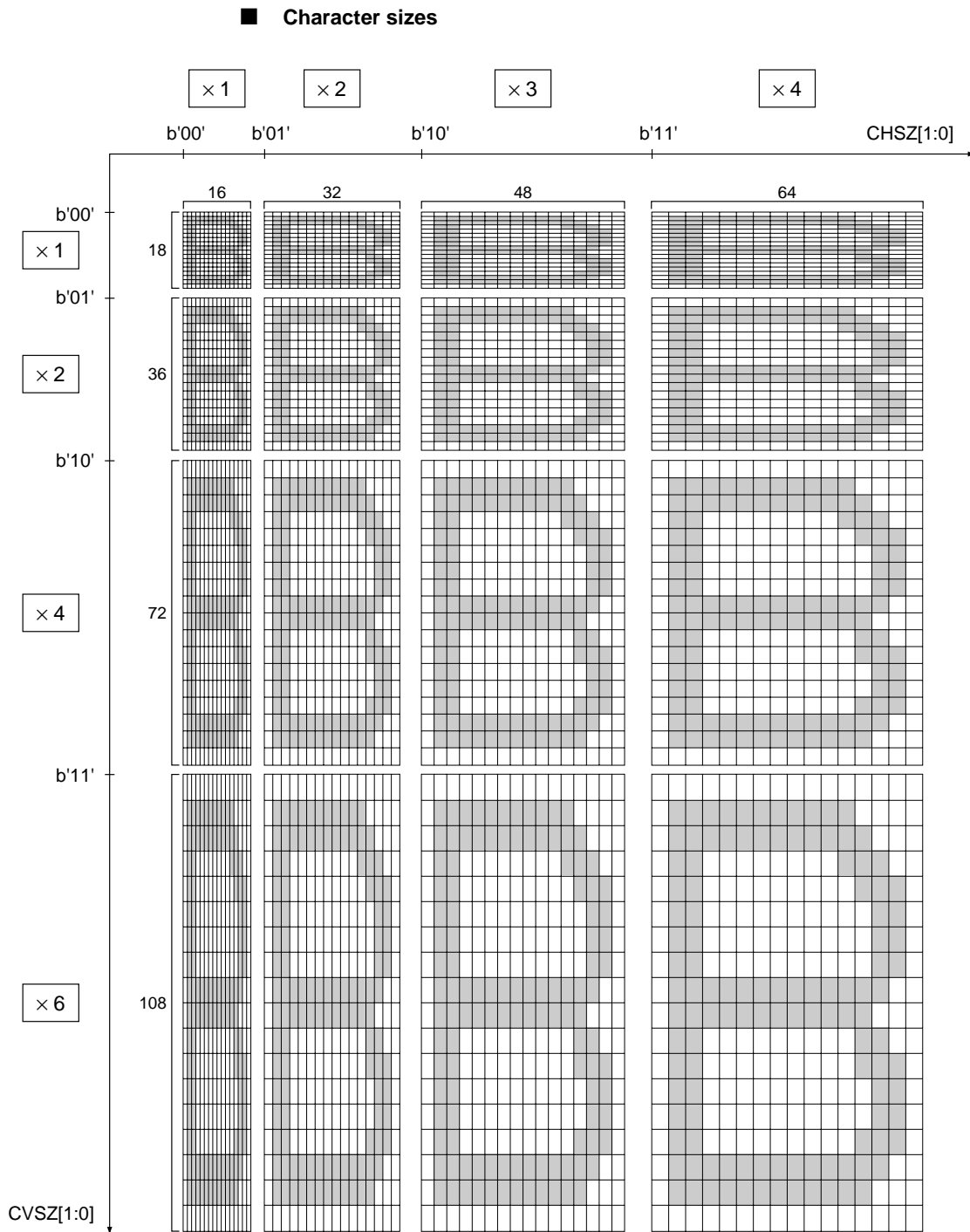
7.10.3 Display Sizes

■ **Graphic tile sizes**



The settings shown are for interlaced displays. In progressive displays, the vertical size settings (GVSZ[1:0]) are as follows: 01 = 1x, 10 = 2x, and 11 = 3x. The 00 setting is reserved.

Figure 7-27 Graphic Tile Size Combinations



The settings shown are for interlaced displays. In progressive displays, the vertical size settings (CVSZ[1:0]) are as follows: 01 = 1x, 10 = 2x, and 11 = 3x. The 00 setting is reserved. In addition, in closed-caption mode, only the b'00', b'01', and b'11' settings are available for CVSZ[1:0]. The b'10' setting is reserved.

Figure 7-28 Character Size Combinations

7.10.4 Setting Up the OSD Display Position

This section describes how to control the positioning of the OSD.

■ **To set up the horizontal position:**

Cursor

- ◆ Write the horizontal position of the cursor to the SHP[9:0] field (x'007F12').
- ◆ Valid range: $SHP \geq x'0C'$

Graphics

- ◆ Write the horizontal position of the first line in the display to the GIHP[9:0] field (x'007F16').
- ◆ Write the position of the second and all following lines in the GHP[9:0] field of the graphics display RAM data for the preceding line.
- ◆ Valid ranges: $x'0C' \leq GHP \leq HP_{max}$ and $x'0C' \leq GIHP \leq HP_{max}$

Text

- ◆ Write the horizontal position of the first line in the display to the CIHP[9:0] field (x'007F1A').
- ◆ Write the position of the second and all following lines in the CHP[9:0] field of the text display RAM data for the preceding line.
- ◆ Valid ranges: $x'0C' \leq CHP \leq HP_{max}$ and $x'0C' \leq CIHP \leq HP_{max}$

■ **To set up HP_{max} equations, write:**

- ◆ $HP_{max} = (T_{hsync} - T_{hw} - 0.8 \mu s) / T_{dot} - (N_{char.} \times 16 \times H_{sz})$; or
- ◆ $HP_{max} = 1024 - (N_{char.} \times 16 \times H_{sz})$



When setting up the horizontal position, you must allow at least 0.8 μs between the end of a line and the leading edge of HSYNC, or the display will flicker.

T_{hsync} is the HSYNC cycle, T_{hw} is the HSYNC pulse width, $N_{char.}$ is the number of characters in the line including repeated characters and blank spaces, T_{dot} is the dot clock cycle, and H_{sz} is the horizontal size. The HP_{max} limit ensures that there is at least 0.8 μs between the end of a line and the leading edge of HSYNC.

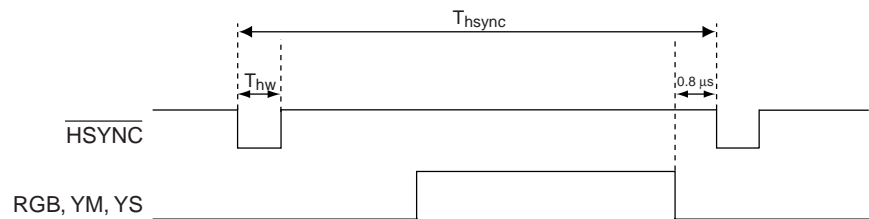


Figure 7-29 HP_{max} of Horizontal Display Position

■ **About the horizontal start position on the screen**

The horizontal position, or HP settings (SHP, GHP, and CHP) determine where the left side of cursor, graphics, and text lines start on the screen. You can set this value for all of the layers in 1-pixel units.

■ To set up the vertical position:

Cursor

- ◆ Write the vertical position of the cursor to the SVP[9:0] field (x'007F14').
- ◆ Valid range: $x'3F0' - (\text{no. of H scan lines}) \geq \text{SHP} \geq x'03'$



When you write new values to the GIVP and CIVP fields, the settings take effect on the next VSYNC pulse. This means that changes are reflected in the next display screen rather than the current one.

Graphics

- ◆ Write the vertical position of the first line in the display to the GIVP[9:0] field (x'007F18').
- ◆ Write the position of the second and all following lines in the GVP[9:0] field of the graphics display RAM data for the preceding line.
- ◆ Valid range: $x'3F0' - (\text{no. of H scan lines}) \geq \text{GIVP/GVP} \geq x'03'$

Text

- ◆ Write the vertical position of the first line in the display to the CIVP[9:0] field (x'007F1C').
- ◆ Write the position of the second and all following lines in the CVP[9:0] field of the text display RAM data for the preceding line.
- ◆ Valid ranges: $x'3F0' - (\text{no. of H scan lines}) \geq \text{CIVP/CVP} \geq x'03'$

VP range calculation example

The base graphics line height is 16 dots, or H scan lines. If the graphics line you are positioning displays at 2x the base height, the number of H scan lines is:

$$16 \times 2 = 32 = x'20' \text{ H scan lines}$$

The valid range of settings for GVP is:

$$x'3F0' - x'20' = x'3D0' \geq \text{GVP} \geq x'03'$$

■ About the vertical start position on the screen

The vertical position, or VP settings (SVP, GVP, and CVP) determine where the upper edge of cursor, graphics, and text lines start on the screen. You can set this value for all of the layers in H scan line units. Using the same VP settings for all the layers makes them all start at the same vertical position.

7.11 DMA and Interrupt Timing

This section describes how the MN102H75K/85K handles the timing of direct memory access (DMA) transfers of OSD data and OSD interrupts.

■ DMA



If you use the OSD function, the DMA function executes for both the text and graphics layers, even if your program does not use one of these layers. To prevent error, program data for the unused layer to meet the restrictions outlined in section 7.1, "Description," on page 153.

On both the text and graphics layers, the microcontroller reads the line 1 data from the RAM as it scans line 1 onto the display. For line 2 and following lines, it reads the data as it scans the display start for the preceding line. The RAM read starts 12 system clock cycles ($12T_S$) after the leading edge of the HSYNC pulse. The DMA transfer takes $4T_S$ for each display data word.

In line 1, or when a graphics and text line begin simultaneously, the data transfer requests for both layers occur simultaneously. The text data transfer always takes priority. The graphics data transfer begins $5T_S$ after the text data transfer ends.

If a DMA transfer occurs at the same time as the leading edge of a VSYNC pulse, the screen flickers. To avoid this, do not set a display position in the last line.

■ Interrupts

For both graphics and text displays, the microcontroller processes the GINT and CINT interrupt request bits of the display data's GVP and CVP fields during DMA transfer. If GINT or CINT is set to 1, when the associated transfer ends (GVP or CVP transfer) the OSD generates an interrupt request.

Note that if the interrupt bit is set to 1 in the line 1 display data, the interrupt occurs at the first scan line. If the interrupt bit is set to 1 in the line 2 display data, the interrupt occurs at the first display line.

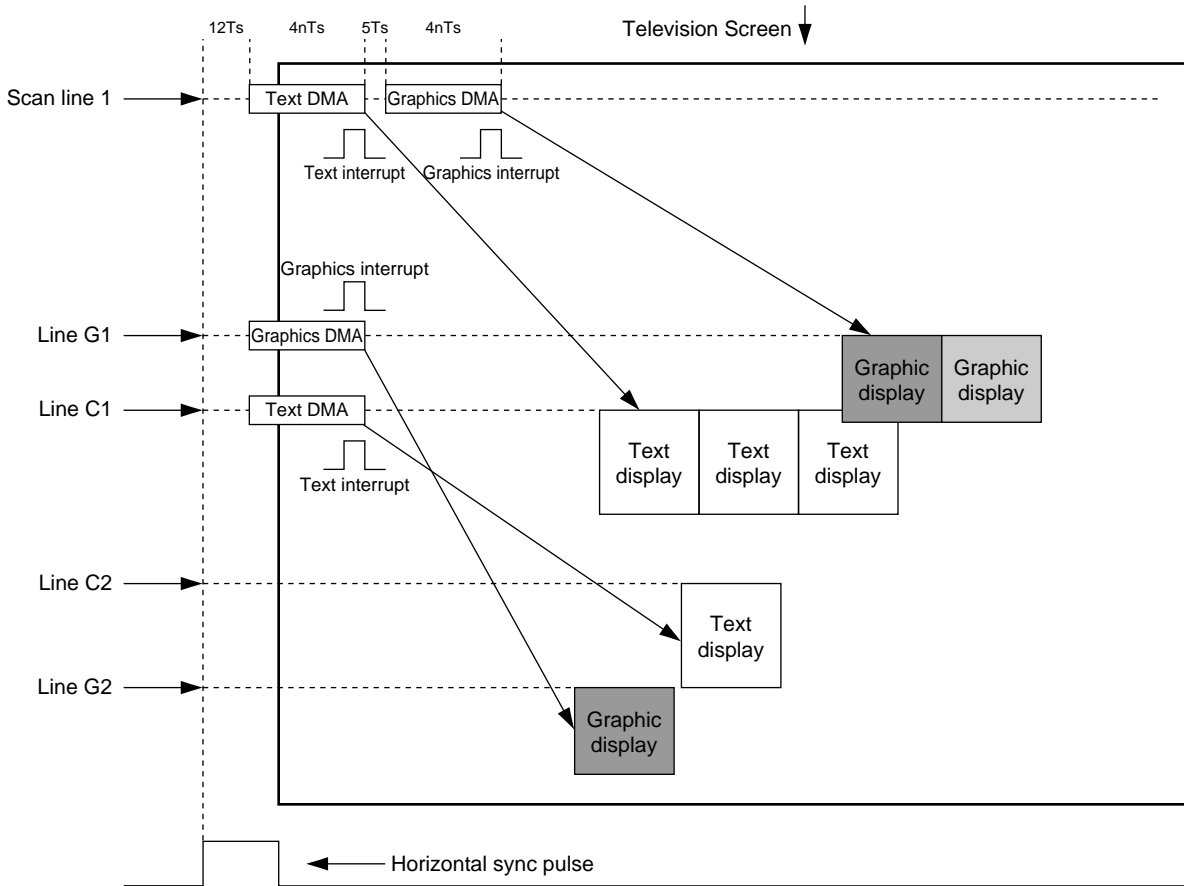


Figure 7-30 DMA and Interrupt Timing for the OSD

7.12 Selecting the OSD Dot Clock

This section describes how to set up the OSD dot clock.

■ Selecting the clock source

The source for the OSD dot clock is programmable to either the 4-MHz clock supplied through the OSC1 and OSC2 pins, then multiplied by the PLL circuit to 48 MHz, or a dedicated clock supplied through the OSDXI and OSDXO pins.

The dedicated OSDX clock can be a crystal, LC oscillator, or other form of excitation that is input through the OSDXI pin and synchronized internally to the HSYNC pulse, or it can be an LC blocking oscillator synchronized to the HSYNC pulse.

Table 7-10 OSD Dot Clock Source Settings

OSCSSEL1 (x'007F06', bit 9)	OSCSSEL0 (x'007F06', bit 8)	Oscillator pins	Oscillator type	Frequency
0	0	OSC1, OSC2	Crystal + PLL	48-MHz (with 4-MHz osc.)
0	1	OSDXI, OSDXO	LC blocking oscillator	f_X
1	0	OSDXI, OSDXO	Crystal, LC oscillator, or other excitation	f_X
1	1	Reserved	—	—

If your design uses the OSDX clock, set the XIO bit (x'007F06', bit 7) for the appropriate frequency.

◆ XIO

0: Less than 20 MHz

1: Greater than 20 MHz

■ Selecting the divide-by ratio

There are five divide-by settings available for any of the clocks described above. Table 7-11 shows the register settings for each ratio.

Table 7-11 OSD Dot Clock Division Settings

VCLK2 (x'007F08', bit 6)	VCLK1 (x'007F08', bit 5)	VCLK0 (x'007F08', bit 4)	Divide by Ratio
0	0	0	1/4 (12 MHz) ⁽¹⁾
0	0	1	1/3 (16 MHz) ⁽¹⁾
0	1	0	1/2 (24 MHz) ⁽¹⁾
0	1	1	2/3 (32 MHz) ⁽¹⁾
1	0	0	1/1 (48 MHz)
1	0	1	Reserved
1	1	0	Reserved
1	1	1	Reserved

Notes: 1. This is the frequency with a 48-MHz clock source.

7.13 Controlling the Shuttering Effect

The MN102H75K/85K OSD achieves a shuttering effect using four programmable shutters—two vertical and two horizontal. With this feature, you can shutter any portion of the OSD display, or you can combine shuttering with a wipe-out effect to create a smooth appearing and disappearing effect.

To prevent flickering and shadows on the display, only write to the registers during the VSYNC cycle.

7.13.1 Controlling the Shuttered Area

The register settings for the two vertical shutters, VSHT0 and VSHT1, and the two horizontal shutters, HSHT0 and HSHT1, control which area of the screen is shuttered. Table 7-12 shows the register settings required for this function, and figure 7-31 shows four setup examples.

Table 7-12 Bit Settings for Controlling the Shuttered Area

Function	VSHT0 Bit	VSHT1 Bit	HSHT0 Bit	HSHT1 Bit	Description
Shutter enable/disable	VSON0	VSON1	HSON0	HSON1	0: Disable shutter (Acts as though there are no shutter lines.) 1: Enable shutter
Shutter position	VST00– VST09	VST10– VST19	HST00– HST09	HST10– HST19	For vertical shutters, this is the number of H scan lines from the top of the screen. For horizontal shutters, it is the number of pixels from the left of the screen.
Shuttering direction	VSP0	VSP1	HSP0	HSP1	0: Shutter below (vertical shutters) or to the right (horizontal shutters) 1: Shutter above (vertical shutters) or to the left (horizontal shutters)
Shuttering mode control	SHTRAD (shared bit)				0: AND the shuttered areas of all the shutters 1: OR the shuttered areas of all the shutters

■ **Determining the vertical shutter positions (VST0 and VST1)**

The top edge of the television screen is x'000'. Each integer higher brings the shutter position down one H scan line.

■ **Determining the horizontal shutter positions (HST0 and HST1)**

The left edge of the television screen is x'000'. Each integer higher brings the shutter position right one pixel. (One pixel, or one dot, is the smallest display unit in the OSD.)

On-Screen Display

Controlling the Shuttering Effect

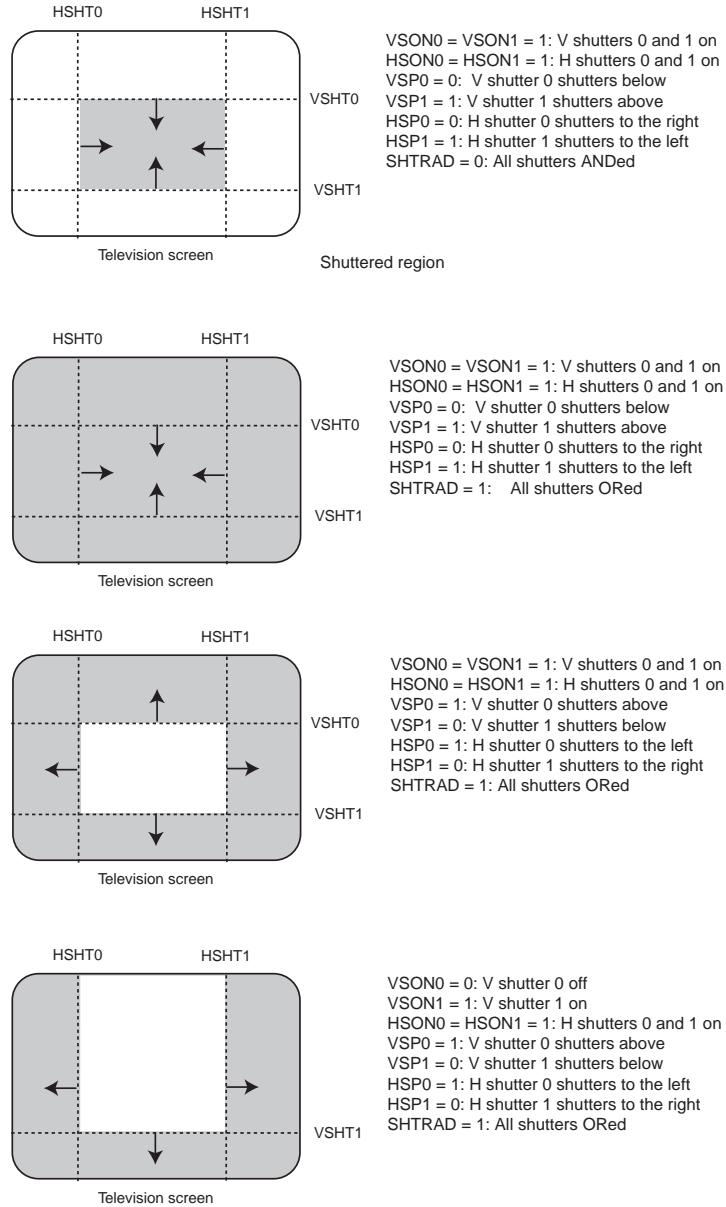


Figure 7-31 Shuttered Area Setup Examples

7.13.2 Controlling Shutter Movement

Enabling the shutter movement function in the registers allows the shuttered area to expand or contract over time, producing a wipe-in or wipe-out effect. This allows the OSD display to appear or disappear without an abrupt transition. Table 7-13 shows the register settings required for this function, and figure 7-32 shows four setup examples. There is no repeat operation for shutter movement, so you must reset the bits each time.

Table 7-13 Bit Settings for Controlling Shutter Movement

Function	V Shutter 0 Bit Name	V Shutter 1 Bit Name	H Shutter 0 Bit Name	H Shutter 1 Bit Name	Description
Shutter movement enable/disable	VSM0	VSM1	HSM0	HSM1	0: Move shutter 1: Don't move shutter
Shuttering movement direction	VSMP0	VSMP1	HSMP0	HSMP1	0: Move from top to bottom (vertical shutters) or from left to right (horizontal shutters) 1: Move from bottom to top (vertical shutters) or from right to left (horizontal shutters)
Shuttering movement speed control	SHSP0, SHTSP1 (shared bits)				00: Move every VSYNC 01: Move every 2 VSYNCS 10: Move every 3 VSYNCS 11: Move every 4 VSYNCS

■ **Considerations for horizontal shutter movement**

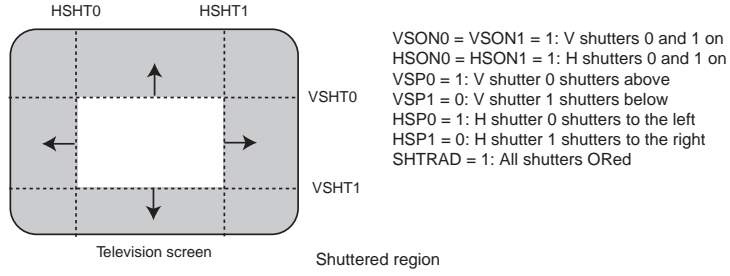
Do not set the horizontal shutter position (HST0, HST1) within the following ranges if you are only moving the horizontal shutter. The shutter cannot move if you do.

x'000' to x'003' and x'3FC' to x'3FF'

You can set these values if you wish to prevent horizontal shutter movement. Note that the horizontal shutter position does not affect vertical movement.

On-Screen Display

Controlling the Shuttering Effect



This example shows V shutter 0 moving downward. It shutters both the text and the background color in the text layer.

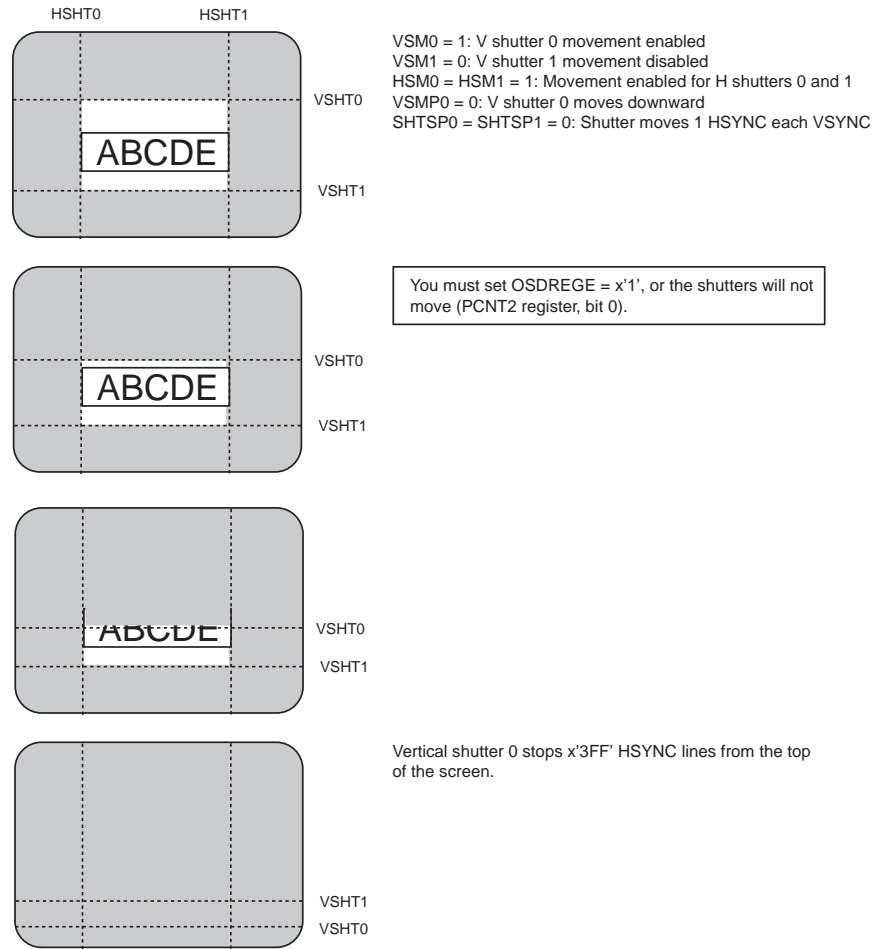


Figure 7-32 Shutter Movement Setup Examples

7.13.3 Controlling Shuttering Effects

Through register settings, you can independently control shuttering for text, text background, graphics, and color background. You can also output blanks to the shuttered area.

You cannot shutter the cursor layer.

Table 7-13 shows the register settings required for these effects. There are three types of shuttering—shuttering of text, text background, and graphics, shuttering of the color background, and shutter blanking. The sections below describe how to control each of these.

Table 7-14 Bit Settings for Controlling Shuttering Effects

Function	Bit Name	Description
Text shuttering	CCSHT	0: Shutter text-layer characters 1: Don't shutter text-layer characters
Text background shuttering	BCSHT	0: Shutter text-layer background 1: Don't shutter text-layer background
Graphics shuttering	GSHT	0: Shutter graphics layer 1: Don't shutter graphics layer
Color background shuttering	COLBSHT	0: Shutter color background 1: Don't shutter color background
Shutter blanking	SHTBLK	0: Don't output blanks to the shuttered area 1: Output blanks to the shuttered area

■ **To shutter text-layer characters, text-layer background, and graphics layer:**



Do not allow the horizontal shuttering boundaries to overlap any italicized portion of a closed-caption display. This distorts the italicized characters.

Text

Set the text shutter control bit, CCSHT, of the shutter control register, SHTC (x'007F28') to 1.

Text background

Set the text background shutter control bit, BCSHT, of SHTC to 1.

Graphics

Set the text background shutter control bit, GSHT, of SHTC to 1.

Figure 7-33 shows three setup examples of text-layer shuttering.

On-Screen Display

Controlling the Shuttering Effect

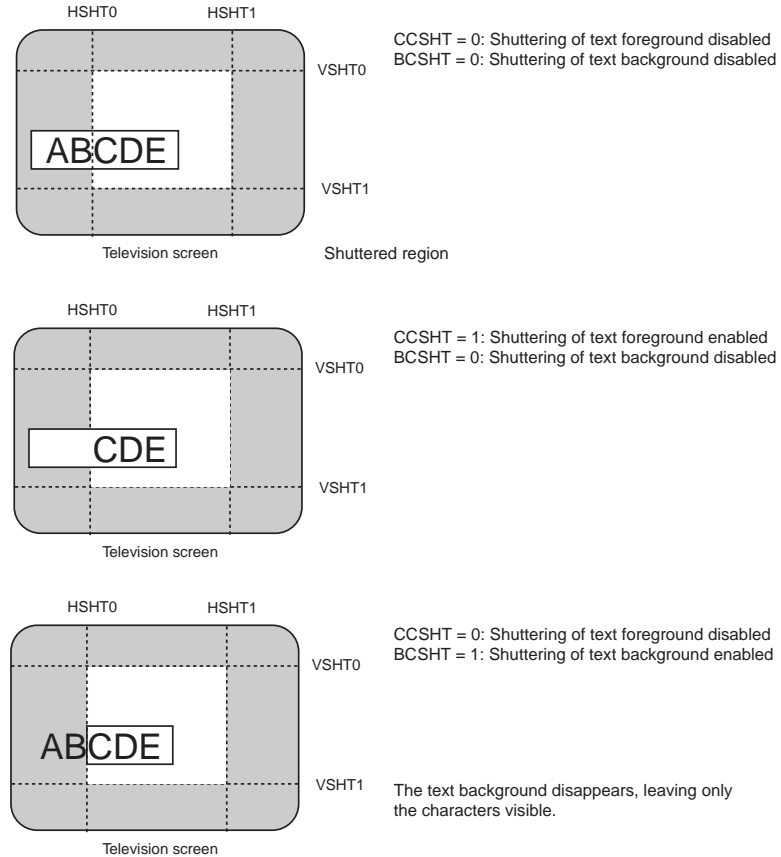


Figure 7-33 Text-Layer Shuttering Setup Examples

■ **To shutter the color background:**

Set the color background shutter control bit, COLBSHT, of the shutter control register, SHTC (x'007F28') to 1.

This function exists only when the program enables a color background. It allows you to limit the area covered by the color background.

■ **To blank out the shuttered area:**

Set the shutter blanking control bit, SHTBLK, of SHTC to 1.

Shutter blanking outputs black to the entire shuttered area. To output blanking to a display that uses a color background, enable the color background shutter (COLBSHT = 1) so that the color background will also be blanked in the shuttered area. Figure 7-34 shows two setup examples.

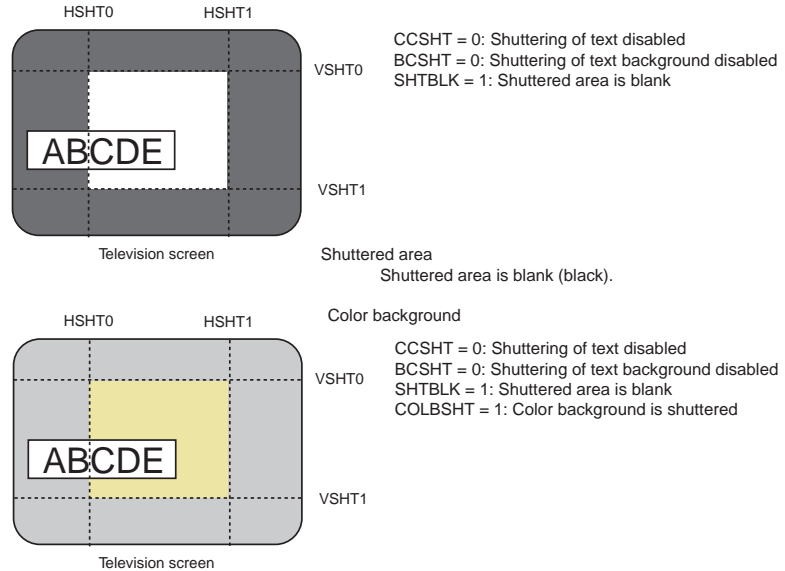


Figure 7-34 Shutter Blanking Setup Examples

7.13.4 Controlling Line Shuttering

It is possible to cancel shuttering of individual lines on the text and graphics layers so that they will be displayed on both shuttered and non-shuttered regions.

- **To disable shuttering on the next line:**
Set the GSHT bit (bit 10 of GHP in the RAM data) and/or the CSHT bit (bit 10 of CHP in the RAM data) to 1.
- **To disable shuttering on the first line:**
Set the GISHT bit of the GIHP register (x'007F16') and/or the CISHT bit of the CIHP register (x'007F14') to 1.

Figure 7-35 shows a setup example for the text layer.

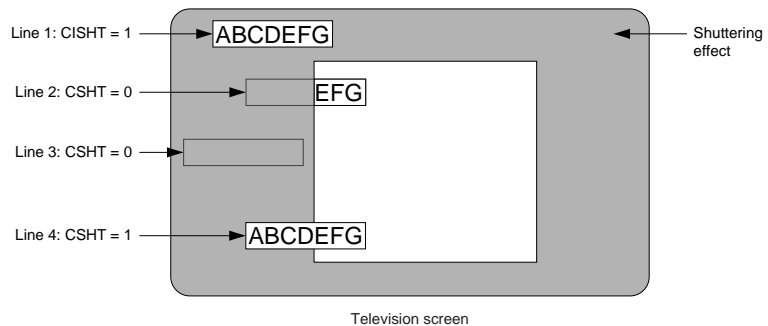


Figure 7-35 Line Shuttering Setup Example

7.14 Field Detection Circuit

7.14.1 Block Diagram

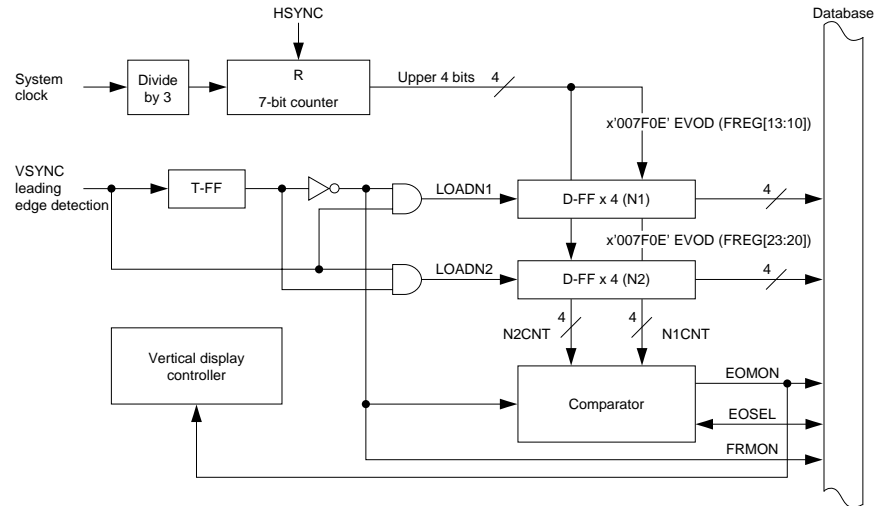


Figure 7-36 Field Detection Circuit Block Diagram

7.14.2 Description

The 7-bit field counter in this block resets every HSYNC interval to count the system clock. At each VSYNC interval, the 4 MSBs of the 7-bit counter are alternately loaded (made readable) to bits 7 to 4 (N2) and 3 to 0 (N1) of the EVOD register (x'007F0E'). The comparator compares the N1 and N2 values and outputs the results to the EOMON bit of EVOD. The OSD identifies the field that sets EOMON to 1 as the display start field. Table 7-15 shows the criteria that the comparator uses.

By reading the FRMON bit of EVOD, the OSD can determine which register the 4 MSBs will load to on the next VSYNC input.

To ensure that the display starts at the right field, you must also set the EOSEL bit of EVOD so that EOMON becomes 1 at the display start field.

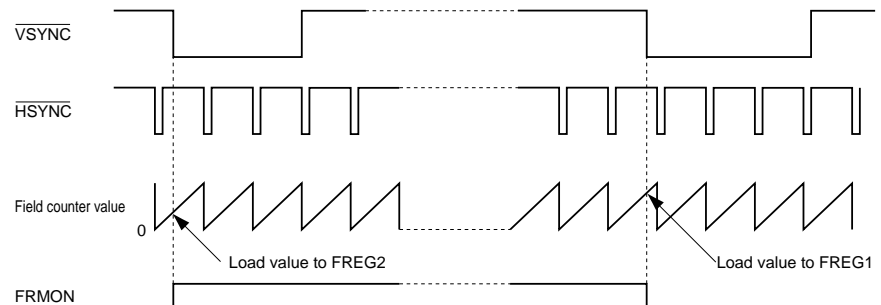


Figure 7-37 Field Detection Timing

Table 7-15 EOMON Output Criteria

FRMON	N1, N2 Relationship	EOMON Output	
		EOSEL = 0	EOSEL = 1
0 Load to N2 next	N1 > N2	0	1
	N1 < N2	1	0
	N1 = N2	Complement previous	Complement previous
1 Load to N1 next	N1 > N2	1	0
	N1 < N2	0	1
	N1 = N2	Complement previous	Complement previous

7.14.3 Considerations for Interlaced Displays

■ **Switching the display start field**

The OSD is constructed so the display start position is the field (field 1) where the EOMON bit is 1. However, interlaced displays may require that the start position be a field (field 2) where the EOMON bit is 0. In this case, merely complementing the EOSEL bit will not result in a correct display. You must set the following two bits to have the display start at field 2.

- ◆ **CANH** (x'007F0A', bit 4): Set to 1.
0: Normal display
1: Slide the field 1 display position down 1 line
- ◆ **EOSEL** (x'007F0E', bit 10): Complement the value EOSEL has for field 1 in a normal display.

■ **Scrolling in the closed-caption mode**

To implement text-layer scrolling in the closed-caption mode, the program must constantly switch the text display fields. This can cause the text lines to display incorrectly. To prevent this, set the following bits to fix the text lines to the even or odd field characters while scrolling.

- ◆ **BFLD** (x'007F0A', bit 2): Set to 1 to enable scrolling.
0: Normal display
1: Display the same characters in fields 1 and 2
- ◆ **EONL** (x'007F0A', bit 3): Set to 0 or 1.
0: Fix to characters in field 1
1: Fix to characters in field 2

7.15 OSD Registers

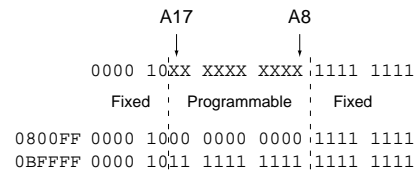
All registers in OSD block cannot be written by byte (by word only). Read by byte is possible.

CROMEND: Text ROM End Address Register

x'007F00'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	A17	A16	A15	A14	A13	A12	A11	A10	A9	A8
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

A[17:8] holds the programmable portion of the text ROM end address. The low-order 8 bits of the address are always x'FF' and the high-order 6 bits are always b'000010'. The available address range is x'0800FF' to x'0BFFFF', with a programmable range from x'000' to x'3FF'.



GROMEND: Graphics ROM End Address Register

x'007F02'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	A17	A16	A15	A14	A13	A12	A11	A10	A9	A8
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

A[17:8] holds the programmable portion of the graphics ROM end address. The low-order 8 bits of the address are always x'FF' and the high-order 6 bits are always b'000010'. The available address range is x'0800FF' to x'0BFFFF', with a programmable range from x'000' to x'3FF'.

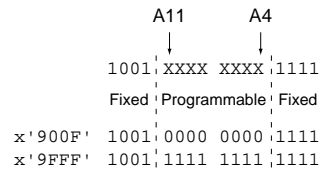
RAMEND: Text and Graphics RAM End Address Register

x'007F04'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	GRAM END A11	GRAM END A10	GRAM END A9	GRAM END A8	GRAM END A7	GRAM END A6	GRAM END A5	GRAM END A4	GRAM END A3	GRAM END A2	GRAM END A1	GRAM END A0	GRAM END A15	GRAM END A14	GRAM END A13	GRAM END A12
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

GRAMENDA[11:4] holds the programmable portion of the graphics RAM end address and CRAMENDA[11:4] holds the programmable portion of the text RAM end address. The low-order 4 bits of the address are always x'F' and the high-order 4 bits are always x'9'. The available address range

is x'900F' to x'9FFF', with a programmable range from x'00' to x'FF'.



STC0: Cursor Tile Code Register 0

x'007F10'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

SPRT0: Cursor 0 color palette select

0: Graphics color palette 1

1: Graphics color palette 2

STC0[8:0]: Cursor 0 Tile Code

Use the same ROM data as that used for the graphics.

STC1: Cursor Tile Code Register 1

x'007F2A'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

SPRT1: Cursor 1 color palette select

0: Graphics color palette 1

1: Graphics color palette 2

STC1[8:0]: Cursor 1 Tile Code

Use the same ROM data as that used for the graphics.

STC2: Cursor Tile Code Register 2

x'007F2C'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

SPRT2: Cursor 2 color palette select

0: Graphics color palette 1

1: Graphics color palette 2

STC2[8:0]: Cursor 2 Tile Code

Use the same ROM data as that used for the graphics.

STC3: Cursor Tile Code Register 3

x'007E2E'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	SPRT3	STC38	STC37	STC36	STC35	STC34	STC33	STC32	STC30	STC30
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

SPRT3: Cursor 3 color palette select

- 0: Graphics color palette 1
- 1: Graphics color palette 2

STC3[8:0]: Cursor 3 Tile Code

Use the same ROM data as that used for the graphics.

SHP: Cursor Horizontal Position Register

x'007F12'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	—	SHSZ1	SHSZ0	—	SHP9	SHP8	SHP7	SHP6	SHP5	SHP4	SHP3	SHP2	SHP1	SHP0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R	R	R	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

SHSZ[1:0]: Cursor horizontal size

- 00: 1 dot = 1 VCLK period
- 01: 1 dot = 2 VCLK periods
- 10: 1 dot = 3 VCLK periods
- 11: 1 dot = 4 VCLK periods

SHP[9:0]: Cursor horizontal position

SVP: Cursor Vertical Position Register

x'007F14'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	—	SVSZ1	SVSZ0	—	SVP9	SVP8	SVP7	SVP6	SVP5	SVP4	SVP3	SVP2	SVP1	SVP0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R	R	R	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

SVSZ[1:0]: Cursor vertical size

Table 7-16 Cursor Vertical Size Settings

SVSZ[1:0] Setting	1 Dot Size	
	Interlaced Displays	Progressive Displays
00	1 H scan line	Reserved
01	2 H scan lines	1 H scan line
10	4 H scan lines	2 H scan lines
11	6 H scan lines	3 H scan lines

SVP[9:0]: Cursor vertical position

GIHP: Graphics Initial Horizontal Position Register

x'007F16'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	—	GIHSZ1	GIHSZ0	GISHT	GIHP9	GIHP8	GIHP7	GIHP6	GIHP5	GIHP4	GIHP3	GIHP2	GIHP1	GIHP0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

GIHSZ[1:0]: Graphics initial horizontal size

00: 1 dot = 1 VCLK period
 01: 1 dot = 2 VCLK periods
 10: 1 dot = 3 VCLK periods
 11: 1 dot = 4 VCLK periods

GISHT: Graphics initial shutter control
 0: Shutter control on
 1: Shutter control off

GIHP[9:0]: Graphics initial horizontal position

GIVP: Graphics Initial Vertical Position Register **x'007F18'**

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	—	GI VSZ1	GI VSZ0	—	GIVP9	GIVP8	GIVP7	GIVP6	GIVP5	GIVP4	GIVP3	GIVP2	GIVP1	GIVP0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R	R	R	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

GIVSZ[1:0]: Graphics initial vertical size

Table 7-17 Graphics Vertical Size Settings

GIVSZ[1:0] Setting	1 Dot Size	
	Interlaced Displays	Progressive Displays
00	1 H scan line	Reserved
01	2 H scan lines	1 H scan line
10	4 H scan lines	2 H scan lines
11	6 H scan lines	3 H scan lines

GIVP[9:0]: Graphics initial vertical position

CIHP: Text Initial Horizontal Position Register **x'007F1A'**

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	—	CI HSZ1	CI HSZ0	CI SHT	CIHP9	CIHP8	CIHP7	CIHP6	CIHP5	CIHP4	CIHP3	CIHP2	CIHP1	CIHP0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

CIHSZ[1:0]: Text initial horizontal size
 00: 1 dot = 1 VCLK period
 01: 1 dot = 2 VCLK periods
 10: 1 dot = 3 VCLK periods
 11: 1 dot = 4 VCLK periods

CISHT: Graphics initial shutter control
 0: Shutter control on
 1: Shutter control off

CIHP[9:0]: Graphics initial horizontal position

CIVP: Text Initial Vertical Position Register **x'007F1C'**

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	—	CI VSZ1	CI VSZ0	—	CIVP9	CIVP8	CIVP7	CIVP6	CIVP5	CIVP4	CIVP3	CIVP2	CIVP1	CIVP0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R	R	R	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

CIVSZ[1:0]: Text initial vertical size

Table 7-18 Text Vertical Size Settings

CIVSZ[1:0] Setting	1 Dot Size	
	Interlaced Displays	Progressive Displays
00	1 H scan line	Reserved
01	2 H scan lines	1 H scan line
10	4 H scan lines	2 H scan lines
11	6 H scan lines	3 H scan lines

CIVP[9:0]: Text initial vertical position

EVOD: Display Start Field Control Register

x'007F0E'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	—	—	—	EO SEL	FR MON	EO MON	FREG 23	FREG 22	FREG 21	FREG 20	FREG 13	FREG 12	FREG 11	FREG 10
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R/W	R	R	R	R	R	R	R	R	R	R

EOSEL: Even/odd field select

- 0: Select the smaller counter value as the display start field
- 1: Select the larger counter value as the display start field

FRMON: Field register monitor

Monitors which field register (FREG) loaded the counter value on the leading edge of VSYNC.

- 0: Loaded to FREG[23:20]
- 1: Loaded to FREG[13:10]

EOMON: Even/odd field monitor

Set between display fields.

- 0: No display start field detected
- 1: Display start field detected

FREG[23:20]: Field register

4-bit register 2 storing field counter value.

FREG[13:10]: Field register

4-bit register 1 storing field counter value.

HCOUNT: HSYNC count

x'007F0C'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	HCNT 9	HCNT 8	HCNT 7	HCNT 6	HCNT 5	HCNT 4	HCNT 3	HCNT 2	HCNT 1	HCNT 0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

This register holds the HSYNC count, which indicates the vertical display position. It is cleared to 0 on the leading edge of VSYNC.

OSD1: OSD Register 1

x'007F06'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	—	—	—	OSD	OSC SEL1	OSC SEL0	XIO	HPOL	VPOL	YS POL	YS PLT	—	YCNT	RGBC
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R	R/W	R/W



A write to the OSD bit of OSD1 takes effect on the next leading edge of VSYNC. If you are turning the OSD on, the OSD starts operating on the next VSYNC after the program writes a 1 to the OSD bit. If you are turning the OSD off, the OSD stops operating on the next VSYNC after the program writes a 0 to the OSD bit.

0: Off

1: On

OSCSEL[1:0]: Oscillator select

00: OSC clock + PLL + internal synchronization

01: OSDX clock (with LC blocking oscillator)

10: OSDX clock (external source) + internal synchronization

11: Reserved

XIO: OSDX frequency select

Frequency range: 12 to 48 MHz

0: Less than 20 MHz

1: Greater than 20 MHz

HPOL: HSYNC input polarity select

0: Active low

1: Active high

VPOL: VSYNC input polarity select

0: Active high

1: Active low

YSPOL: YS output polarity select

0: Active high

1: Active low

YSPLT: YS color palette select

0: Output YS to entire display area (except transparent and translucent areas)

1: Output the MSB (bit 15) of all the color palettes from YS

YCNT: YM DAC/digital output select

0: DAC output

1: Digital output

Digital output is the LSB of YM (bit 12 of the color palette).

RGBC: RGB DAC/digital output select

0: DAC output

1: Digital output

Digital output is the LSB of each color:

R: bit 0 of the color palette

G: bit 4 of the color palette

B: bit 8 of the color palette



To turn off the OSD block to save power:

1. Write a 0 to OSD (OSD1, bit 10).
2. Wait for the next VSYNC input.
3. Write a 0 to OSDPOFF (PCNT0, bit 7), turning the clock off.

If you turn the clock off before the VSYNC input, power usage may not drop or the microcontroller may halt.



In using LC blocking oscillation, the power consumption becomes large. To reduce it, select another oscillation.



In case of using OSDX clock (both LC blocking oscillation and external clock), to prevent the current flow set OSDXI, OSDXO terminal to port function (P45, P46) and output H level before invoking STOP mode.

On-Screen Display

OSD Registers

OSD2: OSD Register 2

x'007F08'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SPEX T	GTHT	GEXTE	PRYM	TRPTF	TRPT	GCOL 1	GCOL 0	COLB	VCLK 2	VCLK 1	VCLK 0	CGPR	SOUT	GOUT	COUT
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

SPEXT: Cursor extended mode select

- 0: Standard mode (16 x 16 pixels)
- 1: Extended mode (32 x 32 pixels)

GTHT: Graphic tile height select

- 0: 16 pixels high
- 1: 18 pixels high

GEXTE: Graphics maximum tiles per line select

- 0: Maximum 18 tiles
- 1: Maximum 28 tiles

PRYM: Translucent color control (YM, YS)

- 0: Normal YS output
- 1: Don't output YS during YM output (when the YM color code is not 0)

TRPTF: Translucency control (all layers)

- Specifies translucency for color 15 in all palettes.
- 0: Make color 15 on all palettes translucent
- 1: Output color 15 as specified

TRPT: Transparency control (all layers)

- Specifies transparency for color 0 in all palettes.
- 0: Make color 0 on all palettes transparent
- 1: Output color 0 as specified

GCOL[1:0]: Graphics color mode

- 00: 16-color mode
- 01: 8-color mode
- 10: 4-color mode
- 11: 2-color mode

COLB: Color background control

- 0: Don't output color background
- 1: Output color background

VCLK[2:0]: VCLK clock select

Divide-by ratio select. The values in parentheses () are applicable when the oscillator in the OSD1 register is set to the OSC clock (OSCSEL[1:0] = 00).

- 000: Divide by 4 (12 MHz)
- 001: Divide by 3 (16 MHz)
- 010: Divide by 2 (24 MHz)
- 011: 2/3 division (32 MHz)
- 100: Don't divide (48 MHz)
- 101: Reserved
- 110: Reserved
- 111: Reserved

CGPR: Graphics/text layer priority

- 0: Graphics layer takes priority
- 1: Text layer takes priority

SOUT/GOUT/COUT: Layer output on/off

SOUT: cursor layer; GOUT: graphics layer; COUT: text layer

- 0: Don't output layer
- 1: Output layer



Setting the YSPLT bit of OSD1 to 1 disables the PRYM bit. With this setting, you must also set the TRPT and TRPTF bits to 1. You can specify transparency for individual color palettes if needed.



You cannot set VCLK[2:0] to 100 (don't divide) when the OSC clock is selected. Only select it when the clock source is 32 MHz or less.

OSD3: OSD Register 3

x'007F0A'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	—	—	—	BLINK	CANH	EONL	BFLD	UNDF	CAPM
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W

BLINK: Character blinking control

Controls blinking for text-layer characters with BLINK set in the COL code.

- 0: Don't blink
- 1: Blink

CANH: Vertical position control for closed captions

Active when interlacing is selected.

- 0: Normal position
- 1: Add 1 to V position of even fields

EONL and BFLD: Closed-caption scrolling control

Use when required for smoother scrolling.

- 00: Normal display
- 01: Fix to characters in odd fields during scrolling
- 10: Fix to characters in even fields during scrolling
- 11: Normal display

UNDF: Underline blinking control

- 0: Don't blink
- 1: Blink

CAPM: Closed-caption mode setting

- 0: Normal display mode
- 1: Closed-caption mode

VSHT0: Vertical Shutter 0 Register

x'007F20'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	VSON0	VSP0	VSMP0	VSM0	VST09	VST08	VST07	VST06	VST05	VST04	VST03	VST02	VST01	VST00
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

VSON0: Vertical shutter 0 on/off

- 0: Off
- 1: On

VSP0: Vertical shutter 0 shuttering direction

- 0: Shutter below
- 1: Shutter above

VSMP0: Vertical shutter 0 movement direction

- 0: Top to bottom
- 1: Bottom to top

VSM0: Vertical shutter 0 movement control

- 0: Don't move
- 1: Move

VST0[9:0]: Vertical shutter 0 position

VSHT1: Vertical Shutter 1 Register

x'007F22'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	VSON 1	VSP1	VSMP 1	VSM1	VST1 9	VST1 8	VST1 7	VST1 6	VST1 5	VST1 4	VST1 3	VST1 2	VST1 1	VST1 0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

VSON1: Vertical shutter 1 on/off

- 0: Off
- 1: On

VSP1: Vertical shutter 1 shuttering direction

- 0: Shutter below
- 1: Shutter above

VSMP1: Vertical shutter 1 movement direction

- 0: Top to bottom
- 1: Bottom to top

VSM1: Vertical shutter 1 movement control

- 0: Don't move
- 1: Move

VST1[9:0]: Vertical shutter 1 position

HSHT0: Horizontal Shutter 0 Register

x'007F24'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	HSON 0	HSP0	HSMP 0	HSM0	HST0 9	HST0 8	HST0 7	HST0 6	HST0 5	HST0 4	HST0 3	HST0 2	HST0 1	HST0 0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

HSON0: Horizontal shutter 0 on/off

- 0: Off
- 1: On

HSP0: Horizontal shutter 0 shuttering direction

- 0: Shutter to the right
- 1: Shutter to the left

HSMP0: Horizontal shutter 0 movement direction

- 0: Left to right
- 1: Right to left

HSM0: Horizontal shutter 0 movement control

- 0: Don't move
- 1: Move

HST0[9:0]: Horizontal shutter 0 position

HSHT1: Horizontal Shutter 1 Register

x'007F26'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	HSO1	HSP1	HSMP1	HSM1	HST1 9	HST1 8	HST1 7	HST1 6	HST1 5	HST1 4	HST1 3	HST1 2	HST1 1	HST1 0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

HSO1: Horizontal shutter 1 on/off

- 0: Off
- 1: On

HSP1: Horizontal shutter 1 shuttering direction

- 0: Shutter to the right
- 1: Shutter to the left

HSMP1: Horizontal shutter 1 movement direction

- 0: Left to right
- 1: Right to left

HSM1: Horizontal shutter 1 movement control

- 0: Don't move
- 1: Move

HST1[9:0]: Horizontal shutter 1 position



The shuttering function does not work in the cursor layer.

SHTC: Shutter Control Register

x'007F28'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	—	SHT BLK	COLB SHT	SHT SP1	SHT SP0	SHT RAD	GSHT	BC SHT	CC SHT
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

SHTBLK: Shutter blanking on/off

- 0: Off
- 1: On

COLBSHT: Color background shutter control

- 0: Disable
- 1: Enable

SHTSP[1:0]: Shutter speed control

- 00: Move every VSYNC
- 01: Move every 2 VSYNCs
- 10: Move every 3 VSYNCs
- 11: Move every 4 VSYNCs

SHTRAD: Shutter mode control

- 0: AND mode
- 1: OR mode

GSHT: Graphics shutter control

- 0: Disable
- 1: Enable

BCSHT: Text background shutter control

- 0: Disable
- 1: Enable

CCSHT: Text shutter control

- 0: Disable
- 1: Enable

CPT0–CPTF: Text Palette Colors 0–15 Registers

x'007F80'–x'007F9E'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CPTn YM3	CPTn YM2	CPTn YM1	CPTn YM0	CPTn B3	CPTn B2	CPTn B1	CPTn B0	CPTn G3	CPTn G2	CPTn G1	CPTn G0	CPTn R3	CPTn R2	CPTn R1	CPTn R0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

These registers contain the colors used in the text layer. When digital output is selected, CPTnYM0 is output as YM, CPTnB0 as B, CPTnG0 as G, and CPTnR0 as R. When the YS color palette is selected, CPTnYM3 is output as YS.

CPTnYM[3:0]: YM color code

CPTnB[3:0]: Blue color code

CPTnG[3:0]: Green color code

CPTnR[3:0]: Red color code

COLB: Color Background Register

x'007FA0'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	COLB YM3	COLB YM2	COLB YM1	COLB YM0	COLB B3	COLB B2	COLB B1	COLB B0	COLB G3	COLB G2	COLB G1	COLB G0	COLB R3	COLB R2	COLB R1	COLB R0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

This register contains the color used for the color background. When digital output is selected, COLBYM0 is output as YM, COLBB0 as B, COLBG0 as G, and COLBR0 as R. When the YS color palette is selected, COLBYM3 is output as YS.

COLBYM[3:0]: YM color code

COLBB[3:0]: Blue color code

COLBG[3:0]: Green color code

COLBR[3:0]: Red color code

FRAME: Outlining and Character Shadowing Color Register

x'007FA2'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FRAM E YM3	FRAM E YM2	FRAM E YM1	FRAM E YM0	FRAM E B3	FRAM E B2	FRAM E B1	FRAM E B0	FRAM E G3	FRAM E G2	FRAM E G1	FRAM E G0	FRAM E R3	FRAM E R2	FRAM E R1	FRAM E R0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

This register contains the color used for text outlining and shadowing. When digital output is selected, FRAMEYM0 is output as YM, FRAMEB0 as B, FRAMEG0 as G, and FRAMER0 as R. When the YS color palette is selected, FRAMEYM3 is output as YS.

FRAMEYM[3:0]: YM color code

FRAMEB[3:0]: Blue color code

FRAMEG[3:0]: Green color code

FRAMER[3:0]: Red color code

BBSHD: Black Box Shadowing Register

x'007FA4'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	BB SHD YM3	BB SHD YM2	BB SHD YM1	BB SHD YM0	BB SHD B3	BB SHD B2	BB SHD B1	BB SHD B0	BB SHD G3	BB SHD G2	BB SHD G1	BB SHD G0	BB SHD R3	BB SHD R2	BB SHD R1	BB SHD R0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

This register contains the color used as black in box shadowing. When digital output is selected, BBSHDYM0 is output as YM, BBSHDB0 as B, BBSHDG0 as G, and BBSHDR0 as R. When the YS color palette is selected, BBSHDYM3 is output as YS.

BBSHDYM[3:0]: YM color code

BBSHDB[3:0]: Blue color code

BBSHDG[3:0]: Green color code

BBSHDR[3:0]: Red color code

WBSHD: White Box Shadowing Register

x'007FA6'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	WB SHD YM3	WB SHD YM2	WB SHD YM1	WB SHD YM0	WB SHD B3	WB SHD B2	WB SHD B1	WB SHD B0	WB SHD G3	WB SHD G2	WB SHD G1	WB SHD G0	WB SHD R3	WB SHD R2	WB SHD R1	WB SHD R0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

This register contains the color used as white in box shadowing. When digital output is selected, WBSHDYM0 is output as YM, WBSHDB0 as B, WBSHDG0 as G, and WBSHDR0 as R. When the YS color palette is selected, WBSHDYM3 is output as YS.

WBSHDYM[3:0]: YM color code

WBSHDB[3:0]: Blue color code

WBSHDG[3:0]: Green color code

WBSHDR[3:0]: Red color code

GPT10–GPT1F: Graphics Palette 1 Colors 0–15 Registers x'007FC0'–x'007FDE'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	GPT1n YM3	GPT1n YM2	GPT1n YM1	GPT1n YM0	GPT1n B3	GPT1n B2	GPT1n B1	GPT1n B0	GPT1n G3	GPT1n G2	GPT1n G1	GPT1n G0	GPT1n R3	GPT1n R2	GPT1n R1	GPT1n R0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

These registers contain one of two sets of colors used in the graphics layer. When digital output is selected, GPT1nYM0 is output as YM, GPT1nB0 as B, GPT1nG0 as G, and GPT1nR0 as R. When the YS color palette is selected, GPT1nYM3 is output as YS.

GPT1nYM[3:0]: YM color code

GPT1nB[3:0]: Blue color code

GPT1nG[3:0]: Green color code

GPT1nR[3:0]: Red color code

On-Screen Display

OSD Registers

GPT20–GPT2F: Graphics Palette 2 Colors 0–15 Registers x'007FE0'–x'007FFE'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	GPT2n YM3	GPT2n YM2	GPT2n YM1	GPT2n YM0	GPT2n B3	GPT2n B2	GPT2n B1	GPT2n B0	GPT2n G3	GPT2n G2	GPT2n G1	GPT2n G0	GPT2n R3	GPT2n R2	GPT2n R1	GPT2n R0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

These registers contain one of two sets of colors used in the graphics layer. When digital output is selected, GPT2nYM0 is output as YM, GPT2nB0 as B, GPT2nG0 as G, and GPT2nR0 as R. When the YS color palette is selected, GPT2nYM3 is output as YS.

GPT2nYM[3:0]: YM color code

GPT2nB[3:0]: Blue color code

GPT2nG[3:0]: Green color code

GPT2nR[3:0]: Red color code

8 IR Remote Signal Receiver

8.1 Description

The MN102H75K/85K contains a remote signal receiver that processes signals in two formats: Household Electrical Appliance Manufacturers Association (HEAMA) format and 5-/6-bit format. This chapter provides an overview of each block in the circuit and describes the operation of the receiver.

The remote signal is input through the RMIN pin. Each time the edge detection circuit detects the active edge of the signal, the 6-bit counter resets and the sampling clock, T_S , starts counting. T_S is formed by dividing PWM3 by the value in the frequency division control register, RMTC. The clock status register, RMCS, which can be read at any time, holds the current value of the 6-bit counter.

The remote signal contains a leader, data, and a trailer, in that order. The microcontroller shifts received data into the LSB of the reception data shift register, RMSR. After it receives 8 bits, it loads the contents of RMSR to the reception data transfer register, RMTR.

fSYSCLK = 12 MHz in all of the examples and descriptions in this section. In addition, fPWM1 = fSYSCLK/23, fPWM3 = fSYSCLK/25, fPWM5 = fSYSCLK/27, fPWM6 = fSYSCLK/28, and fPWM8 = fSYSCLK/210.

8.2 Block Diagram

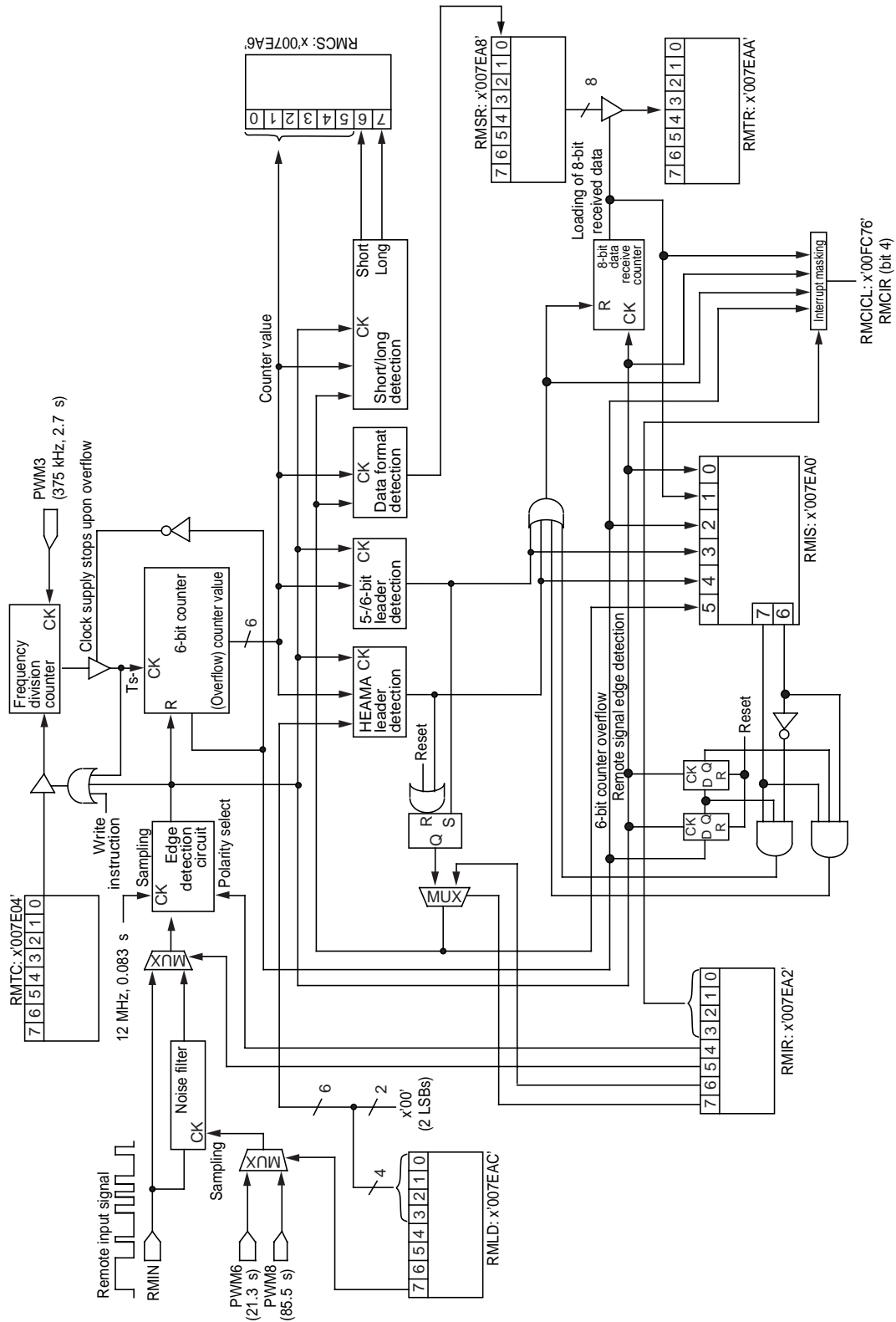


Figure 8-1 IR Remote Signal Receiver Block Diagram

8.3 IR Remote Signal Receiver Operation

8.3.1 Operating Modes

The IR remote signal receiver has three operating modes: HEAMA, 5-/6-bit, and HEAMA–5-/6-bit automatic detect. Set the mode in the MODAUTO and MODSEL bits of the interrupt control register, RMIR. The FMTMON bit of the interrupt status register, RMIS, monitors the operating mode.

In automatic detect mode, the microcontroller checks the interval between remote signal edges. If the interval is $(n - 4T_S)$ to $(n + 3T_S)$, where n is the leader value set in the LD[3:0] field of the RMLD register, it processes the data in HEAMA format. If the interval is 28 to 35 T_S cycles, it processes the data in 5-/6-bit format.

8.3.2 Noise Filter

The IR remote signal receiver contains a noise filter to eliminate noise from the remote signal. To enable the noise filter, set the FILTRE bit of the interrupt control register, RMIR, to 1. The noise filter samples the remote input signal every PWM6 cycle (21.3 μs) or PWM8 cycle (85.5 μs), then outputs the value that it sampled at least three times during the last four sampling cycles. This eliminates any noise occurring during one or two sampling cycles. Select the sampling clock (PWM6 or PWM8) with the SP bit of the RMLD register. (PWM6 is selected at reset.)

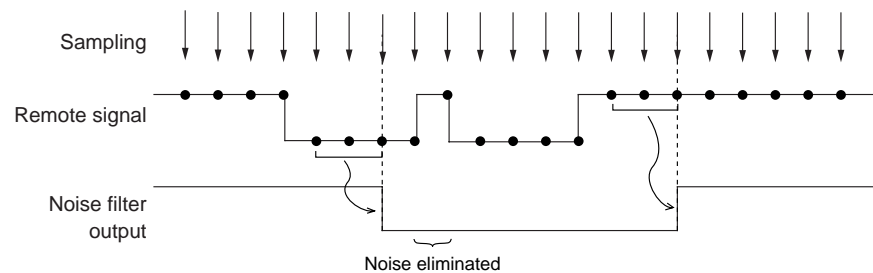


Figure 8-2 IR Remote Signal Noise Filtering

8.3.3 8-Bit Data Reception

Resetting the 8-bit data reception counter allows the microcontroller to receive 8-bit data, either with or without a leader. The software can reset the counter using the BCRSTE and BCEDGS bits of the interrupt status register, RMIS. The counter can also be reset by an external reset or a hardware reset at leader detection.

Set BCRSTE to enable resets to the 8-bit data reception counter. When the BCEDGS bit is 0, the counter resets at the first remote signal edge after each trailer detection. This mode is for data containing no leader. (See figure 8-3.)

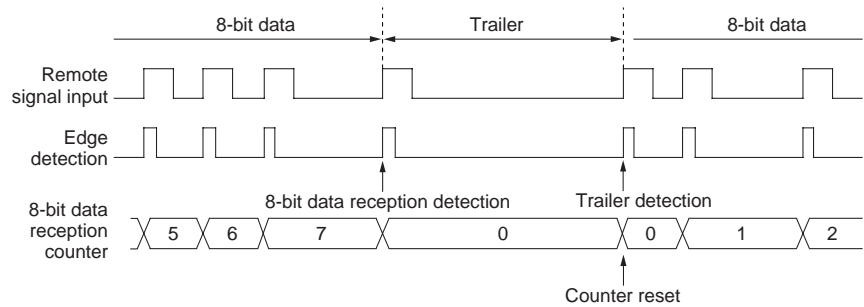


Figure 8-3 Reception of 8-Bit Data with No Leader

When BCEDGS is 1, the counter resets at the second remote signal edge after each trailer detection. By ignoring the leader, this mode allows the microcontroller to receive 8-bit data that contains a leader. (See figure 8-4.)

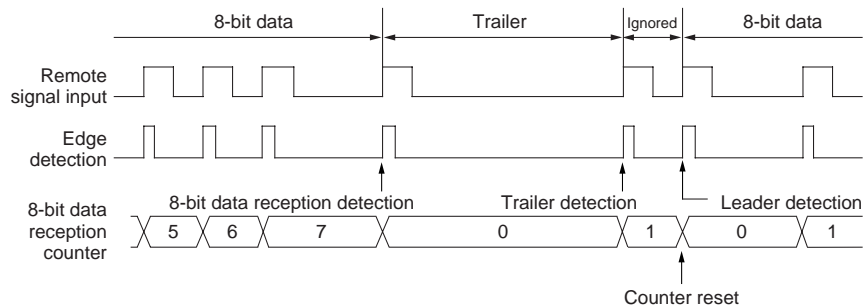


Figure 8-4 Reception of 8-Bit Data with Leader

8.3.4 Identifying the Data Format

The microcontroller determines the logic levels of the data by testing the interval between remote signal edges. Table 8-1 shows the intervals that the microcontroller interprets as 0 and 1 for both HEAMA and 5-/6-bit formats. Table 8-2 shows the conditions for identifying long and short data.


Table 8-1 Logic Level Conditions for Data Formats

Operating Mode	Logic Level Conditions	
	Data = 0	Data = 1
HEAMA format	< 6 T_S cycles	$\geq 6 T_S$ cycles
5-/6-bit format	< 12 T_S cycles	$\geq 12 T_S$ cycles

Table 8-2 Long and Short Data Identification

Operating Mode	Long Data	Short Data
HEAMA format	$\geq 10 T_S$ cycles	< 2 T_S cycles
5-/6-bit format	$\geq 20 T_S$ cycles	< 4 T_S cycles

The 6-bit counter regulates data format detection. When the microcontroller detects a data leader, it sets the LONGDF bit of the clock status register, RMCS, to indicate long data. Figure 8-5 is a graphic representation of all the conditions for identifying the data format.



When the microcontroller detects a data trailer, the hardware automatically shuts off the supply to sampling clock T_S , which the 6-bit counter counts. The counter resets and the clock supply restarts at the next edge detection.

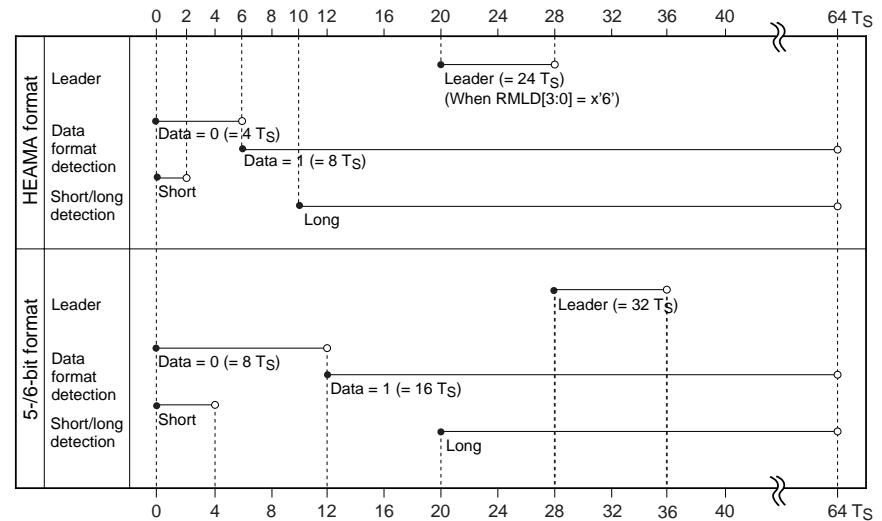


Figure 8-5 Conditions for Detecting Data Formats

8.3.5 Generating Interrupts

The IR remote signal receiver has four interrupt vectors: leader detection, trailer detection, 8-bit data reception detection, and pin edge detection. This section describes the operation for each of them.

8.3.5.1 Leader Detection

An interrupt occurs when the circuit detects a data leader. It detects leaders by testing the interval between remote signal edges. Table 8-3 shows the conditions.

Table 8-3 Leader Detection Conditions

Format	Edge Interval
HEAMA data leader	$(n - 4)T_S \leq \text{interval} < (n + 4)T_S$ ⁽¹⁾
5-/6-bit data leader	$28T_S \leq \text{interval} < 36T_S$

Note: 1. n = the leader value set in LD[3:0] of the RMLD register.

8.3.5.2 Trailer Detection

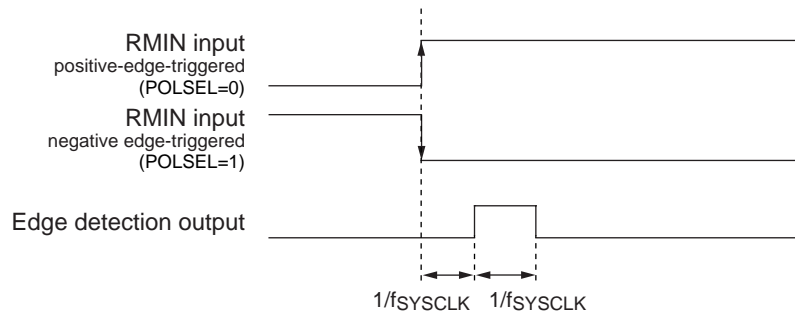
An interrupt occurs when the 6-bit counter overflows.

8.3.5.3 8-Bit Data Reception Detection

An interrupt occurs when the microcontroller loads 8-bit received data to the reception data transfer register, RMTR.

8.3.5.4 Pin Edge Detection

An interrupt occurs when the remote signal input pin, RMIN, is asserted. The POLSEL bit of RMIR sets the polarity of RMIN.



Note: $1/f_{SYSCLK} = 1/12 \text{ MHz} = 0.083 \mu\text{s}$

Figure 8-6 Pin Edge Detection

The detection output for all four interrupt vectors is an active high pulse asserted at intervals of $1/f_{SYSCLK}$. Bits 3 to 0 of the RMIR register control the interrupt vectors individually. A 0 disables the interrupt vector and a 1 enables it.

A remote signal interrupt sets the RMCIR flag of the RMCICL interrupt register (x'00FC76').

Use bit 7 (SP) in the RMLD register to toggle the noise filter sampling frequency between PWM6/PWM8 and PWM3/PWM5.

8.3.6 Controlling the SLOW Mode

The MN102H series microcontrollers have two operating modes: NORMAL and SLOW. (See section 3.1, “CPU Modes,” on page 72.) In SLOW mode, $f_{\text{SYSCLK}} = 2 \text{ MHz}$, which affects the frequencies of the PWM3 clock and noise filter sampling (PWM6/PWM8). Use the SPSLW bit (bit 6) of the RMLD register to change which clocks and noise filter sampling that you use.

Table 8-4 Differences between SLOW and NORMAL Modes

SPSLW (RMLD, bit 6)	Normal Mode ($f_{\text{SYSCLK}} = 12 \text{ MHz}$)			SLOW Mode ($f_{\text{SYSCLK}} = 2 \text{ MHz}$)		
	Noise filter sampling		Clock	Noise filter sampling		Clock
0	PWM6 (21.3 μs)	PWM8 (85.5 μs)	PWM3 (2.7 μs)	PWM6 (128 μs)	PWM8 (512 μs)	PWM3 (16 μs)
1	PWM3 (2.7 μs)	PWM5 (10.7 μs)	PWM1 (0.67 μs)	PWM3 (16 μs)	PWM5 (64 μs)	PWM1 (4.0 μs)

8.4 IR Remote Signal Receiver Control Registers

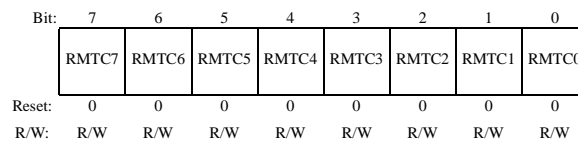
All registers in RMC block cannot be written by byte (by word only). Read by byte is possible.

Table 8-5 IR Remote Signal Receiver Registers

Register	Address	R/W	Function
RMTC	x'007EA4'	R/W	Remote signal frequency division control register
RMIR	x'007EA2'	R/W	Remote signal interrupt control register
RMIS	x'007EA0'	R/W	Remote signal interrupt status register
RMLD	x'007EAC'	R/W	Remote signal leader value set register
RMCS	x'007EA6'	R	Remote signal clock status register
RMSR	x'007EA8'	R	Remote signal reception data shift register
RMTR	x'007EAA'	R	Remote signal reception data transfer register

RMTC: Remote Signal Frequency Division Control Register

x'007EA4'



The edge detection circuit samples the remote signal with fSYSCCLK. Set the frequency divide-by ratio to meet this condition. If you do not, the microcontroller may interpret the data 1s and 0s incorrectly.

To identify the remote signal, the IR signal receiver generates a sampling clock, T_S , by dividing the PWM3 pulse by the value set in RMTC[7:0]. f_{PWM3} is $f_{SYSCCLK}$ divided by 2^5 (= 375 kHz, 2.7 μ s with a 4-MHz oscillator). The T_S cycle is the contents of RMTC + 1, so load a value from 1 to 255 for a division ratio from 2 to 256.

The microcontroller reads the value in the frequency division counter as a ones' complement number (each digit is complemented).

Set the RMTC value so that $T_S = T/2$, where T is the pulse width of the remote input signal. Table 8-6 shows how to define T for the different formats.



After the program sets the divide-by ratio for the frequency in RMTC, the read values may be incorrect until the circuit detects the next active edge of the remote signal.

Table 8-6 HEAMA and 5-/6-Bit Data Pulse Widths

HEAMA format	<div style="text-align: center;"> H L </div> Data 0: Data 1:
5-/6-bit format	Data 0: Data 1:

RMTC is an 8- or 16-bit access register.

All registers in RMC block cannot be written by byte (by word only). Read by byte is possible.

RMIR: Remote Signal Interrupt Control Register

x'007EA2'

Bit:	7	6	5	4	3	2	1	0
	MOD AUTO	MOD SEL	FILTR E	POL SEL	LEADER E	TRAILR E	DAT8 E	EDME E
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

RMIR controls the operating modes and interrupt operations for the receiver circuit. It is a 16-bit access register.

MODAUTO: Automatic operating mode detection on/off

- 0: Automatic detect
- 1: Fixed

MODSEL: Operating mode select

- 0: HEAMA format
- 1: 5-/6-bit format

FILTRE: Noise filter input multiplexer on/off

- 0: Pin level
- 1: Noise filter

POLSEL: Input polarity

- 0: Positive-edge-triggered
- 1: Negative-edge-triggered

LEADERE: Interrupt enable for leader detection

- 0: Disable
- 1: Enable

TRAILRE: Interrupt enable for trailer detection

- 0: Disable
- 1: Enable

DAT8E: Interrupt enable for 8-bit data reception detection

- 0: Disable
- 1: Enable

EDMEE: Interrupt enable for RMIN pin edge detection

- 0: Disable
- 1: Enable

IR Remote Signal Receiver

IR Remote Signal Receiver Control Registers

RMIS: Remote Signal Interrupt Status Register

x'007EA0'

Bit:	7	6	5	4	3	2	1	0
	BC RSTE	BC EDGS	FMT MON	DOMES D	M56BIT D	TRAILR D	DAT8 D	EDGE D
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W

RMIR indicates the detection and operation status of remote signal interrupts. It is a 16-bit access register.

BCRSTE: 8-bit data reception binary counter reset enable

0: Disable

1: Enable

BCEDGS: 8-bit data reception binary counter reset edge select

0: Reset at 1st edge

1: Reset at 2nd edge

FMTMON: Format monitor

0: HEAMA format

1: 5-/6-bit format

DOMESD: Interrupt request on HEAMA leader detection

0: No request

1: Request

M56BITD: Interrupt request on 5-/6-bit leader detection

0: No request

1: Request

TRAILRD: Interrupt request on trailer detection

0: No request

1: Request

DAT8D: Interrupt request on 8-bit reception detection

0: No request

1: Request

EDGED: Interrupt request on RMIN pin edge detection

0: No request

1: Request

RMLD: Remote Signal Leader Value Set Register

x'007EAC'

Bit:	7	6	5	4	3	2	1	0
	SP	SPSLW	—	—	LD3	LD2	LD1	LD0
Reset:	0	0	0	0	0	1	1	0
R/W:	R/W	R/W	R	R	R/W	R/W	R/W	R/W

RMLD is a 16-bit access register.

SP and SPSLW: Switch clock frequencies

00: Filter sampling cycle = f_{PWM6} , clock = PWM3

01: Filter sampling cycle = f_{PWM3} , clock = PWM1

10: Filter sampling cycle = f_{PWM8} , clock = PWM3

11: Filter sampling cycle = f_{PWM5} , clock = PWM1

LD[3:0]: HEAMA data leader value

Set the four MSBs of the 6-bit leader value for HEAMA data in LD[3:0].

This 4-bit setting must be between 0 and 60 T_S cycles. The default value is x'6'. The two LSBs of the leader are always 0.



fPWM1 = fSYSCLK/23,
fPWM3 = fSYSCLK/25,
fPWM5 = fSYSCLK/27,
fPWM6 = fSYSCLK/28, and
fPWM8 = fSYSCLK/210.



Do not set the leader value too small. Leader detection and data detection may occur simultaneously.

RMCS: Remote Signal Clock Status Register

x'007EA6'

Bit:	7	6	5	4	3	2	1	0
	LONG DF	SHORT DF	TSCNT5	TSCNT4	TSCNT3	TSCNT2	TSCNT1	TSCNT0
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R	R	R	R	R	R

RMCS indicates the result of the short/long data detection. It is a 16-bit access register.

LONGDF: Long data format detection

Set to 1 when long data is detected.

SHORTDF: Short data format detection

Set to 1 when short data is detected.

TSCNT[5:0]: 6-bit counter value

RMSR: Remote Signal Reception Data Shift Register

x'007EA8'

Bit:	7	6	5	4	3	2	1	0
	RMSR7	RMSR6	RMSR5	RMSR4	RMSR3	RMSR2	RMSR1	RMSR0
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R

RMTR: Remote Signal Reception Data Transfer Register

x'007EAA'

Bit:	7	6	5	4	3	2	1	0
	RMTR7	RMTR6	RMTR5	RMTR4	RMTR3	RMTR2	RMTR1	RMTR0
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R

The microcontroller shifts received data into RMSR, converting it to parallel data. After it shifts in 8 bits, it loads the data byte to RMTR. The CPU reads the data from RMTR. The data shifts from LSB to MSB. RMSR and RMTR are 8- or 16-bit access registers.

9 Closed-Caption Decoder

9.1 Description

The MN102H75K/85K contains two identical closed-caption decoder circuits, CCD0 and CCD1. The decoders extract encoded captions from composite video signals. Figure 9-1 provides a block diagram of the decoders, and section 9.3, “Functional Description,” on page 228, describes the circuit’s main blocks: the analog-to-digital converter, clamping circuit, sync separator circuit, data slicer, controller, and sampling circuit. Note that this section describes CCD0, but all descriptions apply to CCD1. Table 9-1 provides the pin names for each decoder.

Table 9-1 Pins Used for CCD0 and CCD1

Closed-Caption Decoder	Pin Name			
CCD0	CVBS0	VREFHS	CLH0	CLL0
CCD1	CVBS1	VREFLS	CLH0	CLL0

9.2 Block Diagram

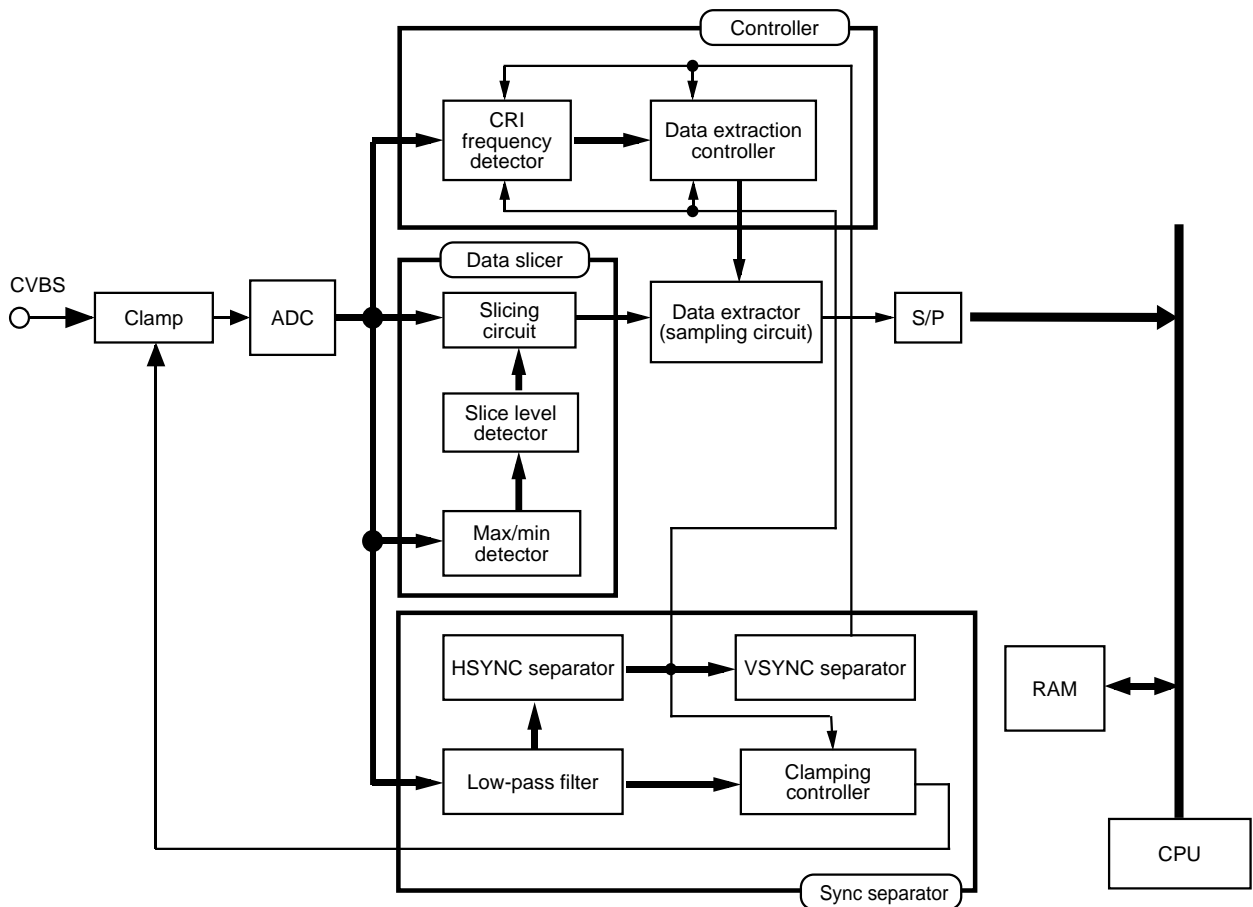


Figure 9-1 Closed-Caption Decoder Block Diagram

9.3 Functional Description

9.3.1 Analog-to-Digital Converter

The analog-to-digital converter (ADC) converts the clamped video signal to 8-bit digital data using a 12-MHz sampling clock. Figure 9-2 shows an example configuration using the recommended external pin connections. In this example, both caption decoders are used. Figure 9-3 shows the recommended connection when neither decoder is used, and figure 9-4 shows that when only CCD0 is used.



The constants shown in figures 9-2 to 9-4 are recommended values only. Operation at these values is not guaranteed.

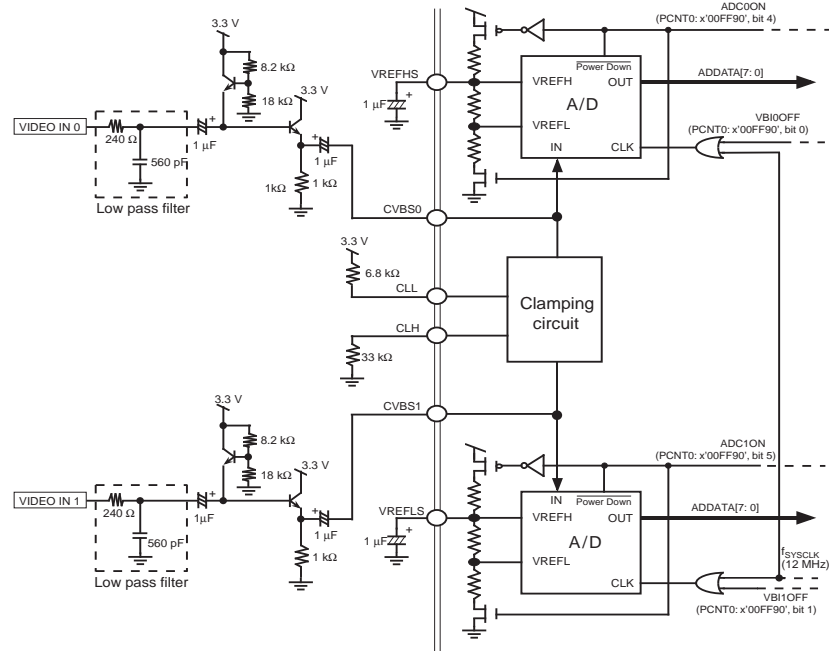


Figure 9-2 Recommended ADC Configuration

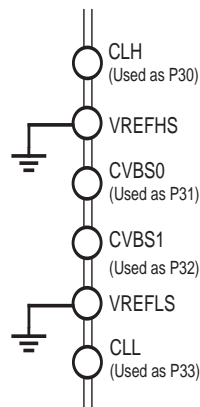


Figure 9-3 External Connection with Both CCD0 and CCD1 Unused

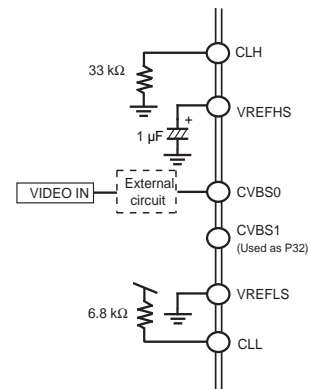


Figure 9-4 External Connection with Only CCD1 Unused

Closed-Caption Decoder

Functional Description

Table 9-2 Caption decoder register setting

		VBI control	ADC control	Clamp control
Use two caption decoders	caption 0	ON(PCNT0.bp0=0)	ON(PCNT0.bp4=1)	(P3MD.bp3,2,1)=(0,1,1)
	caption 1	ON(PCNT0.bp1=0)	ON(PCNT0.bp5=1)	
Use one caption decoder	caption 0	ON(PCNT0.bp0=0)	ON(PCNT0.bp4=1)	(P3MD.bp3,2,1)=(1,0,1)
	no caption 1	OFF(PCNT0.bp1=1)	OFF(PCNT0.bp5=0)	
No use caption decoder	no caption 0	OFF(PCNT0.bp0=1)	OFF(PCNT0.bp4=0)	(P3MD.bp3,2,1)=(0,0,0)
	no caption 1	OFF(PCNT0.bp1=1)	OFF(PCNT0.bp5=0)	

9.3.2 Clamping Circuit

This block clamps the input video signal (CVBS0, CVBS1).

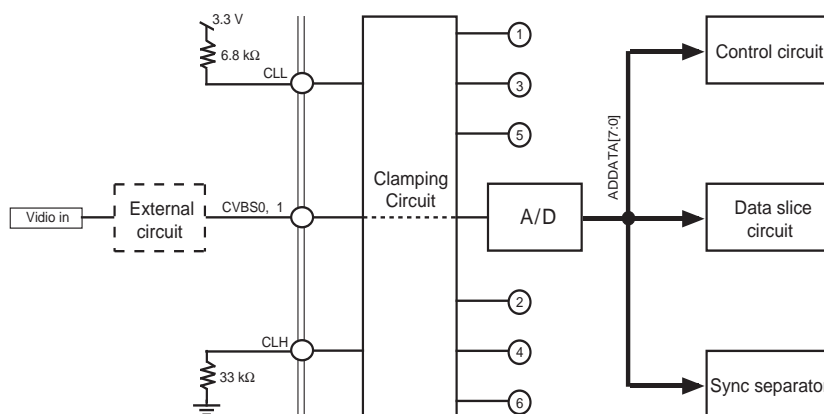


Figure 9-5 Clamping Circuit

The clamping circuit internal to the MN102H75K/85K provides three current sources—high, medium, and low. You can modify these current sources using external resistors R1 and R2. Within the clamping circuit, you can turn each of the current sources on and off in steps. The control bits for these currents are the same for sync tip and pedestal clamping, but the reference and compare levels are different. Table 9-3 provides these values for the two types of clamping, and table 9-4 shows how to control the three current levels so that the video signal matches the reference level.

Table 9-3 Clamping Reference and Compare Levels

Clamping Type	Reference Level		Compare Level	
	CCD0	CCD1	CCD0	CCD1
Sync tip clamping	16 (dec)	16 (dec)	Output from minimum detection circuit (value in SYNCMIN, x'007EC8', bits 6-0)	Output from minimum detection circuit (value in SYNCMINW, x'007EE8', bits 6-0)
Pedestal clamping	Value in PCLV, x'007ECC'	Value in PCLVW, x'007EEC'	Value in SYNCMIN, x'007EC8', bits 14-8	Value in SYNCMINW, x'007EE8', bits 14-8

Table 9-4 Current Level Control

Control Conditions	Current Source					
	Low Current		Medium Current		High Current	
	(1)	(2)	(3)	(4)	(5)	(6)
$10 \leq A$	Off	On	Off	On	Off	On
$4 \leq A \leq 9$	Off	On	Off	On	Off	Off
$1 \leq A \leq 3$	Off	On	Off	Off	Off	Off
$A = 0$	Off	Off	Off	Off	Off	Off
$-3 \leq A \leq -1$	On	Off	Off	Off	Off	Off
$-9 \leq A \leq -4$	On	Off	On	Off	Off	Off
$A \leq -10$	On	Off	On	Off	On	Off

- Notes: 1. A = compare level - reference level
 2. The numbers (1) to (6) correspond to the same number in figure 9-5.

Table 9-5 provides the registers used to control and monitor the clamping circuit. See the page number indicated for register and bit descriptions.

Table 9-5 Control Registers for Clamping Circuit

Register	Page	CCDO Address	CCD1 Address	Description
Register for selecting the low-pass filter				
NFSEL	242	x'007EC0'	x'007EE0'	Noise filter select register
Registers for controlling clamping				
SCMING	243	x'007EC4'	x'007EE4'	Minimum sync level detection interval set register
SYNCMIN	244	x'007EC8'	x'007EE8'	Sync and pedestal level register
BPPST	243	x'007EC6'	x'007EE6'	Backporch position register
CLAMP	245	x'007ECC'	x'007EEC'	Clamping control register
CLPCND 1	248	x'007EDC'	x'007EEC'	Clamping control signal status register 1

9.3.3 Sync Separator Circuit

A low-pass filter and a sync separator comprise this block. The sync separator extracts HSYNC and VSYNC from the composite video signal. Figure 9-6 shows a block diagram of the circuit, and table 9-6 provides the registers used to control and monitor it. See the page number indicated for register and bit descriptions.

Closed-Caption Decoder
Functional Description

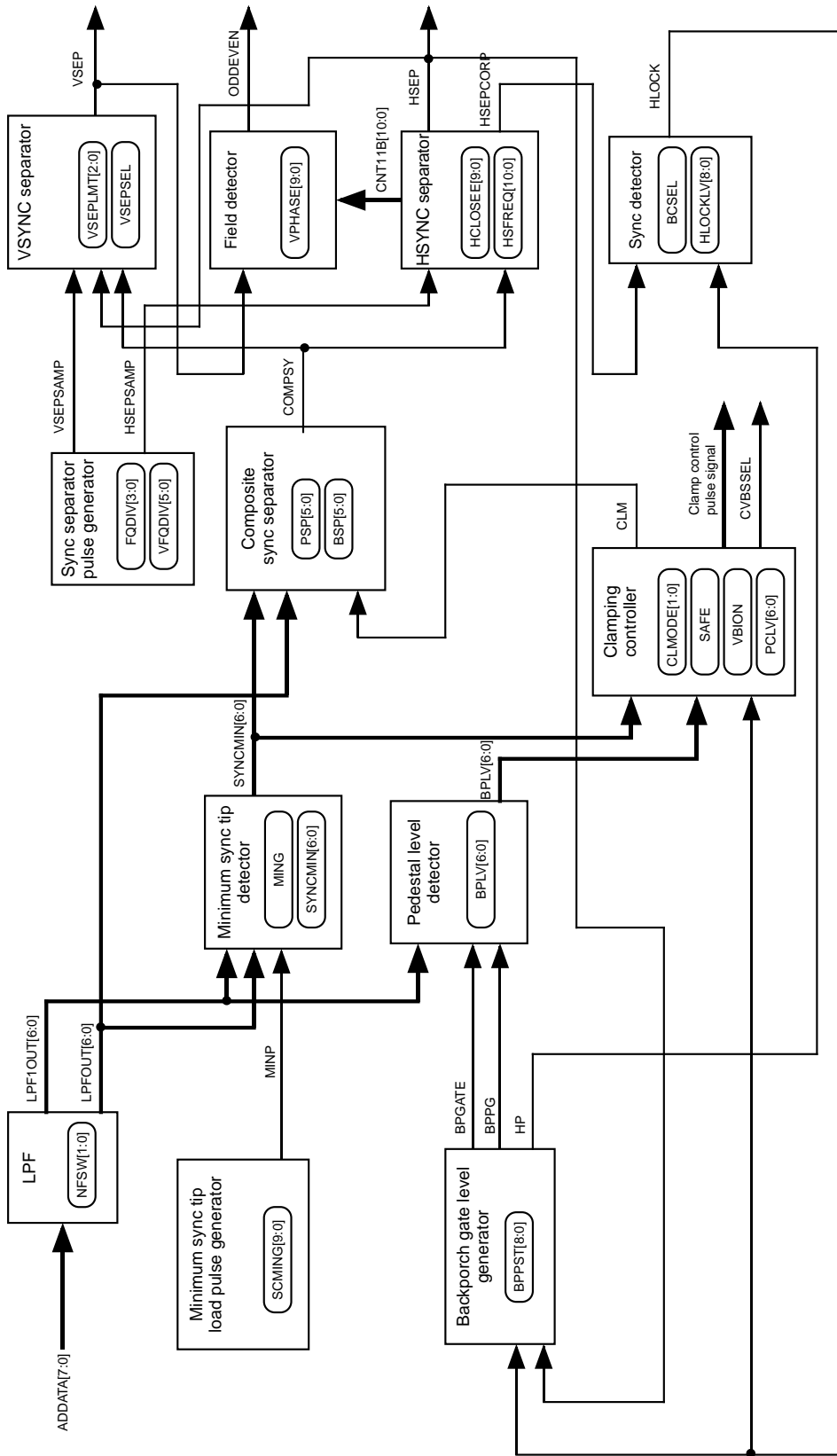


Figure 9-6 Sync Separator Circuit Block Diagram

Table 9-6 Control Registers for Sync Separator Circuit

Register	Page	CCDO Address	CCD1 Address	Description
Register for setting the sync separator level				
SPLV	244	x'007ECA'	x'007EEA'	Sync separator level set register
Register for controlling the sync separator clock				
FQSEL	243	x'007EC2'	x'007EE2'	Frequency select register
Registers for controlling the HSYNC separator				
HSEP1	246	x'007ECE'	x'007EEE'	HSYNC separator control register 1
HSEP2	246	x'007ED0'	x'007EF0'	HSYNC separator control register 2
HLOCKLV	246	x'007ED4'	x'007EF4'	Sync separator detection control register 1
HDISTW	247	x'007ED6'	x'007EF6'	Sync separator detection control register 2
Register for controlling the VSYNC separator				
VCNT	247	x'007ED8'	x'007EF8'	VSYNC separator control register
Register for controlling the field detection				
FIELD	246	x'007ED2'	x'007EF2'	Field detection control register
Register for monitoring the sync separator status				
HVCOND	247	x'007EDA'	x'007EFA'	Sync separator status register

9.3.3.1 HSYNC Separator

The HSYNC separator extracts the HSYNC signal from the composite sync signal using the sampling clock generated by the sync separator clock pulse generator. This circuit also secures and interpolates the HSYNC signal.

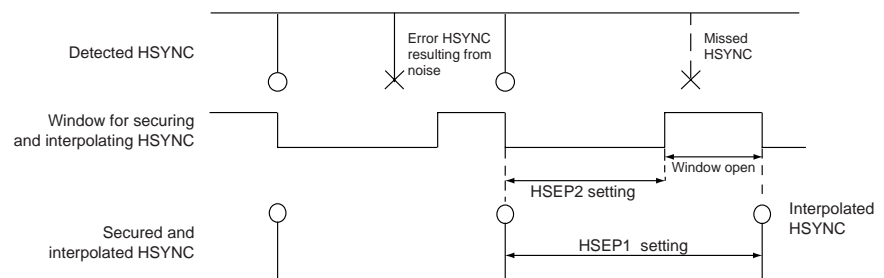


Figure 9-7 HSYNC Securement and Interpolation

As shown in figure 9-7, noise can cause the HSYNC detection circuit to both miss HSYNC pulses and add erroneous ones. The HSYNC separator contains a window circuit to correct these errors. The open and close timing for this window is set in the HSEP1 and HSEP2 registers, and the unit is the sampling clock for the HSYNC separator.

The circuit counts a corrected and interpolated HSYNC signal. If the count is greater than that set in the HLOCKLV register, within the interval set in the HDISTW register, the HLOCK bit of HVCOND sets to 0, indicating an asynchronous state. This allows the device to determine the quality of the signal.

9.3.3.2 VSYNC Separator

The VSYNC separator extracts the VSYNC signal from the composite signal. Like the HSYNC separator, it contains programmable methods for eliminating noise. The VCNT register contains these settings. Masking the 0H to 127H range (by setting the VSEPSSEL bit of VCNT to 0) prevents VSYNC errors due to noise. See figure 9-8.

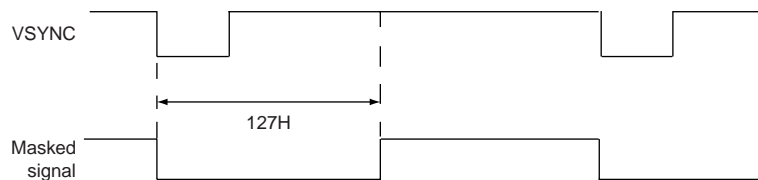


Figure 9-8 VSYNC Masking

9.3.3.3 Field Detection Circuit

The field detection circuit detects the phase difference between VSYNC and HSYNC, based on the setting in the VPHASE[9:0] field of the FIELD register. This setting is in units of the sampling clock for the HSYNC separator. The results of the field detection are stored in the ODDEVEN bit of FIELD.

9.3.4 Data Slicer

The data slicer contains the maximum and minimum detection circuits, the slice level calculator, and the slicer. The circuit compares the 8-bit digital values output from the ADC to the slice level, which can be calculated by the hardware or set in the software. It then outputs the results in serial 0s and 1s.

The data slicer calculates the slice level (the level above which a signal is 1 and below which is 0) from the maximum and minimum clock run-in (CRI) pulses occurring in the interval between the settings in the CRIIS and CRIIE registers.

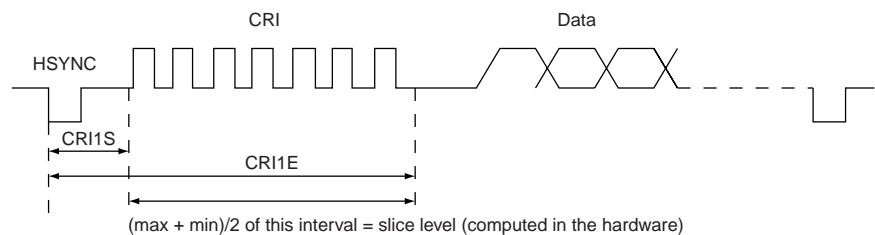


Figure 9-9 Data Slice Level Calculation

Table 9-7 provides the registers used to control and monitor the data slicer. See the page number indicated for register and bit descriptions.

Table 9-7 Control Registers for Data Slicer

Register	Page	CCDO Address	CCD1 Address	Description
CRI1S	240	x'007E10'	x'007E30'	CRI capture start timing control register 1
CRI1E	241	x'007E12'	x'007E32'	CRI capture stop timing control register 1
MAXMIN	238	x'007E02'	x'007E22'	CRI interval maximum and minimum register
SLICE	238	x'007E04'	x'007E24'	VBI data slice level register
FCCNT	237	x'007E00'	x'007E20'	VBI decoding format select register

9.3.5 Controller and Sampling Circuit

The control circuit contains the CRI window generator and the caption data window generator. The sampling circuit extracts the 16-bit caption data (503 kHz) from the serial data output from the data slicer at the 12-MHz ADC sampling rate.

Table 9-8 provides the registers used to control and monitor these two blocks. See the page number indicated for register and bit descriptions.

Table 9-8 Control Registers for Controller and Sampling Circuit

Register	Page	CCDO Address	CCD1 Address	Description
Registers for detecting CRI and generating sampling clock				
CRI2S	241	x'007E14'	x'007E34'	CRI capture start timing control register 2
CRI2E	241	x'007E16'	x'007E36'	CRI capture stop timing control register 2
CRIFA	240	x'007E0C'	x'007E2C'	CRI frequency width register A
CRIFB	240	x'007E0E'	x'007E2E'	CRI frequency width register B
Registers for controlling data capture				
DATAS	241	x'007E18'	x'007E38'	Data capture start timing control register
DATAE	242	x'007E1A'	x'007E3A'	Data capture stop timing control register
CAPDATA	239	x'007E0A'	x'007E2A'	Caption data capture register
HNUM	239	x'007E06'	x'007E26'	HSYNC count register

Closed-Caption Decoder

Functional Description

9.3.5.1 CRI Detection for Sampling Clock Generation

The decoder captures the caption data on the rising edge of the CRI pulse. To achieve this, it contains a circuit to accurately detect the CRI pulse rises and to generate a data sampling clock.

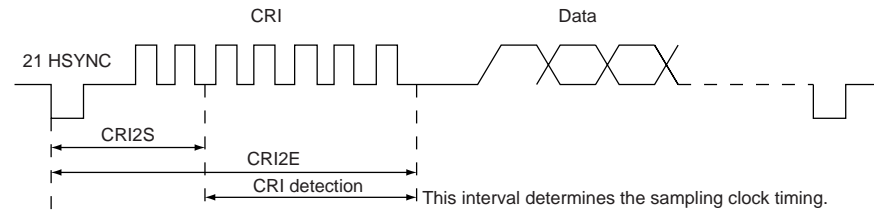


Figure 9-10 Sampling Clock Timing Determination

9.3.5.2 Data Capture Control

The DATAS and DATAE registers control the data capture timing, and the CAPDATA register stores the caption data captured on the sampling clock generated through CRI detection. The HNUM register controls interrupt timing.

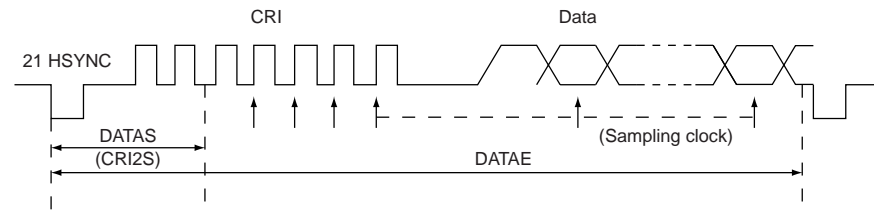


Figure 9-11 Caption Data Capture Timing

9.4 Closed-Caption Decoder Registers

All registers in Closed-caption Decoder block cannot be written by byte (by word only). Read by byte is possible.

Table 9-9 Closed-Caption Decoder Register

Register	CCD0 Address	CCD1 Address	R/W	Description
FCCNT	x'007E00'	x'007E20'	R/W	VBI decoding format select register
MAXMIN	x'007E02'	x'007E22'	R	CRI interval maximum and minimum register
SLICE	x'007E04'	x'007E24'	R/W	VBI data slice level register
HNUM	x'007E06'	x'007E26'	R	HSYNC count register
ACQ1	x'007E08'	x'007E28'	R/W	ACQ capture timing control register 1
CAPDATA	x'007E0A'	x'007E2A'	R	Caption data capture register
CRIFA	x'007E0C'	x'007E2C'	R	CRI frequency width register A
CRIFB	x'007E0E'	x'007E2E'	R	CRI frequency width register B
CRI1S	x'007E10'	x'007E30'	R/W	CRI capture start timing control register 1
CRI1E	x'007E12'	x'007E32'	R/W	CRI capture stop timing control register 1
CRI2S	x'007E14'	x'007E34'	R/W	CRI capture start timing control register 2
CRI2E	x'007E16'	x'007E36'	R/W	CRI capture stop timing control register 2
DATAS	x'007E18'	x'007E38'	R/W	Data capture start timing control register
DATAE	x'007E1A'	x'007E3A'	R/W	Data capture stop timing control register
STAP	x'007E1C'	x'007E3C'	R/W	Sampling start position register (software setting)
FCPNUM	x'007E1E'	x'007E3E'	R	Sampling start position register (hardware calculation)
NFSEL	x'007EC0'	x'007EE0'	R/W	Noise filter select register
FQSEL	x'007EC2'	x'007EE2'	R/W	Frequency select register
SCMING	x'007EC4'	x'007EE4'	R/W	Minimum sync level detection interval set register
BPPST	x'007EC6'	x'007EE6'	R/W	Backporch position register
SYNCMIN	x'007EC8'	x'007EE8'	R	Sync and pedestal level register
SPLV	x'007ECA'	x'007EEA'	R/W	Sync separator level set register
CLAMP	x'007ECC'	x'007EEC'	R/W	Clamping control register
HSEP1	x'007ECE'	x'007EEE'	R/W	HSYNC separator control register 1
HSEP2	x'007ED0'	x'007EF0'	R/W	HSYNC separator control register 2
FIELD	x'007ED2'	x'007EF2'	R/W	Field detection control register
HLOCKLV	x'007ED4'	x'007EF4'	R/W	Sync separator detection control register 1
HDISTW	x'007ED6'	x'007EF6'	R/W	Sync separator detection control register 2
VCNT	x'007ED8'	x'007EF8'	R/W	VSYNC separator control register
HVCOND	x'007EDA'	x'007EFA'	R	Sync separator status register
CLPCND1	x'007EDC'	x'007EFC'	R	Clamping control signal status register 1
SBFNUM	x'007F4C'	x'007F6C'	R	Sampling start position register
TESTA	x'007F4E'	x'007F6E'	R	Test register

Closed-Caption Decoder

Closed-Caption Decoder Registers



For designs using the closed-caption decoder, always tie the FCCNT register to x'0008'.

FCCNT: VBI Decoding Format Select Register (FCCNTW)

x'007E00'
x'007E20'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	FCP SEL	SYNC SEL	HCNT SEL1	HCNT SEL0	SLICE SEL	SLICE LD2	SLICE LD1	SLICE LD0	SLPUL SEL	CRIC SEL	NCRIG SEL	CNT STAP4	CNT STAP3	CNT STAP2	CNT STAP1	CNT STAP0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

This register contains the settings for selecting either a hardware or software slice level and for setting the data capture format. When you do not use a bit or field in this register, tie it to the setting indicated below.

FCPSEL: Hard/soft sampling start position select

- 0: Select hardware calculation
- 1: Select software setting (set in SFTSTAP[10:0] field of STAP)

SYNCSEL: Sync signal select (HSYNC/VSYNC)

Tie this bit to 0.

HCNTSEL[1:0]: HSYNC count value select

When this field is unused, tie it to b'00'.

- 00: When odd, add 1 to the HSYNC count value
- 01: When even, add 1 to the HSYNC count value
- 10: No change to the HSYNC count value
- 11: Reserved

SLICESEL: Hard/soft slice level select

- 0: Select hardware calculation
- 1: Select software setting

SLICELD[2:0]: Slice level load timing select

When this field is unused, tie it to b'000'.

- 000: 1H
- 001: 2H
- 010: 4H
- 011: 8H
- 100: 1 field
- 101: 2 fields
- 110: 4 fields
- 111: 8 fields

SLPULSEL: Polarity select for the CRI cycle transition detection

- 0: Detect 0 to 1 transitions
- 1: Detect 1 to 0 transitions

CRICSEL: Detection interval select for the CRI frequency width

- 0: CRI capture interval only
- 1: CRI capture interval and transition detection interval

NCRIGSEL: Sampling pulse generation interval

- 0: Disable the CRI capture interval
- 1: Enable the CRI capture interval

CNTSTAP[4:0]: Caption data sampling start position

The higher the setting in this field, the later the start position. The valid range is x'00' to x'1F'.

MAXMIN: CRI Interval Maximum and Minimum Register
(MAXMINW)

x'007E02'
x'007E22'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	MAX7	MAX6	MAX5	MAX4	MAX3	MAX2	MAX1	MAX0	MIN7	MIN6	MIN5	MIN4	MIN3	MIN2	MIN1	MIN0
Reset:	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
R/W:	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

MAX[7:0]: Maximum value during the CRI interval
 Valid range: x'00' to x'FF'

MIN[7:0]: Minimum value during the CRI interval
 Valid range: x'00' to x'FF'

SLICE: VBI Data Slice Level Register
(SLICEW)

x'007E04'
x'007E24'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SLSF 7	SLSF 6	SLSF 5	SLSF 4	SLSF 3	SLSF 2	SLSF 1	SLSF 0	SLHD 7	SLHD 6	SLHD 5	SLHD 4	SLHD 3	SLHD 2	SLHD 1	SLHD 0
Reset:																
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R	R	R	R	R	R	R	R

SLSF[7:0]: VBI data slice level—software setting
 Valid range: x'00' to x'FF'

SLHD[7:0]: VBI data slice level—hardware calculation
 Valid range: x'00' to x'FF'

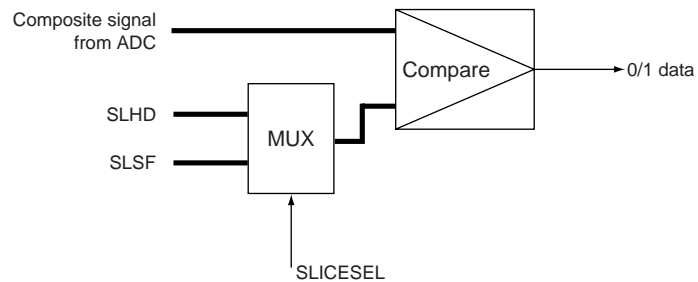


Figure 9-12 SLSF and SLHD Multiplexing

Closed-Caption Decoder

Closed-Caption Decoder Registers

HNUM: HSYNC Count Register (HNUMW)

x'007E06'
x'007E26'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	—	VBI- IRQ 4	VBI- IRQ 3	VBI- IRQ 2	VBI- IRQ 1	VBI- IRQ 0	SB FLAG	—	—	HNU M4	HNU M3	HNU M2	HNU M1	HNU M0
Reset:	0	0	0	1	1	1	1	1	0	0	0	0	0	0	0	0
R/W:	R	R	R	R/W	R/W	R/W	R/W	R/W	R	R	R	R	R	R	R	R

This register allows you to time the interrupt occurring after the line 21 data capture to a line other than line 21.

VBIRQ[4:0]: VBI interrupt timing control

In this field, set the H line number, from 0 to 25, for the VBI interrupt. You must set this field to x'13' or higher.

SBFLAG: Start bit detection flag

0: No start bit detected

1: Start bit detected

HNUM[4:0]: HSYNC count during the VBI interval

This field indicates the H line number, from 0 to 25.



For designs using the closed-caption decoder, always tie the ACQ1 register to x'1312'.

ACQ1: ACQ Capture Timing Control Register 1 (ACQ1W)

x'007E08'
x'007E28'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	—	ACQ1 E4	ACQ1 E3	ACQ1 E2	ACQ1 E1	ACQ1 E0	—	—	—	ACQ1 S4	ACQ1 S3	ACQ1 S2	ACQ1 S1	ACQ1 S0
Reset:	0	0	0	1	1	1	1	1	0	0	0	1	1	1	1	1
R/W:	R	R	R	R/W	R/W	R/W	R/W	R/W	R	R	R	R/W	R/W	R/W	R/W	R/W

ACQ1E[4:0]: Stop position for ACQ capture 1

Valid range: x'00' to x'25'

ACQ1S[4:0]: Start position for ACQ capture 1

Valid range: x'00' to x'25'

CAPDATA: Caption Data Capture Register (CAPDATAW)

x'007E0A'
x'007E2A'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CAP DA15	CAP DA14	CAP DA13	CAP DA12	CAP DA11	CAP DA10	CAP DA9	CAP DA8	CAP DA7	CAP DA6	CAP DA5	CAP DA4	CAP DA3	CAP DA2	CAP DA1	CAP DA0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

CAPDA[15:0]: Caption data

This register stores the 16-bit captured caption data.

CRIFA: CRI Frequency Width Register A
(CRIFAW)

x'007E0C'
x'007E2C'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CRI2 FQW7	CRI2 FQW6	CRI2 FQW5	CRI2 FQW4	CRI2 FQW3	CRI2 FQW2	CRI2 FQW1	CRI2 FQW0	CRI1 FQW7	CRI1 FQW6	CRI1 FQW5	CRI1 FQW4	CRI1 FQW3	CRI1 FQW2	CRI1 FQW1	CRI1 FQW0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

The CRIFA and CRIFB registers store the CRI cycles from rising edge to rising edge, for monitoring whether the CRIs were detected correctly during this period.

CRI2FQW[7:0]: CRI frequency width 2

This field indicates the width, in clock units, from the second to the third rising edge after the CRI.

CRI1FQW[7:0]: CRI frequency width 1

This field indicates the width, in clock units, from the first to the second rising edge after the CRI.

CRIFB: CRI Frequency Width Register B
(CRIFBW)

x'007E0E'
x'007E2E'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CRI4 FQW7	CRI4 FQW6	CRI4 FQW5	CRI4 FQW4	CRI4 FQW3	CRI4 FQW2	CRI4 FQW1	CRI4 FQW0	CRI3 FQW7	CRI3 FQW6	CRI3 FQW5	CRI3 FQW4	CRI3 FQW3	CRI3 FQW2	CRI3 FQW1	CRI3 FQW0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

CRI4FQW[7:0]: CRI frequency width 4

This field indicates the width, in clock units, from the fourth to the fifth rising edge after the CRI.

CRI3FQW[7:0]: CRI frequency width 3

This field indicates the width, in clock units, from the third to the fourth rising edge after the CRI.

CRI1S: CRI Capture Start Timing Control Register 1
(CRI1SW)

x'007E10'
x'007E30'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	—	—	—	CRI1S 10	CRI1S 9	CRI1S 8	CRI1S 7	CRI1S 6	CRI1S 5	CRI1S 4	CRI1S 3	CRI1S 2	CRI1S 1	CRI1S 0
Reset:	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1
R/W:	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

CRI1S[10:0]: Start position for CRI capture 1

Valid range: x'000' to x'7FF'

CRI1E: CRI Capture Stop Timing Control Register 1 **x'007E12'**
(CRI1EW) **x'007E32'**

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	—	—	—	CRI1E 10	CRI1E 9	CRI1E 8	CRI1E 7	CRI1E 6	CRI1E 5	CRI1E 4	CRI1E 3	CRI1E 2	CRI1E 1	CRI1E 0
Reset:	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1
R/W:	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

CRI1E[10:0]: Stop position for CRI capture 1
 Valid range: x'000' to x'7FF'

CRI2S: CRI Capture Start Timing Control Register 2 **x'007E14**
(CRI2SW) **x'007E34'**

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	—	—	—	CRI2S 10	CRI2S 9	CRI2S 8	CRI2S 7	CRI2S 6	CRI2S 5	CRI2S 4	CRI2S 3	CRI2S 2	CRI2S 1	CRI2S 0
Reset:	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1
R/W:	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

CRI2S[10:0]: Start position for CRI capture 2
 Valid range: x'000' to x'7FF'

CRI2E: CRI Capture Stop Timing Control Register 2 **x'007E16'**
(CRI2EW) **x'007E36'**

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	—	—	—	CRI2E 10	CRI2E 9	CRI2E 8	CRI2E 7	CRI2E 6	CRI2E 5	CRI2E 4	CRI2E 3	CRI2E 2	CRI2E 1	CRI2E 0
Reset:	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1
R/W:	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

CRI2E[10:0]: Stop position for CRI capture 2
 Set this field so that the last CRI rising edge is included. The valid range is x'000' to x'7FF'.

DATAS: Data Capture Start Timing Control Register **x'007E18'**
(DATASW) **x'007E38'**

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	—	—	—	DATA S 10	DATA S 9	DATA S 8	DATA S 7	DATA S 6	DATA S 5	DATA S 4	DATA S 3	DATA S 2	DATA S 1	DATA S 0
Reset:	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1
R/W:	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

DATAS[10:0]: Start position for data capture
 Set this field to the same start position as that for CRI detection (set in CRI2S). The valid range is x'000' to x'7FF'.

DATAE: Data Capture Stop Timing Control Register
(DATAEW)

x'007E1A'
x'007E3A'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	—	—	—	DATA E 10	DATA E 9	DATA E 8	DATA E 7	DATA E 6	DATA E 5	DATA E 4	DATA E 3	DATA E 2	DATA E 1	DATA E 0
Reset:	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1
R/W:	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

DATAE[10:0]: Stop position for data capture

Set this value high enough to allow the last data to be captured. The valid range is x'000' to x'7FF'.

STAP: Sampling Start Position Register (Software Setting)
(STAPW)

x'007E1C'
x'007E3C'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	—	—	—	SFT STAP 10	SFT STAP 9	SFT STAP 8	SFT STAP 7	SFT STAP 6	SFT STAP 5	SFT STAP 4	SFT STAP 3	SFT STAP 2	SFT STAP 1	SFT STAP 0
Reset:	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1
R/W:	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

SFTSTAP[10:0]: Software setting for sampling start position (in clock units)

FCPNUM: Sampling Start Position Register (Hardware Calculation)
(FCPNUMW)

x'007E1E'
x'007E3E'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	—	—	—	FCP NUM 10	FCP NUM 9	FCP NUM 8	FCP NUM 7	FCP NUM 6	FCP NUM 5	FCP NUM 4	FCP NUM 3	FCP NUM 2	FCP NUM 1	FCP NUM 0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

FCPNUM[10:0]: Sampling start position calculated by the hardware

NFSEL: Noise Filter Select Register
(NFSELW)

x'007EC0'
x'007EE0'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	MING	—	—	—	—	—	—	NFSW 1	NFSW 0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R/W	R	R	R	R	R	R	R/W	R/W

This register selects the low-pass filter, which eliminates noise and high-frequency signals that are unnecessary to the sync separator and the clamping controller. The recommended setting for NFSEL is x'0000'.

MING: Output select for noise filter detecting minimum sync tip

- 0: Low-pass filter 1
- 1: Low-pass filter 2, 3, or 4 (set in NFSW[1:0])

NFSW[1:0]: Noise filter switch (for composite sync separator)

- 00: Low-pass filter 3
- 01: Low-pass filter 4
- 10: Low-pass filter 2
- 11: Low-pass filter 1

The cutoff frequencies for low-pass filters 1 to 4 are lower in ascending order, so that low-pass filter 4 eliminates the highest amount of noise.

Closed-Caption Decoder
Closed-Caption Decoder Registers

FQSEL: Frequency Select Register
(FQSELW)

x'007EC2'
x'007EE2'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	VFQ DIV5	VFQ DIV4	VFQ DIV3	VFQ DIV2	VFQ DIV1	VFQ DIV0	—	—	—	—	FQ DIV3	FQ DIV2	FQ DIV1	FQ DIV0
Reset:	0	0	1	1	1	1	1	1	0	0	0	0	0	0	0	1
R/W:	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R	R	R	R	R/W	R/W	R/W	R/W

In this register, set the sampling cycle for separating the HSYNC and VSYNC signals from the composite sync signal. The recommended setting is x'1F01'.

VFQDIV[5:0]: Sampling frequency setting for VSYNC separator

In this field, set the ratio by which to divide the sampling frequency for the HSYNC separator.

FQDIV[3:0]: Sampling frequency setting for HSYNC separator

In this field, set the ratio by which to divide the A/D sampling frequency.

SCMING: Minimum Sync Level Detection Interval Set Register
(SCMINGW)

x'007EC4'
x'007EE4'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	SC MING 9	SC MING 8	SC MING 7	SC MING 6	SC MING 5	SC MING 4	SC MING 3	SC MING 2	SC MING 1	SC MING 0
Reset:	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

SCMING[9:0]: Interval setting for the minimum sync level detection

Set the HSYNC cycle in this field in ADC clock units. This is the interval used for detecting the sync tip level for sync tip clamping. The valid range is x'000' to x'3FF'. Note that the HSYNC cycle set in this register is only used for detecting the minimum sync level. You must also set the correction HSYNC cycle in HSEP1.

For the NTSC format, the setting for this register is x'02FA', calculated as follows:

$$(A/D \text{ sampling frequency}) \times (\text{HSYNC cycle}) = 12 \text{ MHz} \times 63 \mu\text{s} = \text{x}'02\text{FA}'$$

BPPST: Backporch Position Register
(BPPSTW)

x'007EC6'
x'007EE6'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	BP PST8	BP PST7	BP PST6	BP PST5	BP PST4	BP PST3	BP PST2	BP PST1	BP PST0
Reset:	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0
R/W:	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

BPPST[8:0]: Backporch start position for the leading edge of HSYNC

Use this register to specify the position for capturing the pedestal level value used during pedestal clamping. Specify a number of ADC clocks after the leading edge of HSYNC. The valid range is x'000' to x'1FF', and the recommended setting is x'003C'.

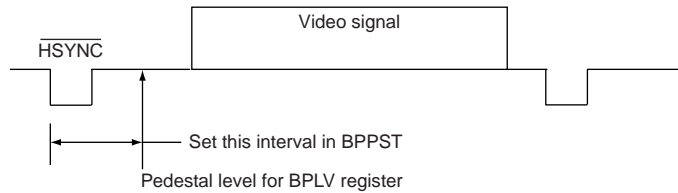


Figure 9-13 Backporch Position Setting

SYNCMIN: Sync and Pedestal Level Register
(SYNCMINW)

x'007EC8'
x'007EE8'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	BPLV6	BPLV5	BPLV4	BPLV3	BPLV2	BPLV1	BPLV0	—	SYNC MIN6	SYNC MIN5	SYNC MIN4	SYNC MIN3	SYNC MIN2	SYNC MIN1	SYNC MIN0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

BPLV[6:0]: Pedestal level

This register stores the pedestal level captured from the position specified in BPPST.

SYNCMIN[6:0]: Minimum sync level

This field stores the minimum level (the sync tip level) detected during the interval set in the SCMING register. For sync tip clamping, you should control clamping so as to make this value 16 (dec).

SPLV: Sync Separator Level Set Register
(SPLVW)

x'007ECA'
x'007EEA'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	BSP5	BSP4	BSP3	BSP2	BSP1	BSP0	—	—	PSP5	PSP4	PSP3	PSP2	PSP1	PSP0
Reset:	0	0	0	1	1	0	0	0	0	0	0	1	1	0	0	0
R/W:	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R	R	R/W	R/W	R/W	R/W	R/W	R/W

The sync separator uses the value set in this register to separate the composite sync signal from the composite video signal.

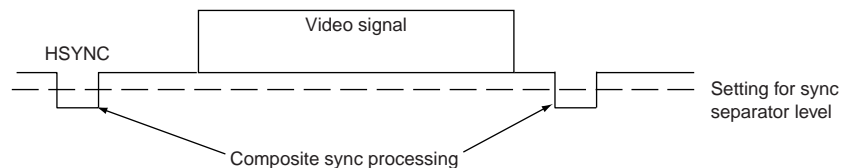


Figure 9-14 Sync Separator Level

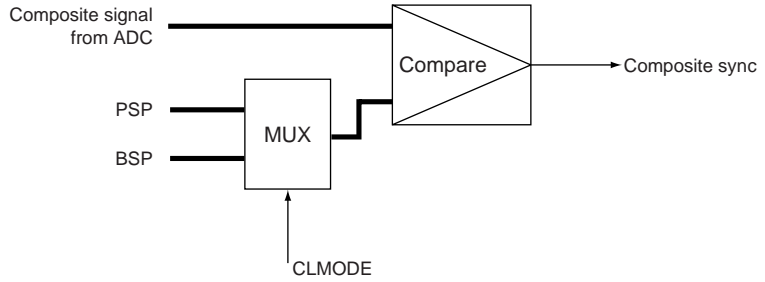


Figure 9-15 BSP and PSP Multiplexing

BSP[5:0]: Sync separator level for pedestal clamping
 Sync separator level = (sync tip level/2) + BSP[5:0]. The valid range is x'00' to x'3F'.

PSP[5:0]: Sync separator level for sync tip clamping
 Valid range: x'00' to x'3F'

CLAMP: Clamping Control Register
(CLAMPW

x'007ECC'
x'007EEC')

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	PCLV6	PCLV5	PCLV4	PCLV3	PCLV2	PCLV1	PCLV0	SAFE	—	—	VBION	—	—	CLMODE	CLMODE
Reset:	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0
R/W:	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R	R	R/W	R	R	R/W	R/W

Use this register to set the clamping mode (sync tip or pedestal clamping).

PCLV[6:0]: Pedestal clamping level setting
 Set the reference level for pedestal clamping in this field. The valid range is x'00' to x'7F'.

VBION: VBI setting
 0: VBI off
 1: VBI on

SAFE: Clamping current source select
 This bit is the capacity switch for (5) and (6) in figure 9-5 on page 229.
 0: High current source ((5) and (6) capacity high)
 1: Medium current source ((5) and (6) capacity low)

CLMODE[1:0]: Clamping mode setting
 00: Automatic switching (depends on the cycle state)
 01: Sync tip clamping only
 10: Pedestal clamping only
 11: Clamping off

HSEP1: HSYNC Separator Control Register 1
(HSEP1W)

x'007ECE'
x'007EEE'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	—	—	—	HS FREQ 10	HS FREQ 9	HS FREQ 8	HS FREQ 7	HS FREQ 6	HS FREQ 5	HS FREQ 4	HS FREQ 3	HS FREQ 2	HS FREQ 1	HS FREQ 0
Reset:	0	0	0	0	0	0	0	1	0	0	0	0	1	1	0	0
R/W:	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

HSFREQ[10:0]: Correction HSYNC frequency

Set the correction HSYNC cycle in this field in HSYNC separator sampling clock units. The valid range is x'000' to x'7FF', and the recommended setting is x'010C'.

HSEP2: HSYNC Separator Control Register 2
(HSEP2E)

x'007ED0'
x'007EF0'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	H CLOSE E9	H CLOSE E8	H CLOSE E7	H CLOSE E6	H CLOSE E5	H CLOSE E4	H CLOSE E3	H CLOSE E2	H CLOSE E1	H CLOSE E0
Reset:	0	0	0	0	0	0	0	0	1	1	1	0	0	1	0	0
R/W:	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

HCLOSEE[9:0]: Start position for HSYNC detection

Set the position in HSYNC separator sampling clock units. The valid range is x'000' to x'3FF', and the recommended setting is x'00E4'.

FIELD: Field Detection Control Register
(FIELDW)

x'007ED2'
x'007EF2'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ODD EVEN	—	—	—	—	—	V PHASE 9	V PHASE 8	V PHASE 7	V PHASE 6	V PHASE 5	V PHASE 4	V PHASE 3	V PHASE 2	V PHASE 1	V PHASE 0
Reset:	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

ODDEVEN: Field detection signal

- 0: Odd field
- 1: Even field

VPHASE[9:0]: Phase difference setting for VSYNC and HSYNC

Set the phase difference in HSYNC separator sampling clock units. The valid range is x'000' to x'3FF'.

HLOCKLV: Sync Separator Detection Control Register 1
(HLOCKLVW)

x'007ED4'
x'007EF4'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	H LOCK LV8	H LOCK LV7	H LOCK LV6	H LOCK LV5	H LOCK LV4	H LOCK LV3	H LOCK LV2	H LOCK LV1	H LOCK LV0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
R/W:	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

HLOCKLV[8:0]: Sync separator detection threshold

This value is compared to the count of the corrected HSYNC. The valid range is x'000' to x'1FF', and the recommended setting is x'0008'.

- HLOCKLV ≤ HSYNC count → asynchronous
- HLOCKLV > HSYNC count → synchronous

HDISTW: Sync Separator Detection Control Register 2
HDISTWW

x'007ED6'
x'007EF6'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	HDIST W8	HDIST W7	HDIST W6	HDIST W5	HDIST W4	HDIST W3	HDIST W2	HDIST W1	HDIST W0
Reset:	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

HDISTW[8:0]: HSYNC count setting the interval for sync separation detection
 In this register, set the interval during which sync separation occurs. The valid range is x'000' to x'1FF' and the recommended setting is x'0100'. For NTSC format, set the register to 525 (dec), indicating an HSYNC count of 525 VSYNC cycles. The recommended setting is x'0100'.

VCNT: VSYNC Separator Control Register
(VCNTW)

x'007ED8'
x'007EF8'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	VSEP SEL	—	—	—	—	—	VSEP LMT2	VSEP LMT1	VSEP LMT0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1
R/W:	R	R	R	R	R	R	R	R/W	R	R	R	R	R	R/W	R/W	R/W

VSEPSEL: VSYNC signal select
 0: 0H to 127H VSYNC separation mask
 1: No mask

VSEP LMT[2:0]: VSYNC separation detection threshold

HVCOND: Sync Separator Status Register
(HVCONDW)

x'007EDA'
x'007EFA'

Bit:	7	6	5	4	3	2	1	0
	—	—	—	STPN	COMP SY	VSEP	HSEP	HLOCK
Reset:	0	0	0	1	1	0	0	0
R/W:	R	R	R	R	R	R	R	R

Use this register to monitor the status of the sync separator.

STPN: Status of clamping control pulse signal during STOP

COMPSY: Composite sync signal status

VSEP: VSYNC signal status

HSEP: HSYNC signal status

HLOCK: Sync detection
 0: Asynchronous
 1: Synchronous

CLPCND1: Clamping Control Signal Status Register 1
(CLPCNDW)

x'007EDC'
x'007EFC'

	Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		—	—	—	—	—	—	—	—	SAFEP	SAFE N	CLPP	CLPN	XPED UP	XPE DOWN	PED UP	PE DOWN
Reset:		0	0	0	0	0	0	0	0	1	0	1	0	1	0	1	0
R/W:		R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

This register is for monitoring the status of the clamping current source switch shown in figure 9-5 on page 229. An N-channel transistor is on when the associated bit (PEDOWN, XPEDOWN, CLPN, or SAFEN) is 1. A P-channel transistor is on when the associated bit (PEDUP, XPEDUP, CLPP, or SAFEP) is 0.

SAFEP: Low clamping control pulse for high current source (P-channel)

SAFEN: Low clamping control pulse for high current source (N-channel)

CLPP: High clamping control pulse for high current source (P-channel)

CLPN: High clamping control pulse for high current source (N-channel)

XPEDUP: Clamping control pulse for medium current source (P-channel)

XPEDOWN: Clamping control pulse for medium current source (N-channel)

PEDUP: Clamping control pulse for low current source (P-channel)

PEDOWN: Clamping control pulse for low current source (N-channel)

SBFNUM: Sampling Start Position Register
(SBFNUMW)

x'007E4C'
x'007E6C'

	Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		—	—	—	—	—	SBF NUM 10	SBF NUM 9	SBF NUM 8	SBF NUM 7	SBF NUM 6	SBF NUM 5	SBF NUM 4	SBF NUM 3	SBF NUM 2	SBF NUM 1	SBF NUM 0
Reset:		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:		R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

SBFNUM[10:0]: Detected position of start bit flag (detected by the hardware)

TESTA: Test Register
(TESTA)

x'007E4E'
x'007E6E'

	Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
		SLICE 7	SLICE 6	SLICE 5	SLICE 4	SLICE 3	SLICE 2	SLICE 1	SLICE 0	DATA G	CRI2G	CRI1G	ACQG	SLD SAMP	FCPIN	FCP	SLD
Reset:		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:		R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

SLICE[7:0]: Slicing value (either from hardware calculation or software setting)

DATAG: Data window (for capturing the caption data)

CRI2G: CRI window 2 (for detecting the sampling cycle position)

CRI1G: CRI window 1 (for calculating the maximum and minimum values)

ACQG: ACQ window (for setting the H interval for data detection)

SLDSAMP: Caption data sampling pulse

FCPIN: Start position for start bit detection (software setting)

FCP: Start position for start bit detection (hardware calculation)

SLD: Sliced data from the CVBS input signal

10 Pulse Width Modulator

10.1 Description

The MN102H75K/85K contains seven 8-bit pulse width modulators (PWMs) with a minimum pulse width of $16/f_{\text{SYSCLK}}$ and an output waveform cycle of $2^{12}/f_{\text{SYSCLK}}$. (With a 4-MHz oscillator, $16/f_{\text{SYSCLK}} = 1.33 \mu\text{s}$ (8 μs for SLOW mode) and $2^{12}/f_{\text{SYSCLK}} = 341.3 \mu\text{s}$ (2 ms for SLOW mode).)

For information on the SLOW mode, see section 3.1, "CPU Modes."

The PWM ports are 3.3-volt, open-drain outputs. To enable the PWM ports, either turn the pullup registers on using the pullup control registers for the associated ports (P15-P17 and P20-P23; see table 10-1) or connect external pullup resistors to these ports.

Table 10-1 Register Settings for Internal PWM Pullup

PWM Block	Register	Bit No.	Setting
PWM0	P1PUP (x'00FFB1')	5	1
PWM1		6	
PWM2		7	
PWM3	P2PUP (x'00FFB2')	0	1
PWM4		1	
PWM5		2	
PWM6		3	

The microcontroller writes the pulsewidth modulated data for a PWM block to its associated 8-bit data register (PWMn). The data register settings determine how long the waveform stays low. With a 4-MHz oscillator, the PWM output pulse width has a resolution of $1.33 \mu\text{s}$ ($1/f_{\text{PWM}}$) and a cycle of $341.3 \mu\text{s}$ ($2^8/f_{\text{PWM}}$). Note that when (and only when) the data changes between x'00' and x'01', the resolution is $1.34 \mu\text{s}$ ($2/f_{\text{PWM}}$).

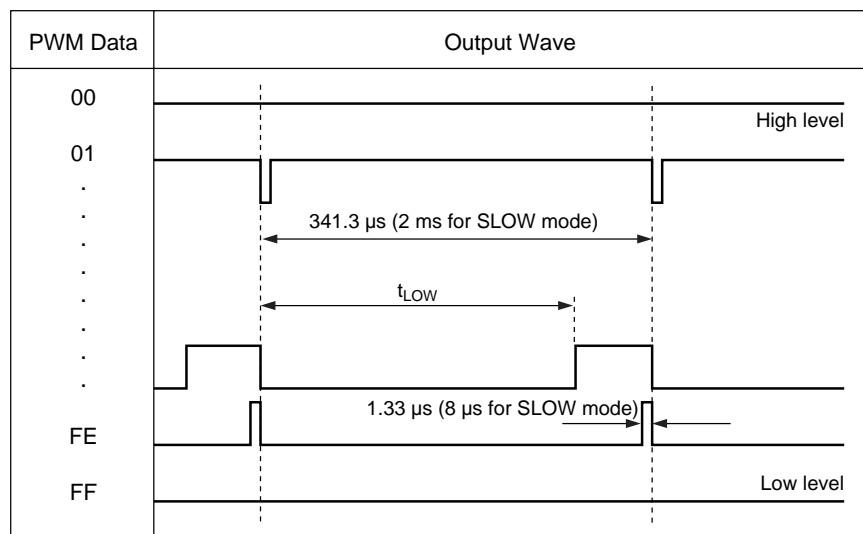
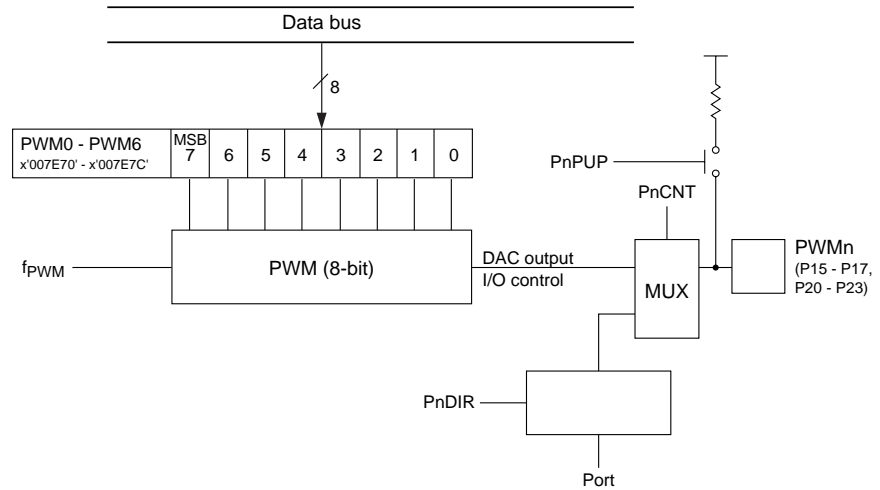


Figure 10-1 PWM Output Waveform

10.2 Block Diagram



Not using internal pullup function, Figure 10-2 connect the external pullup resistance

Note: With a 4-MHz oscillator:
 $f_{PWM} = f_{SYSCLK}/16$
 Output pulse cycle = $2^8/f_{PWM} = 341.3 \mu s$
 Minimum pulse width = $1/f_{PWM} = 1.33 \mu s$
 $t_{LOW} = (PWMn + 1) \times 0.67 \mu s$

Figure 10-2 PWM Block Diagram

10.3 PWM Data Registers

All registers in PWM function cannot be written by byte (be word only). Read by byte is possible.

Bits 7 to 0 of each of the seven PWM data registers (PWM0 to PWM6) hold the 8-bit pulsewidth modulated data to be written to the PWMs. The registers reset to 0, and they set to 1 when PWM output is high.

PWM0–PWM6: PWMn Data Registers

x'007E70'–x'007E7C'

Bit:	7	6	5	4	3	2	1	0
	PWMn7	PWMn6	PWMn5	PWMn4	PWMn3	PWMn2	PWMn1	PWMn0
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R

11 I/O Ports

11.1 Description

The MN102H75K/85K contains 50 pins that form general-purpose I/O ports. Ports 0, 1, 2, 3, 4, and 5 are 8-bit ports, and port 6 is a 2-bit port. All of these pins have alternate functions. (Ports 7 and 8 are only available with the quad flat package.)

Table 11-1 I/O Port Pins

Port	Associated Pins
Port 0	P07–P00
Port 1	P17–P10
Port 2	P27–P20
Port 3	P37–P30
Port 4	P47–P40
Port 5	P57–P50
Port 6	P61–P60
Port 7	P77–P70
Port 8	P87–P80

11.2 I/O Port Circuit Diagrams

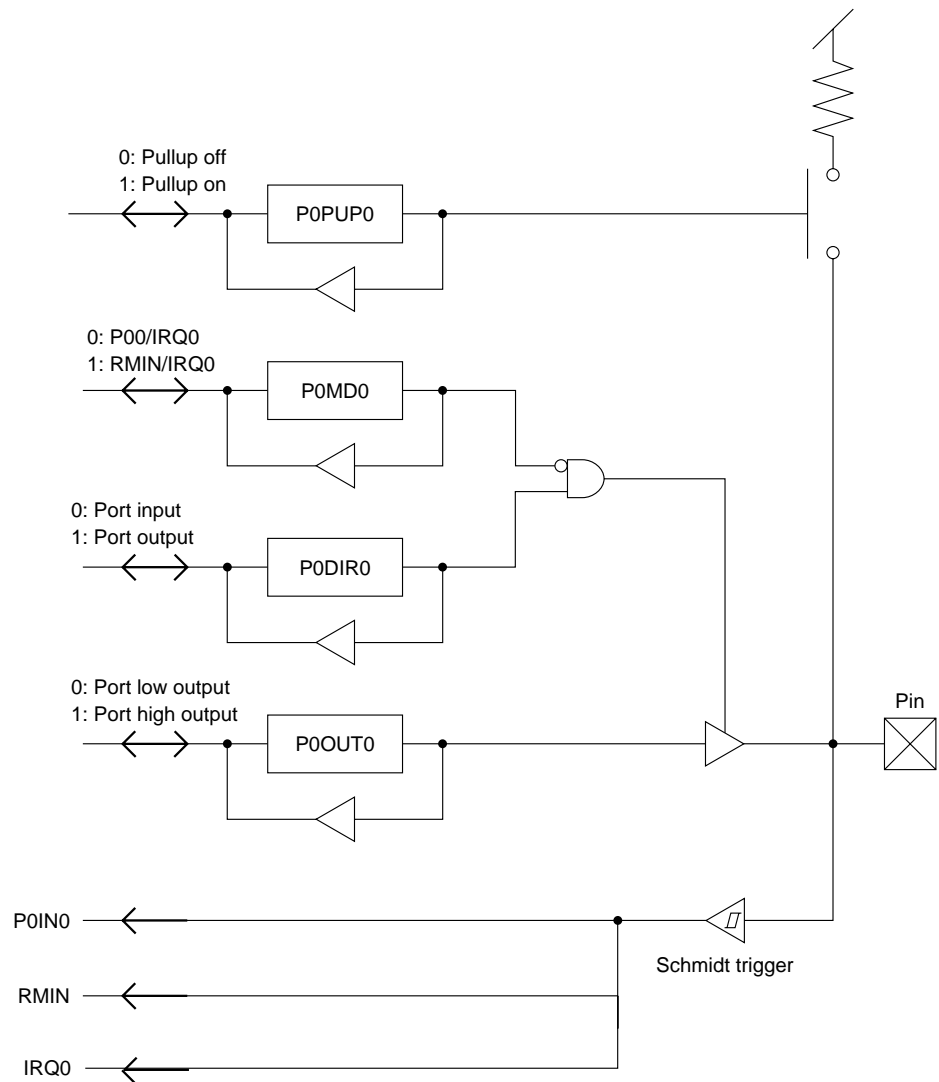


Figure 11-1 P00/RMIN/IRQ0 (Port 0)

I/O Ports

I/O Port Circuit Diagrams

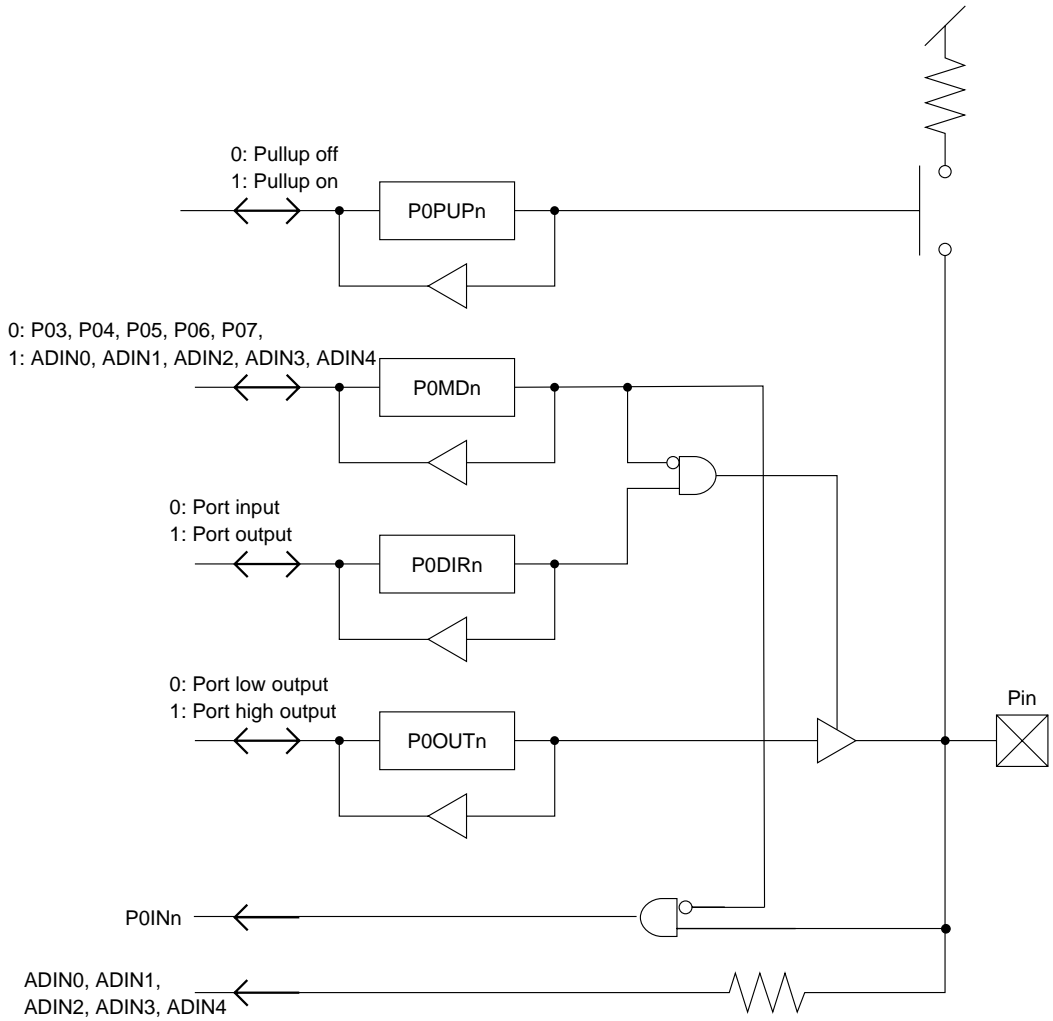


Figure 11-2 P03/ADIN0 to P07/ADIN4 (Port 0)

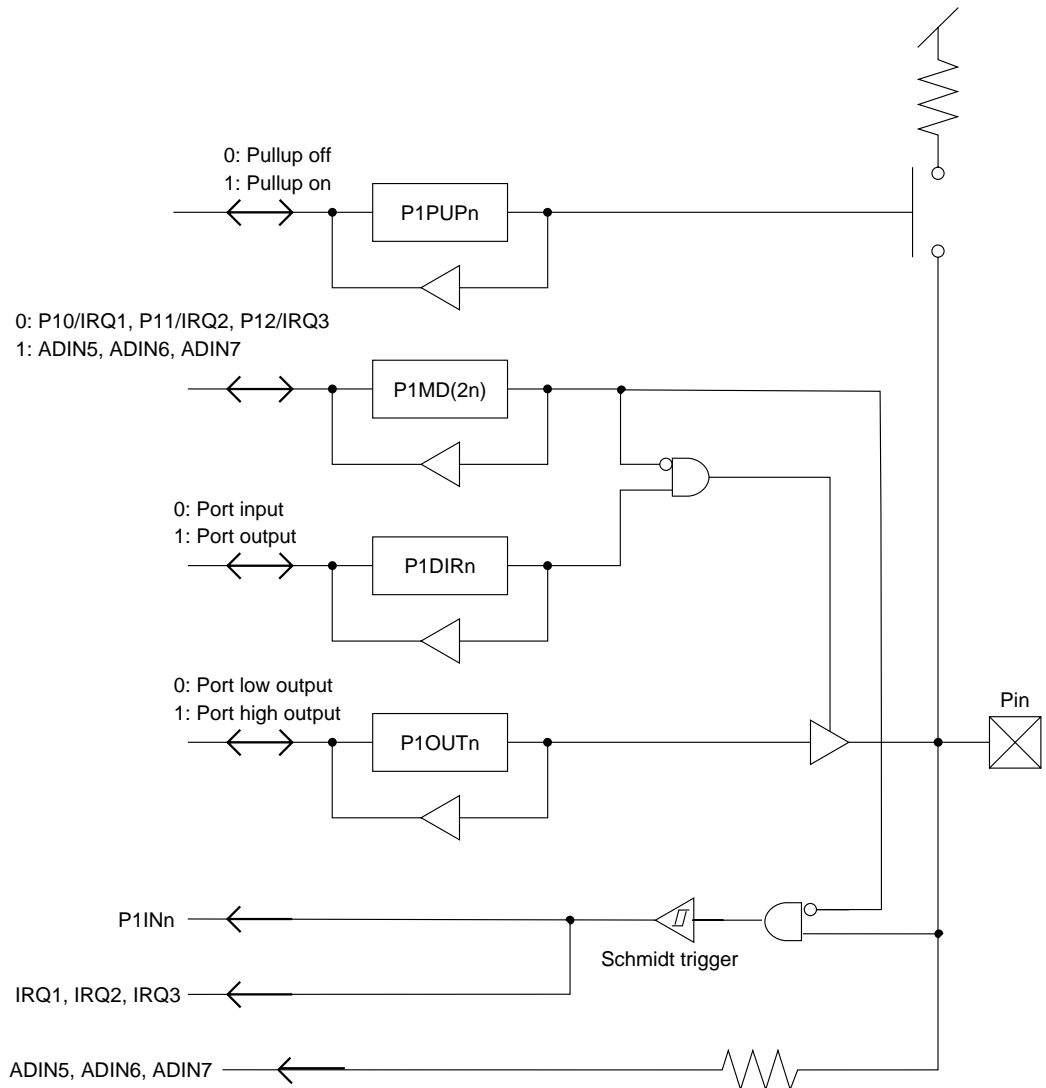


Figure 11-3 P10/ADIN5/IRQ1, P11/ADIN6/IRQ2, and P12/ADIN7/IRQ3 (Port 1)

I/O Ports

I/O Port Circuit Diagrams

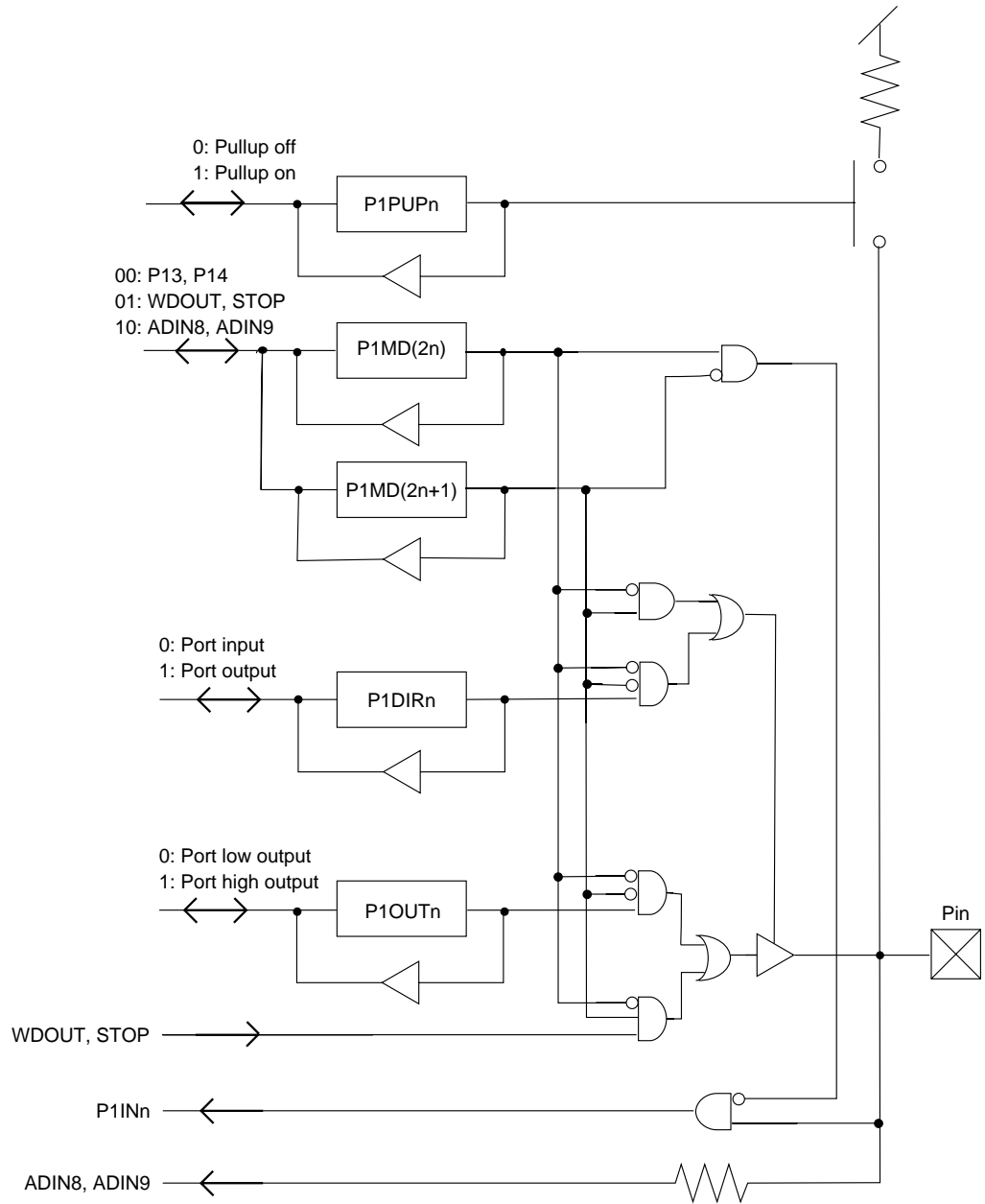


Figure 11-4 P13/ADIN8/WDOUT and P14/ADIN9/STOP (Port 1)

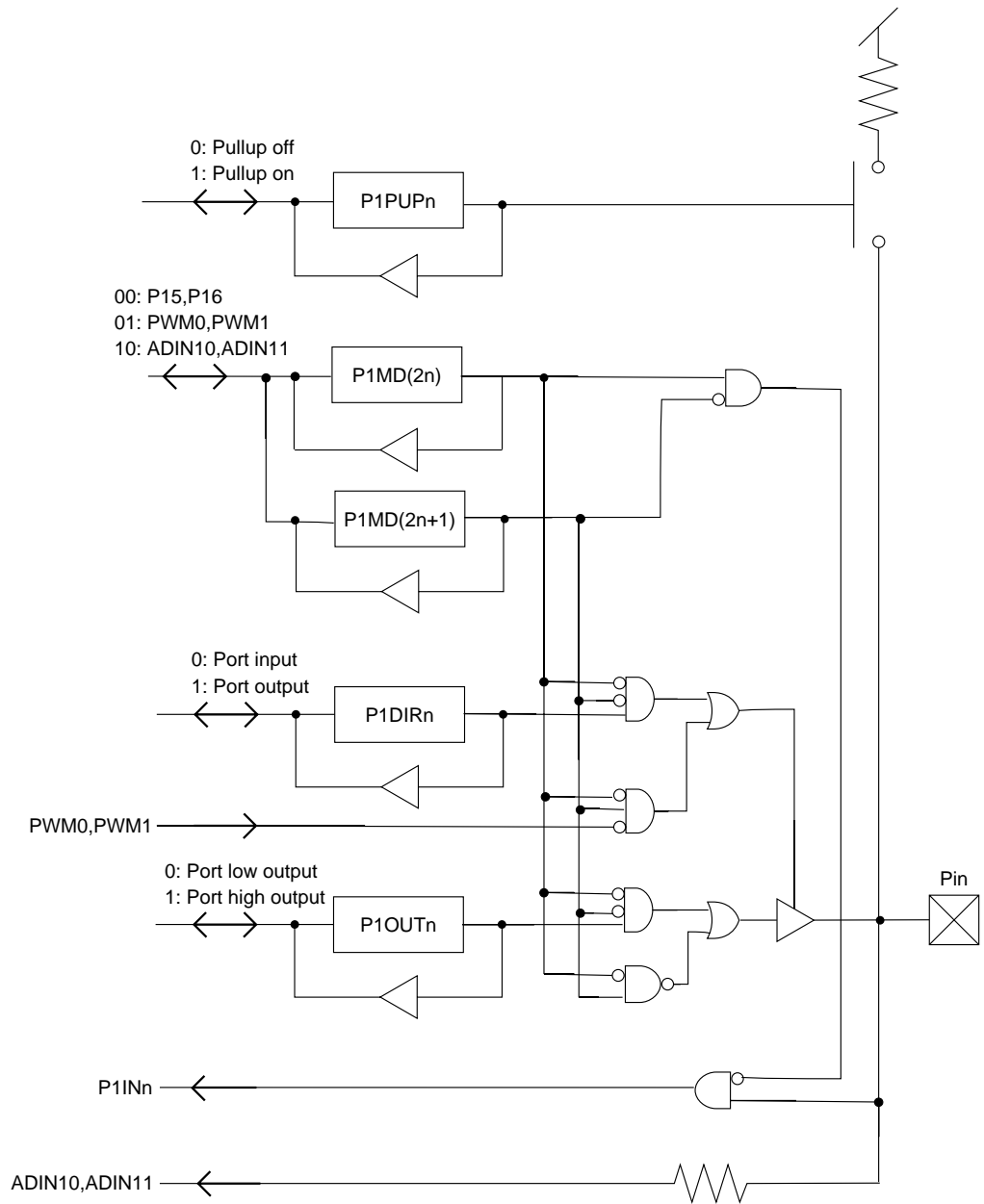


Figure 11-5 P15/ADIN10/PWM0 and P16/ADIN11/PWM1 (Port 1)

I/O Ports

I/O Port Circuit Diagrams

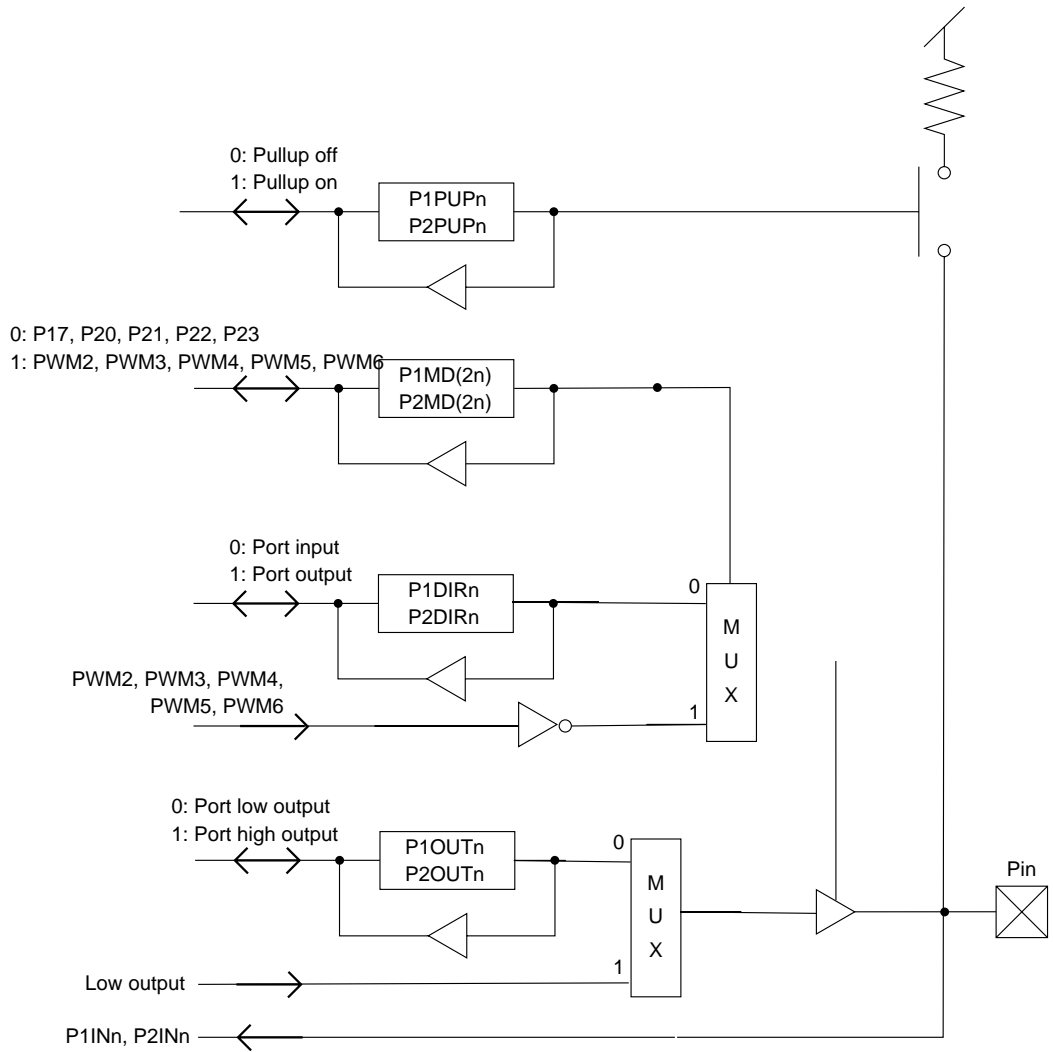


Figure 11-6 /PWM2 (Port 1), P20/PWM3, P21/PWM4, P22/PWM5, and P23/PWM6 (Port 2)

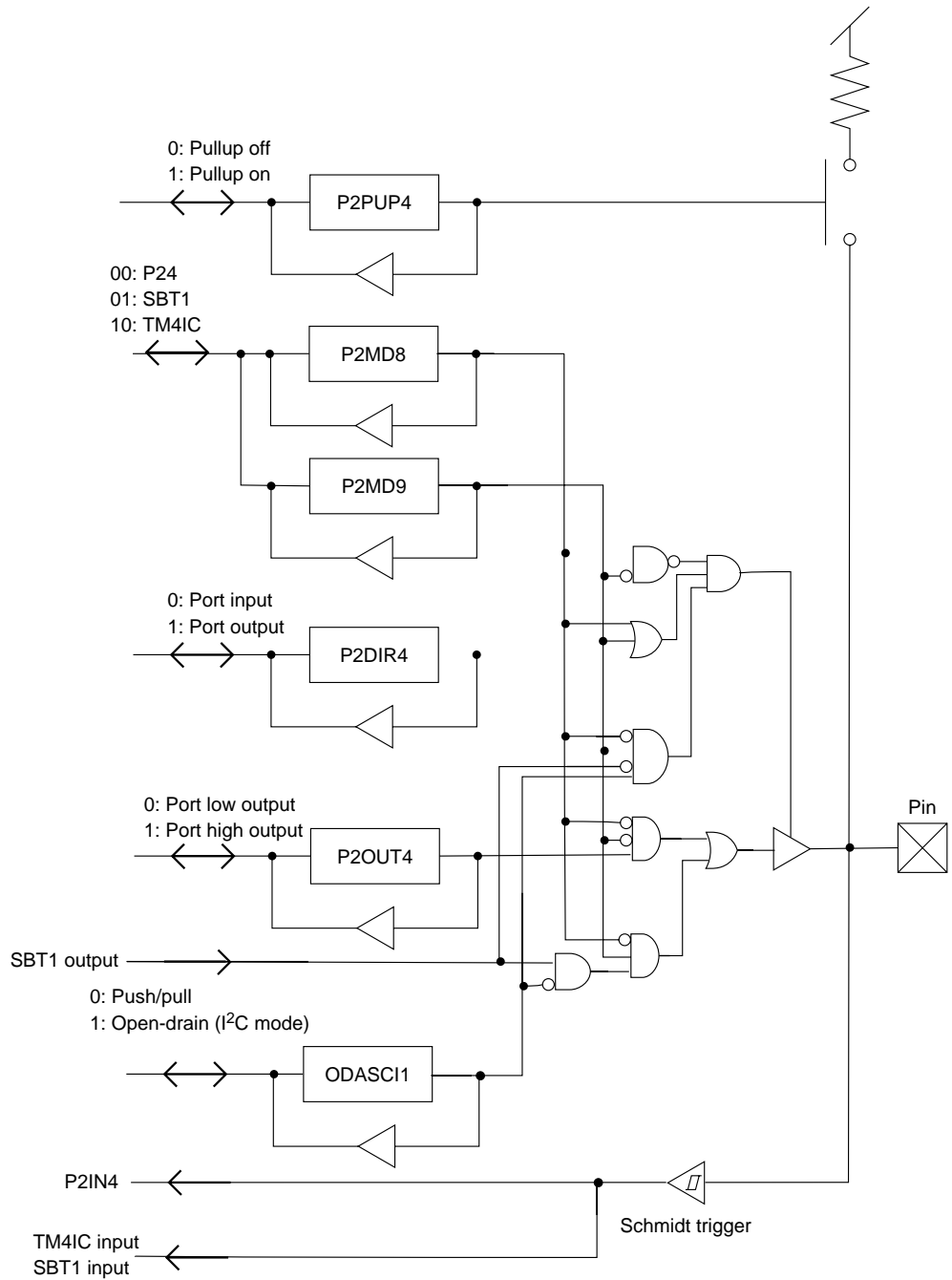


Figure 11-7 P24/TM4IC/SBT1 (Port 2)



To use as SBT1, set P2MD8 and P2MD9 to 0.

I/O Ports

I/O Port Circuit Diagrams

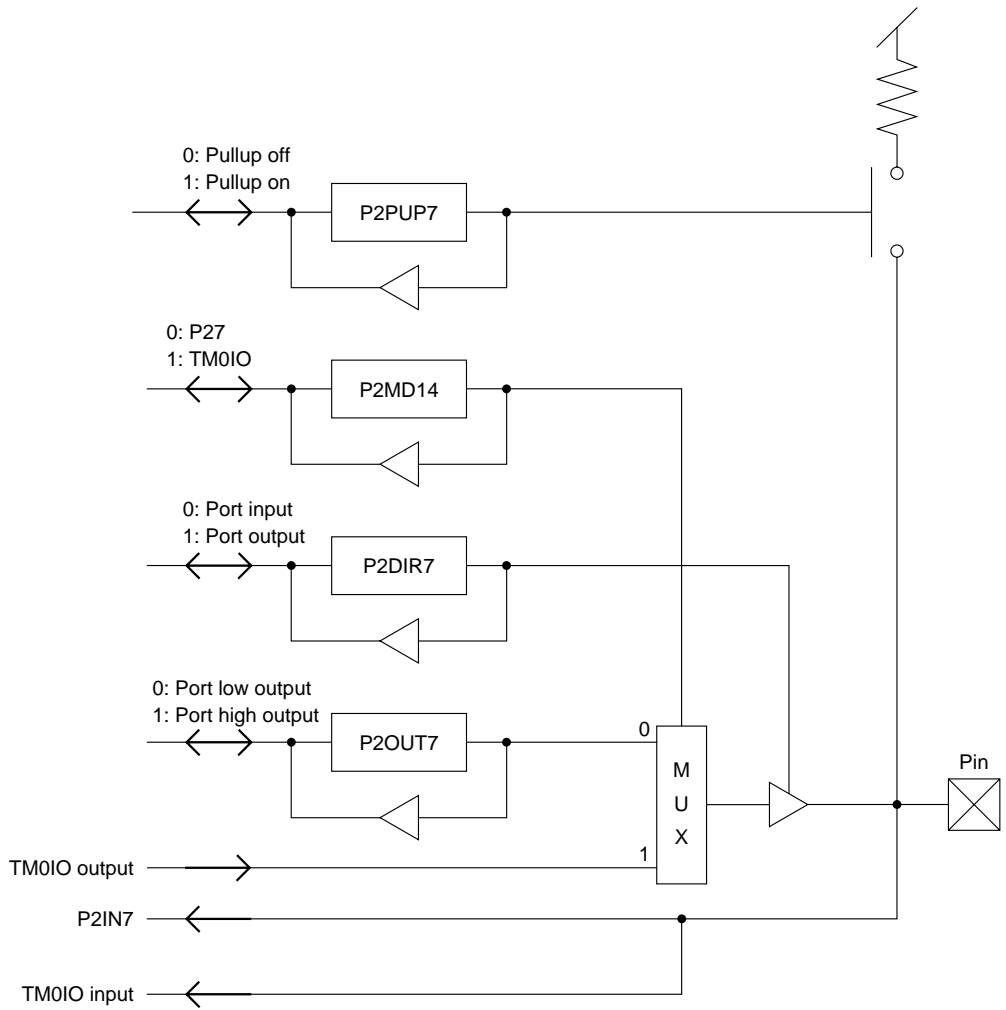


Figure 11-8 P27/TM0IO (Port 2)

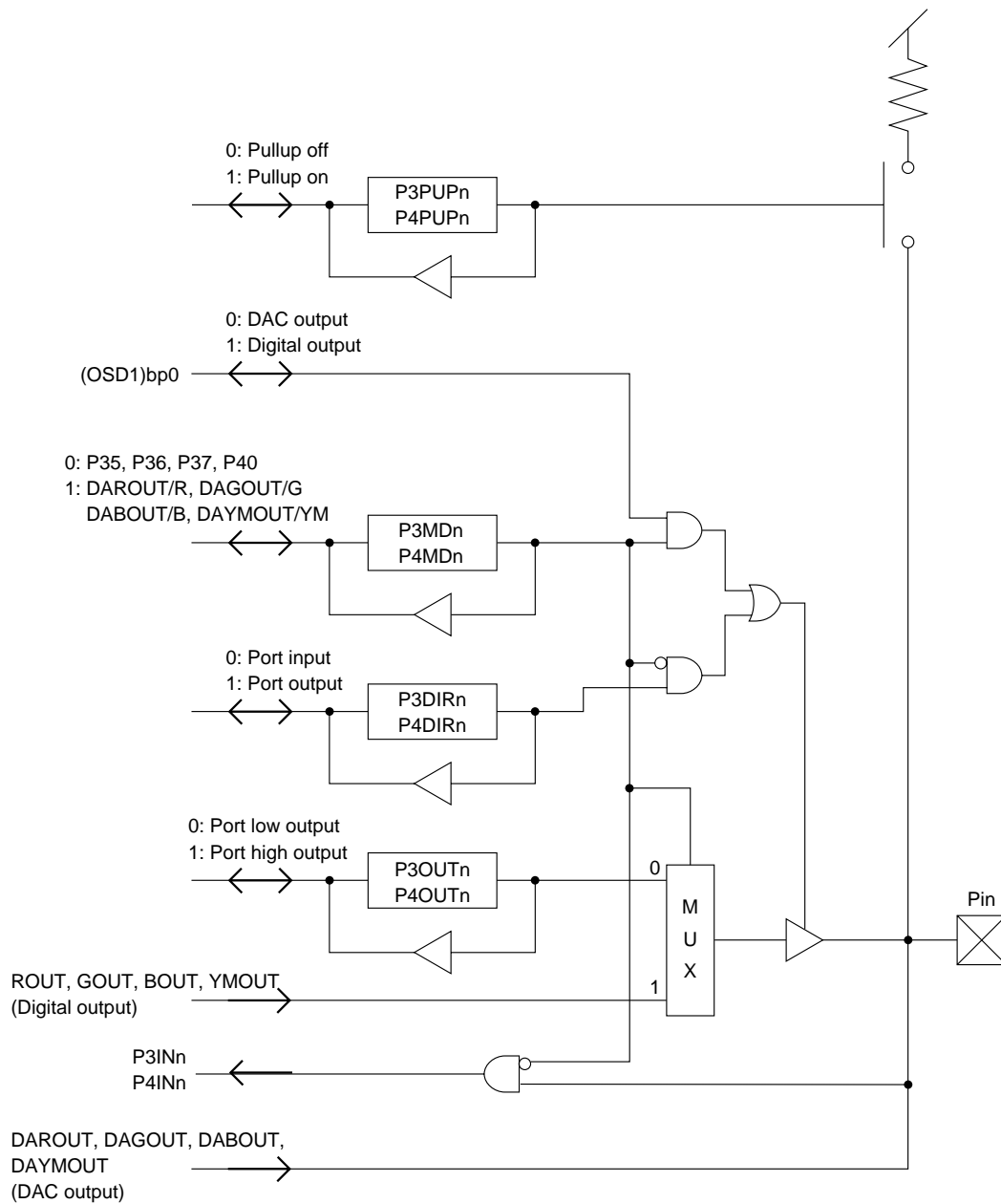


Figure 11-9 P35/DAROUT/R, P36/DAGOUT/G, P37/DABOUT/B (Port 3), and P40/DAYMOUT/YM (Port 4)

I/O Ports

I/O Port Circuit Diagrams

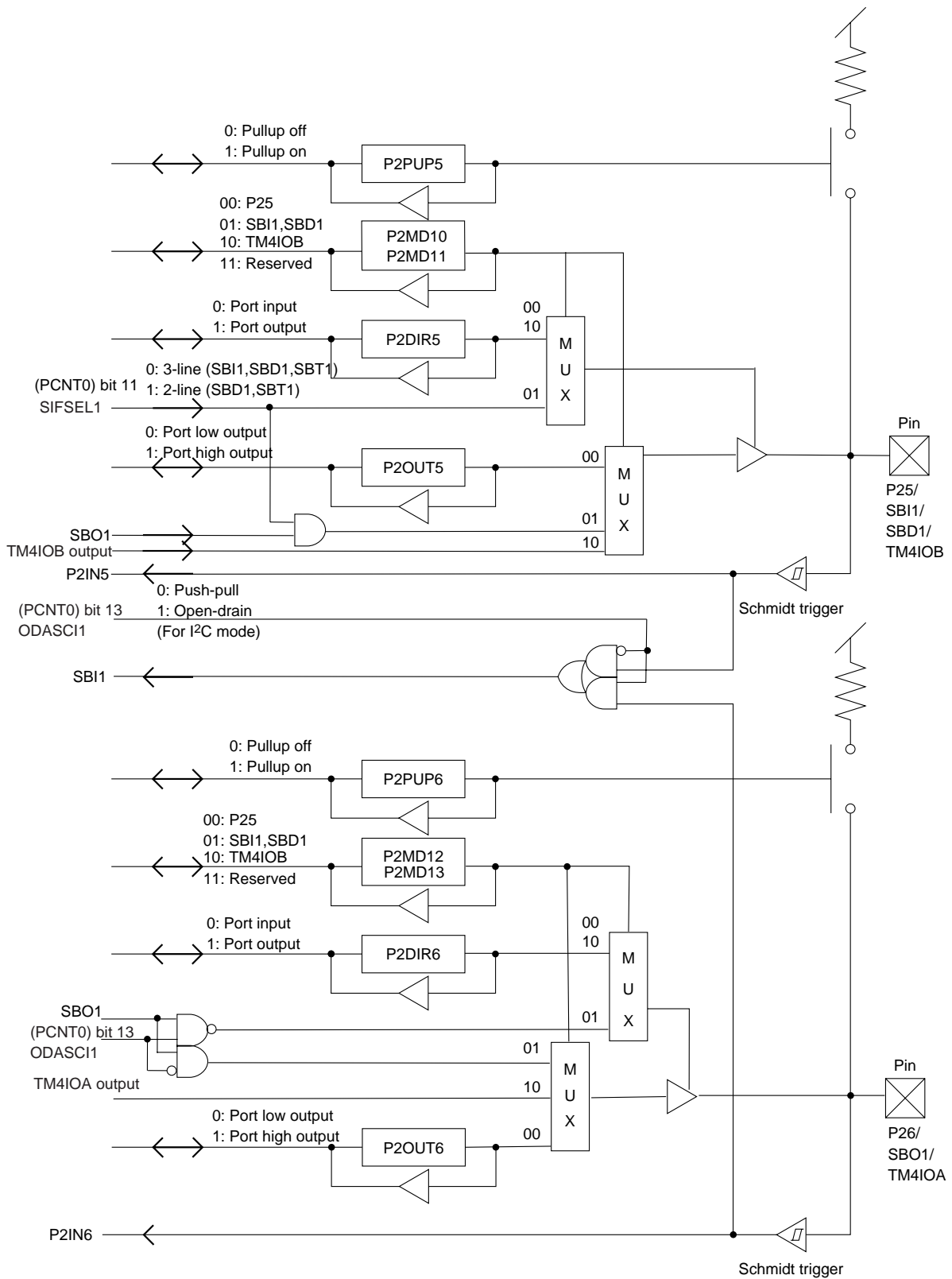


Figure 11-10 P25/TM4IOB/SBI1/SBD1 and P26/TM4IOA/SBO1 (Port 2)

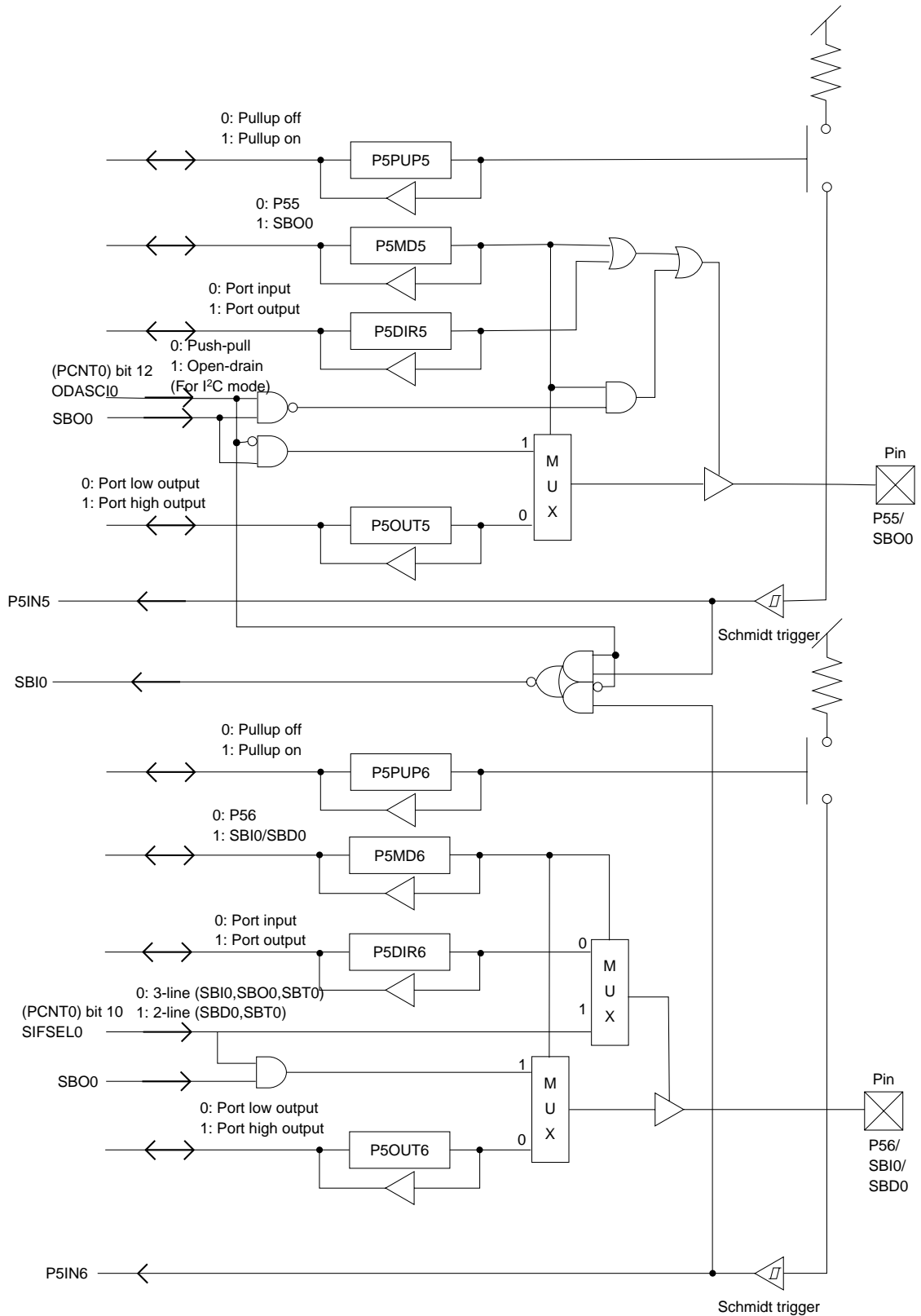


Figure 11-11 P55 and P56 (Port 5)

I/O Ports

I/O Port Circuit Diagrams

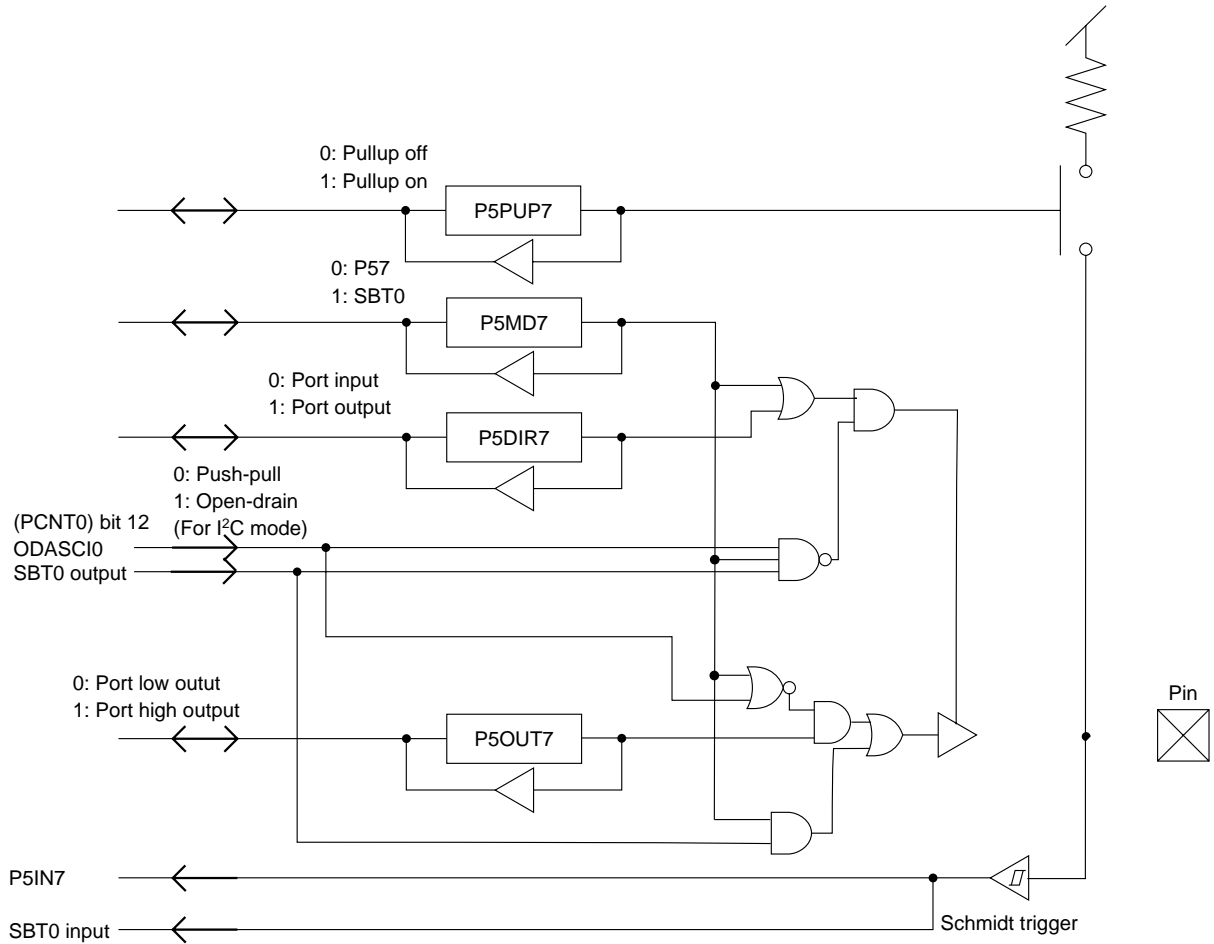


Figure 11-12 P57/SBT0 (Port 5)

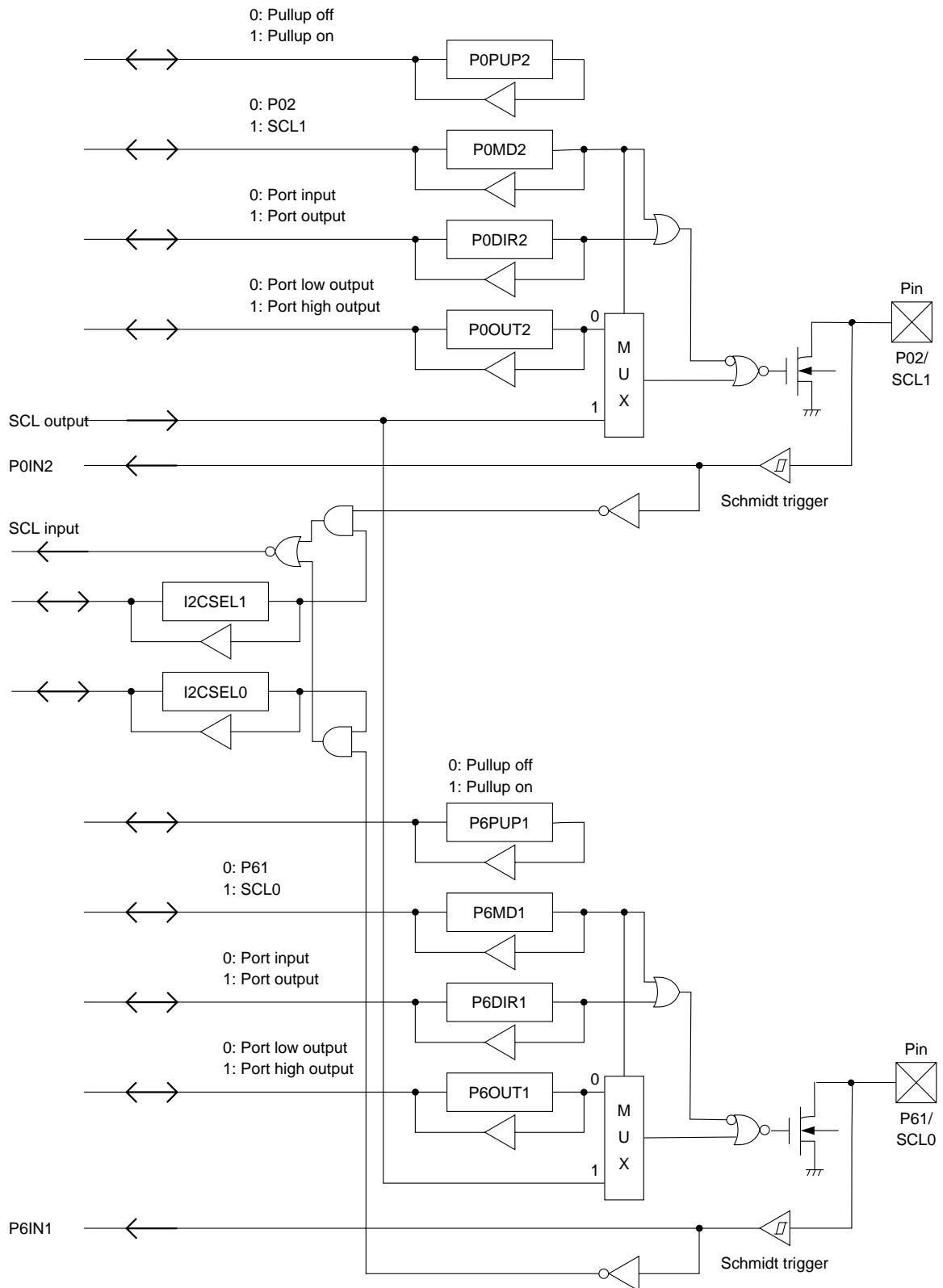


Figure 11-13 P02/SCL1 (Port 0) and P61/SCL0 (Port 6)

I/O Ports

I/O Port Circuit Diagrams

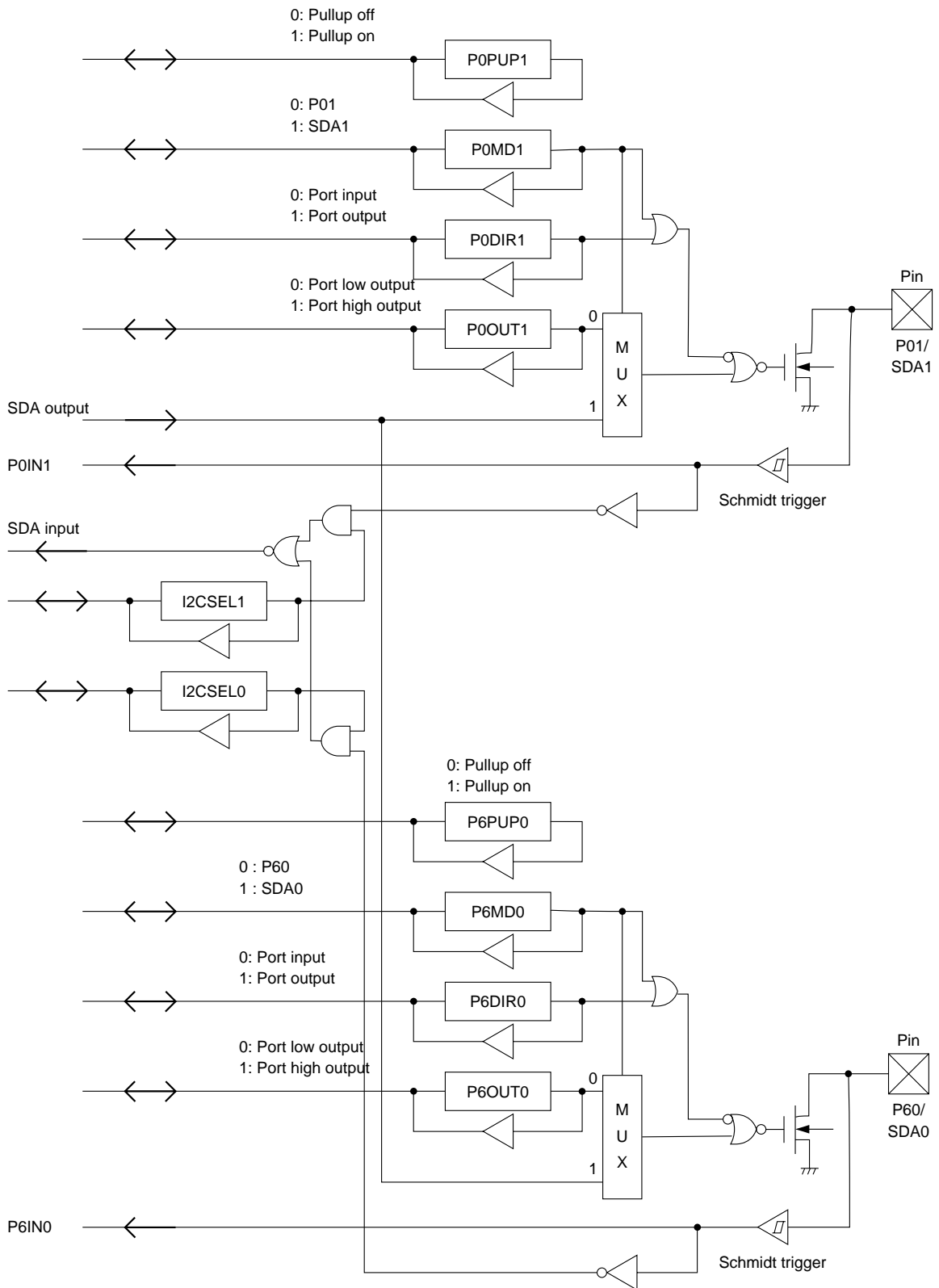


Figure 11-14 P01/SDA1 (Port 1) and P60/SDA0 (Port 6)

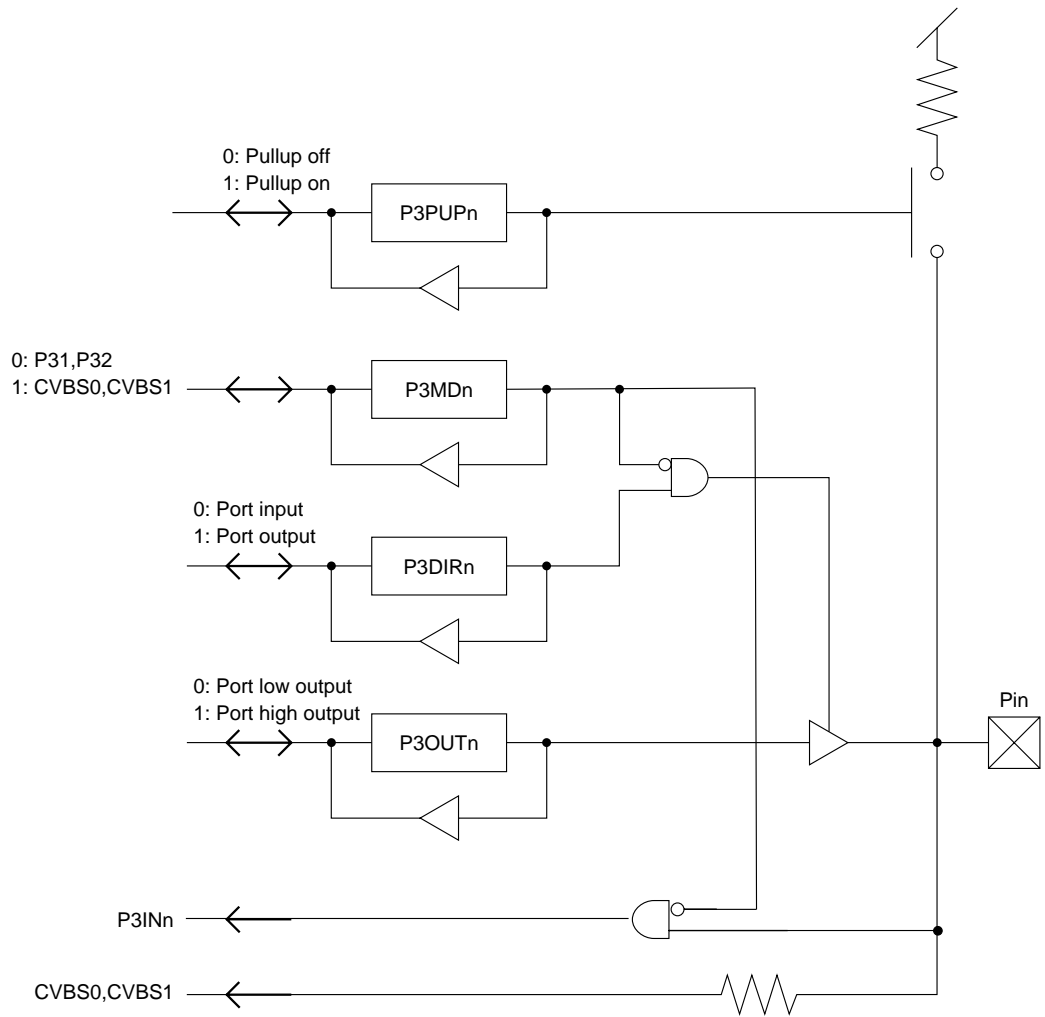


Figure 11-15 P31/CVBS0 and P32/CVBS1 (Port 3)

I/O Ports

I/O Port Circuit Diagrams

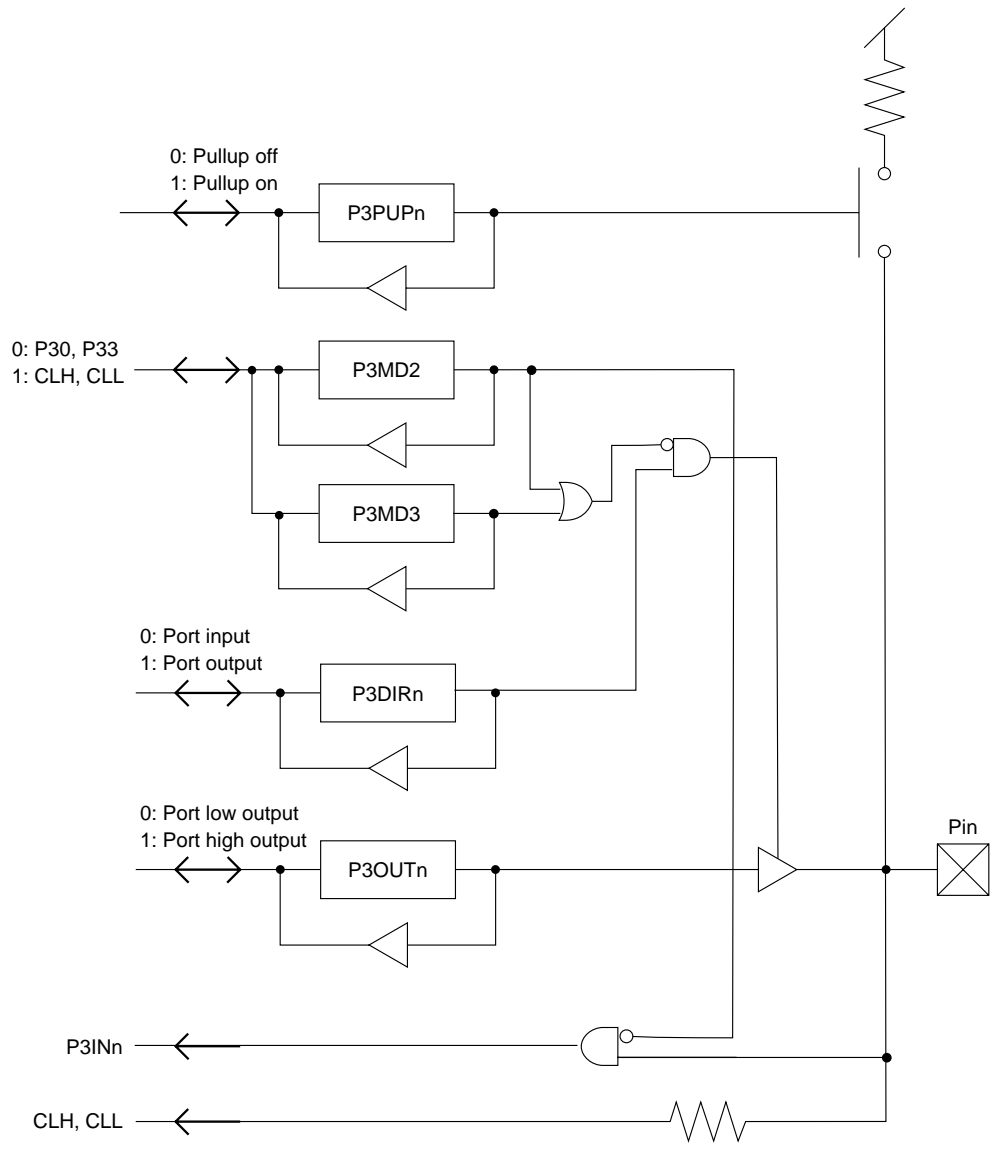


Figure 11-16 P30/CLH and P33/CLL (Port 3)

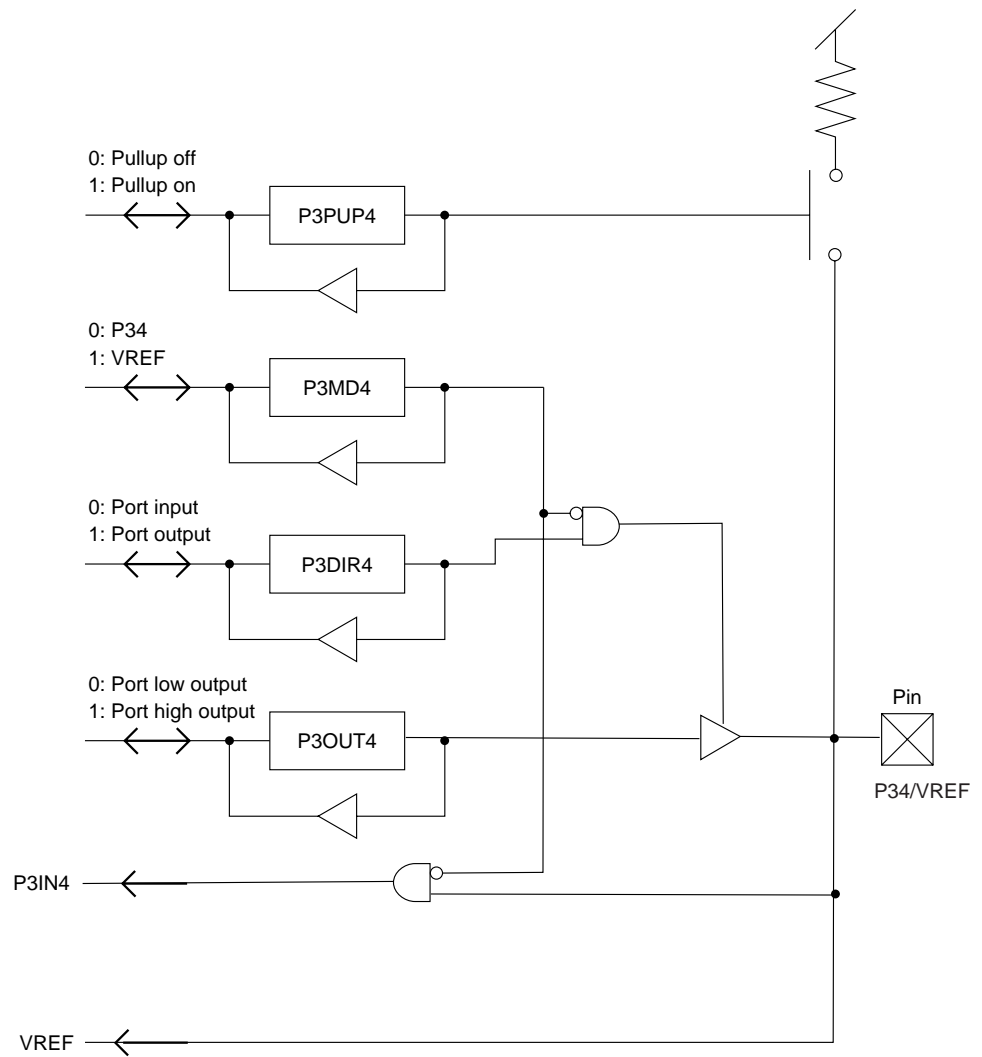


Figure 11-17 P34/VREF (Port 3)

I/O Ports

I/O Port Circuit Diagrams

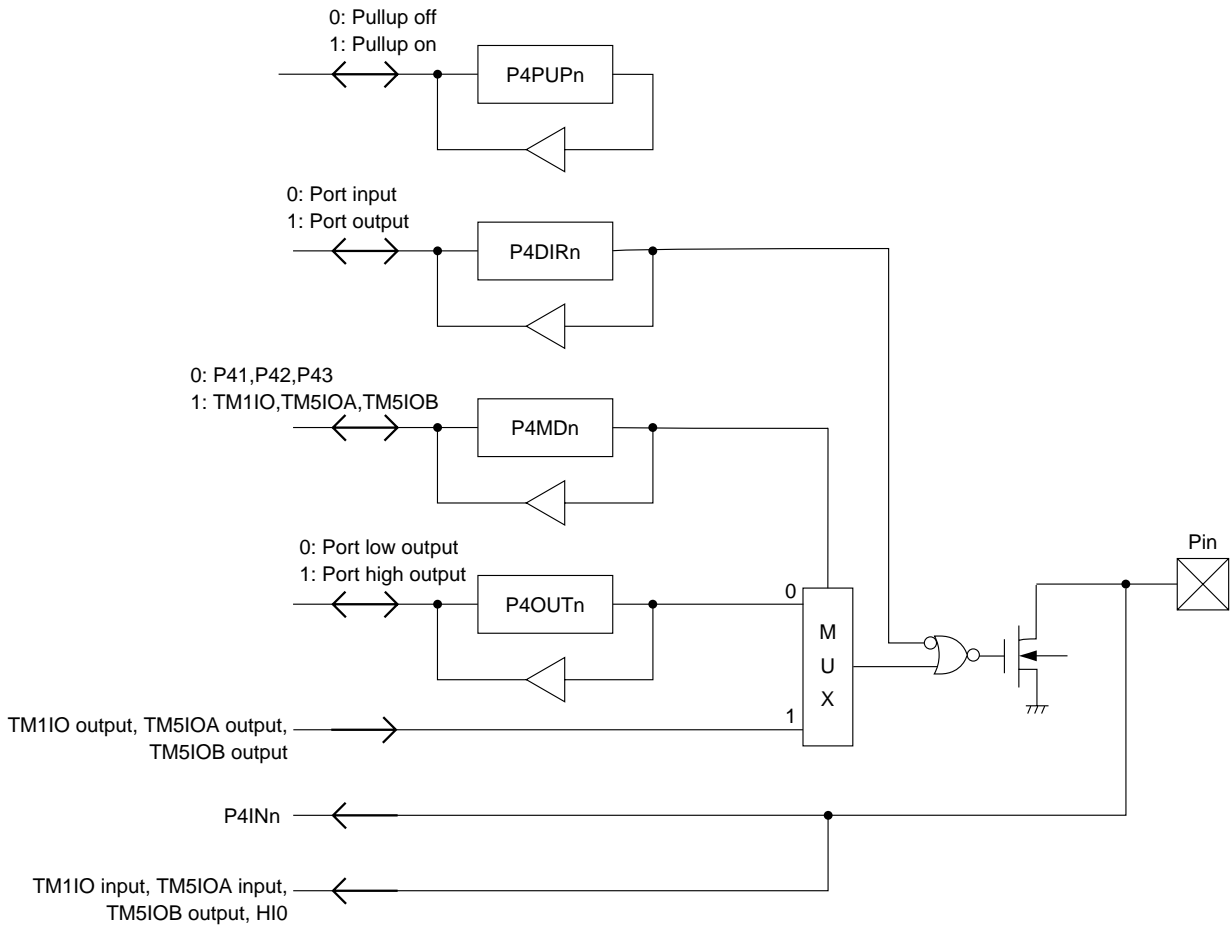


Figure 11-18 P41/TM1IO, P42/TM5IOA, and P43/TM5IOB/HI0 (Port 4)

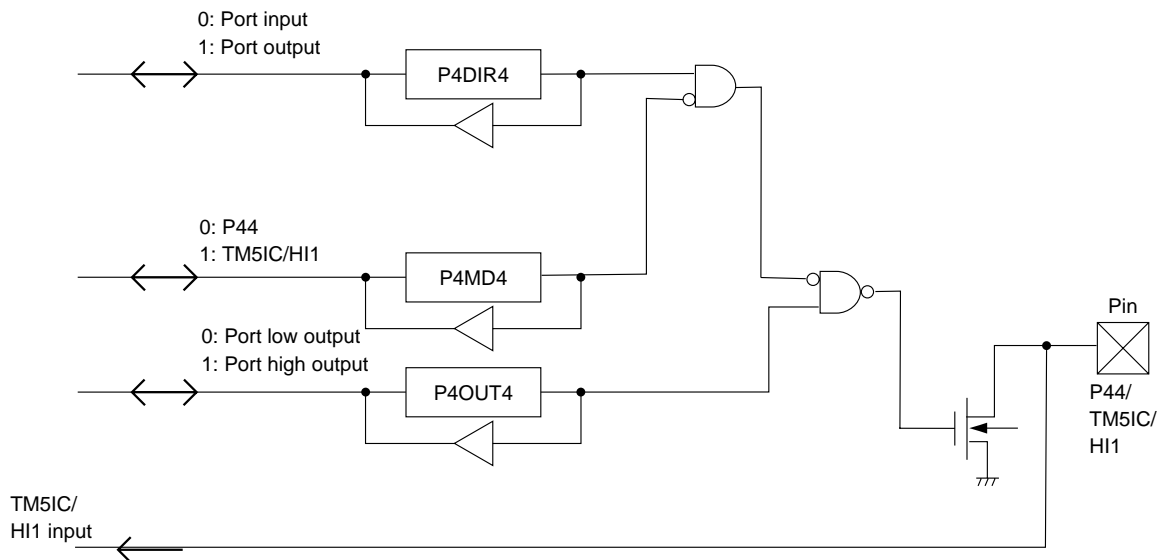


Figure 11-19 P44/TM5IC/HI1 (Port 4)

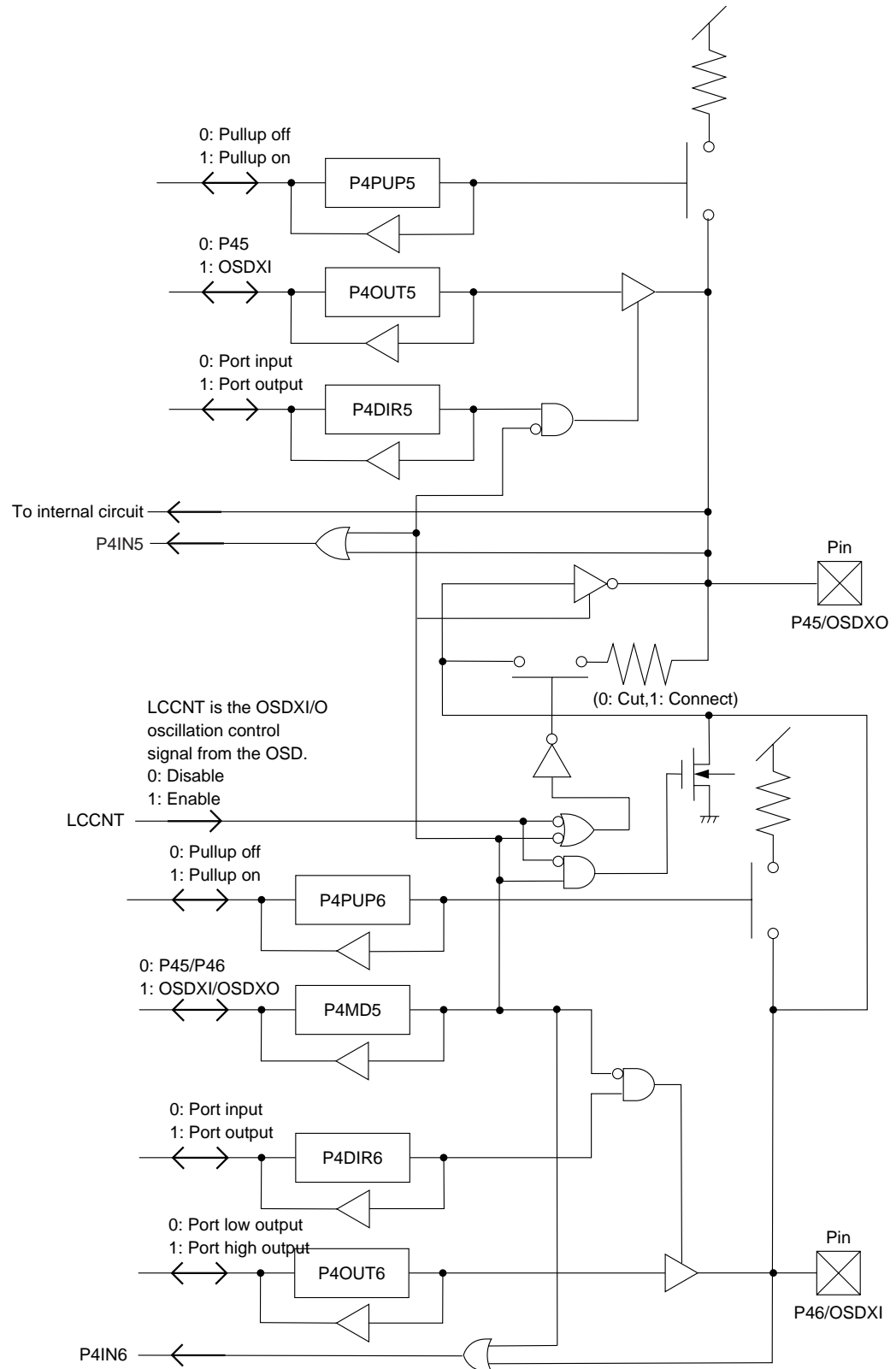


Figure 11-20 P45/OSDXO and P46/OSDXI (Port 4)

I/O Ports

I/O Port Circuit Diagrams

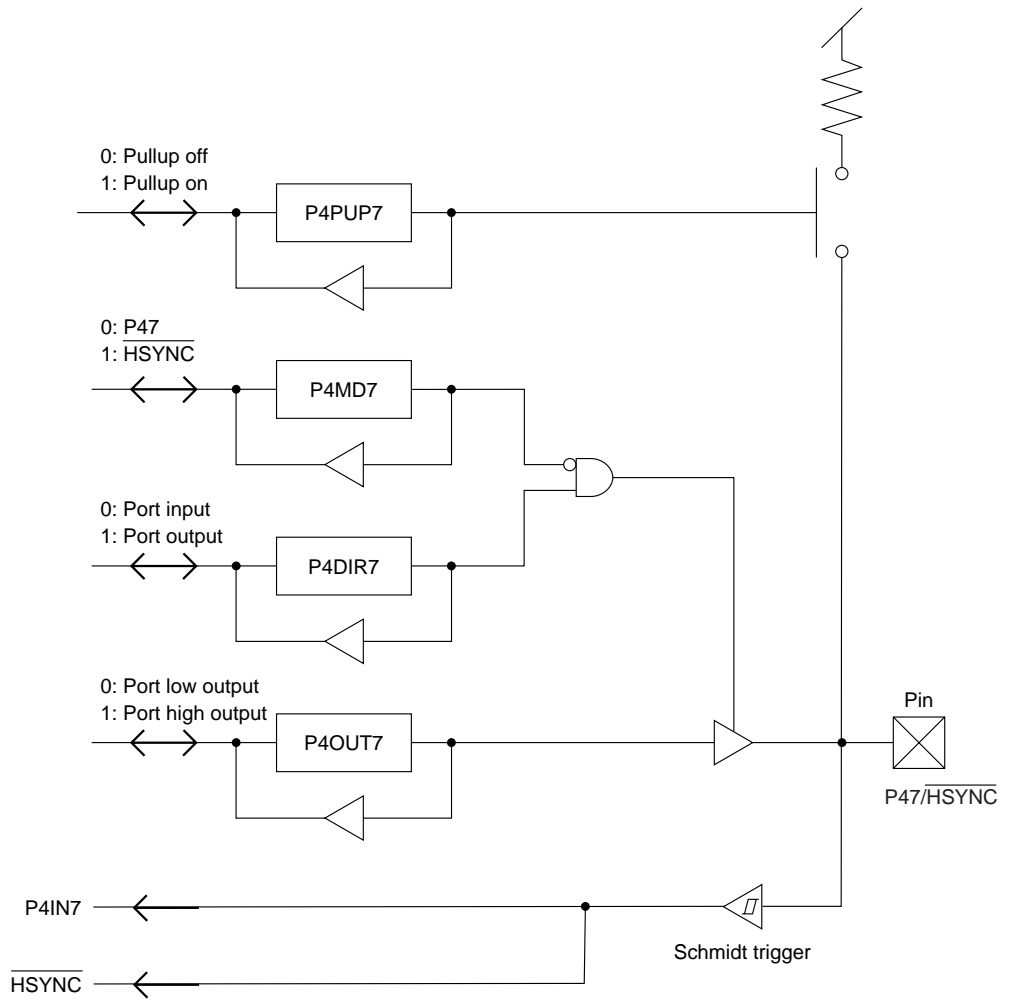


Figure 11-21 P47/HSYNC (Port 4)

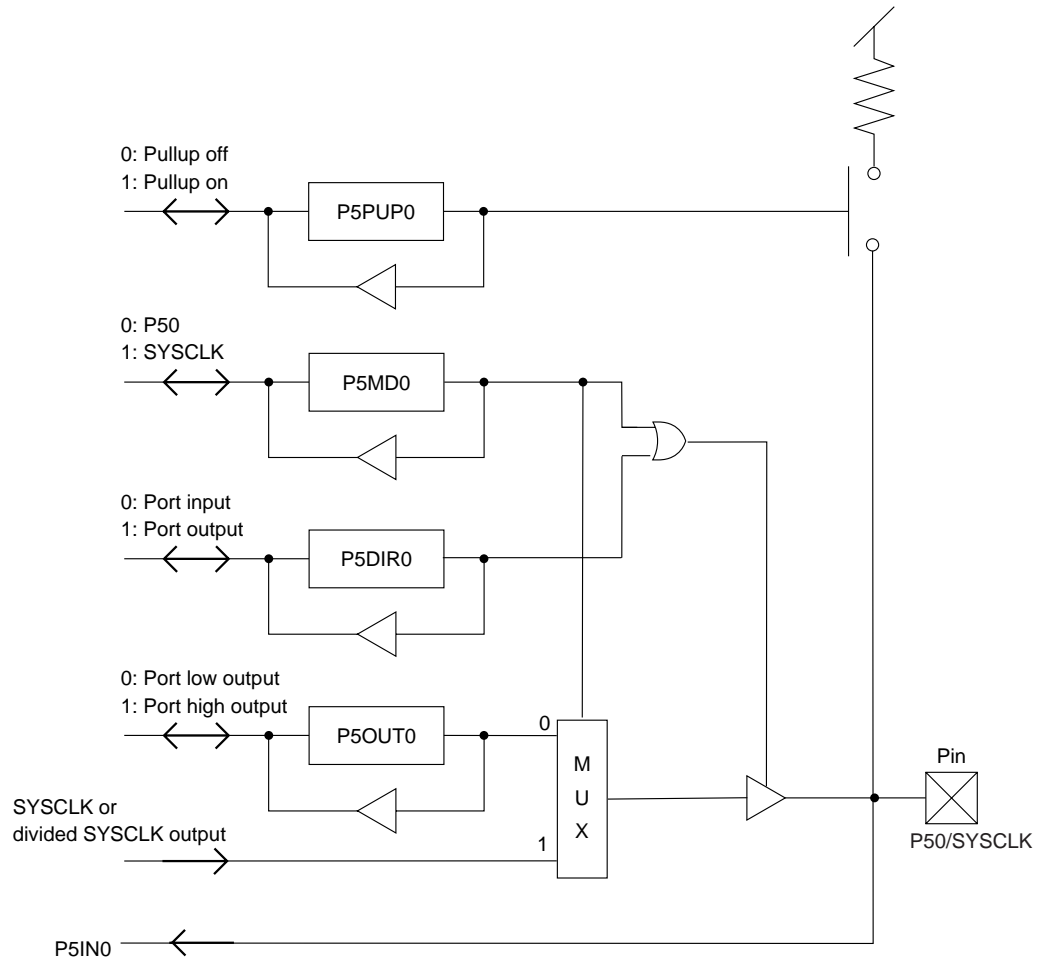


Figure 11-22 P50/SYSCLK (Port 5)

I/O Ports

I/O Port Circuit Diagrams

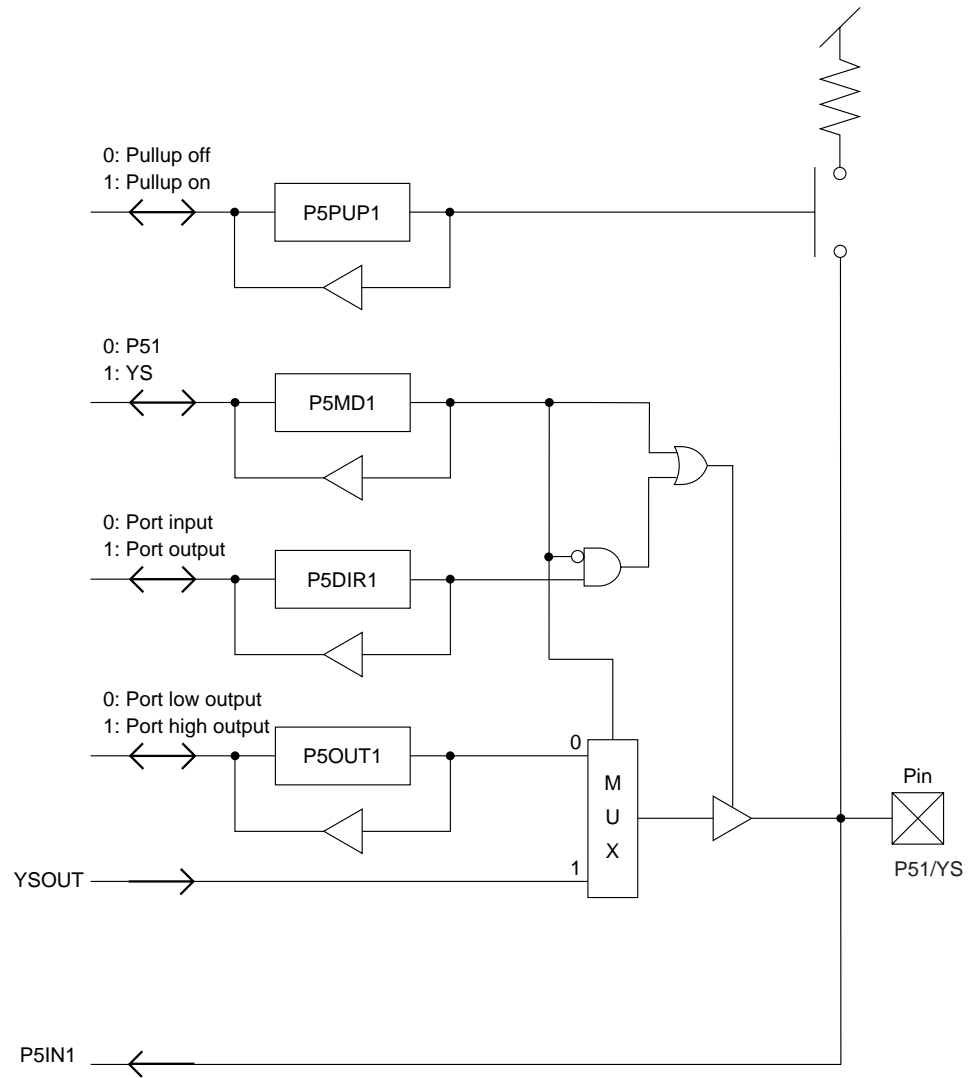


Figure 11-23 P51/YS (Port 5)

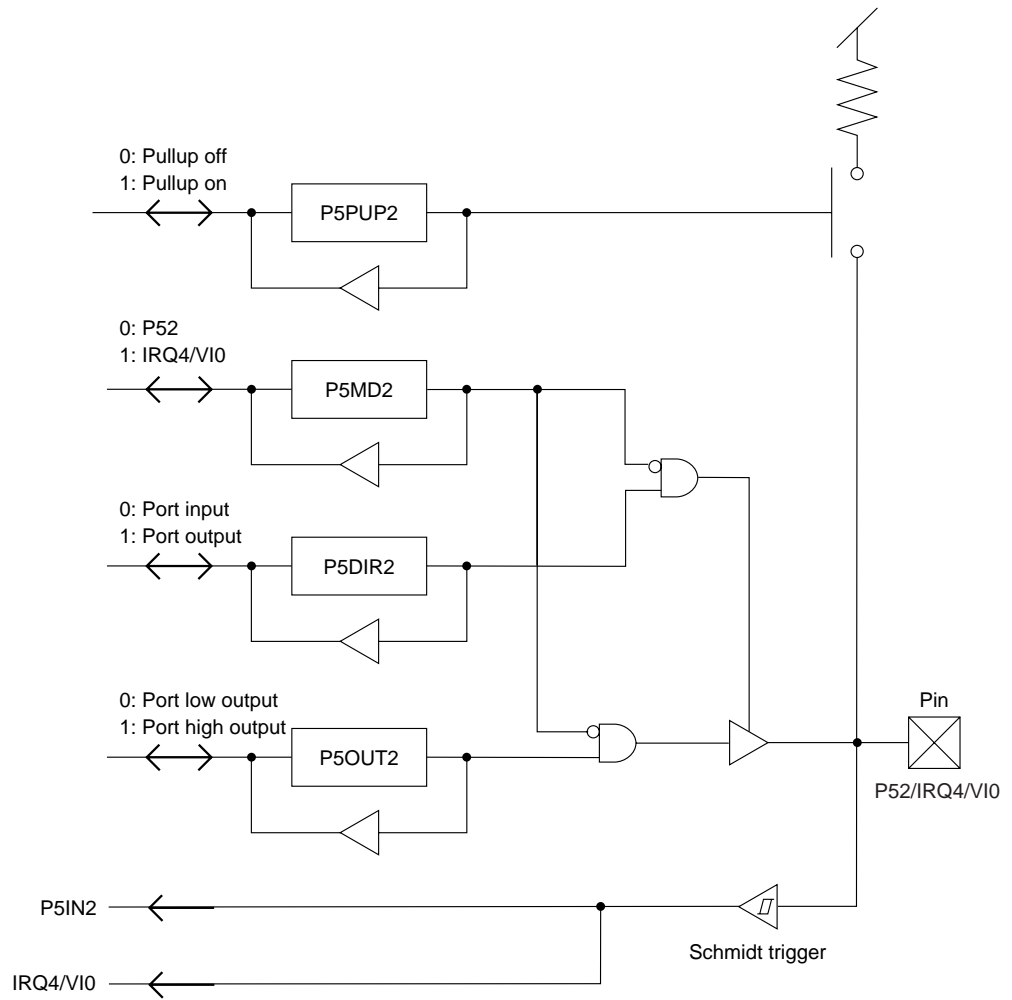


Figure 11-24 P52/IRQ4/VI0 (Port 5)

I/O Ports

I/O Port Circuit Diagrams

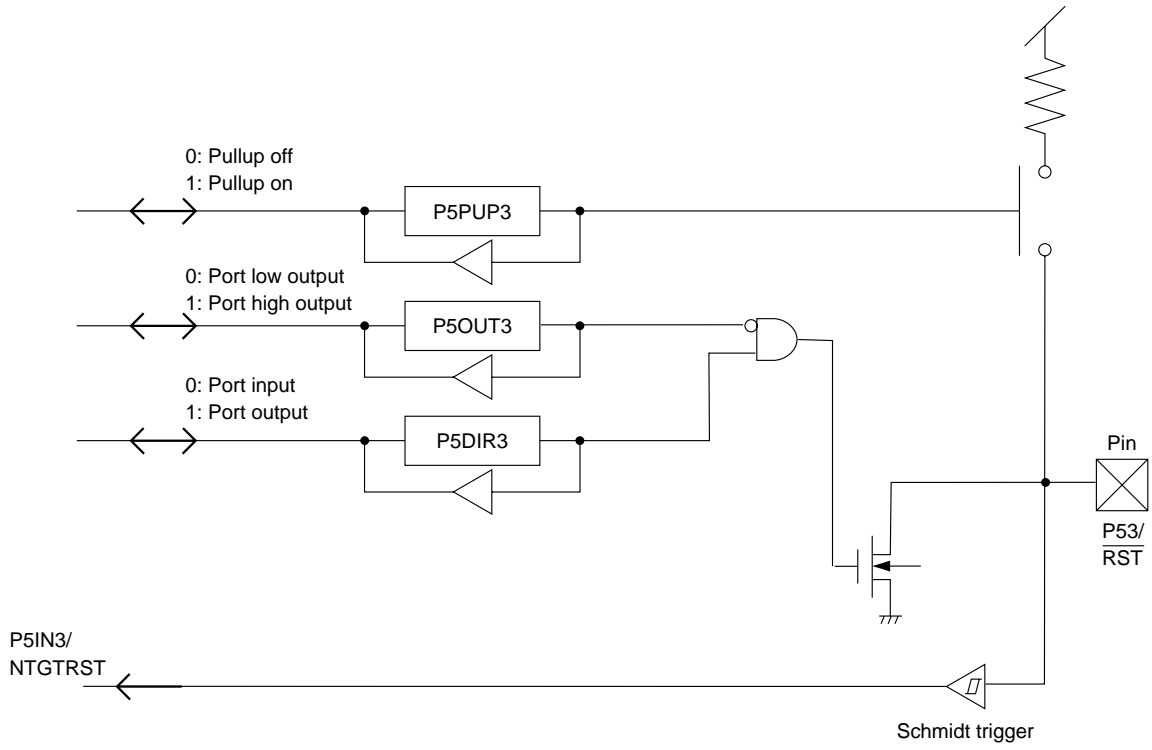


Figure 11-25 P53/RST (Port 5)

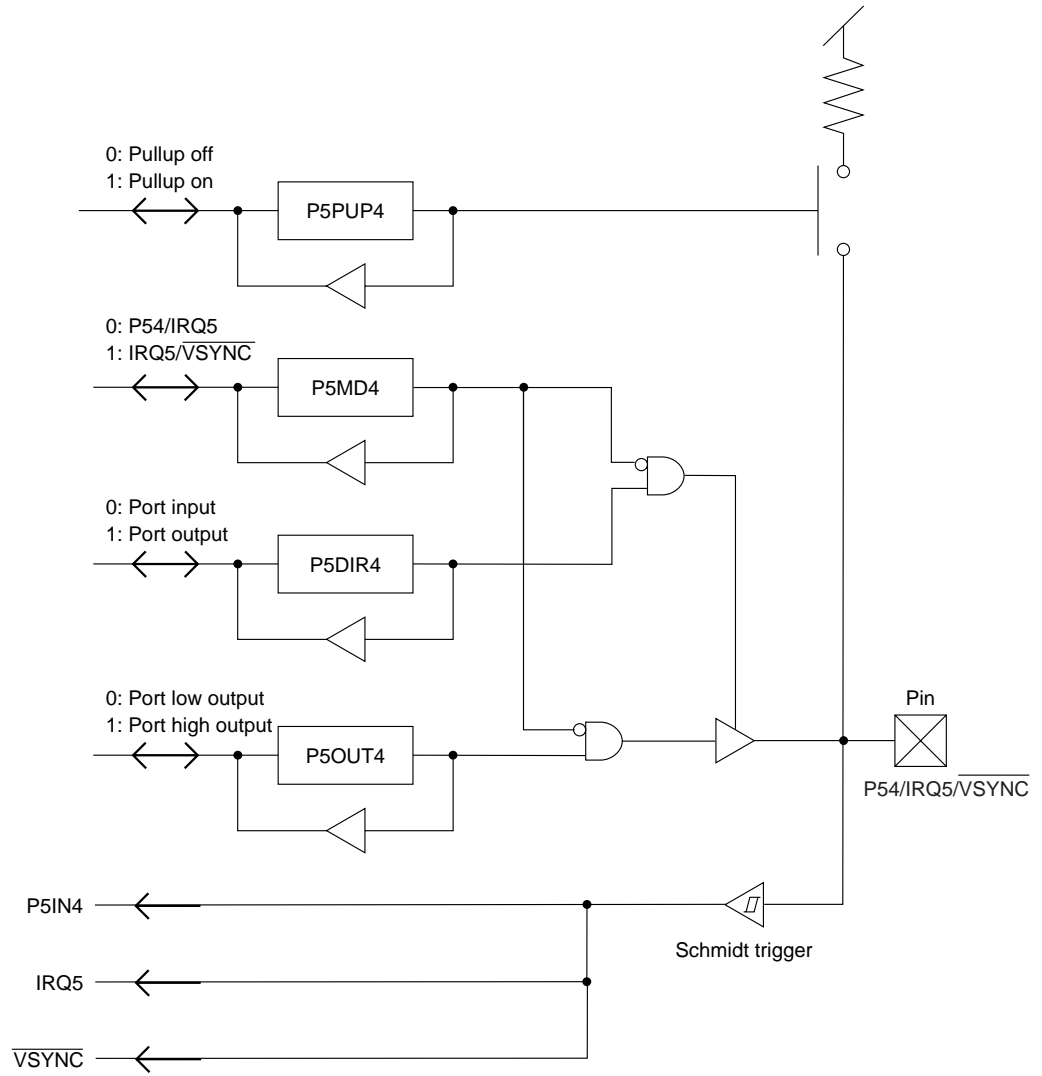


Figure 11-26 P54/IRQ5/VSYNC (Port 5)

11.3 I/O Port Control Registers



Do not activate the pullup resistors when the pins are in output mode. This will cause incorrect output voltage levels and increase power and current consumption.

P0PUP–P5PUP: Ports 0–5 Pullup Resistor Control Registers **x'00FFB0'–x'00FFB5'**
P7PUP–P8PUP: Ports 7–8 Pullup Resistor Control Registers **x'00FFB8'–x'00FFBA'**

Bit:	7	6	5	4	3	2	1	0
	PnPUP7	PnPUP6	PnPUP5	PnPUP4	PnPUP3	PnPUP2	PnPUP1	PnPUP0
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

P6PUP: Port 6 Pullup Resistor Control Register **x'00FFB6'**

Bit:	7	6	5	4	3	2	1	0
	0	0	0	0	0	0	P6PUP1	P6PUP0
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R/W	R/W

The PnPUP registers control the port pullup resistors. The bit number corresponds to the associated pin number. For instance, P0PUP7 applies to the P07 pin. These are 8-bit access registers.

- 0: Pullup resistor off
- 1: Pullup resistor on

Note that by default the P7P8CNT bit of the PCNT2 register forces the pullup resistors on for ports 7 and 8. P7PUP and P8PUP are only valid when P7P8CNT is 1.

P0OUT–P5OUT: Ports 0–5 Output Control Registers **x'00FFC0'–x'00FFC5'**

P7OUT–P8OUT: Ports 7–8 Output Control Registers **x'00FFC8'–x'00FFCA'**



Writing a 0 to P5OUT3 causes a reset to occur.

Bit:	7	6	5	4	3	2	1	0
	PnOUT7	PnOUT6	PnOUT5	PnOUT4	PnOUT3	PnOUT2	PnOUT1	PnOUT0
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

P6OUT: Port 6 Output Control Register **x'00FFC6'**

Bit:	7	6	5	4	3	2	1	0
	0	0	0	0	0	0	P6OUT1	P6OUT0
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R/W	R/W

The PnOUT registers contain the port output data. The bit number corresponds to the associated pin number. For instance, P0OUT7 applies to the P07 pin. These are 8-bit access registers.

P0IN–P5IN: Ports 0–5 Input Registers

x'00FFD0'–x'00FFD5'

P7IN–P8IN: Ports 7–8 Input Registers

x'00FFD8'–x'00FFDA'

Bit:	7	6	5	4	3	2	1	0
	PnIN7	PnIN6	PnIN5	PnIN4	PnIN3	PnIN2	PnIN1	PnIN0
Reset:	Pin	Pin	Pin	Pin	Pin	Pin	Pin	Pin
R/W:	R	R	R	R	R	R	R	R

P6IN: Port 6 Input Register

x'00FFD6'

Bit:	7	6	5	4	3	2	1	0
	0	0	0	0	0	0	P6IN1	P6IN0
Reset:	0	0	0	0	0	0	Pin	Pin
R/W:	R	R	R	R	R	R	R	R

The PnIN registers contain the port input data. The bit number corresponds to the associated pin number. For instance, P0IN7 applies to the P07 pin. These are 8-bit access registers.



When using P57 as a port, set SIFSEL0 (PCNT0 x'FF90' bp12) to '0'.

P0DIR–P5DIR: Ports 0–5 I/O Control Registers

x'00FFE0'–x'00FFE5'

P7DIR–P8DIR: Ports 7–8 I/O Control Registers

x'00FFE8'–x'00FFEA'

Bit:	7	6	5	4	3	2	1	0
	PnDIR7	PnDIR6	PnDIR5	PnDIR4	PnDIR3	PnDIR2	PnDIR1	PnDIR0
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

P6DIR: Port 6 I/O Control Register

x'00FFE6'

Bit:	7	6	5	4	3	2	1	0
	0	0	0	0	0	0	P6DIR1	P6DIR0
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R/W	R/W

The PnDIR registers control the I/O direction of the ports. The bit number corresponds to the associated pin number. For instance, P0DIR7 applies to the P07 pin. These are 8-bit access registers.

0: Input

1: Output

I/O Ports

I/O Port Control Registers

P0MD: Port 0 Output Mode Register

x'00FFF0'

Bit:	7	6	5	4	3	2	1	0
	P0MD7	P0MD6	P0MD5	P0MD4	P0MD3	P0MD2	P0MD1	P0MD0
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

P0MD is an 8-bit access register.

P0MD7: P07 function switch

- 0: P07
- 1: ADIN4

P0MD6: P06 function switch

- 0: P06
- 1: ADIN3

P0MD5: P05 function switch

- 0: P05
- 1: ADIN2

P0MD4: P04 function switch

- 0: P04
- 1: ADIN1

P0MD3: P03 function switch

- 0: P03
- 1: ADIN0

P0MD2: P02 function switch

- 0: P02
- 1: SCL1

P0MD1: P01 function switch

- 0: P01
- 1: SDA1

P0MD0: P00 output switch

Control the IRQ0 interrupt enable settings in the interrupt control registers.

- 0: P00/RMIN/IRQ0
- 1: RMIN/IRQ0

P1MD: Port 1 Output Mode Register**x'00FFF2'**

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	P1MD 14	P1MD 13	P1MD 12	P1MD 11	P1MD 10	P1MD 9	P1MD 8	P1MD 7	P1MD 6	—	P1MD 4	—	P1MD 2	—	P1MD 0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R	R/W	R	R/W	R	R/W

P1MD is a 16-bit access register.

P1MD14: P17 output switch

- 0: P17
- 1: PWM2

P1MD[13:12]: P16 output and function switch

- 00: P16
- 01: ADIN11
- 10: PWM1
- 11: Reserved

P1MD[11:10]: P15 output and function switch

- 00: P15
- 01: ADIN10
- 10: PWM0
- 11: Reserved

P1MD[9:8]: P14 output and function switch

- 00: P14
- 01: ADIN9
- 10: STOP
- 11: Reserved

P1MD[7:6]: P13 output switch

- 00: P13
- 01: ADIN8
- 10: WDOUT
- 11: Reserved

P1MD4: P12 function switch

- 0: P12/IRQ3
- 1: ADIN7

P1MD2: P11 function switch

- 0: P11/IRQ2
- 1: ADIN6

P1MD0: P10 function switch

Control the IRQ3, IRQ2, and IRQ1 interrupt enable settings in the interrupt control registers.

- 0: P10/IRQ1
- 1: ADIN5

I/O Ports

I/O Port Control Registers

P2MD: Port 2 Output Mode Register															x'00FFF4'	
Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	P2MD 14	P2MD 13	P2MD 12	P2MD 11	P2MD 10	P2MD 9	P2MD 8	—	P2MD 6	—	P2MD 4	—	P2MD 2	—	P2MD 0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R	R/W	R	R/W	R	R/W	R	R/W

P2MD is a 16-bit access register.

P2MD14: P27 function switch

To use TM0IO as an output pin, set this bit to 1 and set the P2DIR7 bit to 1.

- 0: P27
- 1: TM0IO

P2MD[13:12]: P26 output and function switch

To use TM4IOA as an output pin, set this field to b'01' and set the P2DIR6 bit to 1.

- 00: P26
- 01: TM4IOA
- 10: SBO1
- 11: Reserved

P2MD[11:10]: P25 output and function switch

If you set this field to b'10', select SBI1 or SBD1 in bit 11 of PCNT0. To use TM4IOB as an output pin, set this field to b'01' and set the P2DIR5 bit to 1.

- 00: P25
- 01: TM4IOB
- 10: SBI1 or SBD1
- 11: Reserved

P2MD[9:8]: P24 output and function switch

To use SBT1 as an input, set this field to b'00' and set the P2DIR4 bit to 0.

- 00: P24
- 01: TM4IC
- 10: SBT1
- 11: Reserved

P2MD6: P23 output switch

- 0: P23
- 1: PWM6

P2MD4: P22 output switch

- 0: P22
- 1: PWM5

P2MD2: P21 output switch

- 0: P21
- 1: PWM4

P2MD0: P20 output switch

- 0: P20
- 1: PWM3

P3MD: Port 3 Output Mode Register

x'00FFF6'

Bit:	7	6	5	4	3	2	1	0
	P3MD7	P3MD6	P3MD5	P3MD4	P3MD3	P3MD2	P3MD1	P3MD0
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

P3MD is an 8-bit access register.

P3MD7: P37 output switch

If you set this field to 1, select DABOUT or B in the RGBC bit of ODS1.

- 0: P37
- 1: DABOUT or B

P3MD6: P36 output switch

If you set this field to 1, select DAGOUT or G in the RGBC bit of ODS1.

- 0: P36
- 1: DAGOUT or G

P3MD5: P35 output switch

If you set this field to 1, select DAROUT or R in the RGBC bit of ODS1.

- 0: P35
- 1: DAROUT or R

P3MD4: P34 function switch

- 0: P34
- 1: V_{REF}

P3MD[3:1]: P33-P30 function switch

Set P3MD3 to 1 only when P3MD[2:1] is 01; otherwise, set it to 0.

- 000: P30/P31/P32/P33
- 101: CLH/CVBS0/P32/CLL
- 010: CLH/P31/CVBS1/CLL
- 011: CLH/CVBS0/CVBS1/CLL

P3MD0

This bit exists, but contains no function.

P4MD: Port 4 Output Mode Register

x'00FFF8'

Bit:	7	6	5	4	3	2	1	0
	P4MD7	P4MD6	P4MD5	P4MD4	P4MD3	P4MD2	P4MD1	P4MD0
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

P4MD is an 8-bit access register.

P4MD7: P47 function switch

- 0: P47/NHSYNC
- 1: NHSYNC

P4MD6

This bit exists, but contains no function.

P4MD5: P45 function switch

- To use P45 or P46, set the OSCSEL[1:0] field of OSD1 to b'00'.
- 0: P45/OSDXO
- 1: P46/OSDXI

P4MD4: P44 output switch

- 0: P44/TM5IC/HI0
- 1: TM5IC/HI0

P4MD3: P43 output switch

- To use TM5IOB as an output pin, set this bit to 1 and set the P4DIR3 bit to 1.
- 0: P43/HI1
- 1: TM5IOB/HI1

P4MD2: P42 output switch

- To use TM5IOA as an output pin, set this bit to 1 and set the P4DIR2 bit to 1.
- 0: P42
- 1: TM5IOA

P4MD1: P41 output switch

- To use TM1IO as an output pin, set this bit to 1 and set the P4DIR1 bit to 1.
- 0: P41
- 1: TM1IO

P4MD0: P40 output switch

- If you set this bit to 1, select DAYMOUT or YM in the YCNT bit of ODS1.
- 0: P40
- 1: YM/DAYMOUT

P5MD: Port 5 Output Mode Register**x'00FFFA'**

Bit:	7	6	5	4	3	2	1	0
	P5MD7	P5MD6	P5MD5	P5MD4	P5MD3	P5MD2	P5MD1	P5MD0
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

P5MD is an 8-bit access register.

P5MD7: P57 output switch

To use SBT0 as an input pin, set this field to 0 and set the P5DIR7 bit to 0.

- 0: P57
- 1: SBT0

P5MD6: P56 output switch

If you set this bit to 1, select SBI0 or SBD0 in the bit 10 of PCNT0.

- 0: P56
- 1: SBI0/SBD0

P5MD5: P55 output switch

- 0: P55
- 1: SBO0

P5MD4: P54 function switch

- 0: P54/IRQ5/VSYNC
- 1: IRQ5/VSYNC

P5MD3

This bit exists, but contains no function.

P5MD2: P52 output switch

- 0: P52/IRQ4/VI0
- 1: IRQ4/VI0

P5MD1: P51 output switch

- 0: P51
- 1: YS

P5MD0: P50 output switch

If you set this bit to 1, set the SYSCLK frequency in bits [15:14] of PCNT0.

- 0: P50
- 1: SYSCLK/divided SYSCLK output

P6MD: Port 6 Output Mode Register**x'00FFFC'**

Bit:	7	6	5	4	3	2	1	0
	0	0	0	0	0	0	P6MD1	P6MD0
Reset:	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R/W	R/W

P6MD is an 8-bit access register.

P6MD1: P61 function switch

- 0: P61
- 1: SCL0

P6MD0: P60 function switch

- 0: P60
- 1: SDA0

I/O Ports

I/O Port Control Registers

PCNT0: Port Control Register 0

x'00FF90'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	SCLK F1	SCLK F0	OD ASCII	OD ASCI0	SIF SEL1	SIF SEL0	I2C SEL1	I2C SEL0	OSD POFF	PLL POFF	ADC1 ON	ADC0 ON	HCNT OFF	RMC OFF	VBII OFF	VBIO OFF
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

PCNT0 is a 16-bit access register.



Enable PWM (set PCNT1 bit 1 to 1) if you are outputting f_{SY} .

$SCLK/2^{14}$.

SCLKF[1:0]: SYSCLK frequency select
 00: $SYSCLK \times 2^{14}$ (732.42 Hz)
 01: $VCOCLK \times 3$
 10: $SYSCLK \times 2$
 11: SYSCLK

ODASCII1: Serial port 1 output switch
 1: Push-pull
 0: Open-drain

ODASCI0: Serial port 0 output switch
 To use P57 as a port, set this bit to 0.
 1: Push-pull
 0: Open-drain

SIFSEL1: Serial port 1 interface select
 0: Three-line (enable SBI1, SBO1, SBT1)
 1: Two-line (enable SBD1, SBT1)

SIFSEL0: Serial port 0 interface select
 0: Three-wire (enable SBI0, SBO0, SBT0)
 1: Two-wire (enable SBD0, SBT0)

I2CSEL1: SDA1, SCL1 enable
 0: Disable
 1: Enable

To use P01/SDA1 and P02/SCL1 as general-purpose port pins, you must both select the ports in the port mode register and disable this bit.

I2CSEL0: SDA0, SCL0 enable
 0: Disable
 1: Enable

To use P60/SDA0 and P61/SCL0 as general-purpose port pins, you must both select the ports in the port mode register and disable this bit.



To turn off the OSD block to save power:

1. Write a 0 to OSD (OSD1, bit 10).
2. Wait for the next VSYNC input.
3. Write a 0 to OSDPOFF (PCNT0, bit 7), turning the clock off.

If you turn the clock off before the VSYNC input, power usage may not drop or the microcontroller may halt.

OSDPOFF: OSD circuit enable

Setting this bit to 0 shuts off the system clock supply to the OSD block, reducing power dissipation. The program must write a 1 to this bit before writing any values to the OSD registers.

0: Disable

1: Enable

PLLPOFF: PLL circuit enable

0: Enable

1: Stop

ADC1ON: ADC circuit enable for closed-caption decoder 1

0: Disable

1: Enable

ADC0ON: ADC circuit enable for closed-caption decoder 0

0: Disable

1: Enable

HCNTOFF: H counter circuit enable

0: Enable

1: Disable

RMCOFF: IR remote signal receiver circuit enable

0: Enable

1: Disable

VBI1OFF: Closed-caption decoder 1 circuit enable

0: Enable

1: Disable

VBI0OFF: Closed-caption decoder 0 circuit enable

0: Enable

1: Disable

Setting the HCNTOFF, RMCOFF, VBI1OFF, or VBI0OFF bits to 1 shuts off the system clock supply to the associated block, which reduces power dissipation.

I/O Ports

I/O Port Control Registers

PCNT2: Port Control Register 2

x'00FF92'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	P7P8 CNT	(Test Bits)					I2C OFF	PWM OFF	OSD REGE
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W



Always set bits 7 to 3 of PCNT2 to 0.



You cannot read from or write to the registers associated with a function that is disabled.

P7P8CNT: Ports 7 and 8 forced pullup

Ports 7 and 8 are only available in the quad flat package.

- 0: Pull up
- 1: Don't pull up

I2COFF: I²C function enable

- 0: Enable
- 1: Disable

PWMOFF: PWM function enable

- 0: Enable
- 1: Disable

Setting the I2COFF or PWMOFF bits to 1 shuts off the system clock supply to the associated block, which reduces power dissipation.

OSDREGE: OSD registers read/write enable

To read or write to the OSD registers, you must first set this bit to 1.

- 0: Disable
- 1: Enable

12 ROM Correction

12.1 Description

The ROM correction function can correct the program data in any address within the 256-kilobyte ROM. (It cannot correct OSD ROM data.) A maximum of sixteen addresses can be corrected. Addresses are set as address match interrupts. This function shortens time-to-market for large-scale designs, since changes can be implemented in the software after the mask ROM is complete.

The ROM correction function has numerous other applications. For instance, you can insert keywords into the functional routines, then use the function to send internal status information to an external location. This enables system-level examination of the internal status even with the mask ROM version.

To use the ROM correction function, embed a routine such as that shown in figure 12-2 in the ROM.

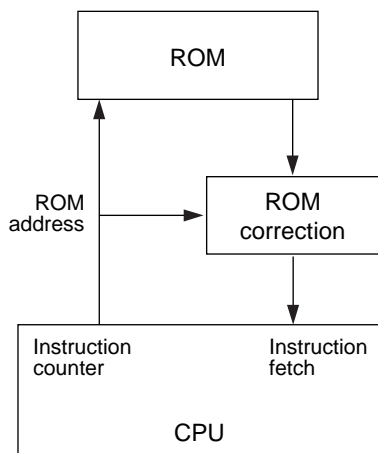


Figure 12-1 ROM Area Schematic Diagram

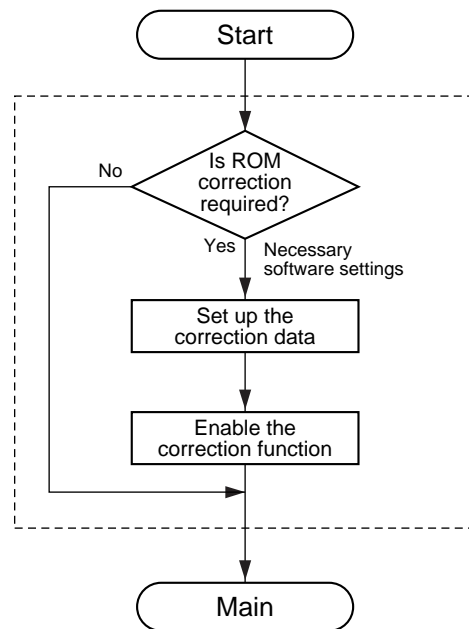


Figure 12-2 ROM Correction Flow

As figure 12-1 shows, the function lies between the microcontroller and ROM blocks. First set the correction data for any sixteen non-OSD addresses in the ROM correction address match and data registers. (Follow the flow shown in figure 12-2.) Once this is done, the circuit will correct the ROM output for the designated addresses.

12.2 Block Diagram

Figure 12-3 is a block diagram of the ROM correction circuit. A match detection circuit constantly monitors the ROM address specified by the CPU instruction pointer (IP). When the value matches a correction address, the circuit replaces the data output from the ROM with the data in the appropriate correction data register. It then sends the corrected data to the CPU.

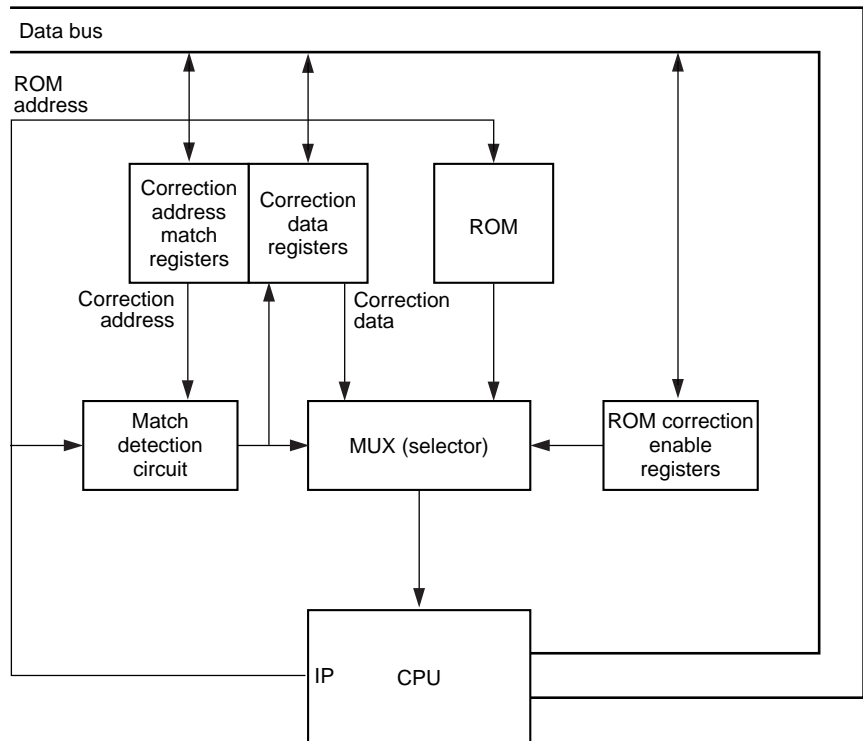


Figure 12-3 ROM Correction Block Diagram

12.3 Programming Considerations

At reset, the ROM correction address match and data registers contain all 0s. Since a reset also disables ROM correction (in ROMCEN), the ROM will still operate normally.

Only read from or write to the address match registers while ROM correction is disabled in ROMCEN. Otherwise, an error may occur in the match detection circuit.

Note that the address match and data registers only allow full-register access (8-bit or 16-bit depending on the register). You cannot write to individual bits.

12.4 ROM Correction Control Registers

Table 12-1 shows the organization of the address match and data registers for ROM correction. Write a ROM address to be corrected to an AMCHIH_n and AMCHIL_n register pair and write the corrected data to the associated CHDAT_n register. Enable ROM correction for the associated address in the ROMCEN register. Enable ROM correction for the associated address in the ROMCEN register.

Table 12-1 ROM Correction Address Match and Data Registers

ROM Address	Address Match Register		Data Register
	High Order	Low Order	
Address 0	AMCHIH0 x'00FD02'	AMCHIL0 x'00FD00'	CHDAT0 x'00FD40'
Address 1	AMCHIH1 x'00FD06'	AMCHIL1 x'00FD04'	CHDAT1 x'00FD44'
Address 2	AMCHIH2 x'00FD0A'	AMCHIL2 x'00FD08'	CHDAT2 x'00FD48'
Address 3	AMCHIH3 x'00FD0E'	AMCHIL3 x'00FD0C'	CHDAT3 x'00FD4C'
Address 4	AMCHIH4 x'00FD12'	AMCHIL4 x'00FD10'	CHDAT4 x'00FD50'
Address 5	AMCHIH5 x'00FD16'	AMCHIL5 x'00FD14'	CHDAT5 x'00FD54'
Address 6	AMCHIH6 x'00FD1A'	AMCHIL6 x'00FD18'	CHDAT6 x'00FD58'
Address 7	AMCHIH7 x'00FD1E'	AMCHIL7 x'00FD1C'	CHDAT7 x'00FD5C'
Address 8	AMCHIH8 x'00FD22'	AMCHIL8 x'00FD20'	CHDAT8 x'00FD60'
Address 9	AMCHIH9 x'00FD26'	AMCHIL9 x'00FD24'	CHDAT9 x'00FD64'
Address 10	AMCHIIA x'00FD2A'	AMCHILA x'00FD28'	CHDAT10 x'00FD68'
Address 11	AMCHIIB x'00FD2E'	AMCHILB x'00FD2C'	CHDAT11 x'00FD6C'
Address 12	AMCHIIHC x'00FD32'	AMCHILC x'00FD30'	CHDAT12 x'00FD70'
Address 13	AMCHIID x'00FD36'	AMCHILD x'00FD34'	CHDAT13 x'00FD74'
Address 14	AMCHIIHE x'00FD3A'	AMCHILE x'00FD38'	CHDAT14 x'00FD78'
Address 15	AMCHIIHF x'00FD3E'	AMCHILF x'00FD3C'	CHDAT15 x'00FD7C'

Note: All registers reset to 0.

ROMCEN: ROM Correction Enable Register

x'00FCF0'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ROMC	ROMC	ROMC	ROMC	ROMC	ROMC	ROMC	ROMC	ROMC	ROMC	ROMC	ROMC	ROMC	ROMC	ROMC	ROMC
	EN15	EN14	EN13	EN12	EN11	EN10	EN9	EN8	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

ROMCEN15: Address 15 ROM correction enable

- 0: Disable
- 1: Enable

ROMCEN14: Address 14 ROM correction enable

- 0: Disable
- 1: Enable

ROMCEN13: Address 13 ROM correction enable

- 0: Disable
- 1: Enable

ROM Correction

ROM Correction Control Registers

ROMCEN12: Address 12 ROM correction enable

0: Disable

1: Enable

ROMCEN11: Address 11 ROM correction enable

0: Disable

1: Enable

ROMCEN10: Address 10 ROM correction enable

0: Disable

1: Enable

ROMCEN9: Address 9 ROM correction enable

0: Disable

1: Enable

ROMCEN8: Address 8 ROM correction enable

0: Disable

1: Enable

ROMCEN7: Address 7 ROM correction enable

0: Disable

1: Enable

ROMCEN6: Address 6 ROM correction enable

0: Disable

1: Enable

ROMCEN5: Address 5 ROM correction enable

0: Disable

1: Enable

ROMCEN4: Address 4 ROM correction enable

0: Disable

1: Enable

ROMCEN3: Address 3ROM correction enable

0: Disable

1: Enable

ROMCEN2: Address 2 ROM correction enable

0: Disable

1: Enable

ROMCEN1: Address 1 ROM correction enable

0: Disable

1: Enable

ROMCEN0: Address 0 ROM correction enable

0: Disable

1: Enable

AMCHIH0–AMCHIH7: ROM Correction Address Match Register n (High)

Bit:	7	6	5	4	3	2	1	0
	CHAD 23	CHAD 22	CHAD 21	CHAD 20	CHAD 19	CHAD 18	CHAD 17	CHAD 16
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

AMCHIH_n is an 8-bit access register.

CHAD[23:16]: Correction address bits A23 to A16 (A23 = MSB)

AMCHIL0–AMCHIL7: ROM Correction Address Match Register n (Low)

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	CHAD 15	CHAD 14	CHAD 13	CHAD 12	CHAD 11	CHAD 10	CHAD 9	CHAD 8	CHAD 7	CHAD 6	CHAD 5	CHAD 4	CHAD 3	CHAD 2	CHAD 1	CHAD 0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

AMCHIL_n is a 16-bit access register.

CHAD[15:0]: Correction address bits A15 to A0

CHDAT0–CHDAT7: ROM Correction Data Register n

Bit:	7	6	5	4	3	2	1	0
	CHD 7	CHD 6	CHD 5	CHD 4	CHD 3	CHD 2	CHD 1	CHD 0
Reset:	0	0	0	0	0	0	0	0
R/W:	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

CHDAT_n is an 8-bit access register.

CHD7: Correction data bit D15 or D7 for address n

CHD6: Correction data bit D14 or D6 for address n

CHD5: Correction data bit D13 or D5 for address n

CHD4: Correction data bit D12 or D4 for address n

CHD3: Correction data bit D11 or D3 for address n

CHD2: Correction data bit D10 or D2 for address n

CHD1: Correction data bit D9 or D1 for address n

CHD0: Correction data bit D8 or D0 for address n

13 I²C Bus Controller

13.1 Description

The MN102H75K/85K contains one I²C bus controller, fully compliant with the I²C specification, that can control one of two I²C bus connections.

An I²C bus is a simple, two-wire bus for transferring data between ICs. Since it requires only two lines, a serial data line (SDA) and a serial clock line (SCL), it minimizes interconnections so ICs have fewer pins and there are less PCB tracks. The result is smaller and less expensive PCBs. Figure 13-1 shows a typical I²C bus application.

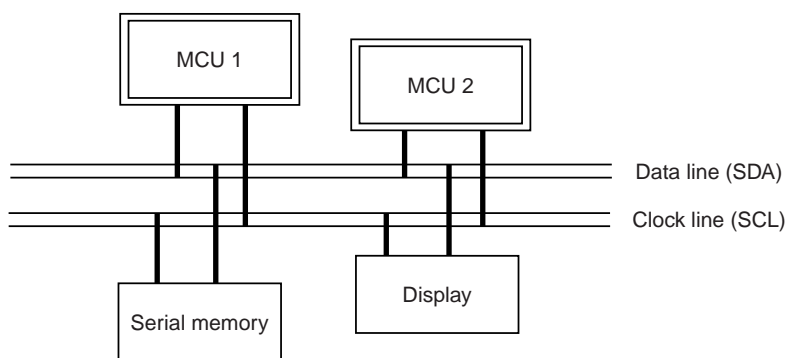


Figure 13-1 Example of I²C Bus Application

In an I²C bus system, devices are considered as masters or slaves when performing data transfers. A master is the device that initiates a data transfer on the bus and generates the clock signals to permit that transfer. At that time, any device addressed is considered a slave. Table 13-1 defines some I²C bus terminology.

Table 13-1 I²C Bus Terminology

Term	Description
Transmitter	The device that sends the data to the bus
Receiver	The device that receives the data from the bus
Master	The device that initiates a transfer, generates clock signals, and terminates a transfer
Slave	The device addressed by a master
Multimaster	More than one device capable of controlling the bus can be connected to it. More than one master can attempt to control the bus at the same time without corrupting the message. The system is not dependent on any single master.
Arbitration	Procedure to ensure that, if more than one master simultaneously tries to control the bus, only one is allowed to do so and the message is not corrupted. The device that loses arbitration becomes the slave of the device that wins.
Synchronization	Procedure to synchronize the clock signals of two or more devices

Figure 13-2 shows an example of an I²C bus configuration using two microcontrollers. Both I²C bus lines, SDA and SCL are bidirectional lines, connected to a positive supply voltage via a pullup resistor. The open-drain output pins of the microcontrollers perform the wired-AND function on the bus. The software controls when each microcontroller operates as a transmitter or receiver, or whether is in master or slave mode.

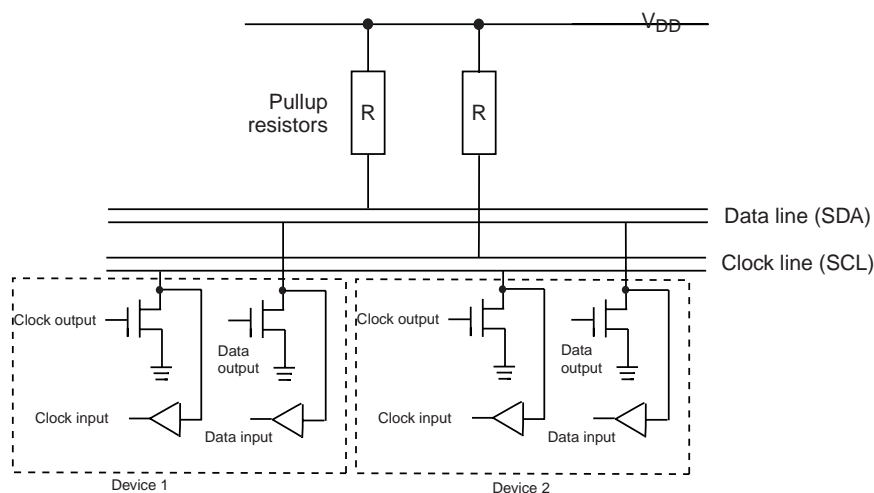


Figure 13-2 Connection of Two Microcontrollers to the I²C Bus

Table 13-2 describes the four possible operating modes for devices on the I²C bus.

Table 13-2 Operating Modes for Devices on an I²C Bus

Operating Mode	Description
Master transmitter	Device that generates the serial transfer clock (SCL) signal and transmits serial data to a slave device in sync with SCL
Master receiver	Device that generates the SCL signal and receives serial data from a slave device in sync with SCL
Slave transmitter	Device that transmits data in sync with the SCL signal from the master
Slave receiver	Device that receives data in sync with the SCL signal from the master

Figure 13-3 shows the MN102H75K/85K operation sequence in each of these modes. In all modes, the I²C bus controller generates an interrupt after each data byte transfer, then the software loads the next data byte.

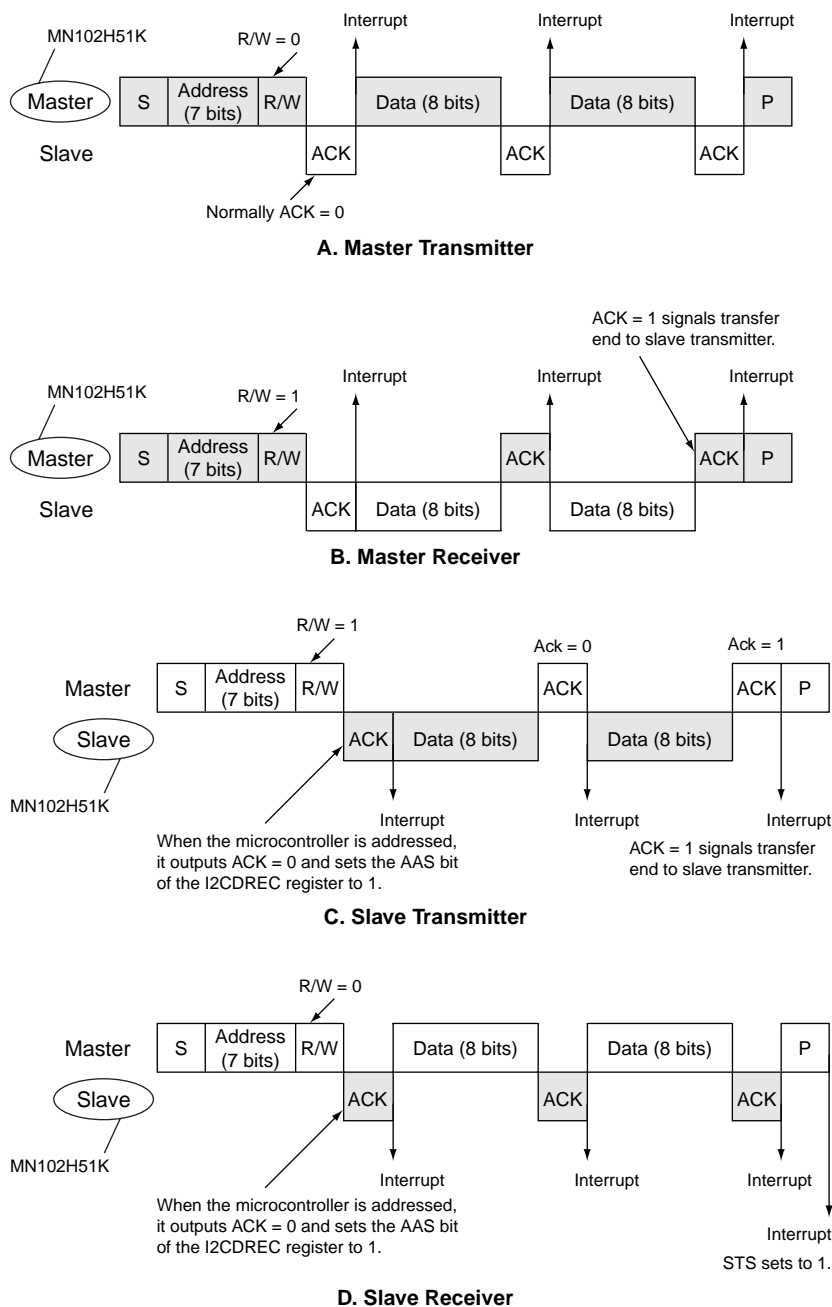


Figure 13-3 I²C Bus Interface Operation

13.2 Block Diagram

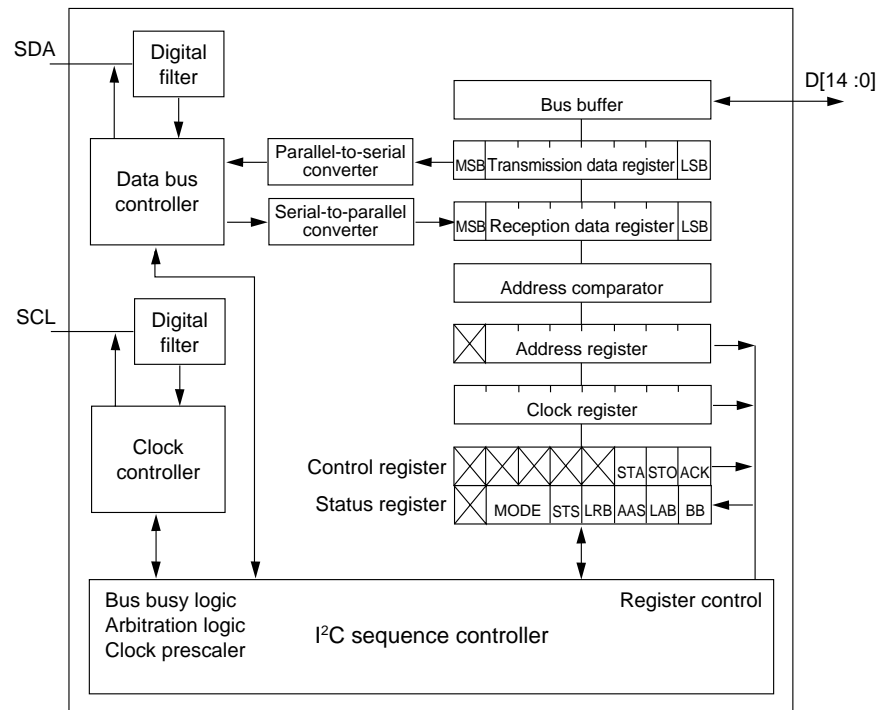


Figure 13-4 I²C Bus Controller Block Diagram

13.3 Functional Description

The I²C bus controller contains the registers shown in table 13-3. See the page number indicated for register and bit descriptions.

Table 13-3 Control Registers for Clamping Circuit

Register	Page	Address	Description
I2CDTRM	304	x'007E40'	I ² C transmission data register
I2CDREC	305	x'007E42'	I ² C reception data register
I2CMYAD	305	x'007E44'	I ² C self address register
I2CCLK	306	x'007E46'	I ² C clock control register
I2CBRST	306	x'007E48'	I ² C bus reset register
I2CBSTS	306	x'007E4A'	I ² C bus status register

■ Arbitration and bus busy control

The I²C bus controller allows software control, but implements communication timing and bus arbitration completely in the hardware.

- ◆ **Arbitration:** Controlled by the software, but implemented completely in the hardware.
- ◆ **Bus busy:** Checked by the hardware. This eliminates the need for the software to check whether the bus is busy. The program can request a transfer to the I²C bus at any time.

■ Register settings conversions to I²C protocol

The I²C bus controller converts the data in the I2CDTRM register to the I²C protocol.

■ Transfer modes changes

A write to the I2CDTRM register indicates the transfer mode (master transmitter/receiver or slave transmitter/receiver) for a new transfer. To minimize software control, the hardware generates an interrupt each time a transfer ends. During interrupt servicing, the SCL line stays low, then clears to high on a write to I2CDTRM. (When the microcontroller is a slave transmitter and the transfer ends, SCL goes high on a read to the I2CDREC register after an ACK = 1 (negative acknowledge) interrupt.)

■ Multimaster support

The hardware performs bus arbitration for a multimaster system. When it loses an arbitration, the hardware immediately stops the data transfer and generates an interrupt.

■ Address decoding

The I²C bus controller decodes the microcontroller's address, set in the I2CMYAD register, when the microcontroller is a slave device. It also decodes the general code address (0).

■ Forced bus reset

Through software control, by a write to the I2CBRST register, the I²C bus controller can force the SCL line to reset to low when a bus error occurs. This resets the entire I²C bus controller circuit, leaving the microcontroller in slave receiver mode. It does not change the contents of the I2CMYAD and I2CCLK registers.

■ Clock frequency adjustment

The I2CCLK register sets the serial clock frequency, allowing synchronization with low-speed devices. With a 12-MHz oscillator, the maximum setting is 100 kHz and the minimum setting is 10 kHz.

■ Bus state monitoring

With the I2CBSTS register, the I²C bus controller determines the logic levels of the SCL and SDA lines.

13.4 Setting Up the I²C Bus Connection

Set the I²C connection in the I2CSEL0 and I2CSEL1 bits of the PCNT0 register (x'00FF90'). Since the SCL0, SDA0, SCL1, and SDA1 pins also serve as general-purpose port pins, and reset to the general-purpose function, you must set these bits every time the program uses the I²C function. You must also select the I²C function in the port mode registers. For I²C bus connection 0, set bits 0 and 1 of the P6MD register (x'00FFFC'). For I²C bus connection 1, set bits 1 and 2 of the P0MD register (x'00FFF0').

Table 13-4 shows the register settings required to use either SDA0/SCL0 or SDA1/SCL1 alone, and figure 13-5 shows the control circuit for this pin setup.

Table 13-4 Registers Settings for SDA0/SCL0 or SDA1/SCL1 Ports

Register	Bit	SDA0, SCL0 Only	SDA1, SCL1 Only
P0MD (x'00FFF0')	1	0 (selects P01)	1 (selects SDA1)
	2	0 (selects P02)	1 (selects SCL1)
P6MD (x'00FFFC')	0	1 (selects SDA0)	0 (selects P60)
	1	1 (selects SCL0)	0 (selects P61)
PCNT0 (x'00FF90')	8	1 (enables SDA0, SCL0)	0 (disables SDA0, SCL0)
	9	0 (disables SDA1, SCL1)	1 (enables SDA1, SCL1)

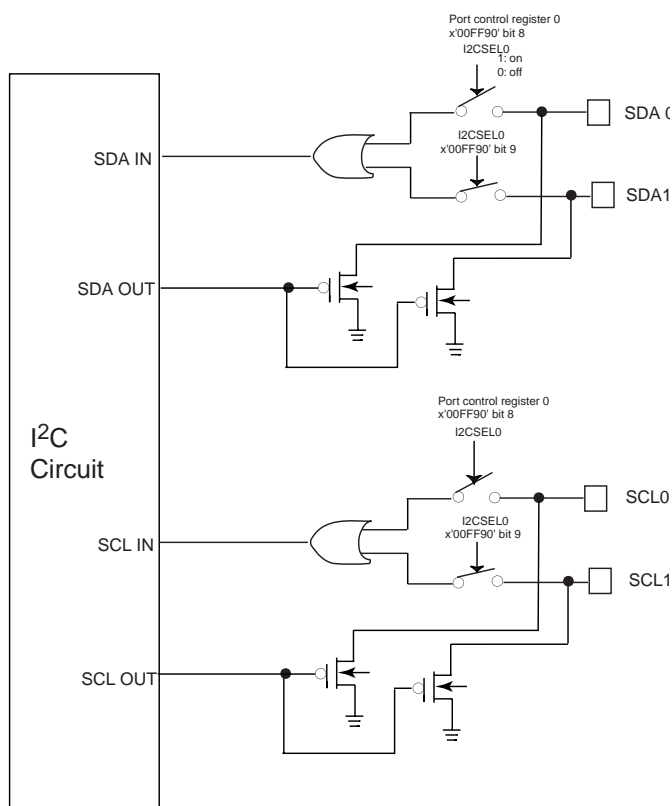


Figure 13-5 Pin Control Circuit for the I²C Bus Controller

13.5 SDA and SCL Waveform Characteristics

Figure 13-6 and table 13-5 provide the timing definitions and specifications for the for the MN102H75K/85K I²C bus interface.

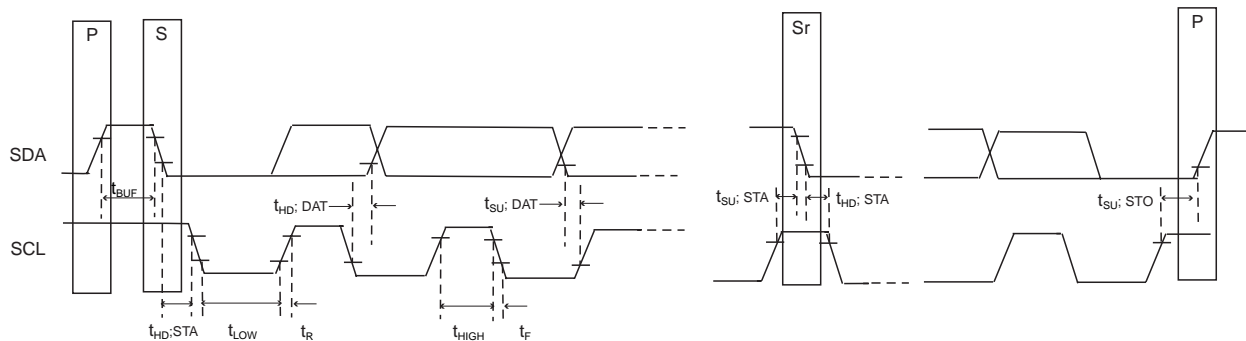


Figure 13-6 SDA and SCL Waveforms

Table 13-5 SDA and SCL Waveform Characteristics

Parameter	Symbol	Min	Max	Unit
SCL clock frequency	f_{SCL}	0	100	kHz
Bus free time between a stop and start condition	t_{BUF}	20	—	μs
Hold time (repeated) start condition	$t_{HD:STA}$	4.0	—	
Low period of the SCL clock	t_{LOW}	4.7	—	
High period of the SCL clock	t_{HIGH}	4.0	—	
Setup time for a repeated start condition	$t_{SU:STA}$	4.7	—	
Data hold time	$t_{HD:DAT}$	300	—	ns
Data setup time	$t_{SU:DAT}$	250	—	
SDA and SCL rise time	t_R	—	1000	
SDA and SCL fall time	t_F	—	300	
Stop condition setup time	$t_{SU:STO}$	4.0	—	μs

13.6 I²C Interface Setup Examples

13.6.1 Setting Up a Transition from Master Transmitter to Master Receiver

This example demonstrates how to set up a data transfer when changing from master transmitter to master receiver. Figure 13-7 shows an example waveform.

13.6.1.1 Pre-configuring

■ **To set up the I/O port:**

Set port control register 0 (PCNT0; x'00FF90') to x'0100' (enabling the SDA0 and SCL0 pins) and set the port 6 output mode register (P6MD; x'00FFFC') to x'0003' (selecting the SDA0 and SCL0 functions).

■ **To enable I²C interrupts:**

Set the I²C interrupt control register pair (I2C0ICH and I2C0ICL; x'00FC9C') to x'0100'.

■ **To set up the I²C registers:**

1. Set the I2CCLK register (x'007E46') to x'0041', selecting a clock frequency of 80 kHz.
2. Set the I2CDTRM register (x'007E40') to x'05FD'. This sets STA to 1, STP to 0, and ACK to 0. Bits 7 to 1 of the transmission data setting (x'FD') indicate the address (b'1111110') of the slave device from which the microcontroller will request the data, and bit 0 indicates the read/write setting (bit 0 = 1 = read).

13.6.1.2 Setting Up the First Interrupt

When an ACK = 0 signal returns from the slave device, the I²C bus controller generates an interrupt. At this point, implement the following settings:

■ **To set up the interrupt:**

Set the I2C0ICH and I2C0ICL register pair (x'00FC9C') to x'0100'. This enables I²C interrupts and clears the previous interrupt request.

■ **To set up the I²C registers:**

1. Read the I2CDREC register (x'007E42') to determine the I²C bus controller status.
2. Since the microcontroller will become a receiver on the next operation, set the I2CDTRM register (x'007E40') to x'0000'. This sets STA, STP, ACK, and the transmission data to 0s. With this setting, the microcontroller returns an ACK = 0 signal on the ninth clock.

13.6.1.3 Setting Up the Second Interrupt

When the microcontroller receives the data x'85' from the slave device, it returns an ACK = 0 signal and the I²C bus controller generates an interrupt. At this point, implement the following settings:

■ **To set up the interrupt:**

Set the I2C0ICH and I2C0ICL register pair (x'00FC9C') to x'0100'. This enables I²C interrupts and clears the previous interrupt request.

■ **To set up the I²C registers:**

1. Read the I2CDREC register (x'007E42') to determine the I²C bus controller status.
2. Since the communication will end when the microcontroller receives the next data byte, set the I2CDTRM register (x'007E40') to x'0100'. This sets STA to 0, STP to 0, ACK to 1, and the transmission data to x'00'. With this setting, the microcontroller returns an ACK = 1 signal on the ninth clock.

13.6.1.4 Setting Up the Third Interrupt

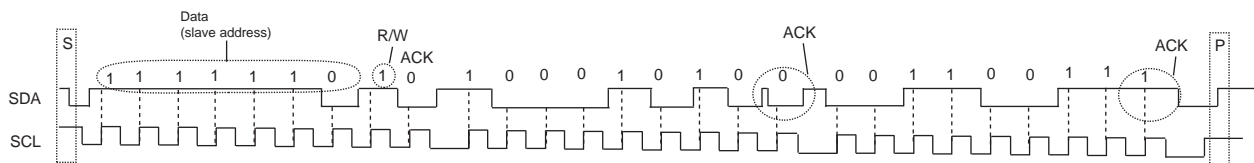
When the microcontroller receives the data x'33' from the slave device, it returns an ACK = 1 signal and the I²C bus controller generates an interrupt. At this point, implement the following settings:

■ **To set up the interrupt:**

Set the I2C0ICH and I2C0ICL register pair (x'00FC9C') to x'0100'. This enables I²C interrupts and clears the previous interrupt request.

■ **To set up the I²C registers:**

1. Read the I2CDREC register (x'007E42') to determine the I²C bus controller status.
2. Since the transfer has ended, set the I2CDTRM register (x'007E40') to x'0300'. This sets STA to 0, STP to 1, ACK to 1, and the transmission data to x'00'. With this setting, the microcontroller issues a stop condition and frees the bus.



Note: The circled areas are signals output from the MN102H75K/85K.

Figure 13-7 Waveform for Master Transmitter Transitioning to Master Receiver

13.6.2 Setting Up a Transition from Slave Receiver to Slave Transmitter

This example demonstrates how to set up a data transfer when changing from slave receiver to slave transmitter. Figure 13-8 shows an example waveform.

13.6.2.1 Pre-configuring

■ **To set up the I/O port:**

Set port control register 0 (PCNT0; x'00FF90') to x'0300' (enabling the SDA1 and SCL1 pins) and set the port 0 output mode register (POMD; x'00FFF0') to x'0006' (selecting the SDA1 and SCL1 functions).

■ **To enable I²C interrupts:**

Set the I²C interrupt control register pair (I2C0ICH and I2C0ICL; x'00FC9C') to x'0100'.

■ **To set up the I²C registers:**

1. Set the I2CMYAD register (x'007E44') to x'0024'. This sets the slave address of the microcontroller.
2. Set the I2CDTRM register (x'007E40') to x'0000'. This sets STA, STP, ACK, and the transmission data to 0s. With this setting, the microcontroller returns an ACK = 0 signal when an address match occurs. The master sends data (the slave address) to the slave microcontroller in sync with the master clock. When the R/W bit = 1, the microcontroller changes from a slave receiver to a slave transmitter.

13.6.2.2 Setting Up the First Interrupt

Once the microcontroller becomes a slave transmitter, set up the transmission data.

■ **To set up the interrupt:**

Set the I2C0ICH and I2C0ICL register pair (x'00FC9C') to x'0100'. This enables I²C interrupts and clears the previous interrupt request.

■ **To set up the I²C registers:**

1. Read the I2CDREC register (x'007E42') to determine the I²C bus controller status. AAS should be 1.
2. Set the I2CDTRM register (x'007E40') to x'0155'. This sets STA to 0, STP to 0, ACK to 1, and the transmission data to x'55'. The microcontroller does not need to issue an ACK signal in this transfer, so the ACK bit should be 1.
3. Begin transmitting data in sync with the clock from the master.

13.6.2.3 Setting Up the Second Interrupt

The master sends an ACK = 0 signal, so the microcontroller must send the next data byte. Set up the transmission data as follows:

■ **To set up the interrupt:**

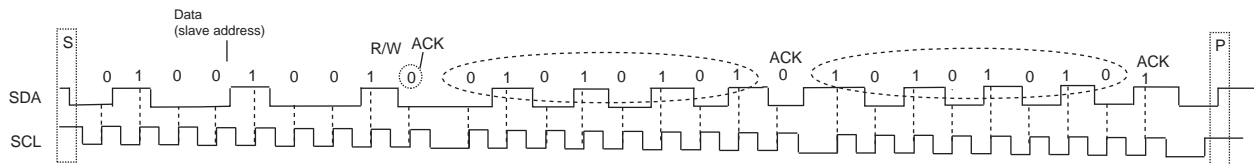
Set the I2C0ICH and I2C0ICL register pair (x'00FC9C') to x'0100'. This enables I²C interrupts and clears the previous interrupt request.

■ **To set up the I²C registers:**

1. Read the I2CDREC register (x'007E42') to determine the I²C bus controller status. The previous read from I2CDREC cleared the AAS, so AAS should be 0.
2. Set the I2CDTRM register (x'007E40') to x'01AA0'. This sets STA to 0, STP to 0, ACK to 1, and the transmission data to x'AA'. The microcontroller does not need to issue an ACK signal in this transfer, so the ACK bit should be 1.
3. Begin transmitting data in sync with the clock from the master.

13.6.2.4 Setting Up the Third Interrupt

The master send an ACK = 1 signal, then issues a stop condition, ending the communication.



Note: The circled areas are signals output from the MN102H75K/85K.

Figure 13-8 Waveform for Slave Receiver Transitioning to Slave Transmitter

13.7 I²C Bus Interface Registers

All registers in I²C block cannot be written by byte (by word only). Read by byte is possible.

I2CDTRM: I²C Transmission Data Register

x'007E40'

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	—	—	—	STA	STO	ACK	DT7	DT6	DT5	DT4	DT3	DT2	DT1	DT0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W



SCL is held low during interrupt servicing, and is cleared high by a write to I2CDTRM.

STA: I²C start control

STO: I²C stop control

Writing to the STA and STO bits allows you to change the state of the transmission or reception operation. Table 13-6 shows the settings for different start and stop conditions.

Table 13-6 STA and STO Settings

STA	STO	Mode	Function	Description
0	0	All	NOP	No state change
1	1	All	NOP	No state change
1	0	Slave receiver	Start	Change to mode indicated by R/W bit. R/W = 0: Change to master transmitter R/W = 1: Change to master receiver
		Master transmitter	Repeat start	
0	1	Slave receiver	Stop read	Change to slave receiver after stop condition.
		Master transmitter	Stop write	

ACK: Acknowledge signal output control

The acknowledge signal is output after every byte transfer, on the ninth clock pulse. ACK is normally 1 and transitions to 0 to output an acknowledge (for instance if the master or slave receiver has received a data byte).

DT[7:0]: Data to be transmitted

The parallel data in this field is converted to serial data for transmission to the I²C bus. It is shifted out MSB first to the interface.

I2CDREC: I²C Reception Data Register

x'007E42'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	MODE1	MODE0	STS	LRB	AAS	LAB	BB	D7	D6	D5	D4	D3	D2	D1	D0
Reset:	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

The I2CDREC register contains the status bits for monitoring the device and the reception data. I2CDREC is a read-only register.

MODE[1:0]: I²C device mode

This field indicates which I²C mode the microcontroller is in. MODE1 indicates slave or master, and MODE0 indicates receiver or transmitter. If the microcontroller loses an arbitration or if a stop condition occurs, the hardware clears MODE[1:0] to b'00'.

- 00: Slave receiver 10: Master receiver
- 01: Slave transmitter 11: Master transmitter

STS: Stop condition at slave receiver

Set to 1 when a stop condition is detected while the microcontroller is in slave receiver mode.

LRB: Last received bit.

Stores the last serial data bit received. LRB normally indicates the ACK cycle data.

AAS: Addressed as slave

Set to 1 when the slave address on the bus matches the contents of the address register or matches the general address (x'00'). AAS resets after a read from the I2CDREC register.

LAB: Lost arbitration bit

Set to 1 when the microcontroller loses a bus arbitration. LAB resets when I2CDTRM indicates a start condition (STA = 1).

BB: Bus busy bit

A start condition on the bus sets this flag to 0, and a stop condition resets it to 1. The microcontroller considers the bus to be busy as long as BB = 0.

D[7:0]: Received data

The serial data received from the I²C bus is shifted into this field MSB first.

I2CMYAD: I²C Self Address Register

x'007E44'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	—	—	A6	A5	A4	A3	A2	A1	A0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

A[6:0]: Microcontroller address

This register is formed from a 7-bit field address latch. It holds the microcontroller's own address, used for a compare when the microcontroller is addressed as a slave. When a match occurs, AAS is set to 1.

I2CCLK: I²C Clock Control Register

x'007E46'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	C9	C8	C7	C6	C5	C4	C3	C2	C1	C0
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W



To conform to the specification, the clock signal must be between 0 and 100 kHz. To satisfy this requirement, always set I2CCLK to x'032' or higher.

C[9:0]: Output clock frequency select

This 10-bit field determines the SCL output. With a 12-MHz system clock, calculate the frequency as follows:

$$f_{SCL} = \frac{12 \text{ MHz}}{2 \times (\text{Register setting} + 10)}$$

In this case, the following settings apply:

x'032': 100 kHz	x'06E': 50 kHz
x'039': 89.6 kHz	x'08C': 40 kHz
x'041': 80 kHz	x'0BE': 30 kHz
x'04C': 69.8 kHz	x'122': 20 kHz
x'05A': 60 kHz	x'24E': 10 kHz

I2CBRST: I²C Bus Reset Register

x'007E48'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	BRST
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
R/W:	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R/W

BRST: Bus reset

When a serious bus error occurs, this bit can be set to 0, forcing the clock line low and resetting the I²C bus. This function works in all I²C modes. After a forced reset, the microcontroller is in slave receiver mode. This reset does not change the contents of the I2CMYAD and I2CCLK registers.

- 0: Force bus to reset
- 1: Steady state

I2CBSTS: I²C Bus Status Register

x'007E4A'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	—	—	—	—	—	—	—	SDAS	SCLS
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

I2CBSTS is a two-bit, read-only register that monitors the status of the I²C bus.

SDAS: SDA data line status

This bit monitors the state of the I²C data line, SDA.

SCLS: SCL clock line status

This bit monitors the state of the I²C clock line, SCL.

14 H Counter

14.1 Description

The MN102H75K/85K contains two H counter circuits that can be used to count the HSYNC signal. Each H counter consists of a 10-bit counter and 10-bit register.

14.2 Block Diagram

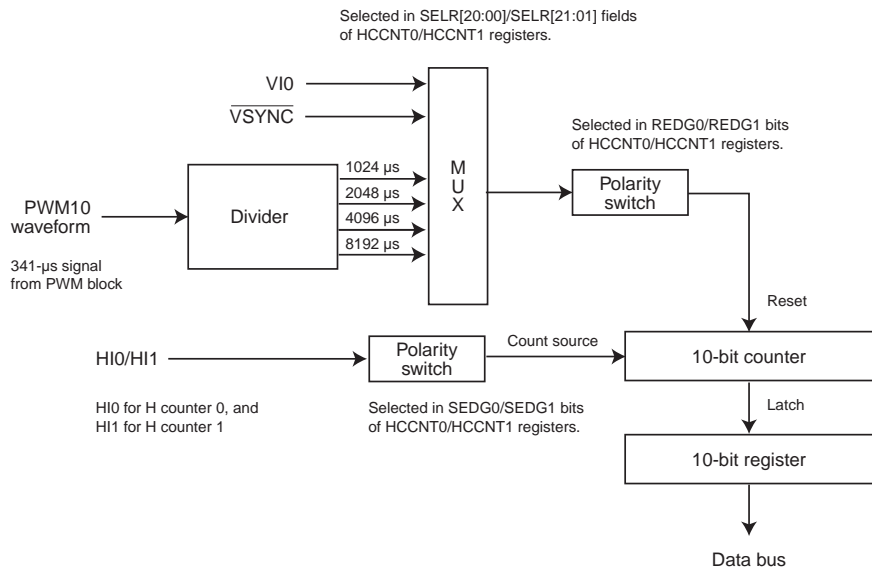
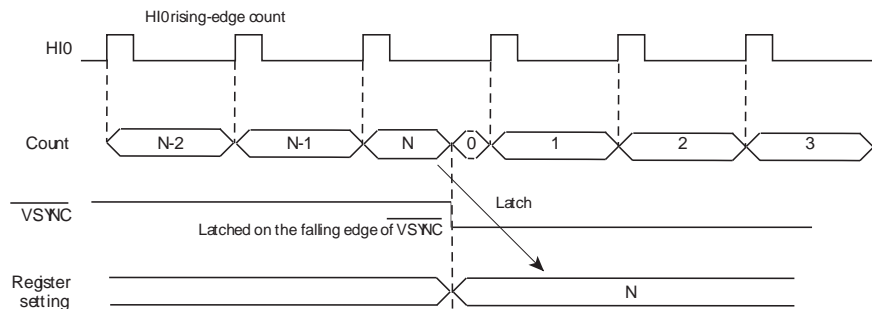


Figure 14-1 H Counter Block Diagram

14.3 H Counter Operation

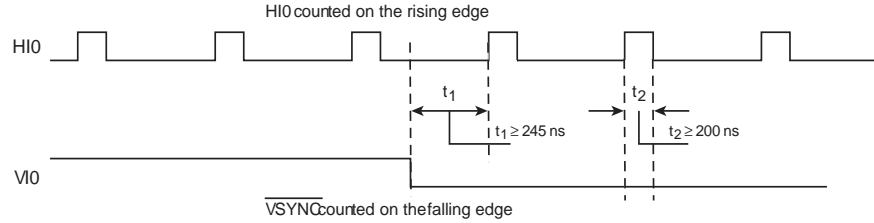
Figure 14-2 provides a schematic diagram of an example counter operation.



Note: In this example, HI0 is active high and $\overline{\text{VSYNC}}$ is active low.

Figure 14-2 H Counter Operation Example

Figure 14-3 shows the input timing for the count source and reset signals. Never input a count source signal in less than 245 ns (t_1) after the reset signal input. Otherwise, the signal may be counted as part of the previous count cycle.



Note: In this example, H10 is active high and $\overline{\text{VSYNC}}$ is active low.

Figure 14-3 H Counter Input Signal Timing

The 10-bit counter counts the HSYNC signal from 0 to x'3FF', and the 10-bit register stores the count. The H counter uses the four pins shown in table 14-1.

Table 14-1 H Counter Pins

Pin Name	Pin No.		Description	Alternative Functions
	H75K	H85K		
H10	17	45	Count source pin	P43/TM5IOB
H11	16	46	Count source pin	P44/TM5IC
V10	6	53	Count reset pin	P52/IRQ4
$\overline{\text{VSYNC}}$	2	55	Count reset pin	P54/IRQ5

To use the H counter, set the port 4 and 5 output control registers (P4OUT and P5OUT) to 0 and set the H counter pins to input.

- To use H10, set the P4DIR3 bit (x'00FFE4') to 0.
- To use H11, set the P4DIR4 bit (x'00FFE4') to 0.
- To use V10, set the P5DIR2 bit (x'00FFE5') to 0.
- To use $\overline{\text{VSYNC}}$, set the P5DIR4 bit (x'00FFE5') to 0.

The H counter counts the HSYNC signal for the interval set in the HCCNT0 (x'007EB0') or HCCNT1 (x'007EB2') register, latches the count value in the 10-bit register, then clears the counter. HCCNT0 and HCCNT1 provide six interval settings:

- 1024- μ s fixed interval obtained by dividing the system clock (12 MHz)
- 2048- μ s fixed interval obtained by dividing the system clock (12 MHz)
- 4098- μ s fixed interval obtained by dividing the system clock (12 MHz)
- 8096- μ s fixed interval obtained by dividing the system clock (12 MHz)
- Interval from active-edge to active-edge input of VI0 pin
- Interval from active-edge to active-edge input of VSYNC pin

If your application uses one of the fixed clocks based on divided PWM output (1024, 2048, 4098, or 8096 μ s), you must also set up the PWM circuit. (See section 10, "Pulse Width Modulator," on page 249.)



To use the H counter, you must always set the HCNTOFF bit of the PCNT0 register to 0. To use the PWM function, always set the PWMOFF bit of the PCNT2 register (x'00FF92') to 0.

When the count overflows (is greater than x'3FF'), the counter stops counting and stores the value x'3FF' in the 10-bit register. At any time, the CPU can obtain the count value stored in the latch by reading the HCD0 (x'007EB4') or HCD1 (x'007EB6') register.

To enable or disable the H counter function, set the HCNTOFF bit of the PCNT0 register (x'00FF90'; see page 286). Disabling this circuit when it is unused can reduce power consumption.

Because the H counter uses the system clock, it does not operate in STOP mode.

14.4 H Counter Control Registers

All registers in H Counter block cannot be written by byte (by word only). Read by byte is possible.

HCCNT0: H Counter Control Register 0

x'007EB0'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	—	—	—	—	SED G0	RED G0	SELR 20	SELR 10	SELR 00
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W

SEDG0: Polarity select for count source signal (HI0)

- 0: Active low
- 1: Active high

SEDG0: Polarity select for reset signal

- 0: Active low
- 1: Active high

SELR20:00]: Reset signal select

- 000: 1024 μ s
- 001: 2048 μ s
- 010: 4096 μ s
- 011: 8192 μ s
- 100: VIO
- 101: $\overline{\text{VSYNC}}$

All other settings default to 1024 μ s.

HCCNT1: H Counter Control Register 1

x'007EB2'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	—	—	—	—	SED G1	RED G1	SELR 21	SELR 11	SELR 01
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R	R	R	R	R/W	R/W	R/W	R/W	R/W

SEDG1: Polarity select for count source signal (HI1)

- 0: Active low
- 1: Active high

SEDG1: Polarity select for reset signal

- 0: Active low
- 1: Active high

SELR[21:01]: Reset signal select

- 000: 1024 μ s
- 001: 2048 μ s
- 010: 4096 μ s
- 011: 8192 μ s
- 100: VIO
- 101: $\overline{\text{VSYNC}}$

All other settings default to 1024 μ s.

H Counter

H Counter Control Registers

HCD0: H Counter Data Register 0

x'007EB4'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	HCD 90	HCD 80	HCD 70	HCD 60	HCD 50	HCD 40	HCD 30	HCD 20	HCD 10	HCD 00
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

HCD[90:00]: Count from HI0 source signal

This field stores the HI0 clock source count. It becomes x'3FF' on overflow.

HCD1: H Counter Data Register 1

x'007EB6'

Bit:	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	HCD 91	HCD 81	HCD 71	HCD 61	HCD 51	HCD 41	HCD 31	HCD 21	HCD 11	HCD 01
Reset:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
R/W:	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R

HCD[91:01]: Count from HI1 source signal

This field stores the HI1 clock source count. It becomes x'3FF' on overflow.

Appendix A Register Map

Table A-1 Register Map: x'007E00' to x'007FFF' (Registers in this area cannot be written by byte only by word.)

20 MSBs	4 LSBs														Description		
	F	E	D	C	B	A	9	8	7	6	5	4	3	2		1	0
007E00	CRI4 FQW	CRI3 FQW	CRI2 FQW	CRI1 FQW	CAPDA		ACQ1		VBI IRQ	HNUM	SLSF	SLHD	MAX	MIN	SL CNT	FC	VBI 1 registers
007E10	FCPNUM		STAP		DATAE		DATAS		CRI2E		CRI2S		CRI1E		CRI1S		
007E20	CRI4 FW W	CRI3 FW W	CRI2 FW W	CRI1 FW W	CAPDAW		ACQ1W		VBI IRQW	HNUM W	SLSF W	SLHD W	MAX W	MINW	SL CNT W	FCW	VBI 2 registers
007E30	FCPNUMW		STAPW		DATAEW		DATASW		CRI2EW		CRI2SW		CRI1EW		CRI1SW		
007E40					I2CBSTS	I2CBRST	I2CCLK	I2CMYAD	I2CDREC	I2CDTRM	I ² C interface registers						
007E50																	
007E60																	
007E70				PWM6	PWM5	PWM4	PWM3	PWM2	PWM1	PWM0	PWM registers						
007E80																	
007E90																	
007EA0				RML D	RMT R	RMS R	RMC S	RMT C	RMIR	RMIS	Remote signal receiver registers						
007EB0								HCD1	HCD0	HCO UNT1	HCO UNT0	H counter registers					
007EC0	HSEP1	CLAMP	SPLV	BPLV	SYNC MIN	BPPST	SCMING	FQSEL	NFSEL	Sync separator 1 registers							
007ED0		CLPCND 1	HVCOND	VCNT	HDISTW	HLOCKLV	FIELD	HSEP2									
007EE0	HSEP1W	CLAMPW	SPLVW	BPLV W	SYNC MINW	BPPSTW	SCMING W	FQSELW	NFSELW	Sync separator 2 registers							
007EF0		CLPCND1W	HVCONDW	VCNTW	HDIS- TWW	HLOCK- LVW	FIELDW	HSEP2W									
007F00	EVOD	HCOVNT	OSD3	OSD2	OSD1	RAMEND	GROMEN D	CROMEN D	OSD control registers								
007F10		CIVP	CIHP	GIVP	GIHP	SVP	SHP	STC0									
007F20	STC3	STC2	STC1	SHTC	HSHT1	HSHT0	VSHT1	VSHT0									
007F30																	
007F40	TESTA (test register)	SBFNUM (test register)	VBI 1 test registers														
007F50																	
007F60	TESTAW (test register)	SBFNUMW (test register)	VBI 2 test registers														
007F70																	
007F80	CPT7	CPT6	CPT5	CPT4	CPT3	CPT2	CPT1	CPT0	OSD text color palette								
007F90	CPTF	CPT E	CPTD	CPTC	CPTB	CPTA	CPT9	CPT8									
007FA0						WBSHD	BBSHD	FRAME	COLB	OSD text effects registers							
007FB0																	
007FC0	GPT17	GPT16	GPT15	GPT14	GPT13	GPT12	GPT11	GPT10	OSD graphics color palette 1								
007FD0	GPT1F	GPT1E	GPT1D	GPT1C	GPT1B	GPT1A	GPT19	GPT18									
007FE0	GPT27	GPT26	GPT25	GPT24	GPT23	GPT22	GPT21	GPT20	OSD graphics color palette 2								
007FF0	GPT2F	GPT2E	GPT2D	GPT2C	GPT2B	GPT2A	GPT29	GPT28									

Register Map

Table A-2 Register Map: x'00FC00' to x'00FDFF'

20 MSBs	4 LSBs															Description		
	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1		0	
00FC00	IAGR														CPUM		Special function registers	
00FC10																		
00FC20																		
00FC30																		
00FC40					IQ1 ICH	IQ1 ICL	IQ0 ICH	IQ0 ICL		EI ICR		PI ICR		WD ICR			Interrupt controller registers	
00FC50					IQ5 ICH	IQ5 ICL	IQ4 ICH	IQ4 ICL					IQ3 ICH	IQ3 ICL	IQ2 ICH	IQ2 ICL		
00FC60	VBIW ICH	VBIW ICL	TM5UD ICH	TM5UD ICL	TM5CA ICH	TM5CA ICL	TM5CB ICH	TM5CB ICL	VBI ICH	VBI ICL	TM4UD ICH	TM4UD ICL	TM4CA ICH	TM4CA ICL	TM4CB ICH	TM4CB ICL		
00FC70	ADMO ICH	ADMO ICL	ADM1 ICH	ADM1 ICL	ADM2 ICH	ADM2 ICL	ADM3 ICH	ADM3 ICL	RMC ICH	RMC ICL	TM0UD ICH	TM0UD ICL	TM1UD ICH	TM1UD ICL	TM2UD ICH	TM2UD ICL		
00FC80			TM3UD ICH	TM3UD ICL	VBIV ICH	VBIV ICL	VBIV ICH	VBIV ICL			SCR0 ICH	SCR0 ICL	SCT0 ICH	SCT0 ICL	AN ICH	AN ICL		
00FC90			I2C ICH	I2C ICL	SCR1 ICH	SCR1 ICL	SCT1 ICH	SCT1 ICL					OSDC ICH	OSDC ICL	OSD G ICH	OSD G ICL		
00FCA0																		
00FCB0																		
00FCC0																		
00FCD0																		
00FCE0																		
00FCF0	ROMCTSTH (test register)		ROMCTSTL (test register)				EXTMD								ROMCEN		ROM correction registers and external interrupt mode register	
00FD00		AMC HIH3	AMCHIL3			AMC HIH2	AMCHIL2			AMC HIH1	AMCHIL1			AMC HIH0	AMCHIL0		ROM correction address match registers	
00FD10		AMC HIH7	AMCHIL7			AMC HIH6	AMCHIL6			AMC HIH5	AMCHIL5			AMC HIH4	AMCHIL4			
00FD20		AMC HIH8	AMCHIL8			AMC HIH9	AMCHIL9			AMC HIH8	AMCHIL8			AMC HIH7	AMCHIL7			
00FD30		AMC HIHF	AMCHILF			AMC HIHE	AMCHILE			AMC HIHD	AMCHILD			AMC HIHC	AMCHILC			
00FD40				CH DAT3				CH DAT2				CH DAT1				CH DAT0	ROM correction data registers	
00FD50				CH DAT7				CH DAT6				CH DAT5				CH DAT4		
00FD60				CH DATB				CH DATA				CH DAT9				CH DAT8		
00FD70				CH DATF				CH DATE				CH DATD				CH DATC		
00FD80					SC1 STR	SC1 TRB	SC1CTR					SCTST (test register)	SC0 STR	SC0 TRB	SC0CTR		Serial interface registers	
00FD90																		
00FDA0																		
00FDB0																		
00FDC0																		
00FDD0																		
00FDE0																		
00FDF0																		

Table A-3 Register Map: x'00FE00' to x'00FFFF'

20 MSBs	4 LSBs														Description		
	F	E	D	C	B	A	9	8	7	6	5	4	3	2		1	0
00FE00													TM3 BC	TM2 BC	TM1 BC	TM0 BC	8-bit timer registers
00FE10													TM3 BR	TM2 BR	TM1 BR	TM0 BR	
00FE20													TM3 MD	TM2 MD	TM1 MD	TM0 MD	
00FE30															TM8TST (test register)		
00FE40																	
00FE50																	
00FE60																	
00FE70																	
00FE80			TM4TST (test register)	(TM4CBX)			TM4CB		(TM4CAX)		TM4CA		TM4BC		TM4MD		16-bit timer 4 registers
00FE90			TM5TST (test register)	(TM5CBX)			TM5CB		(TM5CAX)		TM5CA		TM5BC		TM5MD		16-bit timer 5 registers
00FEA0																	
00FEB0																	
00FEC0																	
00FED0																	
00FEE0																	
00FEF0																	
00FF00	AN3BUF	AN2BUF	AN1BUF	AN0BUF								ANTST (test register)			ANCTR		ADC registers
00FF10	AN11BUF	AN10BUF	AN9BUF	AN8BUF	AN7BUF	AN6BUF	AN5BUF	AN4BUF									
00FF20																	
00FF30																	
00FF40																	
00FF50																	
00FF60																	
00FF70							FBEWER		FAR-EGEX		FAREG		FDREG		FCREG		Flash memory write control registers
00FF80													MEMMD1		EXWMD		External memory wait control registers
00FF90													PCNT2		PCNT0		I/O port control registers
00FFA0															PCNT1 (test register)		
00FFB0					P8 PUP		P7 PUP		P6 PUP	P5 PUP	P4 PUP	P3 PUP	P2 PUP	P1 PUP	P0 PUP		
00FFC0					P8 OUT		P7 OUT		P6 OUT	P5 OUT	P4 OUT	P3 OUT	P2 OUT	P1 OUT	P0 OUT		
00FFD0					P8 IN		P7 IN		P6 IN	P5 IN	P4 IN	P3 IN	P2 IN	P1 IN	P0 IN		
00FFE0					P8 DIR		P7 DIR		P6 DIR	P5 DIR	P4 DIR	P3 DIR	P2 DIR	P1 DIR	P0 DIR		
00FFF0				P6 MD	P5 MD		P4 MD		P3 MD	P2MD		P1MD			P0 MD		

Register Map

Appendix B MN102HF75K Flash EEPROM Version

B.1 Description

The MN102HF75K and MN102HF85K are electrically programmable, 256-kilobyte flash ROM versions of the MN102H75K and MN102H85K. They are programmed in one of two modes:

- PROM writer mode, which uses a dedicated adaptor socket and writer. In this mode, the user program can occupy the entire 256-kilobyte ROM space.
- Onboard serial programming mode, which the CPU controls. This mode requires an 8-kilobyte ROM area to hold a serial writer program.

In onboard serial programming mode, the 256-kilobyte flash memory is divided into three main areas:

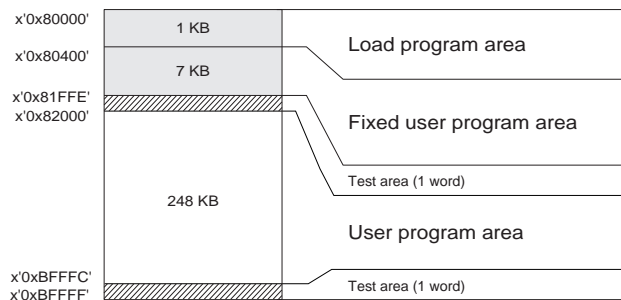
- Load program area (1 kilobyte: x'0x80000 to x'0x803FF) This area stores the load program for the serial writer. It is overwritten in PROM writer mode.
- Fixed user program area (7 kilobytes: x'0x80400 to x'0x81FFD) This area stores a user program that is write-protected in serial programming mode. It is overwritten in PROM writer mode.
- User program area (248 kilobytes: x'0x82000 to x'0xBFFFD) This area stores the user program. It is overwritten in both programming modes.

The two words x'0x81FFE' and x'0xBFFFE' are test areas. Do not use these two words in your programs.

Table B-1 summarizes the programmable areas for the modes. Normal operation is guaranteed with up to ten programmings.

Table B-1 Programmable Areas in Each Programming Mode

Programming Mode	Programmable Area
PROM writer programming mode	Entire memory space (256 KB)
Onboard serial programming mode	User program area (248 KB)



Note: The shaded regions are write-protected in this mode. Cross-hatched regions are test areas (not for use in user programs).

Figure B-1 Memory Map for Onboard Serial Programming Mode



A cycle of erasing to programing is counted as one time no matter how many blocks are rewritten. Even when the multi-block is rewritten separately or the same block is rewritten, each block rewrite is counted. For example, rewriting Block1, Block2 and Block3 respectively is counted as 3 times. Therefore, to program efficiently, rewrite all blocks in the lump.

B.2 Benefits

Because you can maintain and upgrade the program in the MN102HF75K/85K up to and immediately following product release, this version of the device shortens time-to-market by as much as one month. This device is ideal for applications in quickly changing markets, since it allows you to revise the microcontroller program in an existing product.

B.3 Using the PROM Writer Mode

In this mode, the MN102HF75K allows a PROM writer to program the internal flash memory as if it was a standalone memory chip. The microcontroller is inserted into a dedicated adaptor socket, which connects to DATA-I/O's LabSite PROM writer. When the microcontroller connects to the adaptor socket, it automatically enters PROM writer mode. The adaptor socket ties the microcontroller pin states to PROM writer mode, and programming occurs without any reference to the microcontroller pin states.

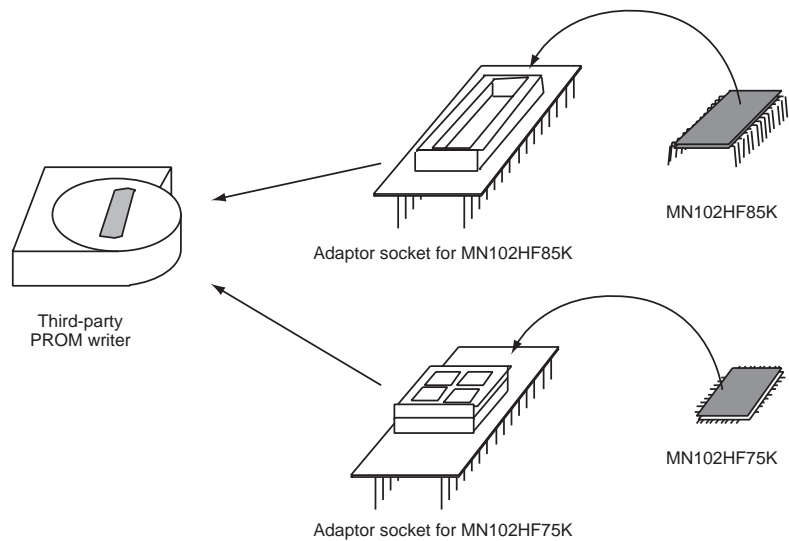


Figure B-2 PROM Writer Hardware Setup

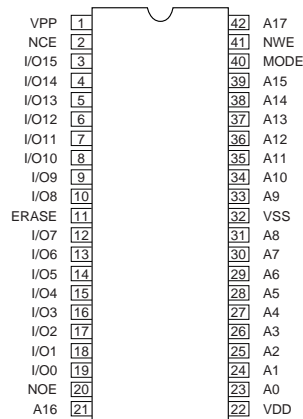
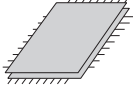
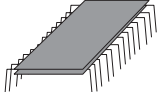
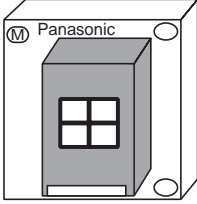
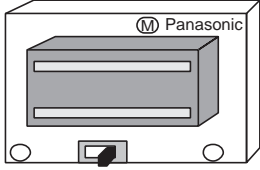
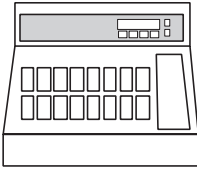
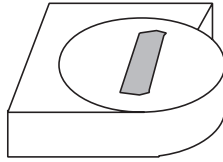


Figure B-3 Pin Configuration for Socket Adaptor

Table B-2 PROM Writer Hardware

Hardware Part	Device	
	MN102HF75KBF	MN102HF85KDP
Package (external view) Installed in ↓	 84-pin QFP	 64-pin SDIP
Adaptor Installed in ↓	 Ordering information: Part no.: FLS84F18--102HF57 OEM: Matsushita Electric Industrial Co., Ltd.	 Set this switch to the right. Ordering information: Part no.: FLS64SD-102HF51 OEM: Matsushita Electric Industrial Co., Ltd.
Third-party PROM writer	Gang writer  Model: 1930 OEM: Minato Electronics 4105 Minami Yamada-cho Tsuzuki-ku Yokohama, Japan Tel: 045-591-5605	Single-unit writer  Model: LabSite DIP 48-1 OEM: DATA-I/O Osaki CN Building, 2F 5-10-10 Osaki, Shinagawa-ku Tokyo, Japan Tel: 03-3779-2040

Check the following web page of our microcomputer division for the writer matching information.

<http://www.mec.panasonic.co.jp/sc/division/micom>

B.4 Using the Onboard Serial Programming Mode

The serial programming mode is primarily used to program the flash ROM in devices that are already installed on a PCB board. Panasonic provides the dedicated hardware and software for this mode. This section describes the microcontroller hardware, system configuration, software register map, and protocol for this type of programming operation.

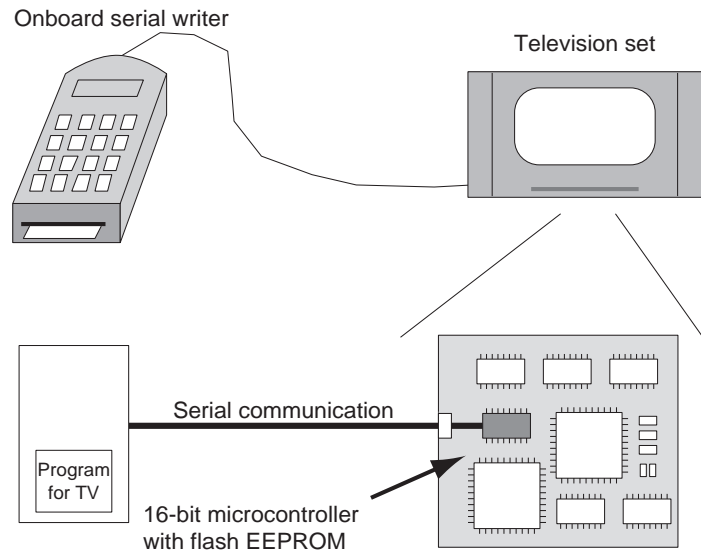


Figure B-4 Serial Writer Programming Configuration

■ **Hardware requirements:**

- ◆ Onboard serial writer (YDC model AF200 (provisional))
- ◆ Add-on circuit for target board
- ◆ Flash programming connectors or pins for target board

■ **Software requirements:**

- ◆ Serial writer load program, installed in first kilobyte of MN102HF75K/85K EEPROM
- ◆ Programming algorithm for operating the onboard serial writer

B.4.1 Configuring the System for Onboard Serial Programming

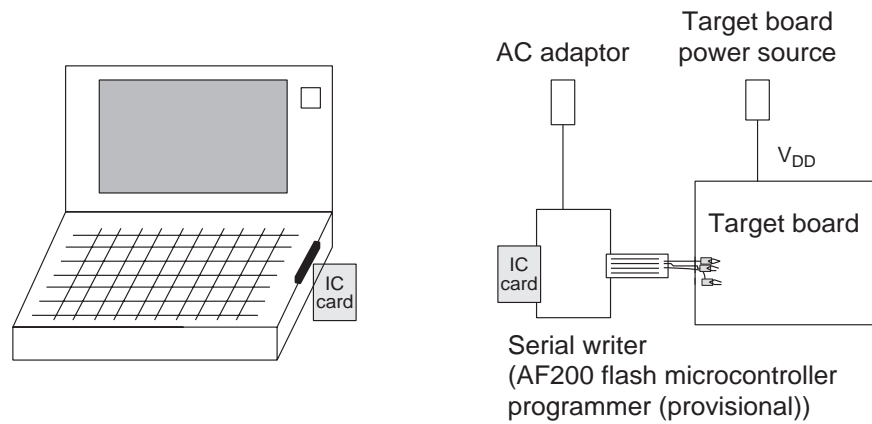


Figure B-5 Serial Writer Hardware Setup

The workstation containing the program data sends the program to the serial writer through an IC card. Through serial communication, the serial writer programs the flash memory inside the microcontroller on the target board. You must supply an external V_{DD} source to the target board. The serial writer supplies the V_{PP} source.

You must provide the personal computer that holds the IC card. To order the serial writer, contact:

(Provisional)
Yokogawa Digital Computer Co., Ltd.
Microcontroller System Joint Headquarters, Equipment Business Center
Keio Fuchu 1-chome Building, 7F
1-9 Fuchu-cho, Fuchu-shi
Tokyo, Japan

Model: AF200 flash microcontroller programmer

B.4.2 Circuit Requirements for the Target Board

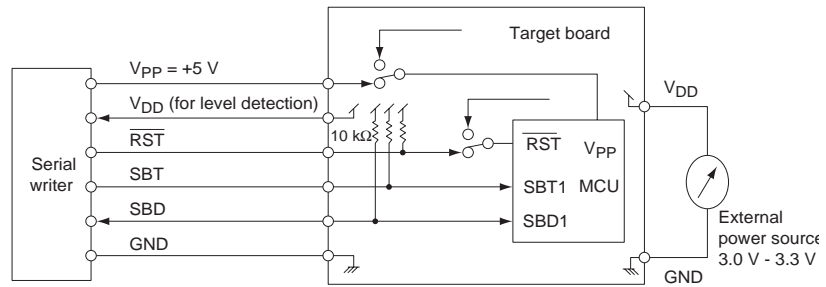


Figure B-6 Target Board–Serial Writer Connection

Table B-3 Pin Descriptions for Target Board–Serial Writer Connection

Pin Name	Description
V _{PP}	5-V power supply
V _{DD}	3.0–3.3 external power supply
V _{DD} (for level detection)	V _{DD} level detection pin for target board
\overline{RST}	Reset
SBT	Serial interface clock supply
SBD	Serial interface data supply
GND	Ground

Note: 1. During normal microcontroller operation, V_{PP} should always be equal to V_{DD} (3.3 ±0.3 V). Apply 5 V to the V_{PP} supply only when programming the flash memory.

- During programming, the serial writer supplies V_{PP} to the microcontroller. *Install a switch on the target board to toggle between V_{PP} supplied by the serial writer and V_{PP} for normal operation.*
- In serial programming a large-scale of current flows through V_{pp}. See to it that 5V is supplied to V_{pp} by checking the connection and reducing the impedance of switches.
- You must supply a V_{DD} source of 3.0 V – 3.3 V externally.
- V_{DD} (for level detection) informs the serial writer of the actual V_{DD} level of the target board. If the V_{DD} level is too low, the serial writer generates an error message.
- Connect pullup resistors on the target board to the \overline{RST} , SBT, and SBD pins. Use a pullup resistor value of 10 kΩ ±10%. *Install a switch on the target board to toggle between \overline{RST} for serial programming and \overline{RST} for normal operation. Alternatively, install a wired-OR connection. For a wired-OR connection, disable \overline{RST} for normal operation during serial programming.*
- \overline{RST} , SBT, and SBD are output from the serial writer through an open connection.

B.4.3 Microcontroller Hardware Used in Onboard Serial Programming

B.4.3.1 Serial Writer Interface Description

The microcontroller contains the following interface hardware for serial programming of the flash ROM:

- One 8-bit serial interface (use serial interface 1):
 - ◆ Data transmission and reception synchronize with external clock
 - ◆ Transmission bit LSB first
 - ◆ Maximum clock speed ≥ 10 MHz
 - ◆ Positive I/O logic
- Two I/O pins:
 - ◆ SBT1 and SBD1 pins (with alternate I/O port functions)

B.4.3.2 Serial Writer Interface Block Diagram

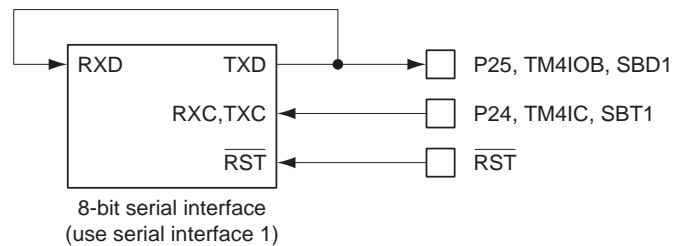


Figure B-7 Serial Writer Interface Block Diagram

When programming the memory, you need not be aware of these microcontroller hardware connections. However, it is vital that you take these connections into account when designing your target board, so that the serial writer can program the device correctly, and so that SBD1 and SBT1 are dedicated pins for the serial writer, preventing other user circuits from communicating with the device.

B.4.4 Microcontroller Memory Map Used During Onboard Serial Programming

B.4.4.1 Flash ROM Address Space

Table B-4 Flash ROM Address Space in Serial Programming Mode

Address	Size	Description
x'0x80000' x'0x803FF'	1 KB	Serial writer load program area
x'0x80400' x'0x81FFF'	7 KB	Fixed user program area
x'0x82000' x'0x82007'	8 bytes	Security code
x'0x82008' x'0x8200F'	8 bytes	Reserved area
x'0x82010' x'0x82017'	8 bytes	Branch instruction to reset service routine (Ex., JMP 82100)
x'0x82018' x'0x8201F'	8 bytes	Branch instruction to interrupt service routine (Ex., JMP 82200)
x'0x82020' x'0xBFFFF'	248 KB	User program area

■ **Serial writer load program area**

This kilobyte of ROM, starting at address x'0x80000', holds the load program for the serial writer. This area is write-protected in the hardware. Panasonic provides the load program.

■ **Fixed user program area**

These 7 kilobytes of ROM starting at address x'0x80400', hold the fixed user program. This area is write-protected in the hardware. (You can program this area with a parallel writer.)

■ **Security code**

This byte holds the password for the serial writer. Enter an 8-character ASCII code in this space.

■ **Reserved area**

Do not write to this area.

■ **Branch instruction to reset service routine**

Normally, reset servicing starts at address x'0x80000', but the soft branch instruction in the serial writer load program branches to x'0x82010'. This address must hold a JMP instruction pointing to the real start address for the reset service routine.

■ **Branch instruction to interrupt service routine**

Normally, interrupt servicing starts at address x'0x80008', but the soft branch instruction in the serial writer load program branches to x'0x82018'. This address must hold a JMP instruction pointing to the real start address for the interrupt service routine.

■ **User program area**

This area stores the user program.

B.4.4.2 RAM Address Space

Table B-5 RAM Address Space in Serial Programming Mode

Address	Size	Description
x'0x8000' x'0x8BFF'	3 KB	Serial writer work area
x'0x8C00' x'0x8FFF'	1 KB	Reserved area

■ **Serial writer work area**

The 3 kilobytes of RAM area starting at address x'0x08000' are used for the serial writer work area. The load program downloads programs to this area that it needs to operate the serial writer and program the EEPROM. No other memory area can be used for this purpose. You do not need to know about RAM allocation to program the EEPROM.

■ **Reserved area**

Do not write to this area.

B.4.5 Microcontroller Clock on the Target Board

For the clock supply to the microcontroller on the target board, use the existing target board clock. The OSC oscillator clock for the microcontroller is 4 MHz. Using the internal PLL circuit, the microcontroller switches to NORMAL mode and operates with a 12-MHz system clock. Table B-6 shows the clock frequencies for the microcontroller during serial programming.

Table B-6 Microcontroller Clock Frequencies during Serial Programming

Oscillator Clock Frequency	Internal System Clock Frequency
4 MHz	12 MHz

B.4.6 Setting Up the Onboard Serial Programming Mode

To enter serial programming mode, the microcontroller must be in write mode.

This section describes the pin setup for the serial writer interface.

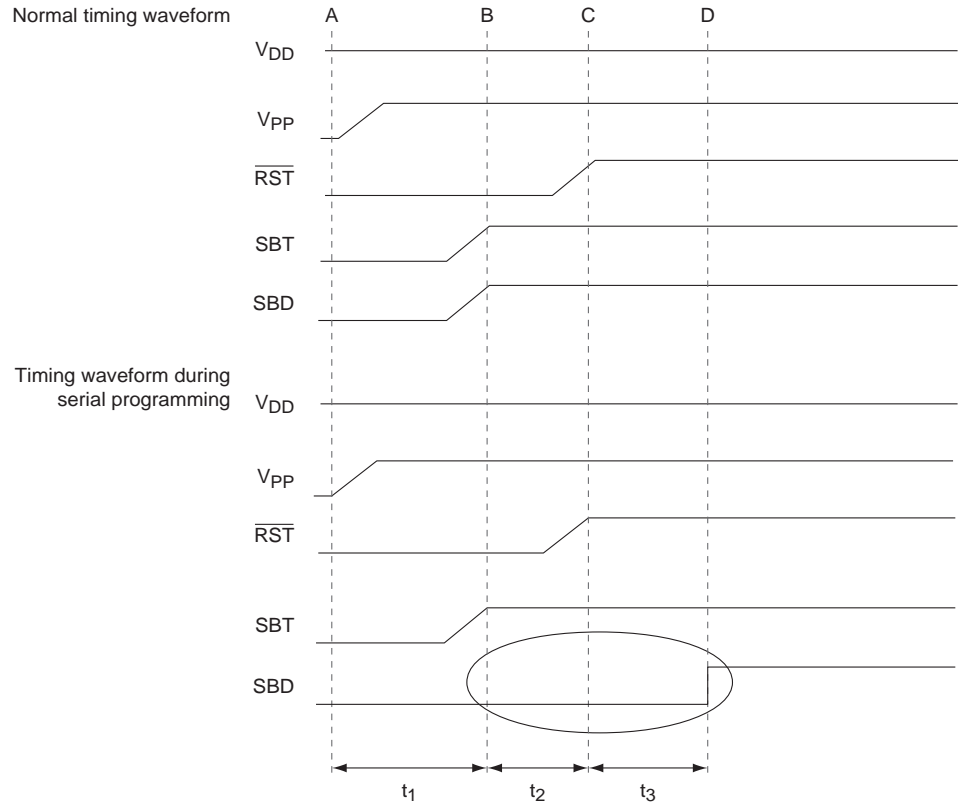


Figure B-8 Timing for the Serial Writer Interface

■ **To set up the serial writer interface:**

1. Turn on the external V_{DD} supply.
2. At time A in figure B-8, turn V_{PP} on. At this point, output $\overline{\text{RST}} = \text{SBD} = \text{low}$.
3. Through the serial writer, drive the $\overline{\text{RST}}$ pin from time B in figure B-8, when SBT goes high on microcontroller power-up, for t₂ cycles. The microcontroller initializes.
4. Through the serial writer, drive the SBD pin low from time C in figure B-8, when RST goes high on microcontroller power-up, for t₃ cycles. This tells the microcontroller that it is connected to the serial writer.
5. Make t₃ long enough to allow the microcontroller oscillator to stabilize.

■ **Start routine for the load program**

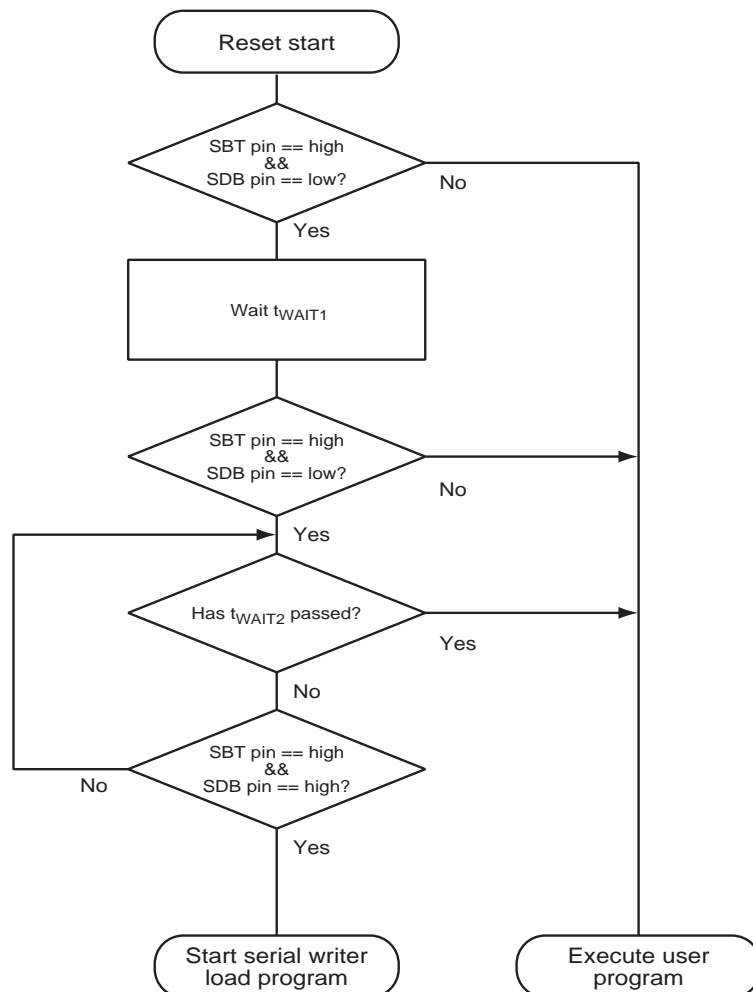


Figure B-9 Load Program Start Flow

Conditions:

1. After the load program initiates a reset start, SBD must be low and SBT high.
2. After the program waits $t_{WAIT1} = 10$ milliseconds, SBD must still be low and SBT high.
3. Within $t_{WAIT2} = 100$ milliseconds, both SBD and SBT must be high.

If any of these conditions is not met, control returns to the user program.

B.4.7 Branching to the User Program

B.4.7.1 Branching to the Reset Start Routine

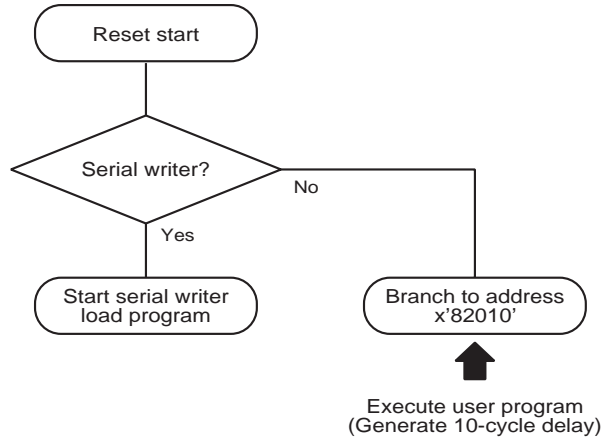


Figure B-10 Flow of Branch to Reset Start Routine

When the reset starts, the serial writer load program initializes only if SBD is low. Otherwise, the program branches to the user program at address x'0x82010'.

B.4.7.2 Branching to the Interrupt Start Routine

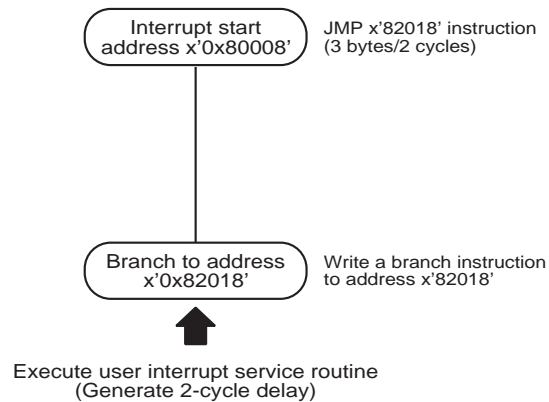


Figure B-11 Flow of Branch to Interrupt Start Routine

In the interrupt start address, place a simple branch instruction pointing to address x'0x82018'.

B.5 Reprogramming Flow

Figure B-12 shows the flow for reprogramming (erasing and programming) the flash memory.

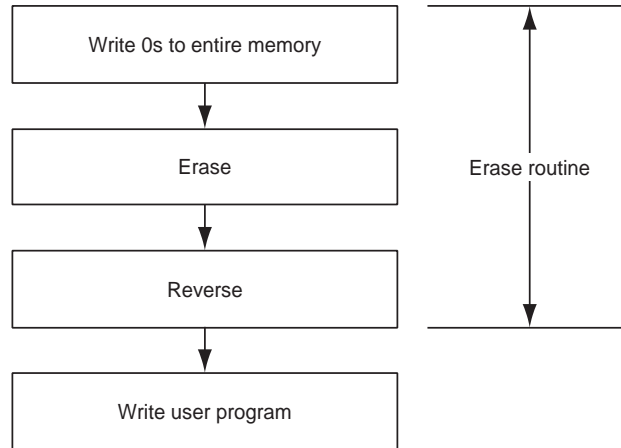


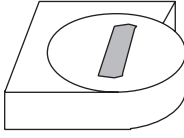

Figure B-12 EEPROM Programming Flow

As the figure shows, the write occurs after the memory is completely erased. The erase routine consists of three steps, first writing all zeros to the entire memory space, next erasing the memory, and finally reversing.

B.6 Programming Times

Table B-7 shows the time required for PROM and serial programming and reprogramming (erasing and programming).

Table B-7 Programming Times for PROM and Serial Writers

Writer	Programming Time (User Program Only)	Reprogramming Time
DATA-I/O LabSite DIP48-1 	TBA	TBA
YDC AF200 (provisional) 	—	3.5–4 minutes

Note: Times indicated are minimum time requirements.



Always program after erasing is completed. Erasing is sometimes not done finely, even though PROM-writer or onboard serial writer shows “PASS” in blank check. And in that situation the data programmed successfully may be incorrect. Rewriting the data to the address where data has already set is forbidden.

MN102H75K/F75K/85K/F85K LSI User's Manual Description Record of Changes (Ver.1.0 to 1.1)

page	Line	defini- tion	Description of Changes	
			Former version	New version
Cover	Pub number	C	22385-010E	22385-011E
Colophon		C	September, 2001 1st Edition	October, 2001 1st Edition 1st Printing
Sales office		C		Latest version

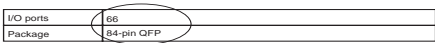

<Definition>

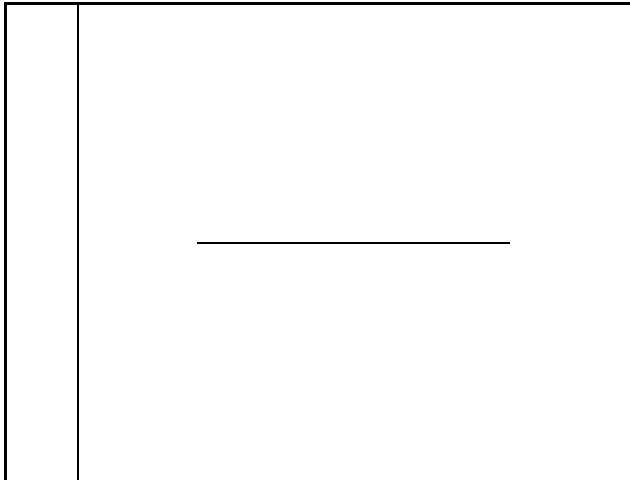
A: add

D: delete

C: modify, change

MN102H75K/F75K/85K/F85K LSI User's Manual
Modified Points From MN102H75K/F75K To MN102H75K/F75K/85K/F85K

page	Before Modify	page	After Modify
P16	<p>This manual is intended for assembly-language programming engineers. It describes the internal configuration and hardware functions of the <u>MN102H75K</u> microcontrollers.</p> <p>Using This Manual</p> <p>The chapters in this manual deal with the internal blocks of the <u>MN102H75K</u>. Chapters 1 to 5 provide an overview of the <u>MN102H75K</u>'s general specifications, interrupts, power modes, timers, and serial connections. Chapters 6 to 10 describe the on-screen display and other specialized functions available with the <u>MN102H75K</u>. Chapter 11 provides the I/O port specifications, chapter 12 describes the ROM correction feature, chapter 13 describes the I²C interface, and chapter 14 describes the H scan line counter. Appendix A provides a register map, and Appendix B describes the flash EEPROM version.</p>	P16	<p>This manual is intended for assembly-language programming engineers. It describes the internal configuration and hardware functions of the <u>MN102H75K</u> and <u>MN102H85K</u> microcontrollers. <u>Except when discussing differing specifications, this manual refers to the two microcontrollers as a single device : MN102H75K/85K.</u></p> <p>Using This Manual</p> <p>The chapters in this manual deal with the internal blocks of the <u>MN102H75K/85K</u>. Chapters 1 to 5 provide an overview of the <u>MN102H75K/85K</u>'s general specifications, interrupts, power modes, timers, and serial connections. Chapters 6 to 10 describe the on-screen display and other specialized functions available with the <u>MN102H75K/85K</u>. Chapter 11 provides the I/O port specifications, chapter 12 describes the ROM correction feature, chapter 13 describes the I²C interface, and chapter 14 describes the H scan line counter. Appendix A provides a register map, and Appendix B describes the flash EEPROM version.</p>
P17	<ul style="list-style-type: none"> ■ <u>MN10200 Series Linear Addressing High-Speed Version LSI User Manual</u> (Describes the core hardware.) ■ <u>MN10200 Series Linear Addressing High-Speed Version Instruction Manual</u> (Describes the instruction set.) ■ <u>MN10200 Series Linear Addressing High-Speed Version C Compiler User Manual: Usage Guide</u> (Describes the installation, commands, and options for the C compiler.) ■ <u>MN10200 Series Linear Addressing High-Speed Version C Compiler User Manual: Language Description</u> (Describes the syntax for the C compiler.) ■ <u>MN10200 Series Linear Addressing High-Speed Version C Compiler User Manual: Library Reference</u> (Describes the standard libraries for the C compiler.) ■ <u>MN10200 Series Linear Addressing High-Speed Version Cross-Assembler User Manual</u> (Describes the assembler syntax and notation.) ■ <u>MN10200 Series Linear Addressing Version C Source Code Debugger User Manual</u> (Describes the use of the C source code debugger.) ■ <u>MN10200 Series Linear Addressing Version PanaX Series Installation Manual</u> (Describes the installation of the C compiler, cross-assembler, and C source code debugger and the procedures for using the in-circuit emulator.) <p>Questions and Comments</p> <p>We welcome your questions, comments, and suggestions. Please contact the semiconductor design center closest to you. See the last page of this manual for a list of addresses and telephone numbers. You can also find contact and product information on the World Wide Web at: http://www.psd.com/</p>	P17	<ul style="list-style-type: none"> ■ <u>MN102H Series LSI User Manual</u> (Describes the core hardware.) ■ <u>MN102H Series Instruction Manual</u> (Describes the instruction set.) ■ <u>MN102H Series C Compiler User Manual: Usage Guide</u> (Describes the installation, commands, and options for the C compiler.) ■ <u>MN102H Series C Compiler User Manual: Language Description</u> (Describes the syntax for the C compiler.) ■ <u>MN102H Series C Compiler User Manual: Library Reference</u> (Describes the standard libraries for the C compiler.) ■ <u>MN102H Series Cross-Assembler User Manual</u> (Describes the assembler syntax and notation.) ■ <u>MN102H Series C Source Code Debugger User Manual</u> (Describes the use of the C source code debugger.) ■ <u>MN102H Series Installation Manual</u> (Describes the installation of the C compiler, cross-assembler, and C source code debugger and the procedures for using the in-circuit emulator.)
P24	<p>Figure 1-5 shows the address space for the <u>MN102H75K</u>. The internal ROM contains the instructions and the font data for the on-screen display (OSD), in any location. The internal RAM contains the MCU data and the VRAM for the OSD, in any location.</p>	P24	<p>Figure 1-5 shows the address space for the <u>MN102H75K/85K</u>. The internal ROM contains the instructions and the font data for the on-screen display (OSD), in any location. The internal RAM contains the MCU data and the VRAM for the OSD, in any location.</p>
P27		P27	



P30

1.6 Pin Descriptions

1.6.1 MN102H85K Pin Description

Notes: 1. Pins marked with an asterisk (*) are N-channel, open-drain pins.
2. Pin 25 is V_{DD} in the MN102H85K and V_{PP} in the MN102HF85K.

Figure 1-9 MN102H85K Pin Configuration in Single-Chip Mode

P30

1.6 Pin Descriptions

1.6.1 MN102H75K Pin Description

Notes: 1. Pins marked with an asterisk (*) are N-channel, open-drain pins.
2. Pin 41 is V_{DD} in the MN102H75K and V_{PP} in the MN102HF75K.

Figure 1-9 MN102H75K Pin Configuration in Single-Chip Mode

P31

1.6.2 MN102H75K Pin Description

Notes: 1. Pins marked with an asterisk (*) are N-channel, open-drain pins.
2. Pin 41 is V_{DD} in the MN102H75K and V_{PP} in the MN102HF75K.

Figure 1-10 MN102H75K Pin Configuration in Single-Chip Mode

P32

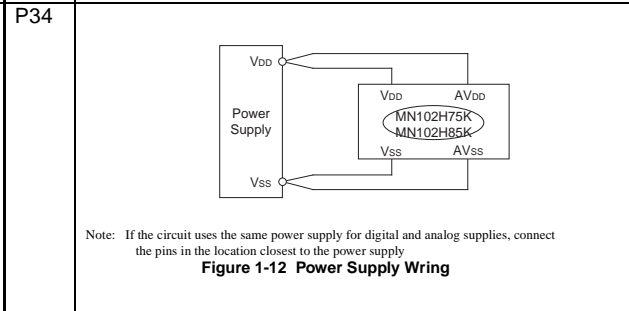
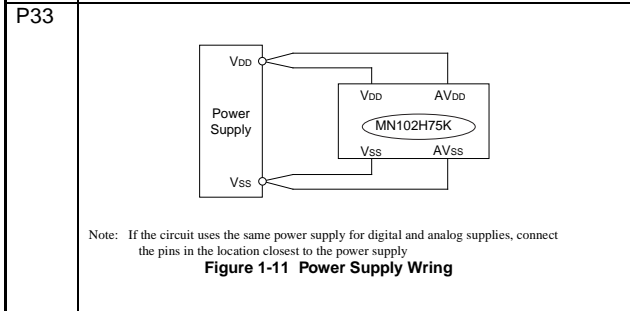
Table 1-3 Pin Function(Continued)



Block	Pin Name
I/O ports	P00-P07
	P10-P17
	P20-P27
	P30-P37
	P40-P47
	P50-P57
	P60-P67
	P70-P77
	P80-P87

P33

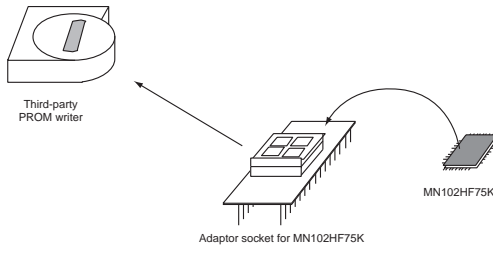
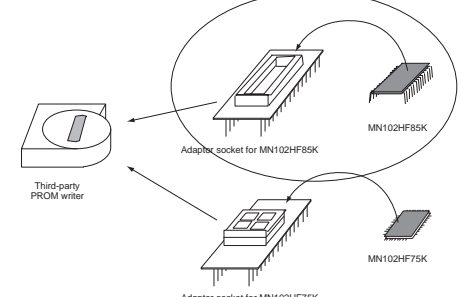
























Table 1-3 Pin Function(Continued)

Block	Pin Name
I/O ports	P00-P07
	P10-P17
	P20-P27
	P30-P37
	P40-P47
	P50-P57
	P60-P67
	P70-P77
	P80-P87



P33	The MN102H75K contains an internal PLL circuit. To use this circuit, you must connect it to an external (lag-lead) filter.	P34	The MN102H75K/85K contains an internal PLL circuit. To use this circuit, you must connect it to an external (lag-lead) filter.																														
P37	<p>The most important factor in real-time control is an MCU's speed in servicing interrupts. The MN102H75K has an extremely fast interrupt response time due to its ability to abort instructions, such as multiply or divide, that require multiple clock cycles. The MN102H75K re-executes an aborted instruction after returning from the interrupt service routine.</p> <p>This section describes the interrupt system in the MN102H75K. The MN102H75K contains 36 interrupt group controllers. Each controls a single interrupt group. Because each group contains only one interrupt vector, the MN102H75K can handle interrupts much quicker than previously possible. Each interrupt group belongs to one of twelve classes, which defines its interrupt priority level.</p> <p>With the exception of reset interrupts, all interrupts from timers, other peripheral circuits, and external pins must be registered in an interrupt group controller. Once they are registered, interrupt requests are sent to the CPU in accordance with the interrupt mask level (0 to 6) set in the interrupt group controller. Groups 1 to 3 are dedicated to system interrupts. Table 2-1 compares the interrupt parameters of the MN102H75K to those of the MN102L35G, the comparable MCU in the previous generation of the 16-bit series</p> <p>Table2-1Comparison of MN102H75K and MN102L35G Interrupt Features</p> <table border="1"> <thead> <tr> <th>Parameter</th> <th>MN102L35G</th> <th>MN102H75K</th> </tr> </thead> <tbody> <tr> <td>interrupt groups (IAGR group numbers)</td> <td>4 vectors per group (Separated by interrupt service routine)</td> <td>1 vector per group (Group number generated for each interrupt)</td> </tr> <tr> <td>interrupt response time</td> <td>Good</td> <td>Excellent</td> </tr> <tr> <td>interrupt level settings</td> <td>4 vectors per level</td> <td>4 vectors per level</td> </tr> <tr> <td>Software compatibility</td> <td>_____</td> <td>Easily modified</td> </tr> </tbody> </table> <p>The MN102H75K has six external interrupt pins. Set the interrupt condition (positive edge, negative edge, either edge, or active low) in the EXTMD register</p>	Parameter	MN102L35G	MN102H75K	interrupt groups (IAGR group numbers)	4 vectors per group (Separated by interrupt service routine)	1 vector per group (Group number generated for each interrupt)	interrupt response time	Good	Excellent	interrupt level settings	4 vectors per level	4 vectors per level	Software compatibility	_____	Easily modified	P37	<p>The most important factor in real-time control is an MCU's speed in servicing interrupts. The MN102H75K/85K has an extremely fast interrupt response time due to its ability to abort instructions, such as multiply or divide, that require multiple clock cycles. The MN102H75K/85K re-executes an aborted instruction after returning from the interrupt service routine.</p> <p>This section describes the interrupt system in the MN102H75K/85K. The MN102H75K/85K contains 36 interrupt group controllers. Each controls a single interrupt group. Because each group contains only one interrupt vector, the MN102H75K/85K can handle interrupts much quicker than previously possible. Each interrupt group belongs to one of twelve classes, which defines its interrupt priority level.</p> <p>With the exception of reset interrupts, all interrupts from timers, other peripheral circuits, and external pins must be registered in an interrupt group controller. Once they are registered, interrupt requests are sent to the CPU in accordance with the interrupt mask level (0 to 6) set in the interrupt group controller. Groups 1 to 3 are dedicated to system interrupts. Table 2-1 compares the interrupt parameters of the MN102H75K/85K to those of the MN102L35G, the comparable MCU in the previous generation of the 16-bit series</p> <p>Table2-1Comparison of MN102H75K/85K and MN102L35G Interrupt Features</p> <table border="1"> <thead> <tr> <th>Parameter</th> <th>MN102L35G</th> <th>MN102H75K/85K</th> </tr> </thead> <tbody> <tr> <td>interrupt groups (IAGR group numbers)</td> <td>4 vectors per group (Separated by interrupt service routine)</td> <td>1 vector per group (Group number generated for each interrupt)</td> </tr> <tr> <td>interrupt response time</td> <td>Good</td> <td>Excellent</td> </tr> <tr> <td>interrupt level settings</td> <td>4 vectors per level</td> <td>4 vectors per level</td> </tr> <tr> <td>Software compatibility</td> <td>_____</td> <td>Easily modified</td> </tr> </tbody> </table> <p>The MN102H75K/85K has six external interrupt pins. Set the interrupt condition (positive edge, negative edge, either edge, or active low) in the EXTMD register</p>	Parameter	MN102L35G	MN102H75K/85K	interrupt groups (IAGR group numbers)	4 vectors per group (Separated by interrupt service routine)	1 vector per group (Group number generated for each interrupt)	interrupt response time	Good	Excellent	interrupt level settings	4 vectors per level	4 vectors per level	Software compatibility	_____	Easily modified
Parameter	MN102L35G	MN102H75K																															
interrupt groups (IAGR group numbers)	4 vectors per group (Separated by interrupt service routine)	1 vector per group (Group number generated for each interrupt)																															
interrupt response time	Good	Excellent																															
interrupt level settings	4 vectors per level	4 vectors per level																															
Software compatibility	_____	Easily modified																															
Parameter	MN102L35G	MN102H75K/85K																															
interrupt groups (IAGR group numbers)	4 vectors per group (Separated by interrupt service routine)	1 vector per group (Group number generated for each interrupt)																															
interrupt response time	Good	Excellent																															
interrupt level settings	4 vectors per level	4 vectors per level																															
Software compatibility	_____	Easily modified																															
P72	<p>The MN102H75K provides two ways to reduce power consumption, controlling CPU operating and standby modes to cut overall consumption and shutting down unused functions by stopping the system clock supplied to them.</p> <p>3.1 CPU Modes</p> <p>3.1.1 Description</p> <p>The MN102H75K has two CPU operating modes, NORMAL and SLOW, and two CPU standby modes, HALT and STOP. Effective use of these modes can significantly reduce power consumption. Figure 3-1 shows the CPU states in the different modes</p>	P72	<p>The MN102H75K/85K provides two ways to reduce power consumption, controlling CPU operating and standby modes to cut overall consumption and shutting down unused functions by stopping the system clock supplied to them.</p> <p>3.1 CPU Modes</p> <p>3.1.1 Description</p> <p>The MN102H75K/85K has two CPU operating modes, NORMAL and SLOW, and two CPU standby modes, HALT and STOP. Effective use of these modes can significantly reduce power consumption. Figure 3-1 shows the CPU states in the different modes</p>																														
P73	 <p>The MN102H75K/85K recovers from power up and reset in SLOW mode. For normal operation, the program must switch the MCU from SLOW to NORMAL mode</p>	P73	 <p>The MN102H75K/85K recovers from power up and reset in SLOW mode. For normal operation, the program must switch the MCU from SLOW to NORMAL mode</p>																														
P73	The MN102H75K contains a PLL circuit that, in NORMAL mode, multiplies the clock input through the OSC1 and OSC2 pins by 12, divides the signal by 2, then sends the resulting clock to the CPU. (See figure 3-2.) The MCU starts in SLOW mode on power up and on recovery from a reset. In SLOW mode (system clock = 2 MHz), the clock from the OSC pins feeds directly to the CPU, without going through the PLL circuit. This means that the program must switch the CPU from SLOW to NORMAL mode (system clock = 12 MHz)	P73	The MN102H75K/85K contains a PLL circuit that, in NORMAL mode, multiplies the clock input through the OSC1 and OSC2 pins by 12, divides the signal by 2, then sends the resulting clock to the CPU. (See figure 3-2.) The MCU starts in SLOW mode on power up and on recovery from a reset. In SLOW mode (system clock = 2 MHz), the clock from the OSC pins feeds directly to the CPU, without going through the PLL circuit. This means that the program must switch the CPU from SLOW to NORMAL mode (system clock = 12 MHz)																														
P73	For information on invoking SLOW mode from NORMAL mode, see MN10200 Series Linear Addressing High-Speed Version LSI User Manual .	P73	For information on invoking SLOW mode from NORMAL mode, see MN102H Series LSI User Manual .																														
P75	The MN102H75K allows you to turn each peripheral function on or off through writing to the registers. You can significantly reduce power consumption by turning off unused functions. Table 3-1 shows the register bits controlling on and off for each function block. The ADC used for the OSD and CCD functions is turned off on reset. Write a 1 to the function to enable it, when necessary	P75	The MN102H75K/85K allows you to turn each peripheral function on or off through writing to the registers. You can significantly reduce power consumption by turning off unused functions. Table 3-1 shows the register bits controlling on and off for each function block. The ADC used for the OSD and CCD functions is turned off on reset. Write a 1 to the function to enable it, when necessary																														

P77	The MN102H75K contains four 8-bit timers that can serve as interval timers, event timer/counters, clock generators (divide-by-2 output of the underflow), reference clocks for the serial interfaces, or start timers for A/D conversions. The clock source can be the internal clock (oscillator frequency divided by 2) or the external clock (1/4 or less the oscillator frequency input). A timer interrupt is generated by a timer underflow	P77	The MN102H75K/85K contains four 8-bit timers that can serve as interval timers, event timer/counters, clock generators (divide-by-2 output of the underflow), reference clocks for the serial interfaces, or start timers for A/D conversions. The clock source can be the internal clock (oscillator frequency divided by 2) or the external clock (1/4 or less the oscillator frequency input). A timer interrupt is generated by a timer underflow
P88	The MN102H75K contains two 16-bit up/down timers, timers 5 and 6. Associated with each timer are two compare/capture registers that can capture and compare the up/down counter values, generate PWM signals, and generate interrupts. The PWM function has a double buffering mode that causes cycle and transition changes to occur at the beginning of the next clock cycle. This prevents PWM signal losses and minimizes waveform distortion during timing changes	P88	The MN102H75K/85K contains two 16-bit up/down timers, timers 5 and 6. Associated with each timer are two compare/capture registers that can capture and compare the up/down counter values, generate PWM signals, and generate interrupts. The PWM function has a double buffering mode that causes cycle and transition changes to occur at the beginning of the next clock cycle. This prevents PWM signal losses and minimizes waveform distortion during timing changes
P127	The MN102H75K contains two general-purpose serial interfaces with synchronous serial, UART, and I ² C modes. The maximum baud rate in synchronous serial mode is 12 Mbps. In UART mode, the maximum baud rate is 375,000 bps, when B _{OSC} = 24 MHz	P127	The MN102H75K/85K contains two general-purpose serial interfaces with synchronous serial, UART, and I ² C modes. The maximum baud rate in synchronous serial mode is 12 Mbps. In UART mode, the maximum baud rate is 375,000 bps, when B _{OSC} = 24 MHz
P143	The MN102H75K contains an 8-bit charge redistribution A/D converter (ADC) that can process up to 12 channels. The reference clock is selectable to B _{OSC} × 1/8 or 1/16. When B _{OSC} is 24 MHz, you must set the reference clock to B _{OSC} /8 (conversion rate = 4 μs) or higher	P143	The MN102H75K/85K contains an 8-bit charge redistribution A/D converter (ADC) that can process up to 12 channels. The reference clock is selectable to B _{OSC} × 1/8 or 1/16. When B _{OSC} is 24 MHz, you must set the reference clock to B _{OSC} /8 (conversion rate = 4 μs) or higher
P153	The MN102H75K contains an on-screen display (OSD) function composed of three layers: a text layer, a graphics layer, and a cursor layer. You can control each layer individually, which gives you great freedom in positioning displays. You can also modify the ROM space that contains the text characters and the graphic tiles and the VRAM space that contains the text and graphics programs. This allows you to adjust the memory space to fit your application	P153	The MN102H75K/85K contains an on-screen display (OSD) function composed of three layers: a text layer, a graphics layer, and a cursor layer. You can control each layer individually, which gives you great freedom in positioning displays. You can also modify the ROM space that contains the text characters and the graphic tiles and the VRAM space that contains the text and graphics programs. This allows you to adjust the memory space to fit your application
P191	This section describes how the MN102H75K handles the timing of direct memory access (DMA) transfers of OSD data and OSD interrupts	P191	This section describes how the MN102H75K/85K handles the timing of direct memory access (DMA) transfers of OSD data and OSD interrupts
P194	The MN102H75K OSD achieves a shuttering effect using four programmable shutters—two vertical and two horizontal. With this feature, you can shutter any portion of the OSD display, or you can combine shuttering with a wipe-out effect to create a smooth appearing and disappearing effect	P194	The MN102H75K/85K OSD achieves a shuttering effect using four programmable shutters—two vertical and two horizontal. With this feature, you can shutter any portion of the OSD display, or you can combine shuttering with a wipe-out effect to create a smooth appearing and disappearing effect
P216	The MN102H75K contains a remote signal receiver that processes signals in two formats: Household Electrical Appliance Manufacturers Association (HEAMA) format and 5-/6-bit format. This chapter provides an overview of each block in the circuit and describes the operation of the receiver	P216	The MN102H75K/85K contains a remote signal receiver that processes signals in two formats: Household Electrical Appliance Manufacturers Association (HEAMA) format and 5-/6-bit format. This chapter provides an overview of each block in the circuit and describes the operation of the receiver
P227	The MN102H75K contains two identical closed-caption decoder circuits, CCD0 and CCD1	P227	The MN102H75K/85K contains two identical closed-caption decoder circuits, CCD0 and CCD1
P229	The clamping circuit internal to the MN102H75K provides three current sources—high, medium, and low.	P229	The clamping circuit internal to the MN102H75K/85K provides three current sources—high, medium, and low.
P249	The MN102H75K contains seven 8-bit pulse width modulators (PWMs) with a minimum pulse width of 16/f _{SYCLK} and an output waveform cycle of 2 ¹² /f _{SYCLK} . (With a 4-MHz oscillator, 16/f _{SYCLK} = 1.33 μs (8 μs for SLOW mode) and 2 ¹² /f _{SYCLK} = 341.3 μs (2 ms for SLOW mode).)	P249	The MN102H75K/85K contains seven 8-bit pulse width modulators (PWMs) with a minimum pulse width of 16/f _{SYCLK} and an output waveform cycle of 2 ¹² /f _{SYCLK} . (With a 4-MHz oscillator, 16/f _{SYCLK} = 1.33 μs (8 μs for SLOW mode) and 2 ¹² /f _{SYCLK} = 341.3 μs (2 ms for SLOW mode).)
P250	The MN102H75K contains 50 pins that form general-purpose I/O ports. Ports 0, 1, 2, 3, 4, and 5 are 8-bit ports, and port 6 is a 2-bit port. All of these pins have alternate functions. (Ports 7 and 8 are only available with the quad flat package.)	P250	The MN102H75K/85K contains 50 pins that form general-purpose I/O ports. Ports 0, 1, 2, 3, 4, and 5 are 8-bit ports, and port 6 is a 2-bit port. All of these pins have alternate functions. (Ports 7 and 8 are only available with the quad flat package.)
P293	The MN102H75K contains one I ² C bus controller, fully compliant with the I ² C specification, that can control one of two I ² C bus connections	P293	The MN102H75K/85K contains one I ² C bus controller, fully compliant with the I ² C specification, that can control one of two I ² C bus connections
P295	Figure 13-3 shows the MN102H75K operation sequence in each of these modes. In all modes, the I ² C bus controller generates an interrupt after each data byte transfer, then the software loads the next data byte	P295	Figure 13-3 shows the MN102H75K/85K operation sequence in each of these modes. In all modes, the I ² C bus controller generates an interrupt after each data byte transfer, then the software loads the next data byte
P299	Figure 13-6 and table 13-5 provide the timing definitions and specifications for the for the MN102H75K I ² C bus interface	P299	Figure 13-6 and table 13-5 provide the timing definitions and specifications for the for the MN102H75K/85K I ² C bus interface
P301	Note: The circled areas are signals output from the MN102H75K	P301	Note: The circled areas are signals output from the MN102H75K/85K
P303	Note: The circled areas are signals output from the MN102H75K	P303	Note: The circled areas are signals output from the MN102H75K/85K

P307	The MN102H75K contains two H counter circuits that can be used to count the HSYNC signal. Each H counter consists of a 10-bit counter and 10-bit register	P307	The MN102H75K/85K contains two H counter circuits that can be used to count the HSYNC signal. Each H counter consists of a 10-bit counter and 10-bit register																																																						
P308	<p>Table 14-1 H Counter Pins</p> <table border="1"> <thead> <tr> <th rowspan="2">Pin Name</th> <th colspan="2">Pin No.</th> <th rowspan="2">Description</th> <th rowspan="2">Alternative Functions</th> </tr> <tr> <th>H75</th> <th>F85</th> </tr> </thead> <tbody> <tr> <td>HI0</td> <td>17</td> <td></td> <td>Count source pin</td> <td>P43/TM5IOB</td> </tr> <tr> <td>HI1</td> <td>16</td> <td></td> <td>Count source pin</td> <td>P44/TM5IC</td> </tr> <tr> <td>VI0</td> <td>6</td> <td></td> <td>Count reset pin</td> <td>P52/IRQ4</td> </tr> <tr> <td>VSINC</td> <td>2</td> <td></td> <td>Count reset pin</td> <td>P54/IRQ5</td> </tr> </tbody> </table>	Pin Name	Pin No.		Description	Alternative Functions	H75	F85	HI0	17		Count source pin	P43/TM5IOB	HI1	16		Count source pin	P44/TM5IC	VI0	6		Count reset pin	P52/IRQ4	VSINC	2		Count reset pin	P54/IRQ5	P308	<p>Table 14-1 H Counter Pins</p> <table border="1"> <thead> <tr> <th rowspan="2">Pin Name</th> <th colspan="2">Pin No.</th> <th rowspan="2">Description</th> <th rowspan="2">Alternative Functions</th> </tr> <tr> <th>H75</th> <th>F85</th> </tr> </thead> <tbody> <tr> <td>HI0</td> <td>17</td> <td>45</td> <td>Count source pin</td> <td>P43/TM5IOB</td> </tr> <tr> <td>HI1</td> <td>16</td> <td>46</td> <td>Count source pin</td> <td>P44/TM5IC</td> </tr> <tr> <td>VI0</td> <td>6</td> <td>53</td> <td>Count reset pin</td> <td>P52/IRQ4</td> </tr> <tr> <td>VSINC</td> <td>2</td> <td>55</td> <td>Count reset pin</td> <td>P54/IRQ5</td> </tr> </tbody> </table>	Pin Name	Pin No.		Description	Alternative Functions	H75	F85	HI0	17	45	Count source pin	P43/TM5IOB	HI1	16	46	Count source pin	P44/TM5IC	VI0	6	53	Count reset pin	P52/IRQ4	VSINC	2	55	Count reset pin	P54/IRQ5
Pin Name	Pin No.		Description	Alternative Functions																																																					
	H75	F85																																																							
HI0	17		Count source pin	P43/TM5IOB																																																					
HI1	16		Count source pin	P44/TM5IC																																																					
VI0	6		Count reset pin	P52/IRQ4																																																					
VSINC	2		Count reset pin	P54/IRQ5																																																					
Pin Name	Pin No.		Description	Alternative Functions																																																					
	H75	F85																																																							
HI0	17	45	Count source pin	P43/TM5IOB																																																					
HI1	16	46	Count source pin	P44/TM5IC																																																					
VI0	6	53	Count reset pin	P52/IRQ4																																																					
VSINC	2	55	Count reset pin	P54/IRQ5																																																					
P316	The MN102HF75K is electrically programmable, 256-kilobyte flash ROM versions of the MN102H75K. It is programmed in one of two modes:	P316	The MN102HF75K and MN102HF85K are electrically programmable, 256-kilobyte flash ROM versions of the MN102H75K and MN102H85K. They are programmed in one of two modes:																																																						
P317	<p>B.2 Benefits</p> <p>Because you can maintain and upgrade the program in the MN102HF75K up to and immediately following product release, this version of the device shortens time-to-market by as much as one month. This device is ideal for applications in quickly changing markets, since it allows you to revise the microcontroller program in an existing product.</p> <p>B.3 Using the PROM Writer Mode</p> <p>In this mode, the MN102HF75K allows a PROM writer to program the internal flash memory as if it was a standalone memory chip. The microcontroller is inserted into a dedicated adaptor socket, which connects to DATA-I/O's LabSite PROM writer. When the microcontroller connects to the adaptor socket, it automatically enters PROM writer mode. The adaptor socket ties the microcontroller pin states to PROM writer mode, and programming occurs without any reference to the microcontroller pin states.</p> 	P317	<p>B.2 Benefits</p> <p>Because you can maintain and upgrade the program in the MN102HF75K/85K up to and immediately following product release, this version of the device shortens time-to-market by as much as one month. This device is ideal for applications in quickly changing markets, since it allows you to revise the microcontroller program in an existing product.</p> <p>B.3 Using the PROM Writer Mode</p> <p>In this mode, the MN102HF75K allows a PROM writer to program the internal flash memory as if it was a standalone memory chip. The microcontroller is inserted into a dedicated adaptor socket, which connects to DATA-I/O's LabSite PROM writer. When the microcontroller connects to the adaptor socket, it automatically enters PROM writer mode. The adaptor socket ties the microcontroller pin states to PROM writer mode, and programming occurs without any reference to the microcontroller pin states.</p> 																																																						
P318	<table border="1"> <thead> <tr> <th>Hardware Part</th> <th>Device</th> </tr> </thead> <tbody> <tr> <td>Package (internal view)</td> <td>MN102HF75KPF </td> </tr> <tr> <td>Adaptor</td> <td>84-pin DIP </td> </tr> <tr> <td>Third-party PROM writer</td> <td>Go!ng writer </td> </tr> </tbody> </table> <p>Model 1030 CSIW Micro Electronics 4100 Mission Road Tustin, CA 92680 Tel: 949-951-5855</p>	Hardware Part	Device	Package (internal view)	MN102HF75KPF 	Adaptor	84-pin DIP 	Third-party PROM writer	Go!ng writer 	P318	<table border="1"> <thead> <tr> <th>Hardware Part</th> <th>Device</th> </tr> </thead> <tbody> <tr> <td>Package (internal view)</td> <td>MN102HF75KPF </td> </tr> <tr> <td>Adaptor</td> <td>MN102HF85KPF  84-pin DIP </td> </tr> <tr> <td>Third-party PROM writer</td> <td>Go!ng writer </td> </tr> <tr> <td></td> <td>Single unit writer </td> </tr> </tbody> </table> <p>Model 1030 CSIW Micro Electronics 4100 Mission Road Tustin, CA 92680 Tel: 949-951-5855</p> <p>Model LabSite DIP 86-1 CSIW Micro Electronics 4100 Mission Road Tustin, CA 92680 Tel: 949-951-5855</p>	Hardware Part	Device	Package (internal view)	MN102HF75KPF 	Adaptor	MN102HF85KPF  84-pin DIP 	Third-party PROM writer	Go!ng writer 		Single unit writer 																																				
Hardware Part	Device																																																								
Package (internal view)	MN102HF75KPF 																																																								
Adaptor	84-pin DIP 																																																								
Third-party PROM writer	Go!ng writer 																																																								
Hardware Part	Device																																																								
Package (internal view)	MN102HF75KPF 																																																								
Adaptor	MN102HF85KPF  84-pin DIP 																																																								
Third-party PROM writer	Go!ng writer 																																																								
	Single unit writer 																																																								

**MN102H75K/F75K/85K/F85K
LSI User's Manual**

October,2001 1st Edition 1st Printing

Issued by Matsushita Electric Industrial Co., Ltd.

© Matsushita Electric Industrial Co., Ltd.

Semiconductor Company, Matsushita Electric Industrial Co., Ltd.

Nagaokakyo, Kyoto, 617-8520 Japan

Tel: (075) 951-8151

<http://www.panasonic.co.jp/semicon/>

SALES OFFICES

■ NORTH AMERICA

● U.S.A. Sales Office:

Panasonic Industrial Company [PIC]

● New Jersey Office:

Two Panasonic Way Secaucus, New Jersey 07094 U.S.A.

Tel: 1-201-348-5257 Fax:1-201-392-4652

● Chicago Office:

1707 N. Randall Road Elgin, Illinois 60123-7847 U.S.A.

Tel: 1-847-468-5720 Fax:1-847-468-5725

● Milpitas Office:

1600 McCandless Drive Milpitas, California 95035 U.S.A.

Tel: 1-408-942-2912 Fax:1-408-946-9063

● Atlanta Office:

1225 Northbrook Parkway Suite 1-151 Suwanee, GA

30024 U.S.A.

Tel: 1-770-338-6953 Fax:1-770-338-6849

● San Diego Office:

9444 Balboa Avenue, Suite 185, San Diego, California

92123 U.S.A.

Tel: 1-619-503-2903 Fax:1-858-715-5545

● Canada Sales Office:

Panasonic Canada Inc. [PCI]

5770 Ambler Drive 27 Mississauga, Ontario, L4W 2T3
CANADA

Tel: 1-905-238-2101 Fax:1-905-238-2414

■ LATIN AMERICA

● Mexico Sales Office:

Panasonic de Mexico, S.A. de C.V. [PANAMEX]

Amores 1120 Col. Del Valle Delegacion Benito Juarez
C.P. 03100 Mexico, D.F. MEXICO

Tel: 52-5-488-1000 Fax:52-5-488-1073

● Guadalajara Office:

SUCURSAL GUADALAJARA

Av. Lazaro Cardenas 2305 Local G-102 Plaza Comercial

Abastos; Col. Las Torres Guadalajara, Jal. 44920

MEXICO

Tel: 52-3-671-1205 Fax:52-3-671-1256

● Brazil Sales Office:

Panasonic do Brasil Ltda. [PANABRAS]

Caixa Postal 1641, Sao Jose dos Campos, Estado de Sao
Paulo

Tel: 55-12-335-9000 Fax:55-12-331-3789

■ EUROPE

● U.K. Sales Office:

Panasonic Industrial Europe Ltd. [PIEL]

Willoughby Road, Bracknell, Berks., RG12 8FP,

THE UNITED KINGDOM

Tel: 44-1344-85-3671 Fax:44-1344-85-3853

● Germany Sales Office:

Panasonic Industrial Europe GmbH [PIEG]

Hans-Pinsel-Strasse 2 85540 Haar, GERMANY

Tel: 49-89-46159-119 Fax:49-89-46159-195

■ ASIA

● Singapore Sales Office:

Panasonic Semiconductor of South Asia [PSSA]

300 Beach Road, #16-01, The Concourse, Singapore

199555 THE REPUBLIC OF SINGAPORE

Tel: 65-390-3688 Fax:65-390-3689

● Malaysia Sales Office:

Panasonic Industrial Company (M) Sdn. Bhd. [PICM]

● Head Office:

Tingkat 16B, Menara PKNS Petaling Jaya, No.17, Jalan

Yong Shook Lin 46050 Petaling Jaya, Selangor Darul

Ehsan, MALAYSIA

Tel: 60-3-7951-6601 Fax:60-3-7954-5968

● Penang Office:

Suite 20-07, 20th Floor, MWE Plaza, No.8, Lebuhraya
Farquhar, 10200 Penang, Malaysia

Tel: 60-4-201-5113 Fax:60-4-261-9989

● Johore Sales Office:

Menara Pelangi, Suite 8.3A, Level 8, No.2, Jalan Kuning

Taman Pelangi, 80400 Johor Bahru, Johor, MALAYSIA

Tel: 60-7-331-3822 Fax:60-7-355-3996

● Thailand Sales Office:

Panasonic Industrial (THAILAND) Ltd. [PICT]

252-133 Muang Thai-Phatra Complex Building, 31st Fl.

Rachadaphisek Rd., Huaykwang, Bangkok 10320,

THAILAND

Tel: 66-2-693-3428 Fax:66-2-693-3422

● Philippines Sales Office:

Panasonic Industrial Sales Philippines Division of

Matsushita Electric Philippines Corporation

102 Laguna Boulevard, Bo. Don Jose Laguna Technopark,

Santa Rosa, Laguna 4026 PHILIPPINES

Tel: 63-2-520-8615 Fax:63-2-520-8629

● India Sales Office:

National Panasonic India Ltd. [NPI]

E Block, 510, International Trade Tower Nehru Place, New
Delhi_110019 INDIA

Tel: 91-11-629-2870 Fax:91-11-629-2877

● Indonesia Sales Office:

P.T.MET & Gobel [M&G]

JL. Dewi Sartika (Cawang 2) Jakarta 13630, INDONESIA

Tel: 62-21-801-5666 Fax:62-21-801-5675

● Taiwan Sales Office:

Panasonic Industrial Sales (Taiwan) Co., Ltd. [PIST]

● Head Office:

6F, 550, Sec. 4, Chung Hsiao E. RD. Taipei, 110, TAIWAN

Tel: 886-2-2757-1900 Fax:886-2-2757-1906

● Kaohsiung Office:

6th Floor, Hsin Kong Bldg. No.251, Chi Hsien 1st Road

Kaohsiung 800, TAIWAN

Tel: 886-7-346-3815 Fax:886-7-236-8362

● China Sales Office:

Panasonic Industrial (Shanghai) Co., Ltd. [PI(SH)]

Floor 6, Zhong Bao Mansion, 166 East Road Lujian Zui,

PU Dong New District, Shanghai, 200120 CHINA

Tel: 86-21-5866-6114 Fax:86-21-5866-8000

Panasonic Industrial (Tianjin) Co., Ltd. [PI(TJ)]

Room No.1001, Tianjin International Building 75, Nanjin

Road, Tianjin 300050, CHINA

Tel: 86-22-2313-9771 Fax:86-22-2313-9770

Panasonic SH Industrial Sales (Shenzhen) Co., Ltd.

[PSI(SZ)]

7A-107, International Business & Exhibition Centre,

Futian Free Trade Zone, Shenzhen 518048, CHINA

Tel: 86-755-359-8500 Fax:86-755-359-8516

Panasonic Shun Hing Industrial Sales (Hong Kong)

Co., Ltd.

[PSI(HK)]

11th Floor, Great Eagle Center 23 Harbour Road,

Wanchai, HONG KONG

Tel: 852-2529-7322 Fax:852-2865-3697

● Korea Sales Office:

Panasonic Industrial Korea Co., Ltd. [PIKL]

Kukje Center Bldg. 11th Fl., 191 Hangangro 2ga,

Yongsan-ku, Seoul 140-702, KOREA

Tel: 82-2-795-9600 Fax:82-2-795-1542