MC9S08SU16 Reference Manual

Supports: MC9S08SU16VFK MC9S08SU8VFK

Document Number: MC9S08SU16RM
Rev. 5, 4/2017

Contents

Section number **Title** **Page**

Chapter 1
About This Document

1.1 Overview....33

1.1.1 Purpose....33

1.1.2 Audience....33

1.2 Conventions....33

1.2.1 Numbering systems....33

1.2.2 Typographic notation....34

1.2.3 Special terms....34

Chapter 2
Introduction

2.1 Introduction....35

2.2 Module functional categories....35

2.2.1 S08L core modules....36

2.2.2 System modules....36

2.2.3 Memories and memory interfaces....37

2.2.4 Clocks....37

2.2.5 Security and integrity modules....37

2.2.6 Analog modules....38

2.2.7 Timer modules....38

2.2.8 Communication interfaces....39

2.2.9 Human-machine interfaces....39

2.3 MCU block diagram....40

2.4 Orderable part numbers....41

Chapter 3
Memory

3.1 Memory map....43

3.2 Reset and interrupt vector assignments....44

3.3 Register addresses assignments 46
3.4 Random-access memory (RAM) 51
3.5 Flash memory 51
3.6 System register file 52

Chapter 4
Interrupt

4.1 Interrupts 53
4.1.1 Interrupt stack frame 54
4.1.2 Hardware nested interrupt 55
4.1.2.1 Interrupt priority level register 57
4.1.2.2 Interrupt priority level comparator set 57
4.1.2.3 Interrupt priority mask update and restore mechanism 57
4.1.2.4 Integration and application of the IPC 58
4.2 IPC memory map and register descriptions 59
4.2.1 IPC Status and Control Register (IPC_SC) 59
4.2.2 Interrupt Priority Mask Pseudo Stack Register (IPC_IPMPS) 60
4.2.3 Interrupt Level Setting Registers n (IPC_ILRS*n*) 61
4.3 IRQ 61
4.3.1 Features 62
4.3.1.1 Configuration options 62
4.3.1.2 Edge and level sensitivity 63
4.4 IRQ Memory Map and Register Descriptions 63
4.4.1 Interrupt Pin Request Status and Control Register (IRQ_SC) 63

Chapter 5
Clock management

5.1 Clock module 65
5.2 System clock distribution 65
5.3 Internal clock source (ICS) 67
5.4 20 kHz low-power oscillator (LPO) 68

5.5 Peripheral clock gating ... 68

Chapter 6
Power Management

6.1 Introduction ... 71
6.2 Features ... 71
6.2.1 Run mode ... 71
6.2.2 Wait mode ... 72
6.2.3 Stop mode ... 72
6.2.4 Active BDM enabled in stop mode ... 72
6.2.5 Power modes behaviors ... 73
6.3 Bandgap reference ... 74

Chapter 7
Signal multiplexing and signal descriptions

7.1 Introduction ... 75
7.2 Port control and interrupt module features ... 75
7.3 Signal multiplexing constraints ... 75
7.4 Pinout ... 76
7.4.1 Signal multiplexing and pin assignments ... 76
7.4.2 Signal description table ... 77
7.4.3 Pinout ... 81

Chapter 8
Port Control (PORT)

8.1 Introduction ... 83
8.2 Port data and data direction ... 84
8.3 Internal pullup/pulldown enable ... 84
8.4 Input glitch filter ... 85
8.5 Memory map and register definition ... 86
8.5.1 Port A Data Register (PORT_PTAD) ... 86
8.5.2 Port B Data Register (PORT_PTBD) ... 87
8.5.3 Port C Data Register (PORT_PTCD) ... 87

8.5.4 Port A Direction Register (PORT_PTADD) ... 88
8.5.5 Port B Direction Register (PORT_PTBDD) ... 89
8.5.6 Port C Direction Register (PORT_PTCDD) ... 89
8.5.7 Port A Pullup Enable Register (PORT_PTAPE) ... 90
8.5.8 Port B Pullup/Pulldown Enable Register (PORT_PTBPE) ... 91
8.5.9 Port C Pullup Enable Register (PORT_PTCPE) ... 91
8.5.10 Port B High Drive Strength Selection Register (PORT_PTBHD) ... 92
8.5.11 Port Clock Division Register (PORT_FCLKDIV) ... 92
8.5.12 Port Filter Register 0 (PORT_IOFLT0) ... 93
8.5.13 Port Filter Register 1 (PORT_IOFLT1) ... 94
8.5.14 Port Filter Register 2 (PORT_IOFLT2) ... 95

Chapter 9
System Integration Module (SIM)

9.1 Chip specific windowed COP ... 97
9.2 System device identification (SDID) ... 98
9.3 Universally unique identification (UUID) ... 98
9.4 Reset and system initialization ... 98
9.5 Computer operating properly (COP) watchdog ... 99
9.6 System options ... 100
9.6.1 BKGD pin ... 100
9.6.2 RESET_b pin enable ... 101
9.7 System interconnection ... 101
9.7.1 Inter Module Crossbar Switch (XBAR) ... 101
9.7.2 Module to module interconnects ... 103
9.8 Memory map and register definition ... 104
9.8.1 System Reset Status Register (SIM_SRS) ... 105
9.8.2 System Background Debug Force Reset Register (SIM_SBDFR) ... 107
9.8.3 System Device Identification Register: High (SIM_SDIDH) ... 107
9.8.4 System Device Identification Register: Low (SIM_SDIDL) ... 108

9.8.5 System Options Register 1 (SIM_SOPT1)..... 108
9.8.6 System Options Register 2 (SIM_SOPT2)..... 110
9.8.7 System Port A Pin Multiplexing Control Register: Low (SIM_MUXPTAL)..... 111
9.8.8 System Port A Pin Multiplexing Control Register: High (SIM_MUXPTAH)..... 112
9.8.9 System Port B Pin Multiplexing Control Register: Low (SIM_MUXPTBL)..... 113
9.8.10 System Port B Pin Multiplexing Control Register: High (SIM_MUXPTBH)..... 114
9.8.11 System Port C Pin Multiplexing Control Register: Low (SIM_MUXPTCL)..... 116
9.8.12 System Clock Gating Control 1 Register (SIM_SCGC1)..... 116
9.8.13 System Clock Gating Control 2 Register (SIM_SCGC2)..... 117
9.8.14 System Clock Gating Control 3 Register (SIM_SCGC3)..... 119
9.8.15 System Clock Divider Register (SIM_SCDIV)..... 120
9.8.16 System POR Register (SIM_PORREG*n*)..... 121
9.8.17 Illegal Address Register: High (SIM_ILLAH)..... 122
9.8.18 Illegal Address Register: Low (SIM_ILLAL)..... 122
9.8.19 Universally Unique Identifier Register 0 (SIM_UUID0)..... 123
9.8.20 Universally Unique Identifier Register 1 (SIM_UUID1)..... 123
9.8.21 Universally Unique Identifier Register 2 (SIM_UUID2)..... 124
9.8.22 Universally Unique Identifier Register 3 (SIM_UUID3)..... 124
9.8.23 Universally Unique Identifier Register 4 (SIM_UUID4)..... 125
9.8.24 Universally Unique Identifier Register 5 (SIM_UUID5)..... 125
9.8.25 Universally Unique Identifier Register 6 (SIM_UUID6)..... 126
9.8.26 Universally Unique Identifier Register 7 (SIM_UUID7)..... 126

Chapter 10
Central processor unit

10.1 Introduction..... 127
10.1.1 Features..... 127
10.2 Programmer's Model and CPU Registers..... 128
10.2.1 Accumulator (A)..... 128
10.2.2 Index Register (H:X)..... 129

10.2.3 Stack Pointer (SP)..... 129
10.2.4 Program Counter (PC)..... 130
10.2.5 Condition Code Register (CCR)..... 130
10.3 Addressing Modes..... 131
10.3.1 Inherent Addressing Mode (INH)..... 132
10.3.2 Relative Addressing Mode (REL)..... 132
10.3.3 Immediate Addressing Mode (IMM)..... 132
10.3.4 Direct Addressing Mode (DIR)..... 133
10.3.5 Extended Addressing Mode (EXT)..... 133
10.3.6 Indexed Addressing Mode..... 134
10.3.6.1 Indexed, No Offset (IX)..... 134
10.3.6.2 Indexed, No Offset with Post Increment (IX+)..... 134
10.3.6.3 Indexed, 8-Bit Offset (IX1)..... 134
10.3.6.4 Indexed, 8-Bit Offset with Post Increment (IX1+)..... 135
10.3.6.5 Indexed, 16-Bit Offset (IX2)..... 135
10.3.6.6 SP-Relative, 8-Bit Offset (SP1)..... 135
10.3.6.7 SP-Relative, 16-Bit Offset (SP2)..... 136
10.3.7 Memory to memory Addressing Mode..... 136
10.3.7.1 Direct to Direct..... 136
10.3.7.2 Immediate to Direct..... 136
10.3.7.3 Indexed to Direct, Post Increment..... 136
10.3.7.4 Direct to Indexed, Post-Increment..... 137
10.4 Operation modes..... 137
10.4.1 Stop mode..... 137
10.4.2 Wait mode..... 137
10.4.3 Background mode..... 138
10.4.4 Security mode..... 139
10.5 HCS08 V6 Opcodes..... 141
10.6 Special Operations..... 141

10.6.1 Reset Sequence 141
10.6.2 Interrupt Sequence 141
10.7 Instruction Set Summary 142

Chapter 11
Flash Memory Module (FTMRH)

11.1 Introduction 155
11.2 Feature 155
11.2.1 Flash memory features 155
11.2.2 Other flash module features 156
11.3 Functional description 156
11.3.1 Modes of operation 156
11.3.1.1 Wait mode 156
11.3.1.2 Stop mode 156
11.3.2 Flash block read access 156
11.3.3 Flash memory map 157
11.3.4 Flash initialization after system reset 157
11.3.5 Flash command operations 157
11.3.5.1 Writing the FCLKDIV register 158
11.3.5.2 Command write sequence 160
11.3.6 Flash interrupts 162
11.3.6.1 Description of flash interrupt operation 162
11.3.7 Protection 162
11.3.8 Security 165
11.3.8.1 Unsecuring the MCU using backdoor key access 166
11.3.8.2 Unsecuring the MCU using BDM 167
11.3.8.3 Mode and security effects on flash command availability 167
11.3.9 Flash commands 167
11.3.9.1 Flash commands 167
11.3.10 Flash command summary 168

11.3.10.1 Erase Verify All Blocks command..... 169
11.3.10.2 Erase Verify Block command..... 169
11.3.10.3 Erase Verify Flash Section command..... 170
11.3.10.4 Read once command..... 171
11.3.10.5 Program Flash command..... 172
11.3.10.6 Program Once command..... 173
11.3.10.7 Erase All Blocks command..... 174
11.3.10.8 Erase flash block command..... 174
11.3.10.9 Erase flash sector command..... 175
11.3.10.10 Unsecure flash command..... 176
11.3.10.11 Verify backdoor access key command..... 177
11.3.10.12 Set user margin level command..... 177
11.3.10.13 Set factory margin level command..... 179
11.4 Memory map and register definition..... 180
11.4.1 Flash Clock Divider Register (FTMRH_FCLKDIV)..... 181
11.4.2 Flash Security Register (FTMRH_FSEC)..... 182
11.4.3 Flash CCOB Index Register (FTMRH_FCCOBIX)..... 183
11.4.4 Flash Configuration Register (FTMRH_FCNFG)..... 183
11.4.5 Flash Status Register (FTMRH_FSTAT)..... 184
11.4.6 Flash Protection Register (FTMRH_FPROT)..... 185
11.4.7 Flash Common Command Object Register:High (FTMRH_FCCOBHI)..... 186
11.4.8 Flash Common Command Object Register: Low (FTMRH_FCCOBLO)..... 187
11.4.9 Flash Option Register (FTMRH_FOPT)..... 187

Chapter 12
Internal Clock Source (ICS)

12.1 Introduction..... 189
12.1.1 Features..... 189
12.1.2 Block diagram..... 190
12.1.3 Modes of operation..... 190

12.1.3.1 FLL engaged internal (FEI) 190
12.1.3.2 FLL engaged external (FEE) 190
12.1.3.3 FLL bypassed internal (FBI) 191
12.1.3.4 FLL bypassed internal low power (FBILP) 191
12.1.3.5 FLL bypassed external (FBE) 191
12.1.3.6 FLL bypassed external low power (FBELP) 191
12.1.3.7 Stop (STOP) 191
12.2 External signal description 191
12.3 Register definition 192
12.3.1 ICS Control Register 1 (ICS_C1) 192
12.3.2 ICS Control Register 2 (ICS_C2) 193
12.3.3 ICS Control Register 3 (ICS_C3) 194
12.3.4 ICS Control Register 4 (ICS_C4) 195
12.3.5 ICS Status Register (ICS_S) 196
12.4 Functional description 197
12.4.1 Operational modes 197
12.4.1.1 FLL engaged internal (FEI) 197
12.4.1.2 FLL engaged external (FEE) 198
12.4.1.3 FLL bypassed internal (FBI) 198
12.4.1.4 FLL bypassed internal low power (FBILP) 198
12.4.1.5 FLL bypassed external (FBE) 199
12.4.1.6 FLL bypassed external low power (FBELP) 199
12.4.1.7 Stop 199
12.4.2 Mode switching 199
12.4.3 Bus frequency divider 200
12.4.4 Low-power field usage 200
12.4.5 Internal reference clock 200
12.4.6 Fixed frequency clock 201
12.4.7 FLL lock and clock monitor 201

12.4.7.1 FLL clock lock ... 201
12.4.7.2 External reference clock monitor ... 202
12.5 Initialization/application information ... 202
12.5.1 Initializing FEI mode ... 202
12.5.2 Initializing FBI mode ... 202
12.5.3 Initializing FEE mode ... 203
12.5.4 Initializing FBE mode ... 203

Chapter 13
Modulo Timer (MTIM)

13.1 Chip specific modulo timer ... 205
13.2 Introduction ... 205
13.3 Features ... 206
13.3.1 Block Diagram ... 206
13.3.2 Modes of Operation ... 207
13.3.2.1 MTIM16 in Wait Mode ... 207
13.3.2.2 MTIM16 in Stop Modes ... 207
13.3.2.3 MTIM16 in Active Background Mode ... 208
13.4 External Signal Description ... 208
13.4.1 TCLK — External Clock Source Input into MTIM16 ... 208
13.5 Memory Map and Register Descriptions ... 209
13.5.1 MTIM16 status and control register (MTIM_SC) ... 209
13.5.2 MTIM16 clock configuration register (MTIM_CLK) ... 210
13.5.3 MTIM16 counter register high (MTIM_CNTH) ... 211
13.5.4 MTIM16 counter register low (MTIM_CNTL) ... 212
13.5.5 MTIM16 modulo register high (MTIM_MODH) ... 213
13.5.6 MTIM16 modulo register low (MTIM_MODL) ... 214
13.6 Functional Description ... 214
13.6.1 MTIM16 Operation Example ... 216

Chapter 14

Power Management Controller (PMC)

14.1 Chip specific power management controller 217

14.2 Introduction..... 217

14.3 Features..... 218

14.4 Overview..... 218

14.5 Modes of operation..... 219

14.5.1 Reduced performance mode..... 219

14.5.2 Full performance mode..... 220

14.6 External signal description..... 220

14.6.1 VDD..... 220

14.6.2 VDDX..... 220

14.6.3 VREFH..... 220

14.6.4 VDDF..... 221

14.6.5 VDD1.8..... 221

14.7 Memory map and register definition..... 221

14.7.1 Control Register (PMC_CTRL)..... 222

14.7.2 Reset Flags Register (PMC_RST)..... 223

14.7.3 Temperature Control and Status Register (PMC_TPCTRLSTAT)..... 223

14.7.4 Temperature Offset Step Trim Register (PMC_TPTM)..... 224

14.7.5 RC Oscillator Offset Step Trim Register (PMC_RC20KTRM)..... 225

14.7.6 Low Voltage Control and Status Register 1 (system 5 V) (PMC_LVCTLSTAT1)..... 226

14.7.7 Low Voltage Control and Status Register 2 (V_{REFH}) (PMC_LVCTLSTAT2)..... 227

14.7.8 V_{REFH} Configuration Register (PMC_VREFHCFG)..... 227

14.7.9 VREFH Low Voltage Warning (LVW) Configuration Register (PMC_VREFHLVW)..... 228

14.7.10 Status Register (PMC_STAT)..... 228

14.8 Functional description..... 229

14.8.1 Voltage regulators..... 229

14.8.1.1 VREGVDDX..... 229

14.8.1.2 VREGVDDF..... 229

14.8.1.3 VREGVDD....230
14.8.1.4 VREGVREFH....230
14.8.2 Power-on reset....230
14.8.3 Low voltage reset (LVR)....230
14.8.3.1 LVR in low power mode....231
14.8.4 Low voltage warning (LVW)....231
14.8.4.1 LVW on VDDX/VDDA....231
14.8.4.2 LVW on VREFH....231
14.8.4.3 LVW in low power mode....231
14.8.5 High-accuracy reference voltage....232
14.8.6 Temperature sensor....232
14.8.6.1 High temperature warning....232
14.8.7 Low-power RC oscillator....233
14.9 Application information....233

Chapter 15
Keyboard Interrupts (KBI)

15.1 Chip specific KBI information....235
15.2 Introduction....235
15.2.1 Features....235
15.2.2 Modes of Operation....235
15.2.2.1 KBI in Wait mode....236
15.2.2.2 KBI in Stop modes....236
15.2.3 Block Diagram....236
15.3 External signals description....237
15.4 Register definition....237
15.5 Memory Map and Registers....237
15.5.1 KBI Status and Control Register (KBI_SC)....238
15.5.2 KBI Pin Enable Register (KBI_PE)....239
15.5.3 KBI Edge Select Register (KBI_ES)....239

15.6 Functional Description....240
15.6.1 Edge-only sensitivity....240
15.6.2 Edge and level sensitivity....240
15.6.3 KBI Pullup Resistor....240
15.6.4 KBI initialization....241

Chapter 16
Cyclic redundancy check (CRC)

16.1 Chip specific cyclic redundancy check (CRC)....243
16.2 Introduction....243
16.2.1 Features....244
16.2.2 Block diagram....244
16.2.3 Modes of operation....244
16.2.3.1 Run mode....244
16.2.3.2 Low-power modes (Wait or Stop)....245
16.3 Memory map and register descriptions....245
16.3.1 CRC Data register: High 1 (CRC_DH1)....245
16.3.2 CRC Data register: High 0 (CRC_DH0)....246
16.3.3 CRC Data register: Low 1 (CRC_DL1)....247
16.3.4 CRC Data register: Low 0 (CRC_DL0)....247
16.3.5 CRC Polynomial Register: High 1 (CRC_PH1)....248
16.3.6 CRC Polynomial Register: High 0 (CRC_PH0)....249
16.3.7 CRC Polynomial Register: Low 1 (CRC_PL1)....249
16.3.8 CRC Polynomial Register: Low 0 (CRC_PL0)....250
16.3.9 CRC Control register (CRC_CTRL)....250
16.4 Functional description....251
16.4.1 CRC initialization/reinitialization....251
16.4.2 CRC calculations....252
16.4.2.1 16-bit CRC....252
16.4.2.2 32-bit CRC....252

16.4.3 Transpose feature ... 253
16.4.3.1 Types of transpose ... 253
16.4.4 CRC result complement ... 254

Chapter 17
Analog-to-digital converter (ADC)

17.1 Chip-specific ADC information ... 255
17.1.1 Analog-to-digital converter (ADC) ... 255
17.1.2 ADC channel assignments ... 256
17.1.3 ADC analog supply and reference connections ... 257
17.1.4 Alternate clock ... 257
17.1.5 Hardware trigger ... 258
17.1.6 Temperature sensor ... 258
17.2 Introduction ... 259
17.2.1 Features ... 259
17.2.2 Block Diagram ... 260
17.3 External Signal Description ... 260
17.3.1 Analog Power (VDDA) ... 261
17.3.2 Analog Ground (VSSA) ... 261
17.3.3 Voltage Reference High (VREFH) ... 261
17.3.4 Voltage Reference Low (VREFL) ... 261
17.3.5 Analog Channel Inputs (ADx) ... 261
17.4 ADC Control Registers ... 262
17.4.1 Status and Control Register 1 (ADC*x*_SC1) ... 262
17.4.2 Status and Control Register 2 (ADC*x*_SC2) ... 264
17.4.3 Status and Control Register 3 (ADC*x*_SC3) ... 265
17.4.4 Status and Control Register 4 (ADC*x*_SC4) ... 266
17.4.5 Conversion Result High Register (ADC*x*_RH) ... 267
17.4.6 Conversion Result Low Register (ADC*x*_RL) ... 268
17.4.7 Compare Value High Register (ADC*x*_CVH) ... 269

17.4.8 Compare Value Low Register (ADC*x*_CVL)....269
17.5 Functional description....270
17.5.1 Clock select and divide control....270
17.5.2 Hardware trigger....271
17.5.3 Conversion control....271
17.5.3.1 Initiating conversions....271
17.5.3.2 Completing conversions....272
17.5.3.3 Aborting conversions....272
17.5.3.4 Power control....273
17.5.3.5 Sample time and total conversion time....273
17.5.4 Automatic compare function....274
17.5.5 FIFO operation....275
17.5.6 MCU wait mode operation....279
17.5.7 MCU Stop mode operation....279
17.5.7.1 Stop mode with ADACK disabled....279
17.5.7.2 Stop mode with ADACK enabled....279
17.6 Initialization information....280
17.6.1 ADC module initialization example....280
17.6.1.1 Initialization sequence....280
17.6.1.2 Pseudo-code example....281
17.6.2 ADC FIFO module initialization example....281
17.6.2.1 Pseudo-code example....282
17.7 Application information....283
17.7.1 External pins and routing....283
17.7.1.1 Analog supply pins....283
17.7.1.2 Analog reference pins....283
17.7.1.3 Analog input pins....284
17.7.2 Sources of error....285
17.7.2.1 Sampling error....285

17.7.2.2 Pin leakage error..... 285
17.7.2.3 Noise-induced errors..... 285
17.7.2.4 Code width and quantization error..... 286
17.7.2.5 Linearity errors..... 287
17.7.2.6 Code jitter, non-monotonicity, and missing codes..... 287

Chapter 18
Chip-specific ACMP information

18.1 CMP configuration information..... 289
18.2 ACMP in stop mode..... 290
18.3 0
18.3.1 0
18.3.2 0
18.4 Introduction..... 290
18.5 CMP Features..... 291
18.6 6-bit DAC Key Features..... 291
18.7 ANMUX Key Features..... 292
18.8 CMP, DAC, and ANMUX Diagram..... 292
18.9 CMP Block Diagram..... 293
18.10 Memory Map/Register Definitions..... 295
18.10.1 CMP Control Register 0 (CMP_CR0)..... 295
18.10.2 CMP Control Register 1 (CMP_CR1)..... 296
18.10.3 CMP Filter Period Register (CMP_FPR)..... 297
18.10.4 CMP Status and Control Register (CMP_SCR)..... 298
18.10.5 DAC Control Register (CMP_DACCR)..... 299
18.10.6 MUX Control Register (CMP_MUXCR)..... 300
18.10.7 MUX Pin Enable Register (CMP_MUXPE)..... 301
18.11 CMP Functional Description..... 301
18.11.1 CMP Functional Modes..... 301
18.11.1.1 Disabled Mode (# 1)..... 303

18.11.1.2 Continuous Mode (#s 2A & 2B)..... 303
18.11.1.3 Sampled, Non-Filtered Mode (#s 3A & 3B)..... 304
18.11.1.4 Sampled, Filtered Mode (#s 4A & 4B)..... 306
18.11.1.5 Windowed Mode (#s 5A & 5B)..... 308
18.11.1.6 Windowed/Resampled Mode (# 6)..... 310
18.11.1.7 Windowed/Filtered Mode (#7)..... 310
18.11.2 Power Modes..... 311
18.11.2.1 Wait Mode Operation..... 311
18.11.2.2 Stop Mode Operation..... 311
18.11.2.3 Background Debug Mode Operation..... 312
18.11.3 Startup and Operation..... 312
18.11.4 Low Pass Filter..... 313
18.11.4.1 Enabling Filter Modes..... 313
18.11.4.2 Latency Issues..... 314
18.12 CMP Interrupts..... 315
18.13 Digital to Analog Converter Block Diagram..... 315
18.14 DAC Functional Description..... 316
18.14.1 Voltage Reference Source Select..... 316
18.15 DAC Resets..... 316
18.16 DAC Clocks..... 316
18.17 DAC Interrupts..... 316

Chapter 19
FlexTimer Module (FTM)

19.1 Chip specific FlexTimer module..... 317
19.2 Introduction..... 318
19.2.1 FlexTimer philosophy..... 318
19.2.2 Features..... 318
19.2.3 Modes of operation..... 319
19.2.4 Block diagram..... 319

19.3 Signal description....320
19.3.1 EXTCLK — FTM external clock....321
19.3.2 CHn — FTM channel (n) I/O pin....321
19.4 Memory map and register definition....321
19.4.1 Module memory map....321
19.4.2 Register descriptions....321
19.4.3 Status and Control (FTM*x*_SC)....322
19.4.4 Counter High (FTM*x*_CNTH)....323
19.4.5 Counter Low (FTM*x*_CNTL)....324
19.4.6 Modulo High (FTM*x*_MODH)....324
19.4.7 Modulo Low (FTM*x*_MODL)....325
19.4.8 Channel Status and Control (FTM*x*_C*n*SC)....326
19.4.9 Channel Value High (FTM*x*_C*n*VH)....327
19.4.10 Channel Value Low (FTM*x*_C*n*VL)....328
19.5 Functional Description....329
19.5.1 Clock Source....329
19.5.1.1 Counter Clock Source....329
19.5.2 Prescaler....330
19.5.3 Counter....330
19.5.3.1 Up counting....331
19.5.3.2 Up-down counting....331
19.5.3.3 Free running counter....332
19.5.3.4 Counter reset....332
19.5.4 Input capture mode....332
19.5.5 Output compare mode....333
19.5.6 Edge-aligned PWM (EPWM) mode....335
19.5.7 Center-aligned PWM (CPWM) mode....336
19.5.8 Update of the registers with write buffers....338
19.5.8.1 MODH:L registers....338

19.5.8.2 CnVH:L registers ... 339
19.5.9 BDM mode ... 339
19.6 Reset overview ... 339
19.7 FTM Interrupts ... 341
19.7.1 Timer overflow interrupt ... 341
19.7.2 Channel (n) interrupt ... 341

Chapter 20
Pules Width Timer (PWT)

20.1 Chip specific pules width timer ... 343
20.2 Introduction ... 344
20.2.1 Features ... 344
20.2.2 Modes of operation ... 344
20.2.3 Block diagram ... 345
20.3 External signal description ... 345
20.3.1 Overview ... 345
20.3.2 PWTIN[3:0] — pulse width timer capture inputs ... 346
20.3.3 ALTCLK— alternative clock source for counter ... 346
20.4 Memory Map and Register Descriptions ... 346
20.4.1 Pulse Width Timer Control and Status Register (PWT*x*_CS) ... 347
20.4.2 Pulse Width Timer Control Register (PWT*x*_CR) ... 348
20.4.3 Pulse Width Timer Positive Pulse Width Register: High (PWT*x*_PPH) ... 349
20.4.4 Pulse Width Timer Positive Pulse Width Register: Loq (PWT*x*_PPL) ... 350
20.4.5 Pulse Width Timer Negative Pulse Width Register: High (PWT*x*_NPH) ... 350
20.4.6 Pulse Width Timer Negative Pulse Width Register: Low (PWT*x*_NPL) ... 351
20.4.7 Pulse Width Timer Counter Register: High (PWT*x*_CNTH) ... 351
20.4.8 Pulse Width Timer Counter Register: Low (PWT*x*_CNTL) ... 351
20.5 Functional description ... 352
20.5.1 PWT counter and PWT clock pre-scaler ... 352
20.5.2 Edge detection and capture control ... 352

20.6 Reset overview....356
20.6.1 Description of reset operation....356
20.7 Interrupts....357
20.7.1 Description of interrupt operation....357
20.7.2 Application examples....358
20.8 Initialization/Application information....359

Chapter 21
Inter-Integrated Circuit (I2C)

21.1 Chip specific inter-integrated circuit....361
21.2 Introduction....361
21.2.1 Features....362
21.2.2 Modes of operation....362
21.2.3 Block diagram....363
21.3 I2C signal descriptions....364
21.4 Memory map/register definition....364
21.4.1 I2C Address Register 1 (I2C_A1)....365
21.4.2 I2C Frequency Divider register (I2C_F)....365
21.4.3 I2C Control Register 1 (I2C_C1)....366
21.4.4 I2C Status register (I2C_S)....368
21.4.5 I2C Data I/O register (I2C_D)....369
21.4.6 I2C Control Register 2 (I2C_C2)....370
21.4.7 I2C Stop Control and Status Register (I2C_SCS)....371
21.4.8 I2C Range Address register (I2C_RA)....372
21.4.9 I2C SMBus Control and Status register (I2C_SMB)....373
21.4.10 I2C Address Register 2 (I2C_A2)....374
21.4.11 I2C SCL Low Timeout Register High (I2C_SLTH)....375
21.4.12 I2C SCL Low Timeout Register Low (I2C_SLTL)....375
21.4.13 I2C Status register 2 (I2C_S2)....376
21.5 Functional description....376

21.5.1 I2C protocol ... 376
21.5.1.1 START signal ... 377
21.5.1.2 Slave address transmission ... 378
21.5.1.3 Data transfers ... 378
21.5.1.4 STOP signal ... 379
21.5.1.5 Repeated START signal ... 379
21.5.1.6 Arbitration procedure ... 379
21.5.1.7 Clock synchronization ... 380
21.5.1.8 Handshaking ... 380
21.5.1.9 Clock stretching ... 380
21.5.1.10 I2C divider and hold values ... 381
21.5.2 10-bit address ... 382
21.5.2.1 Master-transmitter addresses a slave-receiver ... 382
21.5.2.2 Master-receiver addresses a slave-transmitter ... 383
21.5.3 Address matching ... 383
21.5.4 System management bus specification ... 384
21.5.4.1 Timeouts ... 384
21.5.4.2 FAST ACK and NACK ... 386
21.5.5 Resets ... 387
21.5.6 Interrupts ... 387
21.5.6.1 Byte transfer interrupt ... 388
21.5.6.2 Address detect interrupt ... 388
21.5.6.3 Stop Detect Interrupt ... 388
21.5.6.4 Exit from low-power/stop modes ... 388
21.5.6.5 Arbitration lost interrupt ... 388
21.5.6.6 Timeout interrupt in SMBus ... 389
21.5.7 Address matching wake-up ... 389
21.5.8 Double buffering mode ... 390
21.6 Initialization/application information ... 391

Chapter 22
Serial Communications Interface (SCI)

22.1 Chip specific serial communications interface 395

22.2 Introduction 396

22.2.1 Features 396

22.2.2 Modes of operation 396

22.2.3 Block diagram 397

22.3 SCI signal descriptions 399

22.3.1 Detailed signal descriptions 399

22.4 Register definition 399

22.4.1 SCI Baud Rate Register: High (SCI*x*_BDH) 400

22.4.2 SCI Baud Rate Register: Low (SCI*x*_BDL) 401

22.4.3 SCI Control Register 1 (SCI*x*_C1) 401

22.4.4 SCI Control Register 2 (SCI*x*_C2) 403

22.4.5 SCI Status Register 1 (SCI*x*_S1) 404

22.4.6 SCI Status Register 2 (SCI*x*_S2) 406

22.4.7 SCI Control Register 3 (SCI*x*_C3) 407

22.4.8 SCI Data Register (SCI*x*_D) 409

22.5 Functional description 409

22.5.1 Baud rate generation 410

22.5.2 Transmitter functional description 410

22.5.2.1 Send break and queued idle 411

22.5.3 Receiver functional description 412

22.5.3.1 Data sampling technique 413

22.5.3.2 Receiver wake-up operation 414

22.5.4 Interrupts and status flags 415

22.5.5 Baud rate tolerance 416

22.5.5.1 Slow data tolerance 416

22.5.5.2 Fast data tolerance 418

22.5.6 Additional SCI functions..... 419
22.5.6.1 8- and 9-bit data modes..... 419
22.5.6.2 Stop mode operation..... 419
22.5.6.3 Loop mode..... 419
22.5.6.4 Single-wire operation..... 420

Chapter 23
Programmable Delay Block (PDB)

23.1 Chip specific programmable delay block..... 421
23.2 Introduction..... 422
23.3 Features..... 422
23.4 Block diagram..... 422
23.5 Mode of operation..... 423
23.5.1 Single shot delay mode..... 423
23.5.2 Continuous count mode..... 424
23.6 Memory Map and Register Descriptions..... 425
23.6.1 PDB Control Register 0 (PDB_CTRL0)..... 425
23.6.2 PDB Control Register 1 (PDB_CTRL1)..... 426
23.6.3 PDB0 Comparison Low Register (PDB_CMPL0)..... 427
23.6.4 PDB0 Comparison High Register (PDB_CMPH0)..... 428
23.6.5 PDB0 Counter High/Low (PDB_CNT0)..... 428
23.6.6 PDB1 Comparison Low Register (PDB_CMPL1)..... 429
23.6.7 PDB1 Comparison High Register (PDB_CMPH1)..... 429
23.6.8 PDB1 Counter High/Low (PDB_CNT1)..... 430

Chapter 24
Inter-peripheral Crossbar Switch (XBAR)

24.1 Introduction..... 431
24.2 Features..... 431
24.3 Block diagram..... 431
24.4 Memory Map and Register Descriptions..... 432

24.4.1 External Mux Selection Register (XBAR_EXTMUX)..... 433
24.4.2 XBAR Selection Register (XBAR_SEL*n*)..... 434

Chapter 25
Gate Drive Unit (GDU)

25.1 Chip specific GDU information.....435
25.2 Introduction.....435
25.3 Features.....436
25.4 Block diagram.....436
25.5 Modes of operation..... 437
25.6 Memory map and register definition.....438
25.6.1 PHCMP0 Control Register 0 (GDU_PHCMP0CR0)..... 439
25.6.2 PHCMP0 Control Register 1 (GDU_PHCMP0CR1)..... 439
25.6.3 PHCMP0 Filter Period Register (GDU_PHCMP0FPR).....441
25.6.4 PHCMP0 Status and Control Register (GDU_PHCMP0SCR).....441
25.6.5 PHCMP1 Control Register 0 (GDU_PHCMP1CR0)..... 442
25.6.6 PHCMP1 Control Register 1 (GDU_PHCMP1CR1)..... 443
25.6.7 PHCMP1 Filter Period Register (GDU_PHCMP1FPR).....444
25.6.8 PHCMP1 Status and Control Register (GDU_PHCMP1SCR).....445
25.6.9 PHCMP2 Control Register 0 (GDU_PHCMP2CR0)..... 446
25.6.10 PHCMP2 Control Register 1 (GDU_PHCMP2CR1)..... 446
25.6.11 PHCMP2 Filter Period Register (GDU_PHCMP2FPR).....448
25.6.12 PHCMP2 Status and Control Register (GDU_PHCMP2SCR).....448
25.6.13 Clamp Control Register (GDU_CLMPCTRL)..... 449
25.6.14 I/O Control Register (GDU_IOCTRL)..... 450
25.6.15 Virtual Network Phase Detection Control (GDU_PHASECTRL)..... 451
25.6.16 Current Sensor and Overcurrent Protection Control Register (GDU_CURCTRL).....452
25.6.17 LIMIT0 CMP Control Register 0 (GDU_LIMIT0CR0).....452
25.6.18 LIMIT0 CMP Control Register 1 (GDU_LIMIT0CR1).....453
25.6.19 LIMIT0 CMP Filter Period Register (GDU_LIMIT0FPR)..... 454

25.6.20 LIMIT0 CMP Status and Control Register (GDU_LIMIT0SCR)....455
25.6.21 LIMIT0 DAC Control Register (GDU_LIMIT0DACCR)....456
25.6.22 LIMIT1 CMP Control Register 0 (GDU_LIMIT1CR0)....456
25.6.23 LIMIT1 CMP Control Register 1 (GDU_LIMIT1CR1)....457
25.6.24 LIMIT1 CMP Filter Period Register (GDU_LIMIT1FPR)....458
25.6.25 LIMIT1 CMP Status and Control Register (GDU_LIMIT1SCR)....459
25.6.26 LIMIT1 DAC Control Register (GDU_LIMIT1DACCR)....460
25.6.27 PDCS and Clamp Status Register (GDU_STATREG)....460
25.6.28 LIMIT CMP BIAS Register (GDU_SIGBIAS)....461
25.7 Functional description....461
25.7.1 Phase detection function descriptions....462
25.7.1.1 Phase detection diagram....462
25.7.1.2 Phase detection descriptions....462
25.7.2 OpAMP function descriptions....463
25.7.2.1 OpAMP diagram....463
25.7.2.2 OpAMP descriptions....464
25.7.3 Predrive function descriptions....464
25.7.3.1 Predrive diagram....464
25.7.3.2 Predrive descriptions....465
25.7.4 GCMP functional description....465
25.7.4.1 GCMP diagram....465
25.7.4.2 GCMP block diagram....466
25.7.4.3 GCMP functional modes....468
25.7.4.4 Power modes....477
25.7.4.5 Startup and operation....478
25.7.4.6 Low pass filter....478
25.8 GCMP interrupts....480

Chapter 26
Pulse Width Modulator (PWM)

26.1 Chip specific pulse width modulator 481
26.2 Introduction 482
26.2.1 Overview 482
26.2.2 Features 482
26.2.3 Modes of operation 483
26.2.4 Block diagram 483
26.3 Functional description 485
26.3.1 Prescaler 485
26.3.2 Generator 485
26.3.2.1 Alignment and compare output polarity 486
26.3.2.2 Period 487
26.3.2.3 Pulse width duty cycle 487
26.3.3 Independent or complementary channel operation 489
26.3.4 Deadtime generators 491
26.3.5 Asymmetric PWM output 493
26.3.6 Variable edge placement PWM output 493
26.3.7 PWM output polarity 494
26.3.8 Generator loading 497
26.3.8.1 Load enable 497
26.3.8.2 Load frequency 497
26.3.8.3 Reload flag 498
26.3.8.4 Initialization 500
26.3.9 Fault protection 501
26.3.9.1 Fault pin filter 502
26.3.9.2 Automatic fault clearing 503
26.3.9.3 Manual fault clearing 503
26.4 Memory Map and Register Descriptions 504
26.4.1 PWM Control Register: Low (PWM_CTRLL) 506
26.4.2 PWM Control Register: High (PWM_CTRLH) 507

26.4.3 PWM Fault Control Register: Low (PWM_FCTRLL)....508
26.4.4 PWM Fault Control Register: High (PWM_FCTRLH)....510
26.4.5 PWM Fault Status Acknowledge Register: Low (PWM_FLTACKL)....510
26.4.6 PWM Fault Status Acknowledge Register: High (PWM_FLTACKH)....511
26.4.7 PWM Output Control Register: Low (PWM_OUTL)....513
26.4.8 PWM Output Control Register: High (PWM_OUTH)....514
26.4.9 PWM Counter Register: Low (PWM_CNTRL)....514
26.4.10 PWM Counter Register: High (PWM_CNTRH)....515
26.4.11 PWM Counter Register: Low (PWM_CMODL)....515
26.4.12 PWM Counter Register: High (PWM_CMODH)....516
26.4.13 PWM Value Register: Low (PWM_VAL*n*L)....516
26.4.14 PWM Value Register: High (PWM_VAL*n*H)....517
26.4.15 PWM Deadtime Register: Low (PWM_DTIM*n*L)....518
26.4.16 PWM Deadtime Register: High (PWM_DTIM*n*H)....518
26.4.17 PWM Disable Mapping Registers 1: Low (PWM_DMAP1L)....519
26.4.18 PWM Disable Mapping Registers 1: High (PWM_DMAP1H)....520
26.4.19 PWM Disable Mapping Registers 2: Low (PWM_DMAP2L)....520
26.4.20 PWM Configure Register: Low (PWM_CNFGL)....520
26.4.21 PWM Configure Register: High (PWM_CNFGH)....521
26.4.22 PWM Channel Control Register: Low (PWM_CCTRLL)....522
26.4.23 PWM Channel Control Register: High (PWM_CCTRLH)....524
26.4.24 PWM Pulse Edge Control Register: Low (PWM_PECTRLL)....524
26.4.25 PWM Compare Invert Register: High (PWM_CINVH)....525
26.5 Resets....526
26.6 Clocks....526
26.7 Interrupts....527

Chapter 27
Development support

27.1 Introduction....529

27.1.1 Forcing active background....529
27.1.2 Features....529
27.2 Background debug controller (BDC)....530
27.2.1 BKGD pin description....531
27.2.2 Communication details....532
27.2.3 BDC commands....534
27.2.4 BDC hardware breakpoint....537
27.3 On-chip debug system (DBG)....537
27.3.1 Comparators A and B....538
27.3.2 Bus capture information and FIFO operation....538
27.3.3 Change-of-flow information....539
27.3.4 Tag vs. force breakpoints and triggers....540
27.3.5 Trigger modes....541
27.3.6 Hardware breakpoints....542
27.4 Memory map and register description....543
27.4.1 BDC Status and Control Register (BDC_SCR)....543
27.4.2 BDC Breakpoint Match Register: High (BDC_BKPTH)....545
27.4.3 BDC Breakpoint Register: Low (BDC_BKPTL)....546
27.4.4 System Background Debug Force Reset Register (BDC_SBDFR)....546

Chapter 28
Debug module (DBG)

28.1 Introduction....549
28.1.1 Features....549
28.1.2 Modes of operation....550
28.1.3 Block diagram....550
28.2 Signal description....551
28.3 Memory map and registers....551
28.3.1 Debug Comparator A High Register (DBG_CAH)....552
28.3.2 Debug Comparator A Low Register (DBG_CAL)....553

28.3.3 Debug Comparator B High Register (DBG_CBH)....554
28.3.4 Debug Comparator B Low Register (DBG_CBL)....554
28.3.5 Debug Comparator C High Register (DBG_CCH)....555
28.3.6 Debug Comparator C Low Register (DBG_CCL)....556
28.3.7 Debug FIFO High Register (DBG_FH)....556
28.3.8 Debug FIFO Low Register (DBG_FL)....557
28.3.9 Debug Comparator A Extension Register (DBG_CAX)....558
28.3.10 Debug Comparator B Extension Register (DBG_CBX)....559
28.3.11 Debug Comparator C Extension Register (DBG_CCX)....560
28.3.12 Debug FIFO Extended Information Register (DBG_FX)....561
28.3.13 Debug Control Register (DBG_C)....561
28.3.14 Debug Trigger Register (DBG_T)....562
28.3.15 Debug Status Register (DBG_S)....564
28.3.16 Debug Count Status Register (DBG_CNT)....565
28.4 Functional description....566
28.4.1 Comparator....566
28.4.1.1 RWA and RWAEN in full modes....566
28.4.1.2 Comparator C in loop1 capture mode....566
28.4.2 Breakpoints....567
28.4.2.1 Hardware breakpoints....567
28.4.3 Trigger selection....568
28.4.4 Trigger break control (TBC)....568
28.4.4.1 Begin- and end-trigger....569
28.4.4.2 Arming the DBG module....569
28.4.4.3 Trigger modes....570
28.4.5 FIFO....572
28.4.5.1 Storing data in FIFO....573
28.4.5.2 Storing with begin-trigger....573
28.4.5.3 Storing with end-trigger....573

28.4.5.4 Reading data from FIFO..... 573
28.4.6 Interrupt priority..... 574
28.5 Resets..... 575

Chapter 1
About This Document

1.1 Overview

1.1.1 Purpose

This document describes the features, architecture, and programming model of the NXP microcontrollers.

1.1.2 Audience

A reference manual is primarily for system architects and software application developers who are using or considering using an NXP product in a system.

1.2 Conventions

1.2.1 Numbering systems

The following suffixes identify different numbering systems:

This suffix	Identifies a
b	Binary number. For example, the binary equivalent of the number 5 is written 101b. In some cases, binary numbers are shown with the prefix *0b*.
d	Decimal number. Decimal numbers are followed by this suffix only when the possibility of confusion exists. In general, decimal numbers are shown without a suffix.
h	Hexadecimal number. For example, the hexadecimal equivalent of the number 60 is written 3Ch. In some cases, hexadecimal numbers are shown with the prefix *0x*.

1.2.2 Typographic notation

The following typographic notation is used throughout this document:

Example	Description
placeholder, x	Items in italics are placeholders for information that you provide. Italicized text is also used for the titles of publications and for emphasis. Plain lowercase letters are also used as placeholders for single letters and numbers.
`code`	Fixed-width type indicates text that must be typed exactly as shown. It is used for instruction mnemonics, directives, symbols, subcommands, parameters, and operators. Fixed-width type is also used for example code. Instruction mnemonics and directives in text and tables are shown in all caps; for example, BSR.
SR[SCM]	A mnemonic in brackets represents a named field in a register. This example refers to the Scaling Mode (SCM) field in the Status Register (SR).
REVNO[6:4], XAD[7:0]	Numbers in brackets and separated by a colon represent either: • A subset of a register's named field For example, REVNO[6:4] refers to bits 6–4 that are part of the COREREV field that occupies bits 6–0 of the REVNO register. • A continuous range of individual signals of a bus For example, XAD[7:0] refers to signals 7–0 of the XAD bus.

1.2.3 Special terms

The following terms have special meanings:

Term	Meaning
asserted	Refers to the state of a signal as follows: • An active-high signal is asserted when high (1). • An active-low signal is asserted when low (0).
deasserted	Refers to the state of a signal as follows: • An active-high signal is deasserted when low (0). • An active-low signal is deasserted when high (1). In some cases, deasserted signals are described as *negated*.
reserved	Refers to a memory space, register, or field that is either reserved for future use or for which, when written to, the module or chip behavior is unpredictable.

Chapter 2
Introduction

2.1 Introduction

This device is a low-cost, high-performance UHV HCS08 8-bit microcontroller units (MCUs). It uses the enhanced S08L central processor unit with MOSFET predrivers.

The features are as follows:

- Core platform clock up to 40 MHz, bus clock up to 20 MHz
- Memory option is up to 16 KB flash with 8 B flash line buffer, 768 B RAM (256 B is unrestricted, 512 B is restricted during flash erasing and programming) and 8 B register file
- Wide operating voltage ranges from 4.5–18 V
- 24-pin QFN package
- Ambient operating temperature ranges from –40 °C to 105 °C

2.2 Module functional categories

The modules on this device are grouped into functional categories. Information found here describes the modules assigned to each category in more detail.

Table 2-1. Module functional categories

Module category	Description
HCS08 core	• 8-bit MCU core, 40 MHz CPU frequency.
System	• System integration module • Power management and mode controllers • Multiple power modes available based on run, wait, stop, and power-down modes • Interrupt priority controller (IPC) • Inter-module crossbar
Memories	• Up to 16 KB flash memory for SU16 and up to 8 KB flash memory for SU8

Table continues on the next page...

Table 2-1. Module functional categories (continued)

Module category	Description
	• Up to 768 bytes SRAM, including 256 B of which is unrestricted, and the rest 512 B is restricted during flash erasing and programming. • Up to 8 bytes system register file

Clocks	• Internal clock source
	  • 31.25 to 39.0625 kHz IRC
	  • 20 kHz oscillator with functional in all power modes
	  • Frequency-locked loop
	• DC to 40 MHz

Security	• Windowed COP watchdog • One cyclic redundancy check (CRC)
Analog	• Two 12-bit analog-to-digital converters (ADC) • One high speed comparator (CMP) with internal 6-bit digital-to-analog converter (DAC) • One gate drive unit (GDU)
Timers	• One 16-bit PWM • One 16-bit FTM • One 16-bit modulo timer (MTIM) • Two pulse width timers (PWT) • One programmable delay block (PDB)
Communications	• One inter-integrated circuit (I^2C) module • One serial communications interface (SCI)
Human-Machine Interfaces (HMI)	• Port control (PORT) • One keyboard interrupt (KBI)

2.2.1 S08L core modules

The following core modules are available on this device.

Table 2-2. Core modules

Module	Description
HCS08 V6 CPU	The HCS08 CPU is fully source- and object-code-compatible with the M68HC08 CPU. Several instructions and enhanced addressing modes were added to improve C compiler efficiency and to support a new background debug system which replaces the monitor mode of earlier M68HC08 microcontrollers.
Debug module	The DBG module implements an on-chip ICE (in-circuit emulation) system and allows non-intrusive debug of application software by providing an on-chip trace buffer with flexible triggering capability. The trigger can also provide extended breakpoint capacity. The on-chip ICE system is optimized for the HCS08 8-bit architecture and supports 64 KB of memory space.

2.2.2 System modules

The following system modules are available on this device.

Table 2-3. System modules

Module	Description
System integration module (SIM)	The SIM includes integration logic and several module configuration settings.
Power management controller (PMC)	The PMC provides the user with multiple power options. Multiple modes are supported that allow the user to optimize power consumption for the level of functionality needed. Includes power-on-reset (POR) and integrated low voltage detect (LVD) with reset (brownout) capability and selectable LVD trip points.
Development support	Development supports single-wire background debug mode (BDM), which uses the on-chip background debug controller (BDC) module, and the independent on-chip realtime in-circuit emulation (ICE) system, which uses the on-chip debug (DBG) module.
Debug module (DBG)	The DBG module implements an on-chip ICE (in-circuit emulation) system and allows non-intrusive debug of application software by providing an on-chip trace buffer with flexible triggering capability.

2.2.3 Memories and memory interfaces

The following memories and memory interfaces are available on this device.

Table 2-4. Memories and memory interfaces

Module	Description
Flash memory	Program flash memory — up to 16 KB (SU16) or 8 KB (SU8) of the non-volatile flash memory that can execute program code.
SRAM	Up to 768 bytes internal system RAM, including 256 B of which is unrestricted, the rest 512 B is restricted during flash erasing and programming.
System register file	Up to 8 bytes.

2.2.4 Clocks

The following clock modules are available on this device.

Table 2-5. Clock modules

Module	Description
Internal Clock Source (ICS)	ICS module containing an internal reference clock (ICSIRCLK) and a frequency locked loop (FLL).
Low-Power Oscillator (LPO)	The PMC module contains a 20 kHz low-power oscillator which acts as a standalone low-frequency clock source in all modes.

2.2.5 Security and integrity modules

The following security and integrity modules are available on this device:

Table 2-6. Security and integrity modules

Module	Description
Windowed COP watchdog (WCOP)	The COP watchdog is used to force a system reset when the application software fails to execute as expected.
Cyclic Redundancy Check (CRC)	The cyclic redundancy check (CRC) module generates 16/32-bit CRC code for error detection.

2.2.6 Analog modules

The following analog modules are available on this device:

Table 2-7. Analog modules

Module	Description
Analog-to-digital converters (ADC)	12-bit successive-approximation ADC module.
Comparator (CMP)	One comparator that compares two analog input voltages across the full range of the supply voltage and can trigger an ADC acquisition, TPM update, or CPU interrupt.
6-bit digital-to-analog converters (DAC)	64-tap resistor ladder network which provides a selectable voltage reference for comparator.
Gate Drive Unit (GDU)	• 5 V voltage regulator referenced to V_{DD} for HS pre-drivers • Supports wide operation voltages from 4.5 V to 18 V • Overvoltage detection on the system supply VBUS pin • 1/8 V_{DD} drive to ADC

2.2.7 Timer modules

The following timer modules are available on this device:

Table 2-8. Timer modules

Module	Description
FlexTimer Module (FTM)	• Selectable FTM source clock, programmable prescaler • 16-bit counter supporting free-running, and counting is up or up-down • Input capture, output compare, and edge-aligned and center-aligned PWM modes • Operation of FTM channels as pairs with equal outputs or independent channels with independent outputs • Programmable interrupt on input capture, reference compare, overflowed counter

Table continues on the next page...

Table 2-8. Timer modules (continued)

Module	Description
Programmable Delay Block (PDB)	• 16-bit resolution with a shared prescaler • Support software and hardware trigger • Support continuous count mode or single shot delay mode • Supply on-fly delay value update • Selective output mode: logic level or one-shot pulse
Pulse Width Timer (PWT)	• Automatic measurement of pulse width with 16-bit resolution • Separate positive and negative pulse width measurements • Programmable triggering edge for starting measurement • Programmable measuring time between successive alternating edges, rising edges or falling edges • Programmable pre-scaler from clock input as 16-bit counter time base • Two selectable clock sources • Four selectable pulse inputs • Programmable interrupt generation upon pulse width value updated and counter overflow
Pulse width modulator (PWM)	• PWM operation clock runs at system clock • Six PWM signals • Complementary channel operation • Edge- or center-aligned PWM signals • 15 bits of resolution • Half-cycle reload capability • Integral reload rates from 1 to 16 • Individual software controlled PWM output • Programmable fault protection • PWM compare output polarity control • PWM output polarity control • Write-protected registers
16-bit modulo timer (MTIM)	• 16-bit up-counter • Four software selectable clock sources for input to prescaler • Nine selectable clock prescale values • Modulo compare matched can be an output

2.2.8 Communication interfaces

The following communication interfaces are available on this device:

Table 2-9. Communication modules

Module	Description
Inter-integrated circuit (I2C)	Allows communication between a number of devices. Also supports the System Management Bus (SMBus) Specification, version 2.
Serial communications interface (SCI)	SCI is used to connect to the RS232 serial input/output port of a personal computer or workstation and communicate with other embedded controllers.

2.2.9 Human-machine interfaces

The following human-machine interfaces (HMI) are available on this device:

Table 2-10. HMI modules

Module	Description
Port Control (PORT)	Some general purpose input or output pins are capable of interrupt request generation.
Keyboard Interrupts (KBI)	• Up to eight keyboard interrupt pins with individual pin enable bits • Each keyboard interrupt pin is programmable • One software-enabled keyboard interrupt • Exit from low-power modes

2.3 MCU block diagram

The block diagram below shows the structure of the MCUs.

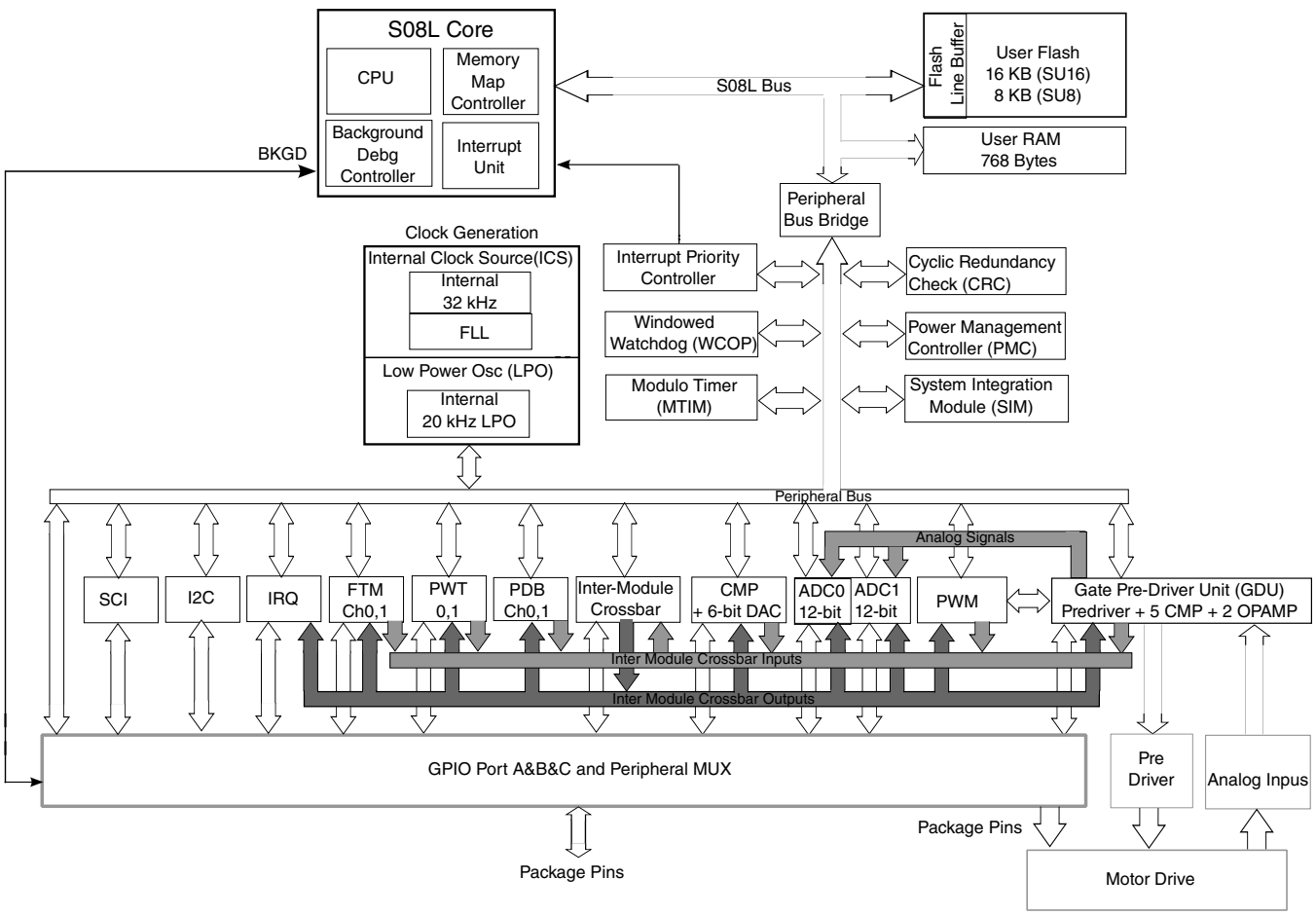


Figure 2-1. Block diagram

2.4 Orderable part numbers

The following table summarizes the part numbers of the devices covered by this document.

Table 2-11. Orderable part numbers summary

Part number	CPU frequency	Pin count	Package	Total flash memory	RAM	Temperature range
MC9S08SU16VFK	40 MHz	24	QFN	16 KB	768 bytes	-40 to 105 °C
MC9S08SU8VFK	40 MHz	24	QFN	8 KB	768 bytes	-40 to 105 °C

Chapter 3
Memory

3.1 Memory map

In this device, the most frequently used registers are placed in the direct page, memory 0x0000 to 0x00FF. Also a portion of RAM is placed in the direct page to take advantage of the direct addressing mode of the CPU. All other register locations in the extended page are the same in terms of instruction cycle times.The placement of the other peripheral modules is not critical because CPU instructions for all other extended memory have the same timing requirements.

As shown below, on-chip memory in this device consists of RAM and flash program memory for nonvolatile data storage, plus I/O and control/status registers. The registers are divided into three groups: direct-page registers, high-page registers and nonvolatile registers.

The entire memory space are partitioned to:

- Direct-page registers: 0x0000–0x007F
- Random access memory: 0x0080–0x037F
- Unimplemented: 0x0380–0x17FF
- High-page registers: 0x1800–0x18FF
- Unimplemented: 0x1900–0xBFFF (SU16) and 0x1900–0xDFFF (SU8)
- Flash memory (SU16): 0xC000–0xFFFF
    - Flash array: 0xC000–0FFBF
    - Vector table: 0xFFC0–0xFFFF
- Flash memory (SU8): 0xE000–0xFFFF
    - Flash array: 0xE000–0FFBF
    - Vector table: 0xFFC0–0xFFFF

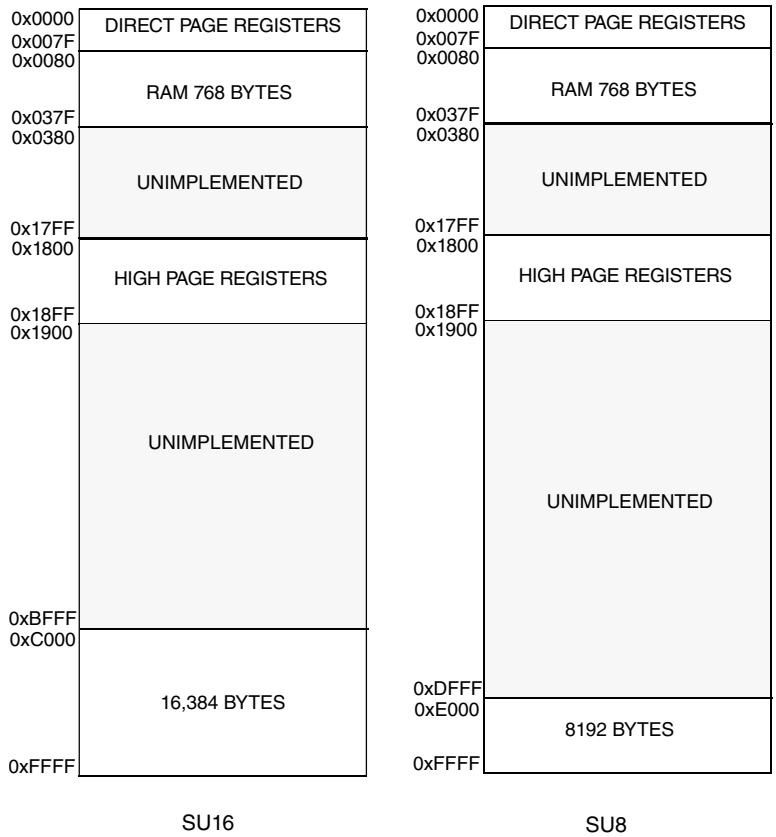


Figure 3-1. Memory map

3.2 Reset and interrupt vector assignments

The following table summarizes the reset and interrupt sources assignments for this device. The high-order byte of the address for the interrupt service routine is located at the first address in the vector address column, and the low-order byte of the address for the interrupt service routine is located at the next higher address. Higher-priority sources are located toward the bottom of the table. The Interrupt Priority Controller module is integrated on this device and provides the ability to alter the default priority scheme.

The vector names shown in this table are the labels used in the NXP-provided header files for the device.

Table 3-1. Reset and interrupt vectors

Vector number	Address (high/low)	Vector	Vector name	Module	Source
31	0xFFC0:FFC1	NVM	Vnvm	NVM	CCIF
30	0xFFC2:FFC3	I2C	Viic	I2C	Or'ed all flags
29	0xFFC4:FFC5	KBI	Vkbi	KBI	KBF

Table continues on the next page...

Table 3-1. Reset and interrupt vectors (continued)

Vector number	Address (high/low)	Vector	Vector name	Module	Source
28	0xFFC6:FFC7	MTIM overflag	Vmtim	MTIM	TOF
27	0xFFC8:FFC9	Lose of lock	Vics	ICS	LOLS
26	0xFFCA:FFCB	PDB successful compare	Vpdb	PDB	TCF0 TCF1
25	0xFFCC:FFCD	SCI transmit	Vscitx	SCI	TDRE TC
24	0xFFCE:FFCF	SCI receive	Vscirx	SCI	RDRF IDLE LBKDIF RXEDGIF
23	0xFFD0:FFD1	SCI error	Vscierr	SCI	OR NF FE PF
22	0xFFD2:FFD3	GDU_ACMPU_PH 2	Vgducmp2	GCMP2	CFRU CFFU
21	0xFFD4:FFD5	GDU_ACMPV_PH 1	Vgducmp1	GCMP1	CFRV CFFV
20	0xFFD6:FFD7	GDU_ACMPW_PH 0	Vgducmp0	GCMP0	CFRW CFFW
19	0xFFD8:FFD9	CMP	Vcmp	ACMP	CFR CFF
18	0xFFDA:FFDB	FTM overflow	Vtpmovf	FTM	TOF
17	0xFFDC:FFDD	FTM channel 1	Vtpmch1	FTM	CH1F
16	0xFFDE:FFDF	FTM channel 0	Vtpmch0	FTM	CH0F
15	0xFFE0:FFE1	Reserved	–	–	–
14	0xFFE2:FFE3	Reserved	–	–	–
13	0xFFE4:FFE5	GDU_AMP_OC	GDU	GDU	AMP_OC0 AMP_OC1
12	0xFFE6:FFE7	ADC1 conversion complete	Vadc1	ADC1	COCO
11	0xFFE8:FFE9	ADC0 conversion complete	Vadc0	ADC0	COCO
10	0xFFEA:FFEB	PWT1 overflow	Vpwt1ovf	PWT1	PWTOV
9	0xFFEC:FFED	PWT1 data ready	Vpwt1rdy	PWT1	PWTRDY
8	0xFFEE:FFEF	PWT0 overflow	Vpwt0ovf	PWT0	PWTOV
7	0xFFF0:FFF1	PWT0 data ready	Vpwt0rdy	PWT0	PWTRDY
6	0xFFF2:FFF3	PWM reload	Vmcpwm	mcPWM	PWMF

Table continues on the next page...

Table 3-1. Reset and interrupt vectors (continued)

Vector number	Address (high/low)	Vector	Vector name	Module	Source
5	0xFFF4:FFF5	PWM fault	Vmcpwmf	mcPWM	FFLAG0 FFLAG1 FFLAG2 FFLAG3
4	0xFFF6:FFF7	Over voltage warning	Vovw	GDU	OVWF
3	0xFFF8:FFF9	Low voltage warning PMC high temp warning	Vlvw	PMC	LVWF HTIF
2	0xFFFA:FFFB	Xbar_IRQ	Virq	Crossbar	IRQF
1	0xFFFC:FFFD	SWI	Vswi	Core	SWI instruction
0	0xFFFE:FFFF	Reset	Vreset	System Control	POR WCOP LVD LOC /Reset pin illegal opcode illegal address flash illegal access

3.3 Register addresses assignments

The register definitions vary in different memory sizes. The register addresses of unused peripherals are reserved. The following table shows the register availability of the devices.

The registers in the devices are divided into two groups:

- Direct-page registers are located in the first 128 locations in the memory map, so they can be accessed with efficient direct addressing mode instructions.
- High-page registers are used much less often, so they are located above 0x1800 in the memory map. This leaves room in the direct page for more frequently used registers and variables.

Direct-page registers can be accessed with efficient direct addressing mode instructions. which requires only the lower byte of the address. Bit manipulation instructions can be used to access any bit in a direct-page register.

Table 3-2. Peripheral registers availability

Address	Bytes	Peripheral	Peripheral registers	Comment
Direct Page Registers				
0x0000—0x0000	1	PTA	PORT_PTAD	Port data
0x0001—0x0001	1	PTB	PORT_PTBD	
0x0002—0x0002	1	PTC	PORT_PTCD	
0x0003—0x0003	1	PTA	PORT_PTADD	Port data direction
0x0004—0x0004	1	PTB	PORT_PTBDD	
0x0005—0x0005	1	PTC	PORT_PTCDD	
0x0006—0x0006	1	XBAR	XBAR_EXTMUX	PWM channel 2-to-1 Mux
0x0007—0x0007	1	–	Reserved	–
0x0008—0x000D	6	MTIM	MTIM_SC, MTIM_CLK, MTIM_CNTH, MTIM_CNTL, MTIM_MODH, MTIM_MODL	–
0x000E—0x00F	2	IPC	IPC_SC, IPC_IPMPS	–
0x0010—0x0017	8	ADC0	ADC0_SC1, ADC0_SC2, ADC0_SC3, ADC0_SC4, ADC0_RH, ADC0_RL, ADC0_CVH, ADC0_CVL	–
0x0018—0x001F	8	ADC1	ADC1_SC1, ADC1_SC2, ADC1_SC3, ADC1_SC4, ADC1_RH, ADC1_RL, ADC1_CVH, ADC1_CVL	–
0x0020—0x002B	12	GDU	GDU_PHCMP0CR0, GDU_PHCMP0CR1, GDU_PHCMP0FPR, GDU_PHCMP0SCR, GDU_PHCMP1CR0, GDU_PHCMP1CR1, GDU_PHCMP1FPR, GDU_PHCMP1SCR, GDU_PHCMP2CR0, GDU_PHCMP2CR1, GDU_PHCMP2FPR, GDU_PHCMP2SCR	–
0x002C—0x002F	4	–	Reserved	–
0x0030—0x0037	8	PWT0	PWT0_CS, PWT0_CR, PWT0_PPH, PWT0_PPL, PWT0_NPH, PWT0_NPL, PWT0_CNTH, PWT0_CNTL	–
0x0038—0x003F	8	PWT1	PWT1_CS, PWT1_CR, PWT1_PPH, PWT1_PPL, PWT1_NPH, PWT1_NPL, PWT1_CNTH, PWT1_CNTL	–
0x0040—0x005F	32	PWM	PWM_CTRLL, PWM_CTRLH, PWM_FCTRLL, PWM_FCTRLH, PWM_FLTACKL, PWM_FLTACKH,	–

Table continues on the next page...

Table 3-2. Peripheral registers availability (continued)

Address	Bytes	Peripheral	Peripheral registers	Comment
			PWM_OUTL, PWM_OUTH, PWM_CNTRL, PWM_CNTRH, PWM_CMODL, PWM_CMODH, PWM_VAL0L, PWM_VAL0H, PWM_VAL1L, PWM_VAL1H, PWM_VAL2L, PWM_VAL2H, PWM_VAL3L, PWM_VAL3H, PWM_VAL4L, PWM_VAL4H, PWM_VAL5L, PWM_VAL5H, PWM_DTIM0L, PWM_DTIM0H , PWM_DTIM1L, PWM_DTIM1H, PWM_DMAP1H, PWM_DMAP1L, PWM_DMAP2H,PWM_DMAP2L	
0x0060—0x0067	8	PDB	PDB_CTRL0, PDB_CTRL1, PDB_CMPL0, PDB_CMPH0, PDB_CNT0, PDB_CMPL1, PDB_CMPH1, PDB_CNT1	Control bit in PDB_CTRL decides read counter high or low
0x0068—0x006E	8	CMP	CMP_CR0, CMP_CR1, CMP_FPR, CMP_SCR, CMP_DACCR, CMP_MUXCR, CMP_MUXPE	–
0x006F—0x006F	1	–	Reserved	–
0x0070—0x007A	11	FTM0	FTM0_SC, FTM0_CNTH, FTM0_CNTL, FTM0_MODH, FTM0_MODL, FTM0_C0SC, FTM0_C0VH, FTM0_C0VL, FTM0_C1SC, FTM0_C1VH, FTM0_C1VL	–
0x007B—0x007B	1	–	Reserved	–
0x007C—0x007E	3	KBI	KBI_SC, KBI_PE, KBI_ES	Port A KBI
0x007F—0x007F	1	IRQ	IRQ_SC	IRQ from Xbar
High Page Registers				
0x1800—0x180F	16	SIM	SIM_SRS, SIM_SBDFR, SIM_SDIDH, SIM_SDIDL, SIM_SOPT1, SIM_SOPT2, SIM_MUXPTAL, SIM_MUXPTAH, SIM_MUXPTBL, SIM_MUXPTBH, SIM_MUXPTCL, SIM_SCGC1, SIM_SCGC2, SIM_SCGC3, SIM_SCDIV	–
0x1810—0x1817	8	SIM register filer	SIM_PORREG0, SIM_PORREG1, SIM_PORREG2, SIM_PORREG3, SIM_PORREG4, SIM_PORREG5, SIM_PORREG6, SIM_PORREG7	–
0x1818—0x181F	8	–	Reserved	–
0x1820—0x1829	10	PWM	PWM_CNFGL, PWM_CNFGH, PWM_CCTRLL, PWM_CCTRLH, PWM_PECTRLL, PWM_CINVH	–
0x182A—0x182F	6	–	Reserved	–

Table continues on the next page...

Table 3-2. Peripheral registers availability (continued)

Address	Bytes	Peripheral	Peripheral registers	Comment
0x1830—0x183C	16	FTMRH	FTMRH_FCLKDIV, FTMRH_FSEC, FTMRH_FCCOBIX, FTMRH_FCNFG, FTMRH_FSTAT, FTMRH_FPROT, FTMRH_FCCOBHI, FTMRH_FCCOBLO, FTMRH_FOPT	–
0x183D—0x1847	11	–	Reserved	–
0x1848—0x184F	8	ICS	ICS_C1, ICS_C2, ICS_C3, ICS_C4, ICS_S	–
0x1850—0x185F	16	PMC	PMC_CTRL, PMC_RST, PMC_TPCTRLSTAT, PMC_TPTM, PMC_RC20KTRM, PMC_LVCTLSTAT1, PMC_LVCTLSTAT2, PMC_VREFHCFG, PMC_VREFHLVW, PMC_STAT	–
0x1860—0x1867	8	IPC	IPC_ILRS0–IPC_ILRS7	–
0x1868—0x186F	8	SCI0	SCI0_BDH, SCI0_BDL, SCI0_C1, SCI0_C2, SCI0_S1, SCI0_S2, SCI0_C3, SCI0_D,	–
0x1868—0x186F	8	–	Reserved	–
0x1870—0x187F	16	GDU	GDU_CLMPCTRL, GDU_IOCTRL, GDU_PHASECTRL, GDU_CURCTRL, GDU_LIMIT0CR0, GDU_LIMIT0CR1, GDU_LIMIT0FPR, GDU_LIMIT0SCR, GDU_LIMIT0DACCR, GDU_LIMIT1CR0, GDU_LIMIT1CR1. GDU_LIMIT1FPR, GDU_LIMIT1SCR, GDU_LIMIT1DACCR, GDU_STATREG, GDU_SIGBIAS	–
0x1880—0x1881	2	ILLA	SIM_ILLAH, SIM_ILLAL	–
0x1882—0x188F	16	–	Reserved	–
0x1890—0x1898	9	CRC	CRC_DH1, CRC_DH0, CRC_DL1, CRC_DL0, CRC_PH1, CRC_PH0, CRC_PL1, CRC_PL0, CRC_CTRL	–
0x1899—0x18AF	23	–	Reserved	–
0x18B0—0x18BC	13	I2C	I2C_A1, I2C_F, I2C_C1, I2C_S1, I2C_D, I2C_C2, I2C_SCS, I2C_RA, I2C_SMB, I2C_A2, I2C_SLTH, I2C_SLTL, I2C_S2	–
0x18BD—0x18BF	3	–	Reserved	–
0x18C0—0x18CF	16	DBG	DBG_CAH, DBG_CAL, DBG_CBH, DBG_CBL, DBG_CCH, DBG_CCL, DBG_FH, DBG_FL, DBG_CAX, DBG_CBX, DBG_CCX, DBG_FX, DBG_C, DBG_T, DBG_S, DBG_CNT	–
0x18D0—0x18DF	16	XBAR	XBAR_SEL0—XBAR_SEL15	–

Table continues on the next page...

Table 3-2. Peripheral registers availability (continued)

Address	Bytes	Peripheral	Peripheral registers	Comment
0x18E0—0x18E0	1	PTA	PORT_PTAPE	Port pull-up enable
0x18E1—0x18E1	1	PTB	PORT_PTBPE	
0x18E2—0x18E2	1	PTC	PORT_PTCPE	
0x18E3—0x18E5	3	–	Reserved	–
0x18E6—0x18E6	1	PTB	PORT_PTBHD	Port high drive enable
0x18E7—0x18EB	5	–	Reserved	–
0x18EC—0x18EF	4	PTx	PORT_FCLKDIV, PORT_IOFLT0, PORT_IOFLT1, PORT_IOFLT2	Port Filter for pins
0x18F0—0x18F7	8	–	Reserved	–
0x18F8—0x18FF	8	SIM	SIM_UUID0—SIM_UUID7	64-bit unique ID

Several reserved flash memory locations, shown in the following table, are used for storing values used by several registers. These registers include an 8-byte backdoor key, NV_BACKKEY, which can be used to gain access to secure memory resources. During reset events, the contents of NVPROT and NVOPT in the reserved flash memory are transferred into corresponding FPROT and FOPT registers in the high-page registers area to control security and block protection options

Table 3-3. Reserved flash memory addresses

Address	Register Name	Bit 7	6	5	4	3	2	1	Bit 0
0xFF6E	NV_FTRIM	—	—	—	—	—	—	—	FTRIM
0xFF6F	NV_ICSTRM	TRIM							
0xFF70	NV_BACKKEY0	BACKKEY0							
0xFF71	NV_BACKKEY1	BACKKEY1							
0xFF72	NV_BACKKEY2	BACKKEY2							
0xFF73	NV_BACKKEY3	BACKKEY3							
0xFF74	NV_BACKKEY4	BACKKEY4							
0xFF75	NV_BACKKEY5	BACKKEY5							
0xFF76	NV_BACKKEY6	BACKKEY6							
0xFF77	NV_BACKKEY7	BACKKEY7							
0xFF78	Reserved	—	—	—	—	—	—	—	—
0xFF79	Reserved	—	—	—	—	—	—	—	—
0xFF7A	Reserved	—	—	—	—	—	—	—	—
0xFF7B	Reserved	—	—	—	—	—	—	—	—
0xFF7C	NV_FPROT	FPOPEN	—	FPHDIS	FPH		—	—	—
0xFF7D	Reserved	—	—	—	—	—	—	—	—
0xFF7E	NV_FOPT	NV							
0xFF7F	NV_FSEC	KEYEN		1	1	1	1	SEC	

The 8-byte comparison key can be used to temporarily disengage memory security provided the key enable field, NV_FSEC[KEYEN], is 10b. This key mechanism can be accessed only through user code running in secure memory. A security key cannot be entered directly through background debug commands. This security key can be disabled completely by programming the NV_FSEC[KEYEN] bit to 0b. If the security key is disabled, the only way to disengage security is by mass erasing the flash if needed, normally through the background debug interface and verifying that flash is blank. To avoid returning to secure mode after the next reset, program the security bits, NV_FSEC[SEC], to the unsecured state (10b).

3.4 Random-access memory (RAM)

The locations in RAM below 0x0100 can be accessed using the more efficient direct addressing mode. Any single bit in this area can be accessed with the bit manipulation instructions (BCLR, BSET, BRCLR, and BRSET).

The RAM retains data when the MCU is in low-power wait, or stop mode. At power-on, the contents of RAM are uninitialized. RAM data is unaffected by any reset provided that the supply voltage does not drop below the minimum value for RAM retention.

For compatibility with older M68HC05 MCUs, the HCS08 resets the stack pointer to 0x00FF. In this series, re-initialize the stack pointer to the top of the RAM so that the direct-page RAM can be used for frequently accessed RAM variables and bit-addressable program variables. Include the following 2-instruction sequence in your reset initialization routine (where RamLast is equated to the highest address of the RAM in the NXP-provided equate file).

```
LDHX     #RamLast+1    ;point one past RAM
TXS                    ;SP<-(H:X-1)
```

When security is enabled, the RAM is considered a secure memory resource and is not accessible through BDM or code executing from non-secure memory.

3.5 Flash memory

This device includes flash memory. The flash memory is ideal for single-supply applications that allow for field reprogramming without requiring external high voltage sources for program or erase operations. The flash module includes a memory controller that executes commands to modify flash memory contents. The user interface to the memory controller consists of the indexed flash common command object (FCCOB)

register, which is written to with the command, global address, data, and any required command parameters. The memory controller must complete the execution of a command before the FCCOB register is written to with a new command.

CAUTION

A flash byte or longword must be in the erased state before being programmed. Cumulative programming of bits within a flash byte or longword is not allowed.

The flash memory is read as two bytes per read. Read access time is one bus cycle for two bytes. For flash memory, an erased bit reads 1 and a programmed bit reads 0.

3.6 System register file

This device includes a 8-byte register file that is powered in all power modes.

Also, it retains contents during low-voltage detect (LVD) events and is only reset during a power-on reset.

Chapter 4
Interrupt

4.1 Interrupts

Interrupts save the current CPU status and registers, execute an interrupt service routine (ISR), and then restore the CPU status so that processing resumes where it left off before the interrupt. Other than the software interrupt (SWI), which is a program instruction, interrupts are caused by hardware events such as an edge on the IRQ pin or a timer-overflow event. The debug module can also generate an SWI under certain circumstances.

If an event occurs in an enabled interrupt source, an associated read-only status flag will be set. The CPU does not respond unless only the local interrupt enable is a logic 1. The CCR [I] is 0 to allow interrupts. The CCR [I] is initially set after reset that masks (prevents) all maskable interrupt sources. The user program initializes the stack pointer and performs other system setups before clearing the CCR [I] to allow the CPU to respond to interrupts.

When the CPU receives a qualified interrupt request, it completes the current instruction before responding to the interrupt. The interrupt sequence obeys the same cycle-by-cycle sequence as the SWI instruction and consists of:

- Saving the CPU registers on the stack.
- Setting the CCR [I] to mask further interrupts.
- Fetching the interrupt vector for the highest-priority interrupt that is currently pending.
- Filling the instruction queue with the first three bytes of program information starting from the address fetched from the interrupt vector locations.

While the CPU is responding to the interrupt, the CCR [I] is automatically set to prevent another interrupt from interrupting the ISR itself, which is called nesting of interrupts. Normally, the CCR [I] is restored to 0 when the CCR is restored from the value stacked

on entry to the ISR. In rare cases, the CCR [I] may be cleared inside an ISR, after clearing the status flag that generated the interrupt, so that other interrupts can be serviced without waiting for the first service routine to finish. This practice is recommended only for the most experienced programmers because it can lead to subtle program errors that are difficult to debug.

The interrupt service routine ends with a return-from-interrupt (RTI) instruction that restores the CCR, A, X, and PC registers to their pre-interrupt values by reading the previously saved information off the stack.

Note

For compatibility with the M68HC08, the H register is not automatically saved and restored. Push H onto the stack at the start of the interrupt service routine (ISR) and restore it immediately before the RTI that is used to return from the ISR.

When two or more interrupts are pending when the CCR [I] is cleared, the highest priority source is serviced first.

4.1.1 Interrupt stack frame

The following figure shows the contents and organization of a stack frame. Before the interrupt, the stack pointer (SP) points at the next available byte location on the stack. The current values of CPU registers are stored on the stack, starting with the low-order byte of the program counter (PC) and ending with the CCR. After stacking, the SP points at the next available location on the stack, which is the address that is one less than the address where the CCR was saved. The PC value that is stacked is the address of the instruction in the main program that would have executed next if the interrupt had not occurred.

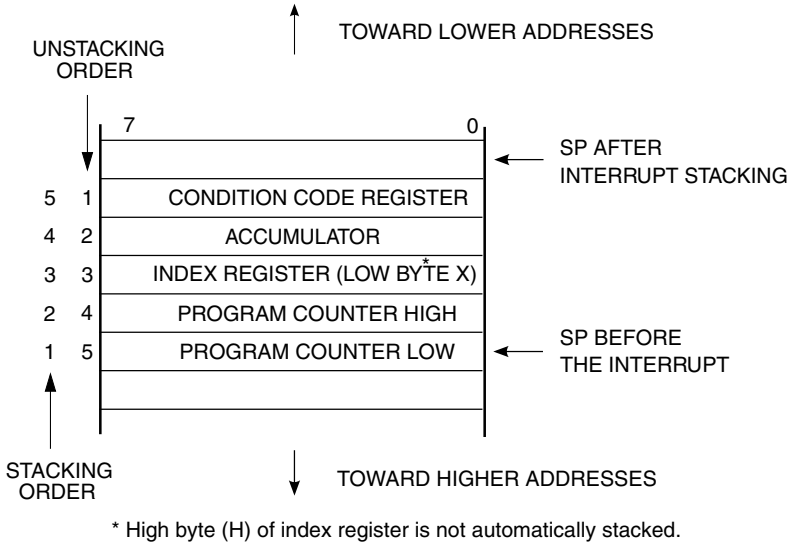


Figure 4-1. Interrupt stack frame

When an RTI instruction executes, these values are recovered from the stack in reverse order. As part of the RTI sequence, the CPU fills the instruction pipeline by reading three bytes of program information, starting from the PC address recovered from the stack.

The status flag causing the interrupt must be acknowledged (cleared) before returning from the ISR. Typically, the flag must be cleared at the beginning of the ISR because if another interrupt is generated by this source it will be registered so that it can be serviced after completion of the current ISR.

4.1.2 Hardware nested interrupt

This device has interrupt priority controller (IPC) module to provide up to four-level nested interrupt capability. IPC includes the following features:

- Four-level programmable interrupt priority for each interrupt source.
- Support for prioritized preemptive interrupt service routines
  - Low-priority interrupt requests are blocked when high-priority interrupt service routines are being serviced.
  - Higher or equal priority level interrupt requests can preempt lower priority interrupts being serviced.
- Automatic update of interrupt priority mask with being serviced interrupt source priority level when the interrupt vector is being fetched.

- Interrupt priority mask can be modified during main flow or interrupt service execution.
- Previous interrupt mask level is automatically stored when interrupt vector is fetched (four levels of previous values accommodated)

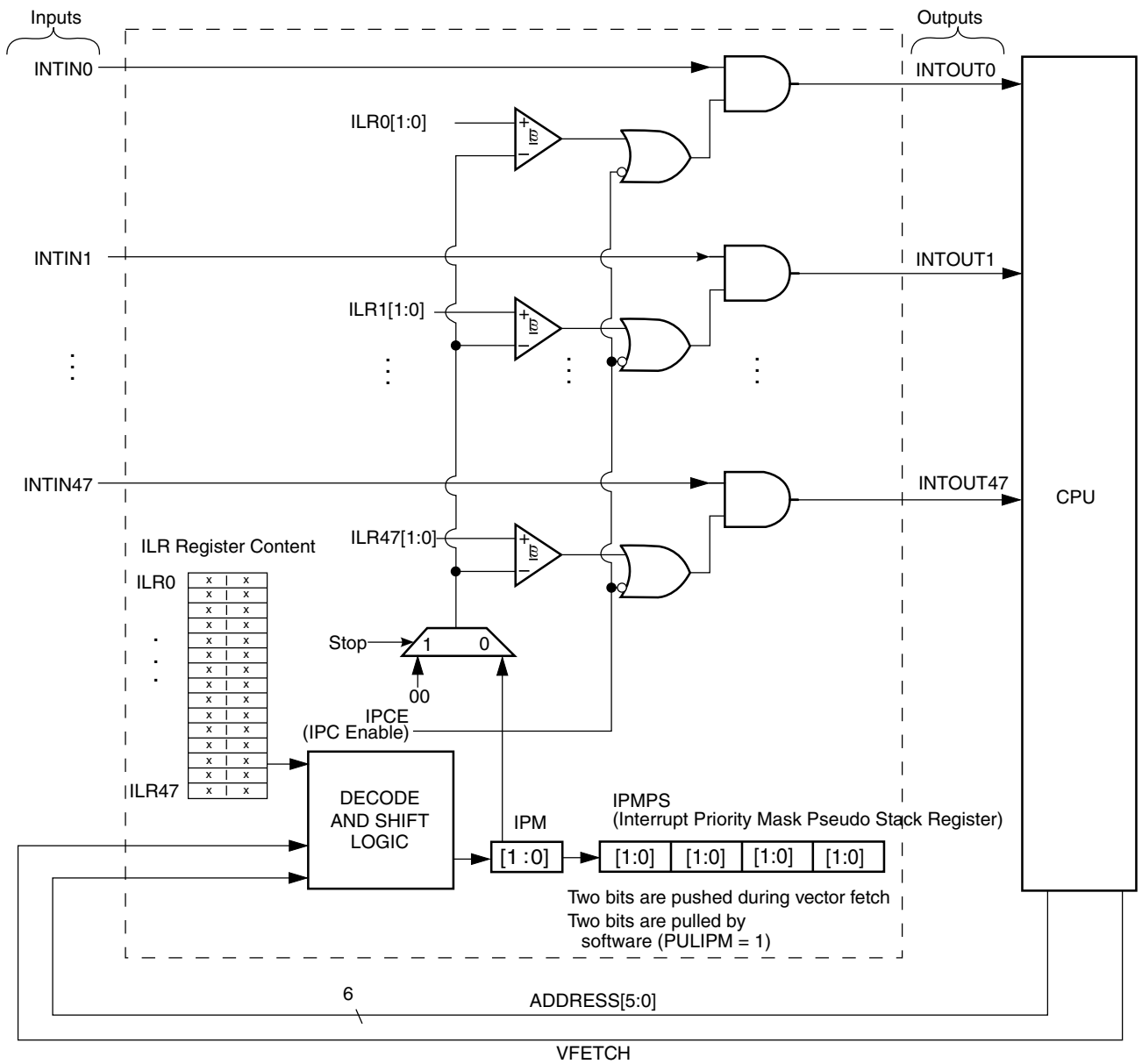


Figure 4-2. Interrupt priority controller block diagram

The IPC works with the existing HCS08 interrupt mechanism to allow nested interrupts with programmable priority levels. This module also allows implementation of preemptive interrupt according to the programmed interrupt priority with minimal software overhead. The IPC consists of three major functional blocks:

- The interrupt priority level registers
- The interrupt priority level comparator set
- The interrupt mask register update and restore mechanism

4.1.2.1 Interrupt priority level register

This set of registers is associated with the interrupt sources to the HCS08 CPU. Each interrupt priority level is a 2-bit value such that a user can program the interrupt priority level of each source to priority 0, 1, 2, or 3. Level 3 has the highest priority while level 0 has the lowest. Software can read or write to these registers at any time. The interrupt priority level comparator set, interrupt mask register update, and restore mechanism use this information.

4.1.2.2 Interrupt priority level comparator set

When the module is enabled, an active interrupt request forces a comparison between the corresponding ILR and the 2-bit interrupt mask IPM[1:0]. In stop mode, the IPM[1:0] is substituted by value 00b. If the ILR value is greater than or equal to the value of the interrupt priority mask (IPM bits in IPCSC), the corresponding interrupt out (INTOUT) signal will be asserted and signals an interrupt request to the HCS08 CPU.

When the module is disabled, the interrupt request signal from the source is directly passed to the CPU.

The interrupt priority level programmed in the interrupt priority register will not affect the inherent interrupt priority arbitration as defined by the HCS08 CPU because the IPC is an external module. Therefore, if two (or more) interrupts are present in the HCS08 CPU at the same time, the inherent priority in HCS08 CPU will perform arbitration by the inherent interrupt priority.

4.1.2.3 Interrupt priority mask update and restore mechanism

The interrupt priority mask (IPM) is two bits located in the least significant end of IPCSC register. These two bits control which interrupt is allowed to be presented to the HCS08 CPU. During vector fetch, the interrupt priority mask is updated automatically with the value of the ILR corresponding to that interrupt source. The original value of the IPM will be saved onto IPMPS for restoration after the interrupt service routine completes execution. When the interrupt service routine completes execution, the user restore the

original value of IPM by writing 1 to the IPCSC[PULIPM] bit. In both cases, the IPMPS is a shift register functioning as a pseudo stack register for storing the IPM. When the IPM is updated, the original value is shifted into IPMPS. The IPMPS can store four levels of IPM. If the last position of IPMPS is written, the PSF flag indicates that the IPMPS is full. If all the values in the IPMPS were read, the PSE flag indicates that the IPMPS is empty.

4.1.2.4 Integration and application of the IPC

All interrupt inputs that comes from peripheral modules are asynchronous signals. None of the asynchronous signals of the interrupts are routed to IPC. The asynchronous signals of the interrupts are routed directly to SIM module to wake system clocks in stop mode.

Additional care must be exercised when IRQ is reprioritized by IPC. CPU instructions BIL and BIH need input from IRQ pin. If IRQ interrupt is masked, BIL and BIH still work but the IRQ interrupt will not occur.

- The interrupt priority controller must be enabled to function. While inside an interrupt service routine, some work has to be done to enable other higher priority interrupts. The following is a pseudo code example written in assembly language:

```
INT_SER :
        BCLR        INTFLAG,INTFLAG_R ; clear flag that generate interrupt
        .                             ; do the most critical part
        .                             ; which it cannot be interrupted
        .
        .
        .
        CLI                           ; global interrupt enable and nested interrupt
enabled
        .                             ; continue the less critical
        .
        .
        .
        BSET        PULIPM, PULIPM_R  ; restore the old IPM value before leaving
        RTI                           ; then you can return
```

- A minimum overhead of six bus clock cycles is added inside an interrupt services routine to enable preemptive interrupts.
- As an interrupt of the same priority level is allowed to pass through IPC to HCS08 CPU, the flag generating the interrupt must be cleared before doing CLI to enable preemptive interrupts.

- The IPM is automatically updated to the level the interrupt is servicing and the original level is kept in IPMPS. Watch out for the full (PSF) bit if nesting for more than four levels is expected.
- Before leaving the interrupt service routine, the previous levels must be restored manually by setting PULIPM bit. Watch out for the full (PSF) bit and empty (PSE) bit.

4.2 IPC memory map and register descriptions

IPC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
E	IPC Status and Control Register (IPC_SC)	8	R/W	20h	4.2.1/59
F	Interrupt Priority Mask Pseudo Stack Register (IPC_IPMPS)	8	R	00h	4.2.2/60
1860	Interrupt Level Setting Registers n (IPC_ILRS0)	8	R/W	00h	4.2.3/61
1861	Interrupt Level Setting Registers n (IPC_ILRS1)	8	R/W	00h	4.2.3/61
1862	Interrupt Level Setting Registers n (IPC_ILRS2)	8	R/W	00h	4.2.3/61
1863	Interrupt Level Setting Registers n (IPC_ILRS3)	8	R/W	00h	4.2.3/61
1864	Interrupt Level Setting Registers n (IPC_ILRS4)	8	R/W	00h	4.2.3/61
1865	Interrupt Level Setting Registers n (IPC_ILRS5)	8	R/W	00h	4.2.3/61
1866	Interrupt Level Setting Registers n (IPC_ILRS6)	8	R/W	00h	4.2.3/61
1867	Interrupt Level Setting Registers n (IPC_ILRS7)	8	R/W	00h	4.2.3/61

4.2.1 IPC Status and Control Register (IPC_SC)

This register contains status and control bits for the IPC.

Address: Eh base + 0h offset = Eh

Bit	7	6	5	4	3	2	1	0
Read	IPCE	0	PSE	PSF	0	0	IPM	
Write					PULIPM			
Reset	0	0	1	0	0	0	0	0

IPC_SC field descriptions

Field	Description
7 IPCE	Interrupt Priority Controller Enable This bit enables/disables the interrupt priority controller module.

Table continues on the next page...

IPC_SC field descriptions (continued)

Field	Description
	0 Disables IPCE. Interrupt generated from the interrupt source is passed directly to CPU without processing (bypass mode). The IPMPS register is not updated when the module is disabled. 1 Enables IPCE and interrupt generated from the interrupt source is processed by IPC before passing to CPU.
6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5 PSE	Pseudo Stack Empty This bit indicates that the pseudo stack has no valid information. This bit is automatically updated after each IPMPS register push or pull operation.
4 PSF	Pseudo Stack Full This bit indicates that the pseudo stack register IPMPS register is full. It is automatically updated after each IPMPS register push or pull operation. If additional interrupt is nested after this bit is set, the earliest interrupt mask value(IPM0[1:0]) stacked in IPMPS will be lost. 0 IPMPS register is not full. 1 IPMPS register is full.
3 PULIPM	Pull IPM from IPMPS This bit pulls stacked IPM value from IPMPS register to IPM bits of IPCSC. Zeros are shifted into bit positions 1 and 0 of IPMPS. 0 No operation. 1 Writing 1 to this bit causes a 2-bit value from the interrupt priority mask pseudo stack register to be pulled to the IPM bits of IPCSC to restore the previous IPM value.
2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
IPM	Interrupt Priority Mask This field sets the mask for the interrupt priority control. If the interrupt priority controller is enabled, the interrupt source with an interrupt level (ILRxx) value that is greater than or equal to the value of IPM will be presented to the CPU. Writes to this field are allowed, but doing this will not push information to the IPMPS register. Writing IPM with PULIPM setting when IPCE is already set, the IPM will restore the value pulled from the IPMPS register, not the value written to the IPM register.

4.2.2 Interrupt Priority Mask Pseudo Stack Register (IPC_IPMPS)

This register is used to store the previous interrupt priority mask level temporarily when the currently active interrupt is executed.

Address: Eh base + 1h offset = Fh

Bit	7	6	5	4	3	2	1	0
Read	IPM3		IPM2		IPM1		IPM0	
Write								
Reset	0	0	0	0	0	0	0	0

IPC_IPMPS field descriptions

Field	Description
7–6 IPM3	Interrupt Priority Mask pseudo stack position 3 This field is the pseudo stack register for IPM3. The most recent information is stored in IPM3.
5–4 IPM2	Interrupt Priority Mask pseudo stack position 2 This field is the pseudo stack register for IPM2. The most recent information is stored in IPM2.
3–2 IPM1	Interrupt Priority Mask pseudo stack position 1 This field is the pseudo stack register for IPM1. The most recent information is stored in IPM1.
IPM0	Interrupt Priority Mask pseudo stack position 0 This field is the pseudo stack register for IPM0. The most recent information is stored in IPM0.

4.2.3 Interrupt Level Setting Registers n (IPC_ILRS*n*)

This set of registers (ILRS0-ILRS7) contains the user specified interrupt level for each interrupt source, and indicates the number of the register (ILRSn is ILRS0 through ILRS7).

Address: Eh base + 1852h offset + (1d × i), where i=0d to 7d

Bit	7	6	5	4	3	2	1	0
Read Write	ILRn3		ILRn2		ILRn1		ILRn0	
Reset	0	0	0	0	0	0	0	0

IPC_ILRS*n* field descriptions

Field	Description
7–6 ILRn3	Interrupt Level Register for Source n*4+3 This field sets the interrupt level for interrupt source n*4+3.
5–4 ILRn2	Interrupt Level Register for Source n*4+2 This field sets the interrupt level for interrupt source n*4+2.
3–2 ILRn1	Interrupt Level Register for Source n*4+1 This field sets the interrupt level for interrupt source n*4+1.
ILRn0	Interrupt Level Register for Source n*4+0 This field sets the interrupt level for interrupt source n*4+0.

4.3 IRQ

The IRQ (interrupt request) module provides a maskable interrupt input.

4.3.1 Features

Features of the IRQ module include:

- IRQ signal is from Intermodule crossbar Output, XBAR_OUT15
- IRQ Interrupt Control Bits
- Programmable Edge-only or Edge and Level Interrupt Sensitivity
- Automatic Interrupt Acknowledge

A low level applied to the interrupt request (IRQ) can latch a CPU interrupt request. The following figure shows the structure of the IRQ module:

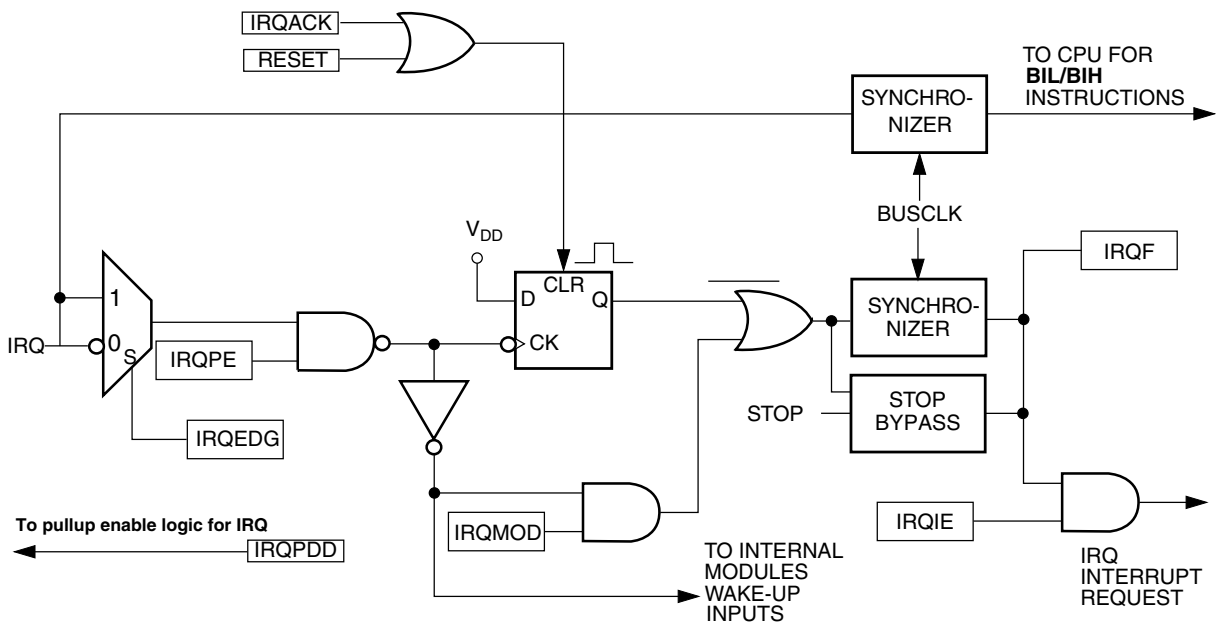


Figure 4-3. IRQ module block diagram

IRQ is managed by the IRQSC status and control register. When the IRQ function is enabled, synchronous logic monitors the Xbar_OUT15 for edge-only or edge-and-level events. When the MCU is in stop mode and system clocks are shut down, a separate asynchronous path is used so that the IRQ, if enabled, can wake the MCU.

4.3.1.1 Configuration options

The IRQ input enable control bit (IRQSC[IRQPE]) must be 1 for the IRQ signal to act as the IRQ input. The user can choose the polarity of edges or levels detected (IRQEDG), whether the pin detects edges-only or edges and levels (IRQMOD), or whether an event causes an interrupt or only sets the IRQF flag, which can be polled by software.

Since IRQ signal is from XBAR_OUT15, it is recommend XBAR_OUT15 is set to logic high after reset.

BIH and BIL instructions may be used to detect the level on the IRQ signal when IRQ is enabled.

4.3.1.2 Edge and level sensitivity

The IRQSC[IRQMOD] control bit reconfigures the detection logic so that it can detect edge events and levels. In this detection mode, the IRQF status flag is set when an edge is detected, if the IRQ signal changes from the de-asserted to the asserted level, but the flag is continuously set and cannot be cleared as long as the IRQ signal remains at the asserted level.

4.4 IRQ Memory Map and Register Descriptions

IRQ memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
7F	Interrupt Pin Request Status and Control Register (IRQ_SC)	8	R/W	00h	4.4.1/63

4.4.1 Interrupt Pin Request Status and Control Register (IRQ_SC)

This direct page register includes status and control bits, which are used to configure the IRQ function, report status, and acknowledge IRQ events.

Address: 7Fh base + 0h offset = 7Fh

Bit	7	6	5	4	3	2	1	0
Read	0	IRQPDD	IRQEDG	IRQPE	IRQF	0	IRQIE	IRQMOD
Write						IRQACK		
Reset	0	0	0	0	0	0	0	0

IRQ_SC field descriptions

Field	Description
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6 IRQPDD	Interrupt Request (IRQ) Pull Device Disable This read/write control bit is used to disable the internal pullup device when the IRQ pin is enabled (IRQPE = 1) allowing for an external device to be used. 0 IRQ pull device enabled if IRQPE = 1. 1 IRQ pull device disabled if IRQPE = 1.
5 IRQEDG	Interrupt Request (IRQ) Edge Select This read/write control bit is used to select the polarity of edges or levels on the IRQ pin that cause IRQF to be set. The IRQMOD control bit determines whether the IRQ pin is sensitive to both edges and levels or only edges. When the IRQ pin is enabled as the IRQ input and is configured to detect rising edges, the optional pullup resistor is disabled. 0 IRQ is falling edge or falling edge/low-level sensitive. 1 IRQ is rising edge or rising edge/high-level sensitive.
4 IRQPE	IRQ Pin Enable This read/write control bit enables the IRQ pin function. When this bit is set the IRQ pin can be used as an interrupt request. 0 IRQ pin function is disabled. 1 IRQ pin function is enabled.
3 IRQF	IRQ Flag This read-only status bit indicates when an interrupt request event has occurred. 0 No IRQ request. 1 IRQ event detected.
2 IRQACK	IRQ Acknowledge This write-only bit is used to acknowledge interrupt request events (write 1 to clear IRQF). Writing 0 has no meaning or effect. Reads always return 0. If edge-and-level detection is selected (IRQMOD = 1), IRQF cannot be cleared while the IRQ pin remains at its asserted level.
1 IRQIE	IRQ Interrupt Enable This read/write control bit determines whether IRQ events generate an interrupt request. 0 Interrupt request when IRQF set is disabled (use polling). 1 Interrupt requested whenever IRQF = 1.
0 IRQMOD	IRQ Detection Mode This read/write control bit selects either edge-only detection or edge-and-level detection. 0 IRQ event on falling/rising edges only. 1 IRQ event on falling/rising edges and low/high levels.

Chapter 5
Clock management

5.1 Clock module

This device has ICS and LPO clock modules.

The internal clock source (ICS) module provides several clock source options for this device. The module contains a frequency-locked loop (FLL) that is controllable by either an internal or external reference clock. The module can select clock from the FLL or bypass the FLL as a source of the MCU system clock. The selected clock source is passed through a reduced bus divider, which allows a lower output clock frequency to be derived.

An internal trimmed 32 kHz is the main clock source for ICS. It also can be used as reference for windowed COP watchdog (WCOP)

An external clock input is available in this device which can be used as the reference of ICS to generate system bus clock and analog-to-digital (ADC) modules.

The low-power oscillator (LPO) module is an on-chip low-power oscillator providing around 20 kHz reference clock to windowed COP watchdog (WCOP).

5.2 System clock distribution

These series contain two on-chip clock sources:

- Internal clock source (ICS) module — The main clock source generator providing bus clock and other reference clocks to core, memory and peripherals
- Low-power oscillator (LPO) module — The on-chip low-power oscillator in PMC module providing 20 kHz reference clock to windowed COP watchdog (WCOP) to meet IEC60730 safety standard.

NOTE

The system clock is bus clock.

NOTE

For this device, the master clock and system/bus clock are the same clock.

The following figure shows a simplified clock connection diagram.

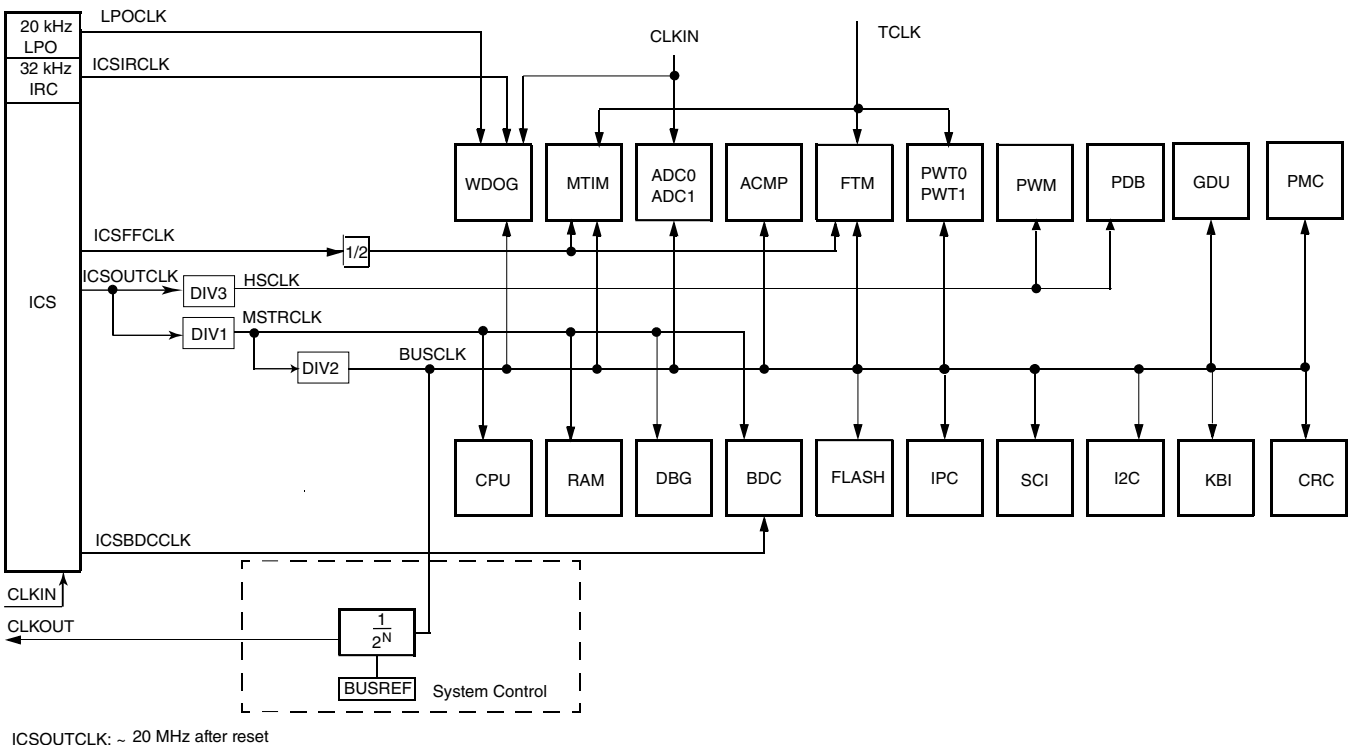


Figure 5-1. System clock distribution diagram

The clock system supplies:

- ICSOUTCLK — This up to 40 MHz clock source is used as the master clock and bus clock and high speed clock that is the reference to CPU and all peripherals. Control bits in the ICS control registers determine which of two clock sources is connected:
  - 32 kHz Internal reference clock
  - Frequency-locked loop (FLL) output

  There are three clocks that are derived from this clock source:
  - MSTRCLK — Master clock is the clock source for CPU and RAM and DBG and system/ bus clock

  - BUSCLK — Bus clock is the clock source for all peripherals and flash module
  - HSCLK — High speed clock is up to 40 MHz. It can be set to 1:1 bus clock or 2:1 bus clock. It does not support other clock ratio. it can be selected as the clock source to the PWM and PDB

  After reset, ICSOUTCLK clock is around 20 MHz.

- ICSLCLK — This clock source is derived from the digitally controlled oscillator (DCO) of the ICS when the ICS is configured to run off of the internal or external reference clock. Development tools can select this internal self-clocked source (20 MHz) to speed up BDC communications in systems where the bus clock is slow.
- ICSIRCLK — This is the internal reference clock and can be selected as the clock source to the WCOP module.
- ICSFFCLK — This generates the fixed frequency clock (FFCLK) after being synchronized to the bus clock. It can be selected as clock source, after being divided by 2, to the FTM and MTIM modules. The frequency of the ICSFFCLK is determined by the setting of the ICS.
- LPOCLK — This clock is generated from an internal low power oscillator (≈20 kHz) that is completely independent of the ICS module. The LPOCLK can be selected as the clock source to the WCOP module.
- TCLK — This is an optional external clock source for the FTM and MTIM and PWT0 and PWT1 modules. The TCLK must be limited to 1/4th frequency of the bus clock for synchronization.
- CLKOUT — This clock is a buffered bus clock to a package pin.
- CLKIN — This is an external clock input from package pins. This clock input cannot be dynamically switched between its inputs (DC - 40 MHz).

5.3 Internal clock source (ICS)

The internal clock source (ICS) module provides clock source options for the MCU. The module contains a frequency-locked loop (FLL) as a clock source that is controllable by an internal or external reference clock. The module can provide this FLL clock or the internal reference clock as a source for the MCU system clock, ICSCLK.

Whichever clock source is chosen, ICSCLK is the output from a bus clock divider (BDIV), which allows a lower clock frequency to be derived.

Key features of the ICS module are:

- Frequency-locked loop (FLL) is trimmable for accuracy

- Internal or external reference clocks can be used to control the FLL
- Reference divider is provided for external clock
- Internal reference clock has nine trim bits available
- Internal or external reference clocks can be selected as the clock source for the MCU
- Whichever clock is selected as the source can be divided down by 1, 2, 4, 8, 16, 32, 64 or 128
- FLL Engaged Internal mode is automatically selected out of reset
- A constant divide by 2 of the DCO output that can be select as BDC clock.
- Digitally-controlled oscillator (DCO) optimized for 32 MHz to 40 MHz frequency range
- FLL lock detector and external clock monitor
    - FLL lock detector with interrupt capability
    - External reference clock monitor with reset capability

5.4 20 kHz low-power oscillator (LPO)

The 20 kHz low-power oscillator acts as a standalone low-frequency clock source in all run, wait, and stop modes.

5.5 Peripheral clock gating

This device includes a clock gating system to manage the bus clock sources to the individual peripherals. Using this system, the user can enable or disable the bus clock to each of the peripherals at the clock source, eliminating unnecessary clocks to peripherals that are not in use, thereby reducing the overall run and wait mode currents.

For lowest possible run wait currents, user software must disable the clock source to any peripheral not in use. The actual clock will be enabled or disabled immediately following the write to the System Clock Gating Control registers (SIM_SCGCx, x=1, 2, 3). Any peripheral with a gated clock cannot be used unless its clock is enabled. Writing to the registers of a peripheral with a disabled clock has no effect.

Note

User software must disable the peripheral before disabling the clocks to the peripheral. When clocks to a peripheral are re-enabled, the peripheral registers need to be re-initialized by user software.

In stop modes, the bus clock is disabled for all gated peripherals, regardless of the setting in SIM_SCGCx (x=1,2,3) registers.

Chapter 6
Power Management

6.1 Introduction

The operating modes of the device are described in this chapter. Entry into each mode, exit from each mode, and functionality while in each of the modes are described.

6.2 Features

These MCUs feature the following power modes:

- Run mode
- Wait mode
  - CPU shuts down to conserve power
  - Bus clocks are running
  - Full voltage regulation is maintained
- Stop modes
  - System clocks stopped; PMC is powered with voltage regulator in standby
  - all internal circuits powered for fast recovery

6.2.1 Run mode

This is the normal operating mode. In this mode, the CPU executes code from internal memory with execution beginning at the address fetched from memory at 0xFFFE: 0xFFFF after reset. The power supply is fully regulating and all peripherals can be active in run mode.

6.2.2 Wait mode

Wait mode is entered by executing a WAIT instruction. Upon execution of the WAIT instruction, the CPU enters a low-power state in which it is not clocked. The CCR [I] is cleared when the CPU enters the wait mode, enabling interrupts. When an interrupt request occurs, the CPU exits the wait mode and resumes processing, beginning with the stacking operations leading to the interrupt service routine.

While the MCU is in wait mode, there are some restrictions on which background debug commands can be used. Only the BACKGROUND command and memory-access-with-status commands are available when the MCU is in wait mode. The memory-access-with-status commands do not allow memory access, but they report an error indicating that the MCU is in either stop or wait mode. The BACKGROUND command can be used to wake the MCU from wait mode and enter active background mode.

6.2.3 Stop mode

To enter stop, the user must execute a STOP instruction with stop mode enabled (SIM_SOPT1[STOPE] = 1). The ICS enters its standby state, as does the voltage regulator and the ADC. The states of all of the internal registers and logic, as well as the RAM content, are maintained. The I/O pin states are not latched at the pin. Instead they are maintained by virtue of the states of the internal logic driving the pins being maintained.

Exit from stop is done by asserting reset or through an interrupt. The interrupt include the asynchronous interrupt from the IRQ or KBI pins or ADC, CMP, I2C, SCI.

If stop is exited by means of the $\overline{\text{RESET}}$ pin, then the MCU will be reset and operation will resume after taking the reset vector. Exit by means of an asynchronous interrupt or the real-time interrupt will result in the MCU taking the appropriate interrupt vector.

Both low voltage detection and low reset are disabled in stop mode.

6.2.4 Active BDM enabled in stop mode

Entry into the active background mode from run mode is enabled if the BDC_SCR[ENBDM] bit is set. This register is described in the development support. If BDC_SCR[ENBDM] is set when the CPU executes a STOP instruction, the system

clocks to the background debug logic remain active when the MCU enters stop mode, so background debug communication is still possible. In addition, the voltage regulator does not enter its low-power standby state but maintains full internal regulation.

Most background commands are not available in stop mode. The memory-access-with-status commands do not allow memory access, but they report an error indicating that the MCU is in either stop or wait mode. The BACKGROUND command can be used to wake the MCU from stop and enter active background mode if the BDC_SCR[ENBDM] bit is set. After entering background debug mode, all background commands are available.

6.2.5 Power modes behaviors

Executing the WAIT or STOP command puts the MCU in a low power consumption mode for standby situations. The system integration module (SIM) holds the CPU in a non-clocked state. The operation of each of these modes is described in the following subsections. Both STOP and WAIT clear the CCR [I], allowing interrupt to occur. The following table shows the low power mode behaviors.

Table 6-1. Low power mode behavior

Mode	Run	Wait	Stop
PMC	Full regulation	Full regulation	Loose regulation
ICS	On	On	Standby
LPO	On	On	Optional On
CPU	On	Standby	Standby
Flash	On	On	Standby
RAM	On	Standby	Standby
ADC	On	On	Optional on
CMP	On	On	Optional on
I/O	On	On	States held
SCI / I2C	On	On	Standby
FTM / PWT / PWM / MTIM	On	On	Standby
WCOP	On	On	Optional on
DBG	On	On	Standby
IPC	On	On	Standby
LVD	On	On	Standby
GDU	On	On	Standby
PDB	On	On	Standby

6.3 Bandgap reference

This device includes an on-chip bandgap reference (≈1.2 V) connected to ADC channels. The bandgap reference voltage does not drop under the full operating voltage even when the operating voltage is falling. This reference voltage acts as an ideal reference voltage for accurate measurements.

This device also includes a high accuracy voltage reference VREFH (~4.2 V). This reference provides high accuracy reference to ADC and 6-bit DAC inside CMP.

Chapter 7
Signal multiplexing and signal descriptions

7.1 Introduction

To optimize functionality in small packages, pins have several functions available via signal multiplexing. Information found here illustrates which of this device's signals are multiplexed on which external pin.

The Port Control block controls which signal is present on the external pin. Refer to that chapter to find which register controls the operation of a specific pin.

7.2 Port control and interrupt module features

- 32-pin ports

NOTE

Not all pins are available on the device. See the following section for details.

The reset state and read/write characteristics of the fields within the PORTx_PCRn registers is summarized in the table below.

7.3 Signal multiplexing constraints

1. A given peripheral function must be assigned to a maximum of one package pin. Do not program the same function to more than one pin.
2. To ensure the best signal timing for a given peripheral's interface, choose the pins in closest proximity to each other.

7.4 Pinout

7.4.1 Signal multiplexing and pin assignments

The following table shows the signals available on each pin and the locations of these pins on the devices supported by this document. The Port Control Module is responsible for selecting which ALT functionality is available on each pin.

24 QFN	Pin Name	Default/ALT0	ALT1	ALT2	ALT3
1	PTB5	PWM_WL			PTB5
2	PWM_UH	PWM_UH			
3	PWM_VH	PWM_VH			
4	PWM_WH	PWM_WH			
5	VCLAMP	VCLAMP			
6	VDD	VDD			
7	VDDX	VDDX			
8	VSS	VSS			
9	PTB6/ RESET_b	RESET_b		TCLK	PTB6
10	PTC0	CMP_REF/ VREFH	PWM_FAULT0	CLK_IN	PTC0
11	PTB7/ BKGD/ MS	BKGD/ MS		CLKOUT	PTB7
12	PTA7	PWT1	TX	XB_OUT1	PTA7/ KBI7
13	PTA6	PWT0	RX	XB_IN1	PTA6/ KBI6
14	PTA5	TX	SDA	XB_OUT0	PTA5/ KBI5
15	PTA4	RX	SCL	XB_IN0	PTA4/ KBI4
16	PTA3	AMP1_M/ ADC1AD1	CLKOUT	XB_OUT1	PTA3/ KBI3
17	PTA2	AMP1_P/ CMP2/ ADC1AD0	XB_IN1	XB_OUT0	PTA2/ KBI2
18	PTA1	AMP0_M/ CMP1/ ADC0AD1	XB_OUT0	XB_IN1	PTA1/ KBI1
19	PTA0	AMP0_P/ CMP0/ ADC0AD0	CLK_IN	XB_IN0	PTA0/ KBI0

24 QFN	Pin Name	Default/ALT0	ALT1	ALT2	ALT3
20	PTB0	GDU_CMP0/ ADC0AD2/ ADC1AD2			PTB0
21	PTB1	GDU_CMP1/ ADC0AD3/ ADC1AD3			PTB1
22	PTB2	GDU_CMP2/ ADC0AD4/ ADC1AD4			PTB2
23	PTB3	PWM_UL			PTB3
24	PTB4	PWM_VL			PTB4

7.4.2 Signal description table

Table 7-1. Pin signal description

24 QFN	Chip signal name	Module	Module signal name	Operating voltage range (V)	Alt function	Type [1]	State during reset	Signal description
1	PWM_WL	PWM	PWM5	0–5	Default	O	Internally pull to low	PWM output 5 for driving N-MOSFET
	PTB5	PORT	PTB5		ALT3	I/O		This GPIO pin can be individually programmed as an input or output pin.
2	PWM_UH	PWM	PWM0	0–18	Default	O	Internally Pull to High	PWM output 0 for driving P-MOSFET
3	PWM_VH	PWM	PWM2	0–18	Default	O	Internally Pull to High	PWM output 2 for driving P-MOSFET
4	PWM_WH	PWM	PWM4	0–18	Default	O	Internally Pull to High	PWM output 2 for driving P-MOSFET
5	VCALMP	GDU	V_{o_clamp}	0–13	Default	S	Floating 5 V regulator output	Floating 5 V regulator for PMOS Vgs clamp. It outputs 5 V below VDD. Recommend to connect 1 µF low ESR ceramic capacitor, such as X7R capacitor between VDD and this pin to stabilize the voltage regulator output required for proper device operation.
6	VDD	All	VDD	4.5–18	Default	S	Supply	Power supplies 4.5–18 V
7	VDDX	PMC	VDDX	0–5	Default	S	5 V regulator output	Connect a 4.7 µF or greater bypass capacitor between this pin and VSS to stabilize the voltage regulator output required for proper device operation.
8	VSS	All	VSS	0	Default	S	Supply	Ground
9	RESET_b	All	$\overline{\text{RESET}}$	0–5	Default	I	Reset, internal	A direct hardware reset on the processor. When $\overline{\text{RESET}}$ is

Table continues on the next page...

Table 7-1. Pin signal description (continued)

24 QFN	Chip signal name	Module	Module signal name	Operating voltage range (V)	Alt function	Type [1]	State during reset	Signal description
							pullup enabled	asserted low, the device is initialized and placed in the reset state. A Schmitt-trigger input is used for noise immunity.
	TCLK	MTIM	TCLK		ALT2	I		An optional external clock source for the FTM, MTIM, PWT0 and PWT1 modules. The TCLK must be limited to 1/4th frequency of the bus clock for synchronization.
	PTB6	PORT	PTB6		ALT3	I/O		This GPIO pin can be individually programmed as an input or output pin.
10	CMP_REF/ VREFH			0–5	Default	S	Input	CMP_REF: Common input of GDU Phase comparator A, B and C; VREFH: On-chip 4.2V Voltage reference output. When this pin is configured as VREFH, Connect a 2.2 µF bypass capacitor between this pin and VSS to stabilize the voltage reference output required for proper device operation.
	PWM_FAULT0	PWM	FAULT0		ALT1	I		PWM fault input is used for disabling selected PWM outputs in case where fault conditions originate off-chip
	CLK_IN				ALT2	I		An optional external clock source. This clock input cannot be dynamically switched between its inputs (DC - 40 MHz)
	PTC0	PORT	PTC0		ALT3	I/O		This GPIO pin can be individually programmed as an input or output pin.
11	BKGD/MS	Core	BKGD/MS	0–5	Default	I/O	BKGD/MS	Background / Mode Select
	CLKOUT	SIM	CLKOUT		ALT2	O		A buffered bus clock output
	PTB7	PORT	PTB7		ALT3	O		This GPIO pin is an output pin only.
12	PWT1	PWT	PWT1IN0	0–5	Default	I	Tri-State	Input 0 of PWT1
	TX	SCI	TxD		ALT1	I/O		SCI transmit data output
	XB_OUT1	XBAR	XBAR_OUT1		ALT2	O		Crossbar module output 1
	PTA7/ KBI7	PORT/KBI	PTA7/KBIP7		ALT3	I/O		This GPIO pin can be individually programmed as an input or output pin with KBI functionality

Table continues on the next page...

Table 7-1. Pin signal description (continued)

24 QFN	Chip signal name	Module	Module signal name	Operating voltage range (V)	Alt function	Type [1]	State during reset	Signal description
13	PWT0	PWT	PWT0IN0	0–5	Default	Input	Tri-State	Input 0 of PWT0
	RX	SCI	RxD		ALT1	Input		SCI receive data input
	XB_IN1	XBAR	XBAR_IN1		ALT2	Input		Crossbar module input 1
	PTA6/ KBI6	PORT/KBI	PTA6/KBIP6		ALT3	I/O		This GPIO pin can be individually programmed as an input or output pin with KBI functionality
14	TX	SCI	TxD	0–5	Default	I/O	Tri-State	SCI transmit data output
	SDA	I2C	SDA		ALT1	I/O		I2C serial data line
	XB_OUT0	XBAR	XBAR_OUT 0		ALT2	O		Crossbar module output 0
	PTA5/KBI5	PORT/KBI	PTA5/KBIP5		ALT3	I/O		This GPIO pin can be individually programmed as an input or output pin with KBI functionality
15	RX	SCI	RxD	0–5	Default	I	Tri-State	SCI receive data input
	SCL	I2C	SCL		ALT1	I/O		I2C serial clock
	XB_IN0	XBAR	XBAR_IN 0		ALT2	I		Crossbar module input 0
	PTA4/ KBI4	PORT/KBI	PTA4/KBIP4		ALT3	I/O		This GPIO pin can be individually programmed as an input or output pin with KBI functionality
16	ADC1AD1/ AMP1_M[2]	ADC/ GDU	AD1	0–5	Default	I	Input	ADC1AD1: Input to channel 1 of ADC1; AMP0_M: GDU analog operational amplifier 1 negative input;
	CLKOUT				ALT1	O		A buffered bus clock output
	XB_OUT1	XBAR	XBAR_OUT 1		ALT2	O		Crossbar module output 1
	PTA3/ KBI3	PORT/KBI	PTA3/KBIP3		ALT3	I/O		This GPIO pin can be individually programmed as an input or output pin with KBI functionality
17	ADC1AD0/ AMP1_P / CMP2[2]	ADC/ GCMP/ CMP	AD0/INP/ Reference Input 2	0–5	Default	I	Input	ADC1AD0: Input to channel 0 of ADC1; AMP1_P: GDU analog operational amplifier 1 positive input; CMP2: Input 2 of comparator;
	XB_IN1	XBAR	XBAR_IN 1		ALT1	I		Crossbar module input 1
	XB_OUT0	XBAR	XBAR_OUT 0		ALT2	O		Crossbar module output 0
	PTA2/KBI2	PORT/KBI	PTA2/KBIP2		ALT3	I/O		This GPIO pin can be individually programmed as an input or output pin with KBI functionality

Table continues on the next page...

Table 7-1. Pin signal description (continued)

24 QFN	Chip signal name	Module	Module signal name	Operating voltage range (V)	Alt function	Type [1]	State during reset	Signal description
18	ADC0AD1/ AMP0_M / CMP1[2]	ADC/ GCMP/ CMP	AD1/INM/ Reference input 1	0–5	Default	I	Input	ADC0AD1: Input to channel 1 of ADC0; AMP0_M: GDU analog operational amplifier 0 negative input CMP1: Input 1 of analog comparator;
	XB_OUT0	XBAR	XBAR_OUT 1		ALT1	O		Crossbar module output 0
	XB_IN1	XBAR	XBAR_IN 1		ALT2	I		Crossbar module input 1
	PTA1/KB1	PORT/KBI	PTA1/KBIP1		ALT3	I/O		This GPIO pin can be individually programmed as an input or output pin with KBI functionality
19	ADC0AD0 / AMP0_P / CMP0[2]	ADC/ GCMP/ CMP	AD0/INP/ Reference input 0	0–5	Default	I	Input	ADC0AD0: Input to channel 0 of ADC0; AMP0_P: GDU analog operational amplifier 0 positive input; CMP0: Input 0 of analog comparator;
	CLK_IN				ALT1	I		An optional external clock source. This clock input cannot be dynamically switched between its inputs (DC - 40 MHz)
	XB_IN0	XBAR	XBAR_IN 0		ALT2	I		Crossbar module input 0
	PTA0/ KBI0	PORT/KBI	PTA0/KBIP0		ALT3	I/O		This GPIO pin can be individually programmed as an input or output pin with KBI functionality
20	ADC0AD2 / ADC1AD2 / GDU_CMP0[2]	ADC0/ ADC1/GDU	AD2/AD2/ CMP0	0–5	Default	I	Internal pull down	ADC0AD2: Input to channel 2 of ADC0; ADC1AD2: Input to channel 2 of ADC1; GDU_CMP0: GDU phase comparator 0 positive input
	PTB0	PORT	PTB0		ALT3	I/O		This GPIO pin can be individually programmed as an input or output pin.
21	ADC0AD3 / ADC1AD3 / GDU_CMP1[2]	ADC0/ ADC1/GDU	AD3/AD3/ CMP1	0–5	Default	I	Internal pull down	ADC0AD3: Input to channel 3 of ADC0; ADC1AD3: Input to channel 3 of ADC1; GDU_CMP1: GDU Phase comparator 1 positive input

Table continues on the next page...

Table 7-1. Pin signal description (continued)

24 QFN	Chip signal name	Module	Module signal name	Operating voltage range (V)	Alt function	Type [1]	State during reset	Signal description
	PTB1	PORT	PTB1		ALT3	I/O		This GPIO pin can be individually programmed as an input or output pin.
22	ADC0AD4 / ADC1AD4 / GDU_CMP2 [2]	ADC0/ ADC1/GDU	AD4/AD4/ CMP2	0–5	Default	I	Internal pull down	ADC0AD4: Input to channel 4 of ADC0; ADC1AD4: Input to channel 4 of ADC1; GDU_CMP2: GDU Phase comparator 2 positive input
	PTB2	PORT	PTB2		ALT3	I/O		This GPIO pin can be individually programmed as an input or output pin.
23	PWM_UL	PWM	PWM1	0–5	Default	O	Internally Pull to Low	PWM output 1 for driving N-MOSFET
	PTB3	PORT	PTB3		ALT3	I/O		This GPIO pin can be individually programmed as an input or output pin.
24	PWM_VL	PWM	PWM3	0–5	Default	O	Internally Pull to Low	PWM output 3 for driving N-MOSFET
	PTB4	PORT	PTB4		ALT3	I/O		This GPIO pin can be individually programmed as an input or output pin.

1. S: supply; I: input; O: output; I/O: input/output
2. When used as an analog input, the signal goes to all analog modules, but the glitch during ADC sampling on this pin may interfere with other analog inputs shared on this pin.

7.4.3 Pinout

The following figures show the pinout diagrams for the devices supported by this document. Many signals may be multiplexed onto a single pin. To determine what signals can be used on which pin, see Signal multiplexing and pin assignments.

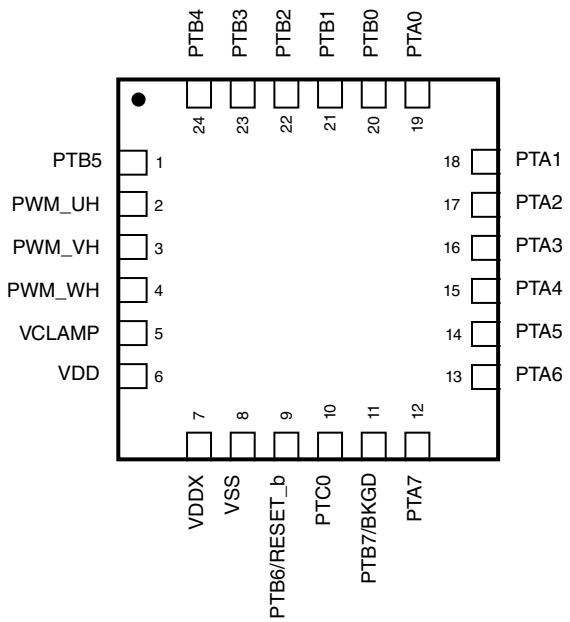


Figure 7-1. 24-pin QFN pinout diagram

Chapter 8
Port Control (PORT)

8.1 Introduction

This device has three sets of I/O ports, which include up to 17 general-purpose I/O pins.

To enable the GPIO function, SIM_MUXPTxL/SIM_MUXPTxH (x=A, B, or C) registers need be adjusted to select ALT3.

The pin control register configures the following functions for each pin within the 8-bit port.

- Out data on selected pins
- In/out direction on selected pins
- Pullup/pulldown enable on selected pins

KBI shares with PTA on ALT3, so PTA function works only when KBI_PE is disabled;

The following figure show the structure of normal I/O pin(PTA0 as example).

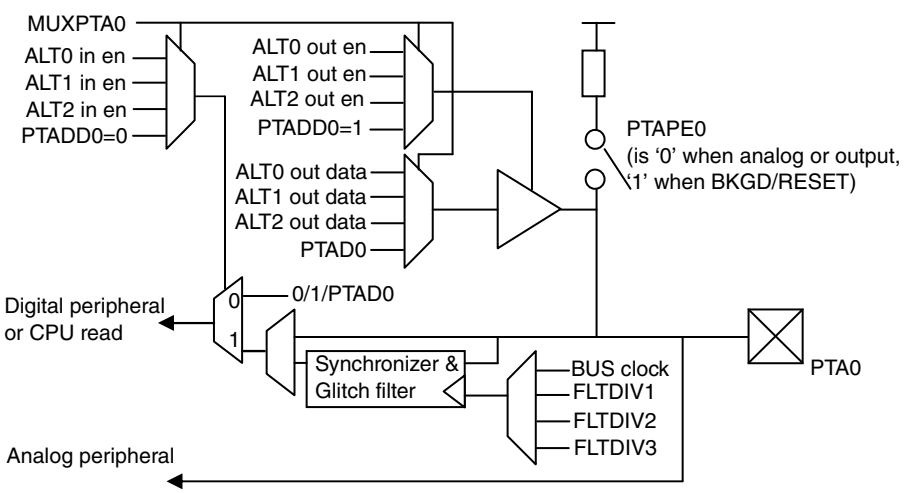


Figure 8-1. Normal I/O structure

NOTE

For PTB6/RESET_b pin, output is true open-drain drive.

8.2 Port data and data direction

Reading and writing of parallel I/O is accomplished through the port data registers (PORT_PTxD, x=A, B, or C). The input or output is controlled by the direction registers. Each port pin has an input enable bit and an output enable bit. When PORT_PTxDD[n] = 0 (x=A, B, or C; n=0–7), a read from PORT_PTxD[n] returns the input value of the associated pin; when PORT_PTxDD[n] = 1, a read from PORT_PTxD[n] returns the last value written to the port data register.

NOTE

When a digital peripheral module or system function is selected and enabled on a pin, reads of the port data register still returns the pin value of the associated pin if PORT_PTxDD[n] = 0. When a shared analog function is selected for a pin, all digital pin functions are disabled. A read of the port data register returns a value of 0 for any bits that have shared analog functions enabled.

A write of valid data to a port data register must occur before setting the direction control bit of an associated port pin. This ensures that the pin will not be driven with an incorrect data value.

8.3 Internal pullup/pulldown enable

An internal pullup or pulldown device can be enabled for each port pin by setting the corresponding bit in one of the pullup enable registers (PORT_PTxPE[n], x=A, B, or C, n=0–7). The internal pullup device is disabled if the pin is configured as an output direction, or by selecting any output peripheral functions, or input peripheral function like PWT, regardless of the state of the corresponding pullup enable register bit. The internal pullup device is also disabled if the pin is controlled by an analog function.

If an SDA, SCL, RX, TX, KBI, or TCLK function is selected and enabled on a pin, the pullup configuration for that pin still works. The internal pullup device is enabled when pin select as BKGD/$\overline{\text{RESET}}$ function.

NOTE

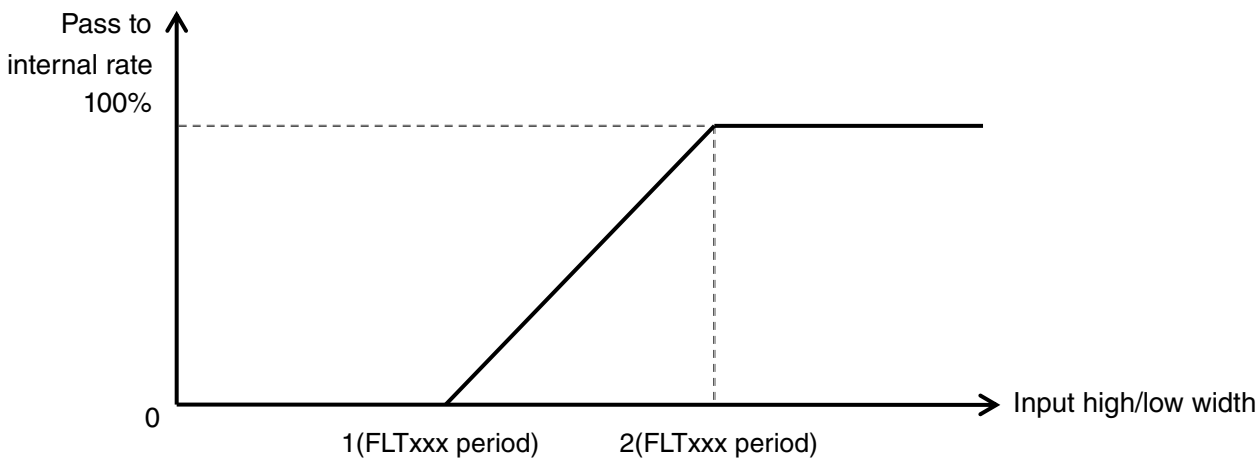
When configuring I2C to use "SDA(PTA5) and SCL(PTA4)" pins, and if an application uses internal pullups instead of external pullups, the internal pullups remain present setting when the pins are configured as outputs, but they are automatically disabled to save power when the output values are low.

8.4 Input glitch filter

A filter is implemented for each port pin that is configured as a digital input. It can be used as a simple low-pass filter to filter any glitch that is introduced from the pins of PTx (x=A,B, or C), I2C, PWT, XBI, $\overline{\text{RESET}}$, and KBI. The glitch width threshold can be adjusted easily by setting registers PORT_IOFLTn (n=0–2) and PORT_FCLKDIV between 1–4096 BUSCLKs (or 1–128 LPOCLKs). This configurable glitch filter can take the place of an on board external analog filter, and greatly improve the EMC performance because any glitch will not be wrongly sampled or ignored.

Setting register PORT_IOFLTn (n=0–2) can configure the filter of the whole port. For example, setting PORT_IOFLT0[FLTA] affects all PTA pins.

Glitches that are shorter than the selected clock period are filtered out; Glitches that are twice more than the selected clock period are filtered out. It passes to internal circuitry.



Note: FLTxxx is contents in register PORT_IOFLTn (n=0-2).

Figure 8-2. Input glitch filter

8.5 Memory map and register definition

PORT memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
0	Port A Data Register (PORT_PTAD)	8	R/W	00h	8.5.1/86
1	Port B Data Register (PORT_PTBD)	8	R/W	00h	8.5.2/87
2	Port C Data Register (PORT_PTCD)	8	R/W	00h	8.5.3/87
3	Port A Direction Register (PORT_PTADD)	8	R/W	00h	8.5.4/88
4	Port B Direction Register (PORT_PTBDD)	8	R/W	00h	8.5.5/89
5	Port C Direction Register (PORT_PTCDD)	8	R/W	00h	8.5.6/89
18E0	Port A Pullup Enable Register (PORT_PTAPE)	8	R/W	00h	8.5.7/90
18E1	Port B Pullup/Pulldown Enable Register (PORT_PTBPE)	8	R/W	00h	8.5.8/91
18E2	Port C Pullup Enable Register (PORT_PTCPE)	8	R/W	00h	8.5.9/91
18E6	Port B High Drive Strength Selection Register (PORT_PTBHD)	8	R/W	00h	8.5.10/92
18EC	Port Clock Division Register (PORT_FCLKDIV)	8	R/W	00h	8.5.11/92
18ED	Port Filter Register 0 (PORT_IOFLT0)	8	R/W	00h	8.5.12/93
18EE	Port Filter Register 1 (PORT_IOFLT1)	8	R/W	00h	8.5.13/94
18EF	Port Filter Register 2 (PORT_IOFLT2)	8	R/W	00h	8.5.14/95

8.5.1 Port A Data Register (PORT_PTAD)

Reading and writing of parallel I/O is accomplished through this register.

When a digital peripheral module or system function is selected and enabled on a pin, reads of this register still returns the pin value of the associated pin if PORT_PTADD[n] = 0. (n=0-7) When a shared analog function is selected for a pin, all digital pin functions are disabled. A read of this register returns a value of 0 for any bits that have shared analog functions enabled.

A write of valid data to this register must occur before setting the direction control bit of an associated port pin. This ensures that the pin will not be driven with an incorrect data value.

Address: 0h base + 0h offset = 0h

Bit	7	6	5	4	3	2	1	0
Read Write	PTAD							
Reset	0	0	0	0	0	0	0	0

PORT_PTAD field descriptions

Field	Description
PTAD	Port A Data Register Bits For port A pins that are configured as inputs, a read returns the logic level on the pin. For port A pins that are configured as outputs, a read returns the last value that was written to this register. Writes are latched into all bits of this register. For port A pins that are configured as outputs, the logic level is driven out of the corresponding MCU pin.

8.5.2 Port B Data Register (PORT_PTBD)

Reading and writing of parallel I/O is accomplished through this register.

When a digital peripheral module or system function is selected and enabled on a pin, reads of this register still returns the pin value of the associated pin if PORT_PTBDD[n] = 0 (n=1-7). When a shared analog function is selected for a pin, all digital pin functions are disabled. A read of this register returns a value of 0 for any bits that have shared analog functions enabled.

A write of valid data to this register must occur before setting the direction control bit of an associated port pin. This ensures that the pin will not be driven with an incorrect data value.

Address: 0h base + 1h offset = 1h

Bit	7	6	5	4	3	2	1	0
Read Write	PTBD							
Reset	0	0	0	0	0	0	0	0

PORT_PTBD field descriptions

Field	Description
PTBD	Port B Data Register Bits For port B pins that are configured as inputs, a read returns the logic level on the pin. For port B pins that are configured as outputs, a read returns the last value that was written to this register. Writes are latched into all bits of this register. For port B pins that are configured as outputs, the logic level is driven out of the corresponding MCU pin.

8.5.3 Port C Data Register (PORT_PTCD)

Reading and writing of parallel I/O is accomplished through this register.

When a digital peripheral module or system function is selected and enabled on a pin, reads of this register still returns the pin value of the associated pin if PTCDD[0] = 0. When a shared analog function is selected for a pin, all digital pin functions are disabled. A read of this register returns a value of 0 for any bits that have shared analog functions enabled.

A write of valid data to this register must occur before setting the direction control bit of an associated port pin. This ensures that the pin will not be driven with an incorrect data value.

Address: 0h base + 2h offset = 2h

Bit	7	6	5	4	3	2	1	0
Read	0							PTCD
Write								
Reset	0	0	0	0	0	0	0	0

PORT_PTCD field descriptions

Field	Description
7–1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 PTCD	Port C Data Register Bits For port C pins that are configured as inputs, a read returns the logic level on the pin. For port C pins that are configured as outputs, a read returns the last value that was written to this register. Writes are latched into all bits of this register. For port C pins that are configured as outputs, the logic level is driven out of the corresponding MCU pin.

8.5.4 Port A Direction Register (PORT_PTADD)

Address: 0h base + 3h offset = 3h

Bit	7	6	5	4	3	2	1	0
Read Write	PTADD							
Reset	0	0	0	0	0	0	0	0

PORT_PTADD field descriptions

Field	Description
PTADD	Port A Direction Register Bits These bits control the direction of port A pins and what is read for PTAD reads. 0 Pin is configured as general-purpose input, for the GPIO function. 1 Pin is configured as general-purpose output, for the GPIO function.

8.5.5 Port B Direction Register (PORT_PTBDD)

Address: 0h base + 4h offset = 4h

Bit	7	6	5	4	3	2	1	0
Read Write	PTBDD							
Reset	0	0	0	0	0	0	0	0

PORT_PTBDD field descriptions

Field	Description
PTBDD	Port B Direction Register Bits These bits control the direction of port B pins and what is read for PTBD reads. **NOTE:** PTB7 pin is output only. 0 Pin is configured as general-purpose input, for the GPIO function. 1 Pin is configured as general-purpose output, for the GPIO function.

8.5.6 Port C Direction Register (PORT_PTCDD)

Address: 0h base + 5h offset = 5h

Bit	7	6	5	4	3	2	1	0
Read	0							PTCDD
Write								
Reset	0	0	0	0	0	0	0	0

PORT_PTCDD field descriptions

Field	Description
7–1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 PTCDD	Port C Direction Register Bits These bits control the direction of port C pins and what is read for PTCD reads. 0 Pin is configured as general-purpose input, for the GPIO function. 1 Pin is configured as general-purpose output, for the GPIO function.

8.5.7 Port A Pullup Enable Register (PORT_PTAPE)

An internal pullup device can be enabled for each port pin by setting the corresponding bit in one of the pullup enable registers (PTAPE[n], n=0-7). The internal pullup device is disabled regardless of the state of the corresponding pullup enable register bit if:

- the pin is configured as an output direction
- by selecting any output peripheral functions
- input peripheral function like PWT

The internal pullup device is also disabled if the pin is controlled by an analog function.

If an SDA, SCL, RX, TX, KBI, or TCLK function is selected and enabled on a pin, the pullup configuration for that pin still works. The internal pullup device is enabled when pin select as BKGD/$\overline{\text{RESET}}$ function.

NOTE

When configuring I2C to use "SDA(PTA5) and SCL(PTA4)" pins, and if an application uses internal pullups instead of external pullups, the internal pullups remain present setting when the pins are configured as outputs, but they are automatically disabled to save power when the output values are low.

Address: 0h base + 18E0h offset = 18E0h

Bit	7	6	5	4	3	2	1	0
Read Write	PTAPE							
Reset	0	0	0	0	0	0	0	0

PORT_PTAPE field descriptions

Field	Description
PTAPE	Pull Enable for Port A Bit These bits determines if the internal pullup device is enabled for the associated PTA pin. For port A pins that are configured as outputs, these bits have no effect and the internal pullup devices are disabled. 0 Internal pullup device disabled for port A bit n. 1 Internal pullup device enabled for port A bit n.

8.5.8 Port B Pullup/Pulldown Enable Register (PORT_PTBPE)

Address: 0h base + 18E1h offset = 18E1h

Bit	7	6	5	4	3	2	1	0
Read Write	PTBPE							
Reset	0	0	0	0	0	0	0	0

PORT_PTBPE field descriptions

Field	Description
PTBPE	Pull Enable for Port B Bit These bits determines if the internal pullup/pulldown device is enabled for the associated PTB pin. For port B pins that are configured as outputs, these bits have no effect and the internal pullup/pulldown devices are disabled. **NOTE:** PTB0-PTB2 is default pulldown, can not be pullup enabled. PTB3-PTB5 can be pulldown enabled. 0 Internal pullup//pulldown device disabled for port B bit n. 1 Internal pullup/pulldown device enabled for port B bit n.

8.5.9 Port C Pullup Enable Register (PORT_PTCPE)

Address: 0h base + 18E2h offset = 18E2h

Bit	7	6	5	4	3	2	1	0
Read	0							PTCPE
Write								
Reset	0	0	0	0	0	0	0	0

PORT_PTCPE field descriptions

Field	Description
7–1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 PTCPE	Pull Enable for Port C Bit These bits determines if the internal pullup device is enabled for the associated PTC pin. For port C pins that are configured as outputs, these bits have no effect and the internal pullup devices are disabled. 0 Internal pullup device disabled for port C bit n. 1 Internal pullup device enabled for port C bit n.

8.5.10 Port B High Drive Strength Selection Register (PORT_PTBHD)

Output extreme high drive strength sink/source current can be enabled by setting the corresponding bit in the PORT_PTBHD register for PTB7. Output extremely high sink/source current is enabled when they are operated as output. Extreme high drive function is disabled if the pin is configured as an input by the parallel I/O control logic. When configured as any shared peripheral function, extreme high drive function still works on these pins, but only when they are configured as outputs.

Address: 0h base + 18E6h offset = 18E6h

Bit	7	6	5	4	3	2	1	0
Read	HD7	0						
Write								
Reset	0	0	0	0	0	0	0	0

PORT_PTBHD field descriptions

Field	Description
7 HD7	Output High Drive Strength Selection for Port B Bit 7 This bit enables the extreme high drive capability of associated PTB pin. 0 Low output drive enabled for port B bit 7. 1 High output drive enabled for port B bit 7.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

8.5.11 Port Clock Division Register (PORT_FCLKDIV)

Configure the high/low level glitch width threshold. Glitches that are shorter than the selected clock width will be filtered out; glitches that are more than twice the selected clock width will not be filtered out (they will pass to the internal circuitry).

Address: 0h base + 18ECh offset = 18ECh

Bit	7	6	5	4	3	2	1	0
Read	FLTDIV3			FLTDIV2			FLTDIV1	
Write								
Reset	0	0	0	0	0	0	0	0

PORT_FCLKDIV field descriptions

Field	Description
7–5 FLTDIV3	Filter Division Set 3

Table continues on the next page...

PORT_FCLKDIV field descriptions (continued)

Field	Description
	Port Filter Division Set 3 000 LPOCLK. 001 LPOCLK/2. 010 LPOCLK/4. 011 LPOCLK/8. 100 LPOCLK/16. 101 LPOCLK/32. 110 LPOCLK/64. 111 LPOCLK/128.
4–2 FLTDIV2	Filter Division Set 2 Port Filter Division Set 2 000 BUSCLK/32. 001 BUSCLK/64. 010 BUSCLK/128. 011 BUSCLK/256. 100 BUSCLK/512. 101 BUSCLK/1024. 110 BUSCLK/2048. 111 BUSCLK/4096.
FLTDIV1	Filter Division Set 1 Port Filter Division Set 1 00 BUSCLK/2. 01 BUSCLK/4. 10 BUSCLK/8. 11 BUSCLK/16.

8.5.12 Port Filter Register 0 (PORT_IOFLT0)

This register sets the filters for input from PTA to PTC.

Address: 0h base + 18EDh offset = 18EDh

Bit	7	6	5	4	3	2	1	0
Read	0		FLTC		FLTB		FLTA	
Write								
Reset	0	0	0	0	0	0	0	0

PORT_IOFLT0 field descriptions

Field	Description
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

Table continues on the next page...

PORT_IOFLT0 field descriptions (continued)

Field	Description
5–4 FLTC	Filter selection for input from PTC 00 BUSCLK 01 FLTDIV1 10 FLTDIV2 11 FLTDIV3
3–2 FLTB	Filter selection for input from PTB 00 BUSCLK 01 FLTDIV1 10 FLTDIV2 11 FLTDIV3
FLTA	Filter selection for input from PTA 00 BUSCLK 01 FLTDIV1 10 FLTDIV2 11 FLTDIV3

8.5.13 Port Filter Register 1 (PORT_IOFLT1)

This register sets the filters for input.

Address: 0h base + 18EEh offset = 18EEh

Bit	7	6	5	4	3	2	1	0
Read Write	FLTKBI		FLTRST		0			
Reset	0	0	0	0	0	0	0	0

PORT_IOFLT1 field descriptions

Field	Description
7–6 FLTKBI	Filter selection for input from KBI 00 No filter 01 Select FLTDIV1, and will switch to FLTDIV3 in stop mode automatically. 10 Select FLTDIV2, and will switch to FLTDIV3 in stop mode automatically. 11 FLTDIV3
5–4 FLTRST	Filter selection for input from RESET 00 No filter. 01 Select FLTDIV1, and will switch to FLTDIV3 in stop mode automatically. 10 Select FLTDIV2, and will switch to FLTDIV3 in stop mode automatically. 11 FLTDIV3
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

8.5.14 Port Filter Register 2 (PORT_IOFLT2)

This register sets the filters for input from PWT, I2C and XB.

Address: 0h base + 18EFh offset = 18EFh

Bit	7	6	5	4	3	2	1	0
Read Write	FLTPWT1		FLTPWT0		FLTI2C		FLTXBI	
Reset	0	0	0	0	0	0	0	0

PORT_IOFLT2 field descriptions

Field	Description
7–6 FLTPWT1	Filter Selection For Input from PWT1 00 No filter 01 FLTDIV1 10 FLTDIV2 11 BUSCLK
5–4 FLTPWT0	Filter Selection For Input from PWT0 00 No filter 01 FLTDIV1 10 FLTDIV2 11 BUSCLK
3–2 FLTI2C	Filter Selection For Input from SDA and SCL. 00 No filter 01 FLTDIV1 10 FLTDIV2 11 BUSCLK
FLTXBI	Filter Selection For Input from XB_IN0 and XB_IN1 00 No filter 01 FLTDIV1 10 FLTDIV2 11 FLTDIV3

Chapter 9
System Integration Module (SIM)

9.1 Chip specific windowed COP

The windowed COP (WCOP) module triggers a system reset if it is allowed to time out. The program is expected to periodically reload the COP timer, thereby preventing it from timing out. However, if a fault occurs that causes the program to stop working, the timer will not be reloaded and it will time out. The resulting trigger of a system reset brings the system back from an unresponsive state into a normal state.

After any reset, the WCOP is enabled. If the WCOP is not used in an application, it can be disabled by clearing SIM_SOPT1[COPT].

The WCOP counter is reset by writing 0x55 and 0xAA (in that order) to the address of the SIM_SRS during the selected timeout period. Writes do not affect the data in that field. As soon as the write sequence is complete, the WCOP timeout period is restarted. If the program fails to perform this restart during the timeout period, the microcontroller resets. Also, if any value other than 0x55 or 0xAA is written to the SIM_SRS register, the microcontroller immediately resets.

Windowed watchdog operation is available by setting SIM_SOPT1[COPW], In this mode, writes to service watchdog register SIM_SRS to clear WCOP counter must be in a selected timeout period. A premature write immediately resets the chip.

WCOP has four clock selections: BUSCLK (20 MHz), ICSIRCLK (up to the 32 kHz) and an independent clock source LPOCLK (up to the 20 kHz), CLKIN

Customization:

- Primary clock: BUSCLK (20 MHz)
- Input clock option: BUSCLK (20 MHz); ICSIRCLK (up to the 32 kHz), LPOCLK (up to the 20 kHz), CLKIN (40 MHz)
- WCOP is a part of SIM. Its register set is a subset of SIM registers.

Module Instances:

- One

9.2 System device identification (SDID)

This device is hard coded to the value 0x45 in the SIM_SDID registers.

9.3 Universally unique identification (UUID)

This device contains up to 64-bit UUID to identify each device in this family. The intent of UUID is to enable distributed systems to uniquely identify information without significant central coordination.

9.4 Reset and system initialization

Resetting the MCU provides a way to start processing from a set of known initial conditions. During reset, most control and status registers are forced to initial values and the program counter is loaded from the reset vector (0xFFFE:0xFFFF). On-chip peripheral modules are disabled and I/O pins are initially configured as default functions on ALT0. The CCR [I] bit is set to block maskable interrupts so that the user program has a chance to initialize the stack pointer (SP) and system control settings. SP is forced to 0x00FF at reset.

This device has the following sources for reset:

- Power-on reset (POR)
- Low-voltage detect (LVD)
- Windowed COP (COP) timer
- Illegal opcode detect (ILOP)
- Illegal address detect (ILAD)
- Background debug forced reset
- External reset pin (RESET)
- Loss of clock reset (LOC)
- Flash illegal access detect (FILA)

Each of these sources, with the exception of the background debug forced reset, has an associated bit in the system reset status (SRS) register.

When the MCU is reset by SIM_SRS[ILAD], the address of illegal address is captured in illegal address register, which is a 16-bit register consisting of SIM_ILLAL and SIM_ILLAH that contains the LSB and MSB 8-bit of the address, respectively.

9.5 Computer operating properly (COP) watchdog

The COP watchdog is used to force a system reset when the application software fails to execute as expected. To prevent a system reset from the COP timer (when it is enabled), application software must reset the COP counter periodically. If the application program gets lost and fails to reset the COP counter before it times out, a system reset is generated to force the system back to an known starting point. After any reset, the COP watchdog is enabled (see System Options Register 1 (SIM_SOPT1) for additional information). If the COP watchdog is not used in an application, it can be disabled by clearing SOPT1[COPT].

The COP counter is reset by writing 0x55 and 0xAA (in this order) to the address of SRS during the selected timeout period. Writes do not affect the data in the read-only SRS. As soon as the write sequence is done, the COP timeout period is restarted. If the program fails to do this during the time-out period, the MCU will reset. Also, if any value other than 0x55 or 0xAA is written to SRS, the MCU is immediately reset. The SOPT1[COPCLKS] selects the clock source used for the COP timer. The clock source options are

- bus clock;
- internal 20 kHz LPO clock source
- 32 kHz ICSIRCLK
- CLK_IN from external pin

With each clock source, there are three associated time-outs controlled by SOPT1[COPT]. The following table summaries the control functions of the SOPT1[COPCLKS] and SOPT1[COPT] bits. The COP watchdog defaults to operation from the 20 kHz LPO clock source and the longest time-out (2^{10} cycles).

When the bus clock source is selected, windowed COP operation is available by setting SOPT1[COPW]. In this mode, writes to the SRS register to clear the COP timer must occur in the last 25% of the selected timeout period. A premature write immediately resets the MCU. When the LPO, ICSIRCLK, or CLK_IN clock source is selected, windowed COP operation is not available. The COP counter is initialized by the first writes to the SOPT1 registers and after any system reset. Subsequent writes to SOPT1 have no effect on COP operation. Even if the application will use the reset default settings of SOPT1[COPT], SOPT1[COPCLKS], and SOPT1[COPW] bits, the user must write to the write-once SOPT1 register during reset initialization to lock in the settings.

This will prevent accidental changes if the application program gets lost. The write to SRS that services (clears) the COP counter must not be placed in an interrupt service routine (ISR) because the ISR could continue to be executed periodically even if the main application program fails.

If the bus clock source is selected, the COP counter does not increment while the MCU is in background debug mode or while the system is in stop mode. The COP counter resumes when the MCU exits background debug mode or stop mode.

If the 20 kHz LPO, ICSIRCLK, or CLK_IN clock source is selected, the COP counter is re-initialized to zero upon entry to either background debug mode or stop mode and begins from zero upon exit from background debug mode or stop mode.

Table 9-1. Configuration option

Control bits		Clock source	COP window opens[1]	COP overflow count
SOPT1[COPCLKS]	SOPT1[COPT]			
N/A	00	N/A	N/A	COP is disabled
00/10	01	20 kHz LPOCLK / ICSIRCLK	N/A	2^5 cycles
00/10	10	20 kHz LPOCLK / ICSIRCLK	N/A	2^8 cycles
00/10	11	20 kHz LPOCLK / ICSIRCLK	N/A	2^{10} cycles
01	01	BUSCLK	6,144 cycles	2^{13} cycles
01	10	BUSCLK	49,152 cycles	2^{16} cycles
01	11	BUSCLK	196,608 cycles	2^{18} cycles
11	01	CLK_IN	N/A	2^{13} cycles
11	10	CLK_IN	N/A	2^{16} cycles
11	11	CLK_IN	N/A	2^{18} cycles

1. Windowed COP operation requires the user to clear the COP timer in the last 25% of the selected timeout period. This column displays the minimum number of clock counts required before the COP timer can be reset when in windowed COP mode (SOPT1[COPW] = 1).

9.6 System options

9.6.1 BKGD pin

After POR, PTB7/CLKOUT/BKGD/MS pin functions as BKGD output. Other functions are selected by SIM_MUXPTBH[MUXPTB7]. This pin is an output only when configured as PTB7.

9.6.2 RESET_b pin enable

After POR reset, PTB6/TCLK0/RESET_b functions as $\overline{\text{RESET}}$. Other functions are selected by SIM_MUXPTBH[MUXPTB6]. The pin is a true open-drain output when it is configured as PTB6. internal pullup can pull the output to logic high if it is enabled.

9.7 System interconnection

This device contains a set of system-level logics for module-to-module interconnection for flexible configuration.

9.7.1 Inter Module Crossbar Switch (XBAR)

The Inter Module Crossbar Switch module provides a generic mechanism for making connections between on-chip peripherals as well as between peripherals and pins. It provides a purely combinational path from input to output. The module groups 16 identical multiplexes with 19 shared inputs.

In general, the crossbar module connects the mcPWM, ADC, Timers, and comparators together, which allows synchronization between PWM pulse generation and ADC sampling. In addition, several crossbar inputs and outputs are routed to package pins. For example, the user can define an XB_INn pin as a PWM fault protection input that is routed to the PWM module through the crossbar, increasing the flexibility of pin use and reducing the complexity of PCB layout.

Default XBAR output is connected to logic low.

Customization:

- Primary clock: No
- Alternate clock: No
- The following table summarizes the signal connection of XBAR.

Table 9-2. XBAR module input signals from

Module	XBAR_INx	Function
Package Pin	XBAR_IN0	XB_IN0
Package Pin	XBAR_IN1	XB_IN1

Table continues on the next page...

Table 9-2. XBAR module input signals from (continued)

Module	XBAR_INx	Function
Logic Low	XBAR_IN2	Logic Low
Logic High	XBAR_IN3	Logic High
PWM_Sync	XBAR_IN4	XBAR_IN4
PWM0 or PWM1	XBAR_IN5	PWM0 or PWM1, selected by a 2-to-1 Mux
PWM2 or PWM3	XBAR_IN6	PWM2 or PWM3, selected by a 2-to-1 Mux
PWM4 or PWM5	XBAR_IN7	PWM4 or PWM5, selected by a 2-to-1 Mux
PDB Channel0	XBAR_IN8	PDB_Ch0 Output
PDB Channel1	XBAR_IN9	PDB_Ch1 Output
FTM Channel0	XBAR_IN10	FTM_Ch0 Output
FTM Channel1	XBAR_IN11	FTM_Ch1 Output
GDU Phase A ACMP	XBAR_IN12	GDU phase A detection comparator output
GDU Phase B ACMP	XBAR_IN13	GDU phase B detection comparator output
GDU Phase C ACMP	XBAR_IN14	GDU phase C detection comparator output
CMP	XBAR_IN15	High speed analog comparator output

Table 9-3. XBAR module output signals to

Module	XBAR_OUTx	Function
Package Pin	XBAR_OUT0	XB_OUT0
Package Pin	XBAR_OUT1	XB_OUT1
ADC0	XBAR_OUT2	ADC0 hardware trigger input
ADC1	XBAR_OUT3	ADC1 Hardware trigger input
FTM Channel0	XBAR_OUT4	FTM_Ch0 input capture or output
FTM Channel1	XBAR_OUT5	FTM_Ch1 input capture or output

Table continues on the next page...

Table 9-3. XBAR module output signals to (continued)

Module	XBAR_OUTx	Function
PWT0	XBAR_OUT6	PWT0 input capture
PWT1	XBAR_OUT7	PWT1 input capture
PDB Channel0	XBAR_OUT8	PDB_Ch0 hardware trigger input
PDB Channel1	XBAR_OUT9	PDB_Ch1 hardware trigger input
GDU Phase A ACMP	XBAR_OUT10	GDU phase A detection comparator window
GDU Phase B ACMP	XBAR_OUT11	GDU phase B detection comparator window
GDU Phase C ACMP	XBAR_OUT12	GDU phase C detection comparator window
CMP	XBAR_OUT13	High speed analog comparator window
PWM FAULT1	XBAR_OUT14	PWM Fault1
IRQ	XBAR_OUT15	IRQ

9.7.2 Module to module interconnects

In addition to Inter module crossbar connection, There are signals that are directly connected between modules. The following lists the module to module interconnections

Table 9-4. module to module signals connection

Module	Signal	Connect to
PWM	PWM0	GDU Predriver Phase A Top
PWM	PWM1	GDU Predriver Phase A Bottom
PWM	PWM2	GDU Predriver Phase B Top
PWM	PWM3	GDU Predriver Phase B Bottom
PWM	PWM4	GDU Predriver Phase C Top
PWM	PWM5	GDU Predriver Phase C Bottom
GDU	Limit CMP0 OUT	PWM Fault2
GDU	Limit CMP1 OUT	PWM Fault3

9.8 Memory map and register definition

The SIM module contains many fields for selecting the clock source and dividers for various module clocks.

SIM memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
1800	System Reset Status Register (SIM_SRS)	8	R	82h	9.8.1/105
1801	System Background Debug Force Reset Register (SIM_SBDFR)	8	W (always reads 0)	00h	9.8.2/107
1802	System Device Identification Register: High (SIM_SDIDH)	8	R	00h	9.8.3/107
1803	System Device Identification Register: Low (SIM_SDIDL)	8	R	45h	9.8.4/108
1804	System Options Register 1 (SIM_SOPT1)	8	R/W	C0h	9.8.5/108
1805	System Options Register 2 (SIM_SOPT2)	8	R/W	00h	9.8.6/110
1806	System Port A Pin Multiplexing Control Register: Low (SIM_MUXPTAL)	8	R/W	00h	9.8.7/111
1807	System Port A Pin Multiplexing Control Register: High (SIM_MUXPTAH)	8	R/W	00h	9.8.8/112
1808	System Port B Pin Multiplexing Control Register: Low (SIM_MUXPTBL)	8	R/W	00h	9.8.9/113
1809	System Port B Pin Multiplexing Control Register: High (SIM_MUXPTBH)	8	R/W	00h	9.8.10/114
180A	System Port C Pin Multiplexing Control Register: Low (SIM_MUXPTCL)	8	R/W	00h	9.8.11/116
180C	System Clock Gating Control 1 Register (SIM_SCGC1)	8	R/W	30h	9.8.12/116
180D	System Clock Gating Control 2 Register (SIM_SCGC2)	8	R/W	00h	9.8.13/117
180E	System Clock Gating Control 3 Register (SIM_SCGC3)	8	R/W	00h	9.8.14/119
180F	System Clock Divider Register (SIM_SCDIV)	8	R/W	20h	9.8.15/120
1810	System POR Register (SIM_PORREG0)	8	R/W	00h	9.8.16/121
1811	System POR Register (SIM_PORREG1)	8	R/W	00h	9.8.16/121
1812	System POR Register (SIM_PORREG2)	8	R/W	00h	9.8.16/121
1813	System POR Register (SIM_PORREG3)	8	R/W	00h	9.8.16/121
1814	System POR Register (SIM_PORREG4)	8	R/W	00h	9.8.16/121
1815	System POR Register (SIM_PORREG5)	8	R/W	00h	9.8.16/121
1816	System POR Register (SIM_PORREG6)	8	R/W	00h	9.8.16/121
1817	System POR Register (SIM_PORREG7)	8	R/W	00h	9.8.16/121
1880	Illegal Address Register: High (SIM_ILLAH)	8	R	Undefined	9.8.17/122
1881	Illegal Address Register: Low (SIM_ILLAL)	8	R	Undefined	9.8.18/122
18F8	Universally Unique Identifier Register 0 (SIM_UUID0)	8	R	Undefined	9.8.19/123

Table continues on the next page...

SIM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
18F9	Universally Unique Identifier Register 1 (SIM_UUID1)	8	R	Undefined	9.8.20/123
18FA	Universally Unique Identifier Register 2 (SIM_UUID2)	8	R	Undefined	9.8.21/124
18FB	Universally Unique Identifier Register 3 (SIM_UUID3)	8	R	Undefined	9.8.22/124
18FC	Universally Unique Identifier Register 4 (SIM_UUID4)	8	R	Undefined	9.8.23/125
18FD	Universally Unique Identifier Register 5 (SIM_UUID5)	8	R	Undefined	9.8.24/125
18FE	Universally Unique Identifier Register 6 (SIM_UUID6)	8	R	Undefined	9.8.25/126
18FF	Universally Unique Identifier Register 7 (SIM_UUID7)	8	R	Undefined	9.8.26/126

9.8.1 System Reset Status Register (SIM_SRS)

This register includes read-only status flags to indicate the source of the most recent reset. When a debug host forces reset by writing 1 to the SIM_SBDFR[BDFR] bit, none of the status bits in SRS will be set. The reset state of these bits depends on what caused the MCU to reset.

NOTE

For PIN, COP, and ILOP, any of these reset sources that are active at the time of reset (not including POR or LVR) will cause the corresponding bit(s) to be set; bits corresponding to sources that are not active at the time of reset will be cleared.

The $\overline{\text{RESET}}$ values in the figure are values for power on reset; for other resets, the values depend on the trigger causes.

Address: 1800h base + 0h offset = 1800h

Bit	7	6	5	4	3	2	1	0
Read	POR	PIN	WCOP	ILOP	ILAD	LOC	LVD	FILA
Write								
Reset	1	0	0	0	0	0	1	0

SIM_SRS field descriptions

Field	Description
7 POR	Power-On Reset Reset was caused by the power-on detection logic. When the internal supply voltage was ramping up at the time, the low-voltage reset (LVR) status bit is also set to indicate that the reset occurred while the internal supply was below the LVR threshold. **NOTE:** This bit POR to 1, LVR to uncertain value and reset to 0 at any other conditions.

Table continues on the next page...

SIM_SRS field descriptions (continued)

Field	Description
	0 Reset not caused by POR. 1 POR caused reset.
6 PIN	External Reset Pin Reset was caused by an active low level on the external reset pin. 0 Reset not caused by external reset pin. 1 Reset came from external reset pin.
5 WCOP	Windowed COP Reset was caused by the WCOP timer timing out. This reset source may be blocked by SIM_SOPT1[COPT] =00. 0 Reset not caused by WCOP timeout. 1 Reset caused by WCOP timeout.
4 ILOP	Illegal Opcode Reset was caused by an attempt to execute an unimplemented or illegal opcode. The STOP instruction is considered illegal if stop is disabled by SIM_SOPT1[STOPE] = 0. The BGND instruction is considered illegal if active background mode is disabled by BDC_SCR[ENBDM] = 0. 0 Reset not caused by an illegal opcode. 1 Reset caused by an illegal opcode.
3 ILAD	Illegal Address Reset was caused by an attempt to access a illegal address. The illegal address is captured in illegal address register (ILLAH:ILLAL). 0 Reset not caused by an illegal address. 1 Reset caused by an illegal address.
2 LOC	Internal Clock Source Module Reset Reset was caused by an ICS module reset. 0 Reset not caused by ICS module. 1 Reset caused by ICS module.
1 LVD	Low Voltage Detect This bit is set by POR. **NOTE:** This bit reset to 1 on POR and LVR and reset to 0 on other reset. 0 Reset not caused by LVD trip or POR. 1 Reset caused by LVD trip or POR.
0 FILA	Flash Illegal Access Reset was caused by an flash illegal access. 0 Reset not caused by an flash illegal access. 1 Reset caused by flash illegal access.

9.8.2 System Background Debug Force Reset Register (SIM_SBDFR)

This register contains a single write-only control bit. A serial background command such as WRITE_BYTE must be used to write to SIM_SBDFR. Attempts to write this register from a user program are ignored. Reads always return 0x00.

NOTE

This register is the same as the BDC_SBDFR.

Address: 1800h base + 1h offset = 1801h

Bit	7	6	5	4	3	2	1	0
Read	0							0
Write								BDFR
Reset	0	0	0	0	0	0	0	0

SIM_SBDFR field descriptions

Field	Description
7–1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 BDFR	Background Debug Force Reset A serial background command such as WRITE_BYTE may be used to allow an external debug host to force a target system reset. Writing logic 1 to this bit forces an MCU reset. This bit cannot be written from a user program. **NOTE:** BDFR is writable only through serial background debug commands, not from user programs.

9.8.3 System Device Identification Register: High (SIM_SDIDH)

This read-only register, together with SDIDL, is included so that host development systems can identify the HCS08 derivative and revision number. This allows the development software to recognize where specific memory blocks, registers, and control bits are located in a target MCU.

Address: 1800h base + 2h offset = 1802h

Bit	7	6	5	4	3	2	1	0
Read	Reserved				ID			
Write								
Reset	0	0	0	0	0	0	0	0

SIM_SDIDH field descriptions

Field	Description
7–4 Reserved	This field is reserved.
ID	Part Identification Number These bits, together with the SDIDL, indicate part identification number. Each derivative in the HCS08 family has a unique identification number. This device is hard coded to the value 0x45.

9.8.4 System Device Identification Register: Low (SIM_SDIDL)

This read-only register, together with SDIDH, is included so host development systems can identify the HCS08 derivative and revision number. This allows the development software to recognize where specific memory blocks, registers, and control bits are located in a target MCU.

Address: 1800h base + 3h offset = 1803h

Bit	7	6	5	4	3	2	1	0
Read	ID							
Write								
Reset	0	1	0	0	0	1	0	1

SIM_SDIDL field descriptions

Field	Description
ID	Part Identification Number These bits, together with the SDIDH, indicate part identification number. Each derivative in the HCS08 family has a unique identification number. This device is hard coded to the value 0x45.

9.8.5 System Options Register 1 (SIM_SOPT1)

Address: 1800h base + 4h offset = 1804h

Bit	7	6	5	4	3	2	1	0
Read Write	COPT		COPCLKS		COPW	RANGE	FWAKE	STOPE
Reset	1	1	0	0	0	0	0	0

SIM_SOPT1 field descriptions

Field	Description
7–6 COPT	COP Timeout These write-once bits selects the timeout period of the COP. COPT along with SOPT1[COPCLKS] defines the COP timeout period as described in Computer operating properly (COP) watchdog.

Table continues on the next page...

SIM_SOPT1 field descriptions (continued)

Field	Description
	00 COP is disabled. 01 2^5(LPOCLK/ICSIRCLK) or 2^{13}(BUSCLK/CLK_IN) cycles. 10 2^8(LPOCLK/ICSIRCLK) or 2^{16}(BUSCLK/CLK_IN) cycles. 01 2^{10}(LPOCLK/ICSIRCLK) or 2^{18}(BUSCLK/CLK_IN) cycles
5–4 COPCLKS	COP Clock Source Select These write-once bits selects the COP clock source. 00 COP logic runs from independent on-chip LPO clock source. 01 COP logic runs from system bus clock. 10 COP logic runs from ICSIRCLK. 11 COP logic runs from pin CLK_IN.
3 COPW	COP Window Mode Enable This write-once bit specifies whether the COP operates in Normal or Window mode. In Window mode, the 0x55-0xAA write sequence to the SRS register must occur within the last 25% of the selected period; any write to the SRS register during the first 75% of the selected period resets the microcontroller. 0 Normal mode. 1 Window mode.
2 RANGE	Frequency Range Select This write-once bit work with ICS_C1[RDIV] to divide external reference clock to 31.25–39.0625 kHz.. 0 Low frequency range of external clock(31.25 kHz–1 MHz). 1 High frequency range of external clock(1 MHz–40 MHz).
1 FWAKE	Fast Wakeup Enable This write once bit can set CPU wakeup without any interrupt subroutine serviced. This action saved more than 11 cycles(whole interrupt subroutine time). After wake up CPU continue the address before wait or stop. **NOTE:** When FWAKE is set, user should avoid generating interrupt 0~8 bus clock cycles after issuing the stop instruction, or the MCU may stuck at Stop mode and cannot wake up by interrupts. 0 CPU wakes up as normal. 1 CPU wakes up without any interrupt subroutine serviced.
0 STOPE	Stop Mode Enable This write-once bit defaults to 0 after reset, which disables stop mode. If stop mode is disabled and a user program attempts to execute a STOP instruction, an illegal opcode reset occurs. 0 Stop mode disabled. 1 Stop mode enabled.

9.8.6 System Options Register 2 (SIM_SOPT2)

This register may be read and written at any time.

Address: 1800h base + 5h offset = 1805h

Bit	7	6	5	4	3	2	1	0
Read	ESFC	WPXB	0			BUSREF		
Write								
Reset	0	0	0	0	0	0	0	0

SIM_SOPT2 field descriptions

Field	Description
7 ESFC	Enable Stalling Flash Controller Enables stalling flash controller when flash is busy. When software needs to access the flash memory while a flash memory resource is being manipulated by a flash command, software can enable a stall mechanism to avoid a read collision. The stall mechanism allows software to execute code from the same block on which flash operations are being performed. However, software must ensure the sector the flash operations are being performed on is not the same sector from which the code is executing. ESFC enables the stall mechanism. This bit must be set only just before the flash operation is executed and must be cleared when the operation completes. 0 Disable stalling flash controller when flash is busy. 1 Enable stalling flash controller when flash is busy.
6 WPXB	Write Protection for XBAR registers This field configures the write-protection for XBAR registers 0 XBAR registers are writable. 1 XBAR registers are not writable.
5–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
BUSREF	BUS Output select This bit enables bus clock output on PTB7 via an optional prescalar. 000 Bus. 001 Bus divided by 2. 010 Bus divided by 4. 011 Bus divided by 8. 100 Bus divided by 16. 101 Bus divided by 32. 110 Bus divided by 64. 111 Bus divided by 128.

9.8.7 System Port A Pin Multiplexing Control Register: Low (SIM_MUXPTAL)

Many of the I/O pins are shared with on-chip peripheral functions. This register selects the multiplexing pin functions from ALT0 to ALT3. Default is ALT0 function, When the Pin Muxing mode is configured for analog pins, all the digital functions on that pin are disabled, including the pullup/output/input.

The shared analog pin functions can work together if they're enabled separately because they're directly connected to internal analog modules separately. They still work even when pin Muxing mode is configured for digital pins.

NOTE

KBI share with PTA on ALT3 and can be enable/disable by KBI_PE register.

"RX and CLK_IN" function at different location must not be selected at the same time to prevent signal override or modulation.

Address: 1800h base + 6h offset = 1806h

Bit	7	6	5	4	3	2	1	0
Read Write	MUXPTA3		MUXPTA2		MUXPTA1		MUXPTA0	
Reset	0	0	0	0	0	0	0	0

SIM_MUXPTAL field descriptions

Field	Description
7–6 MUXPTA3	Pin Mux Control The corresponding pin is configured in the following pin muxing slot: 00 Alternative 0. 01 Alternative 1. 10 Alternative 2. 11 Alternative 3.
5–4 MUXPTA2	Pin Mux Control The corresponding pin is configured in the following pin muxing slot: 00 Alternative 0. 01 Alternative 1. 10 Alternative 2. 11 Alternative 3.
3–2 MUXPTA1	Pin Mux Control

Table continues on the next page...

SIM_MUXPTAL field descriptions (continued)

Field	Description
	The corresponding pin is configured in the following pin muxing slot: 00 Alternative 0. 01 Alternative 1. 10 Alternative 2. 11 Alternative 3.
MUXPTA0	Pin Mux Control The corresponding pin is configured in the following pin muxing slot: 00 Alternative 0. 01 Alternative 1. 10 Alternative 2. 11 Alternative 3.

9.8.8 System Port A Pin Multiplexing Control Register: High (SIM_MUXPTAH)

Many of the I/O pins are shared with on-chip peripheral functions. This register selects the multiplexing pin functions from ALT0 to ALT3. Default is ALT0 function, When the Pin Muxing mode is configured for analog pins, all the digital functions on that pin are disabled, including the pullup/output/input.

The shared analog pin functions can work together if they're enabled separately because they're directly connected to internal analog modules separately. They still work even when pin Muxing mode is configured for digital pins.

NOTE

KBI share with PTA on ALT3 and can be enable/disable by KBI_PE register.

"RX and CLK_IN" function at different location must not be selected at the same time to prevent signal override or modulation.

Address: 1800h base + 7h offset = 1807h

Bit	7	6	5	4	3	2	1	0
Read Write	MUXPTA7		MUXPTA6		MUXPTA5		MUXPTA4	
Reset	0	0	0	0	0	0	0	0

SIM_MUXPTAH field descriptions

Field	Description
7–6 MUXPTA7	Pin Mux Control The corresponding pin is configured in the following pin muxing slot: 00 Alternative 0. 01 Alternative 1. 10 Alternative 2. 11 Alternative 3.
5–4 MUXPTA6	Pin Mux Control The corresponding pin is configured in the following pin muxing slot: 00 Alternative 0. 01 Alternative 1. 10 Alternative 2. 11 Alternative 3.
3–2 MUXPTA5	Pin Mux Control The corresponding pin is configured in the following pin muxing slot: 00 Alternative 0. 01 Alternative 1. 10 Alternative 2. 11 Alternative 3.
MUXPTA4	Pin Mux Control The corresponding pin is configured in the following pin muxing slot: 00 Alternative 0. 01 Alternative 1. 10 Alternative 2. 11 Alternative 3.

9.8.9 System Port B Pin Multiplexing Control Register: Low (SIM_MUXPTBL)

Many of the I/O pins are shared with on-chip peripheral functions. This register selects the multiplexing pin functions from ALT0 to ALT3. Default is ALT0 function, When the Pin Muxing mode is configured for analog pins, all the digital functions on that pin are disabled, including the pullup/output/input.

The shared analog pin functions can work together if they’re enabled separately because they’re directly connected to internal analog modules separately. They still work even when pin Muxing mode is configured for digital pins.

Address: 1800h base + 8h offset = 1808h

Bit	7	6	5	4	3	2	1	0
Read Write	MUXPTB3		MUXPTB2		MUXPTB1		MUXPTB0	
Reset	0	0	0	0	0	0	0	0

SIM_MUXPTBL field descriptions

Field	Description
7–6 MUXPTB3	Pin Mux Control The corresponding pin is configured in the following pin muxing slot: 00 Alternative 0. 01 Alternative 1. 10 Alternative 2. 11 Alternative 3.
5–4 MUXPTB2	Pin Mux Control The corresponding pin is configured in the following pin muxing slot: 00 Alternative 0. 01 Alternative 1. 10 Alternative 2. 11 Alternative 3.
3–2 MUXPTB1	Pin Mux Control The corresponding pin is configured in the following pin muxing slot: 00 Alternative 0. 01 Alternative 1. 10 Alternative 2. 11 Alternative 3.
MUXPTB0	Pin Mux Control The corresponding pin is configured in the following pin muxing slot: 00 Alternative 0. 01 Alternative 1. 10 Alternative 2. 11 Alternative 3.

9.8.10 System Port B Pin Multiplexing Control Register: High (SIM_MUXPTBH)

Many of the I/O pins are shared with on-chip peripheral functions. This register selects the multiplexing pin functions from ALT0 to ALT3. Default is ALT0 function, When the Pin Muxing mode is configured for analog pins, all the digital functions on that pin are disabled, including the pullup/output/input.

The shared analog pin functions can work together if they're enabled separately because they're directly connected to internal analog modules separately. They still work even when pin Muxing mode is configured for digital pins.

Address: 1800h base + 9h offset = 1809h

Bit	7	6	5	4	3	2	1	0
Read Write	MUXPTB7		MUXPTB6		MUXPTB5		MUXPTB4	
Reset	0	0	0	0	0	0	0	0

SIM_MUXPTBH field descriptions

Field	Description
7–6 MUXPTB7	Pin Mux Control The corresponding pin is configured in the following pin muxing slot: 00 Alternative 0. 01 Alternative 1. 10 Alternative 2. 11 Alternative 3.
5–4 MUXPTB6	Pin Mux Control The corresponding pin is configured in the following pin muxing slot: **NOTE:** This field is is POR/LVR only. 00 Alternative 0. 01 Alternative 1. 10 Alternative 2. 11 Alternative 3.
3–2 MUXPTB5	Pin Mux Control The corresponding pin is configured in the following pin muxing slot: 00 Alternative 0. 01 Alternative 1. 10 Alternative 2. 11 Alternative 3.
MUXPTB4	Pin Mux Control The corresponding pin is configured in the following pin muxing slot: 00 Alternative 0. 01 Alternative 1. 10 Alternative 2. 11 Alternative 3.

9.8.11 System Port C Pin Multiplexing Control Register: Low (SIM_MUXPTCL)

Many of the I/O pins are shared with on-chip peripheral functions. This register selects the multiplexing pin functions from ALT0 to ALT3. Default is ALT0 function, When the Pin Muxing mode is configured for analog pins, all the digital functions on that pin are disabled, including the pullup/output/input.

The shared analog pin functions can work together if they're enabled separately because they're directly connected to internal analog modules separately. They still work even when pin Muxing mode is configured for digital pins.

Address: 1800h base + Ah offset = 180Ah

Bit	7	6	5	4	3	2	1	0
Read	0							MUXPTC0
Write								
Reset	0	0	0	0	0	0	0	0

SIM_MUXPTCL field descriptions

Field	Description
7–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
MUXPTC0	Pin Mux Control The corresponding pin is configured in the following pin muxing slot: 00 Alternative 0. 01 Alternative 1. 10 Alternative 2. 11 Alternative 3.

9.8.12 System Clock Gating Control 1 Register (SIM_SCGC1)

This register contains control bits to enable or disable the bus clock to the PMC, DBG, NVM, IPC and CRC modules. Gating off the clocks to unused peripherals is used to reduce the MCU's run and wait currents.

NOTE

User software must disable the peripheral before disabling the clocks to the peripheral. When clocks are re-enabled to a peripheral, the peripheral registers need to be re-initialized by user software.

Address: 1800h base + Ch offset = 180Ch

Bit	7	6	5	4	3	2	1	0
Read	0	PMC	DBG	NVM	IPC	CRC	0	
Write								
Reset	0	0	1	1	0	0	0	0

SIM_SCGC1 field descriptions

Field	Description
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6 PMC	PMC Clock Gate Control This bit controls the clock gate to the PMC module. 0 Bus clock to the PMC module is disabled. 1 Bus clock to the PMC module is enabled.
5 DBG	DBG Clock Gate Control This bit controls the clock gate to the DBG module. 0 Bus clock to the DBG module is disabled. 1 Bus clock to the DBG module is enabled.
4 NVM	NVM Clock Gate Control This bit controls the clock gate to the NVM module. 0 Bus clock to the NVM module is disabled. 1 Bus clock to the NVM module is enabled.
3 IPC	IPC Clock Gate Control This bit controls the clock gate to the IPC module. 0 Bus clock to the IPC module is disabled. 1 Bus clock to the IPC module is enabled.
2 CRC	CRC Clock Gate Control This bit controls the clock gate to the CRC module. 0 Bus clock to the CRC module is disabled. 1 Bus clock to the CRC module is enabled.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

9.8.13 System Clock Gating Control 2 Register (SIM_SCGC2)

This register contains control bits to enable or disable the bus clock to the CMP0, GDU, ADC, IRQ, PDB and KBI modules. Gating off the clocks to unused peripherals is used to reduce the MCU's run and wait currents.

NOTE

User software must disable the peripheral before disabling the clocks to the peripheral. When clocks are re-enabled to a peripheral, the peripheral registers need to be re-initialized by user software.

Address: 1800h base + Dh offset = 180Dh

Bit	7	6	5	4	3	2	1	0
Read	CMP0	GDU_CMP	ADC1	ADC0	IRQ	0	PDB	KBI
Write								
Reset	0	0	0	0	0	0	0	0

SIM_SCGC2 field descriptions

Field	Description
7 CMP0	CMP0 Clock Gate Control This bit controls the clock gate to the CMP0 module. 0 Bus clock to the CMP0 module is disabled. 1 Bus clock to the CMP0 module is enabled.
6 GDU_CMP	GDU_CMP Clock Gate Control This bit controls the clock gate to the GDU_CMP module. 0 Bus clock to the GDU_CMP module is disabled. 1 Bus clock to the GDU_CMP module is enabled.
5 ADC1	ADC1 Clock Gate Control This bit controls the clock gate to the ADC1 module. 0 Bus clock to the ADC1 module is disabled. 1 Bus clock to the ADC1 module is enabled.
4 ADC0	ADC0 Clock Gate Control This bit controls the clock gate to the ADC0 module. 0 Bus clock to the ADC0 module is disabled. 1 Bus clock to the ADC0 module is enabled.
3 IRQ	IRQ Clock Gate Control This bit controls the clock gate to the IRQ module. 0 Bus clock to the IRQ module is disabled. 1 Bus clock to the IRQ module is enabled.
2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 PDB	PDB Clock Gate Control This bit controls the clock gate to the PDB module.

Table continues on the next page...

SIM_SCGC2 field descriptions (continued)

Field	Description
	0 Bus clock to the PDB module is disabled. 1 Bus clock to the PDB module is enabled.
0 KBI	KBI Clock Gate Control This bit controls the clock gate to the KBI module. 0 Bus clock to the KBI module is disabled. 1 Bus clock to the KBI module is enabled.

9.8.14 System Clock Gating Control 3 Register (SIM_SCGC3)

This page register contains control bits to enable or disable the bus clock to the I2C, PWM, MTIM, PWT, SCI and FTM modules. Gating off the clocks to unused peripherals is used to reduce the MCU's run and wait currents.

NOTE

User software must disable the peripheral before disabling the clocks to the peripheral. When clocks are re-enabled to a peripheral, the peripheral registers need to be re-initialized by user software.

Address: 1800h base + Eh offset = 180Eh

Bit	7	6	5	4	3	2	1	0
Read	I2C	PWM	MTIM	PWT1	PWT0	SCI	FTM	0
Write								
Reset	0	0	0	0	0	0	0	0

SIM_SCGC3 field descriptions

Field	Description
7 I2C	I2C Clock Gate Control This bit controls the clock gate to the I2C module. 0 Bus clock to the I2C module is disabled. 1 Bus clock to the I2C module is enabled.
6 PWM	PWM Clock Gate Control This bit controls the clock gate to the PWM module. 0 Bus clock to the PWM module is disabled. 1 Bus clock to the PWM module is enabled.
5 MTIM	MTIM Clock Gate Control This bit controls the clock gate to the MTIM module.

Table continues on the next page...

SIM_SCGC3 field descriptions (continued)

Field	Description
	0 Bus clock to the MTIM module is disabled. 1 Bus clock to the MTIM module is enabled.
4 PWT1	PWT1 Clock Gate Control This bit controls the clock gate to the PWT1 module. 0 Bus clock to the PWT1 module is disabled. 1 Bus clock to the PWT1 module is enabled.
3 PWT0	PWT0 Clock Gate Control This bit controls the clock gate to the PWT0 module. 0 Bus clock to the PWT0 module is disabled. 1 Bus clock to the PWT0 module is enabled.
2 SCI	SCI Clock Gate Control This bit controls the clock gate to the SCI module. 0 Bus clock to the SCI module is disabled. 1 Bus clock to the SCI module is enabled.
1 FTM	FTM Clock Gate Control This bit controls the clock gate to the FTM module. 0 Bus clock to the FTM module is disabled. 1 Bus clock to the FTM module is enabled.
0 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

9.8.15 System Clock Divider Register (SIM_SCDIV)

This register sets the divide value for the clock.

Address: 1800h base + Fh offset = 180Fh

Bit	7	6	5	4	3	2	1	0
Read	0		DIV1	0		DIV2	0	DIV3
Write								
Reset	0	0	1	0	0	0	0	0

SIM_SCDIV field descriptions

Field	Description
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5 DIV1	Clock 1 output divider value This field sets the divide value for the core/system clock.

Table continues on the next page...

SIM_SCDIV field descriptions (continued)

Field	Description
	0 Same as ICSOUTCLK. 1 ICSOUTCLK divides by 2.
4–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 DIV2	Clock 2 output divider value This field sets the divide value for the bus/flash, follow DIV1. **NOTE:** The bus clock max speed is 20 MHz. When CPU clock is up to 40 MHz, CPU:Bus could not be 1:1,but only 2:1. 0 Not divided from divider1. 1 Divide by 2 from divider1.
1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 DIV3	Clock 3 output divider value This field sets the divide value for the PDB/PWM. 0 Same as ICSOUTCLK. 1 ICSOUTCLK divides by 2.

9.8.16 System POR Register (SIM_PORREG*n*)

Address: 1800h base + 10h offset + (1d × i), where i=0d to 7d

Bit	7	6	5	4	3	2	1	0
Read Write	PORREG							
Reset	0	0	0	0	0	0	0	0

SIM_PORREG*n* field descriptions

Field	Description
PORREG	Power-on-reset only registers These 8 byte registers can only be reset by power-on.

9.8.17 Illegal Address Register: High (SIM_ILLAH)

The ILLAH is a read-only register containing the high 8-bit of the illegal address of ILAD reset.

Address: 1800h base + 80h offset = 1880h

Bit	7	6	5	4	3	2	1	0
Read	ADDR[15:8]							
Write								
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

SIM_ILLAH field descriptions

Field	Description
ADDR[15:8]	High 8-bit of illegal address **NOTE:** For ILAD, it reset to the high 8-bit of the illegal address; in other cases, the reset to values are undetermined.

9.8.18 Illegal Address Register: Low (SIM_ILLAL)

The ILLAL is a read-only register containing the low 8-bit of the illegal address of ILAD reset.

Address: 1800h base + 81h offset = 1881h

Bit	7	6	5	4	3	2	1	0
Read	ADDR[7:0]							
Write								
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

SIM_ILLAL field descriptions

Field	Description
ADDR[7:0]	Low 8-bit of illegal address **NOTE:** For ILAD, it resets to the low 8-bit of the illegal address; in other cases, the reset to values are undetermined.

9.8.19 Universally Unique Identifier Register 0 (SIM_UUID0)

The read-only UUIDx registers contain a series of 64-bit number to identify the unique device in the family.

Address: 1800h base + F8h offset = 18F8h

Bit	7	6	5	4	3	2	1	0
Read	ID[63:56]							
Write								
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

SIM_UUID0 field descriptions

Field	Description
ID[63:56]	Universally Unique Identifier

9.8.20 Universally Unique Identifier Register 1 (SIM_UUID1)

The read-only UUIDx registers contain a series of 63-bit number to identify the unique device in the family.

Address: 1800h base + F9h offset = 18F9h

Bit	7	6	5	4	3	2	1	0
Read	ID[55:48]							
Write								
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

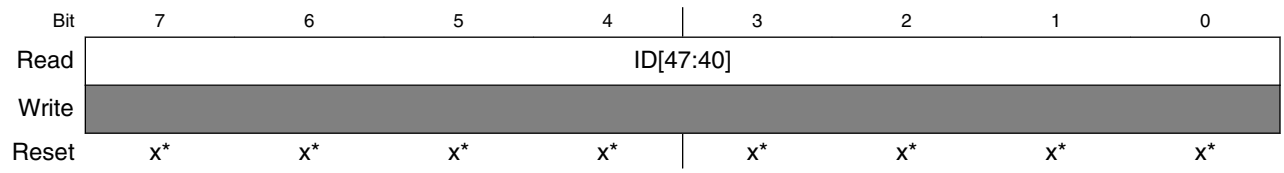
SIM_UUID1 field descriptions

Field	Description
ID[55:48]	Universally Unique Identifier

9.8.21 Universally Unique Identifier Register 2 (SIM_UUID2)

The read-only UUIDx registers contain a series of 63-bit number to identify the unique device in the family.

Address: 1800h base + FAh offset = 18FAh



* Notes:
- x = Undefined at reset.

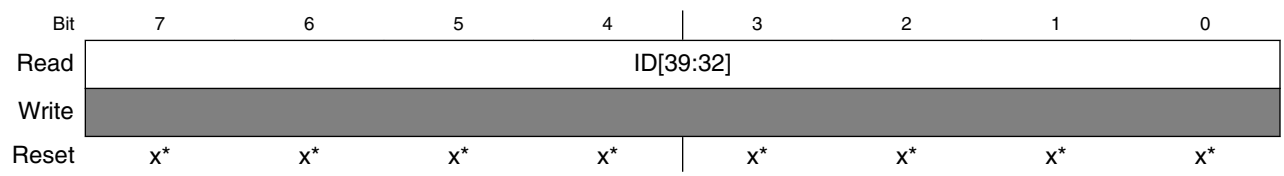
SIM_UUID2 field descriptions

Field	Description
ID[47:40]	Universally Unique Identifier

9.8.22 Universally Unique Identifier Register 3 (SIM_UUID3)

The read-only UUIDx registers contain a series of 63-bit number to identify the unique device in the family.

Address: 1800h base + FBh offset = 18FBh



* Notes:
- x = Undefined at reset.

SIM_UUID3 field descriptions

Field	Description
ID[39:32]	Universally Unique Identifier

9.8.23 Universally Unique Identifier Register 4 (SIM_UUID4)

The read-only UUIDx registers contain a series of 64-bit number to identify the unique device in the family.

Address: 1800h base + FCh offset = 18FCh

Bit	7	6	5	4	3	2	1	0
Read	ID[31:24]							
Write								
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

SIM_UUID4 field descriptions

Field	Description
ID[31:24]	Universally Unique Identifier

9.8.24 Universally Unique Identifier Register 5 (SIM_UUID5)

The read-only UUIDx registers contain a series of 64-bit number to identify the unique device in the family.

Address: 1800h base + FDh offset = 18FDh

Bit	7	6	5	4	3	2	1	0
Read	ID[23:16]							
Write								
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

SIM_UUID5 field descriptions

Field	Description
ID[23:16]	Universally Unique Identifier

9.8.25 Universally Unique Identifier Register 6 (SIM_UUID6)

The read-only UUIDx registers contain a series of 64-bit number to identify the unique device in the family.

Address: 1800h base + FEh offset = 18FEh

Bit	7	6	5	4	3	2	1	0
Read	ID[15:8]							
Write								
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

SIM_UUID6 field descriptions

Field	Description
ID[15:8]	Universally Unique Identifier

9.8.26 Universally Unique Identifier Register 7 (SIM_UUID7)

The read-only UUIDx registers contain a series of 64-bit number to identify the unique device in the family.

Address: 1800h base + FFh offset = 18FFh

Bit	7	6	5	4	3	2	1	0
Read	ID[7:0]							
Write								
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

SIM_UUID7 field descriptions

Field	Description
ID[7:0]	Universally Unique Identifier

Chapter 10
Central processor unit

10.1 Introduction

This section provides summary information about the registers, addressing modes, special operations, instructions and exceptions processing of the HCS08 V6 CPU.

The HCS08 V6 CPU is fully source- and object-code-compatible with the HCS08 CPU.

10.1.1 Features

Features of the HCS08 V6 CPU include:

- Object code fully upward-compatible with M68HC05 and M68HC08 families
- 16-bit stack pointer (any size stack anywhere in 64 KB CPU address space)
- 16-bit index register (H:X) with powerful indexed addressing modes
- 8-bit accumulator (A)
- Many instructions treat X as a second general-purpose 8-bit register
- Seven addressing modes:
  - Inherent — Operands in internal registers
  - Relative — 8-bit signed offset to branch destination
  - Immediate — Operand in next object code byte(s)
  - Direct — Operand in memory at 0x0000–0x00FF
  - Extended — Operand anywhere in 64-Kbyte address space

- Indexed relative to H:X — Five submodes including auto increment
- Indexed relative to SP — Improves C efficiency dramatically

- Memory-to-memory data move instructions with four address mode combinations
- Overflow, half-carry, negative, zero, and carry condition codes support conditional branching on the results of signed, unsigned, and binary-coded decimal (BCD) operations
- Efficient bit manipulation instructions
- STOP and WAIT instructions to invoke low-power operating modes

10.2 Programmer's Model and CPU Registers

Figure 10-1 shows the five CPU registers. CPU registers are not part of the memory map.

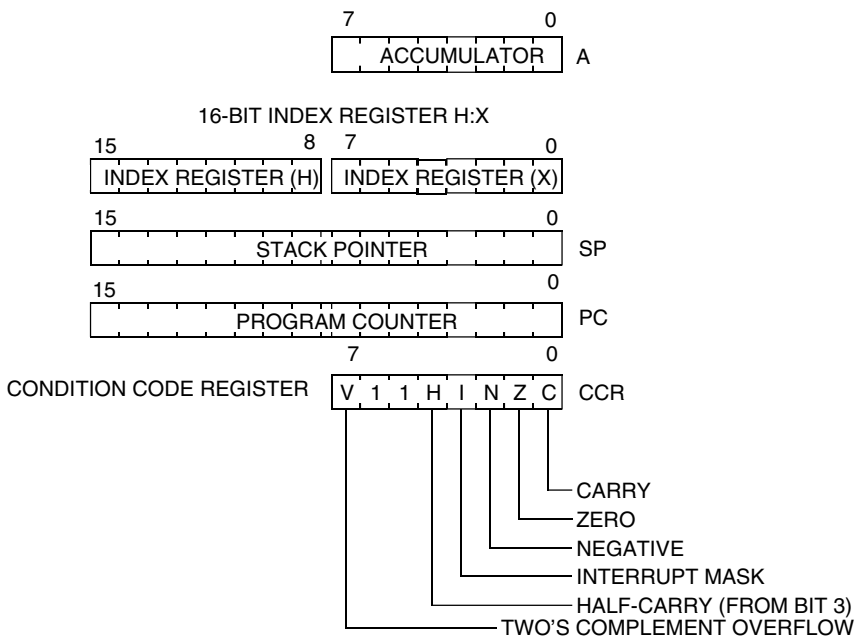


Figure 10-1. CPU Registers

10.2.1 Accumulator (A)

The A accumulator is a general-purpose 8-bit register. One input operand from the arithmetic logic unit (ALU) is connected to the accumulator, and the ALU results are often stored into the A accumulator after arithmetic and logical operations. The

accumulator can be loaded from memory using various addressing modes to specify the address where the loaded data comes from, or the contents of A can be stored to memory using various addressing modes to specify the address where data from A will be stored.

Reset has no effect on the contents of the A accumulator.

10.2.2 Index Register (H:X)

This 16-bit register is actually two separate 8-bit registers (H and X), which often work together as a 16-bit address pointer where H holds the upper byte of an address and X holds the lower byte of the address. All indexed addressing mode instructions use the full 16-bit value in H:X as an index reference pointer; however, for compatibility with the earlier M68HC05 family, some instructions operate only on the low-order 8-bit half (X).

Many instructions treat X as a second general-purpose 8-bit register that can be used to hold 8-bit data values. X can be cleared, incremented, decremented, complemented, negated, shifted, or rotated. Transfer instructions allow data to be transferred from A or transferred to A where arithmetic and logical operations can then be performed.

For compatibility with the earlier M68HC05 family, H is forced to 0x00 during reset. Reset has no effect on the contents of X.

10.2.3 Stack Pointer (SP)

This 16-bit address pointer register points at the next available location on the automatic last-in-first-out (LIFO) stack. The stack may be located anywhere in the 64 KB address space that has RAM, and can be any size up to the amount of available RAM. The stack is used to automatically save the return address for subroutine calls, the return address and CPU registers during interrupts, and for local variables. The AIS (add immediate to stack pointer) instruction adds an 8-bit signed immediate value to SP. This is most often used to allocate or deallocate space for local variables on the stack.

SP is forced to 0x00FF at reset for compatibility with the earlier M68HC05 family. HCS08 V6 programs normally change the value in SP to the address of the last location (highest address) in on-chip RAM during reset initialization to free up direct page RAM (from the end of the on-chip registers to 0x00FF).

The RSP (reset stack pointer) instruction was included for compatibility with the M68HC05 family and is seldom used in new HCS08 V6 programs because it affects only the low-order half of the stack pointer.

10.2.4 Program Counter (PC)

The program counter is a 16-bit register that contains the address of the next instruction or operand to be fetched.

During normal program execution, the program counter automatically increments to the next sequential memory location every time an instruction or operand is fetched. Jump, branch, interrupt, and return operations load the program counter with an address other than that of the next sequential location. This is called a change-of-flow.

During reset, the program counter is loaded with the reset vector that is located at 0xFFFE and 0xFFFF. The vector stored there is the address of the first instruction that will be executed after exiting the reset state.

10.2.5 Condition Code Register (CCR)

The 8-bit condition code register contains the interrupt mask (I) and five flags that indicate the results of the instruction just executed. Bits 6 and 5 are set permanently to 1. The following paragraphs describe the functions of the condition code bits in general terms.

Table 10-1. CCR Register Field Descriptions

Field	Description
7 V	**Two's Complement Overflow Flag** — The CPU sets the overflow flag when a two's complement overflow occurs. The signed branch instructions BGT, BGE, BLE, and BLT use the overflow flag. 0 No overflow 1 Overflow
4 H	**Half-Carry Flag** — The CPU sets the half-carry flag when a carry occurs between accumulator bits 3 and 4 during an add-without-carry (ADD) or add-with-carry (ADC) operation. The half-carry flag is required for binary-coded decimal (BCD) arithmetic operations. The DAA instruction uses the states of the H and C condition code bits to automatically add a correction value to the result from a previous ADD or ADC on BCD operands to correct the result to a valid BCD value. 0 No carry between bits 3 and 4 1 Carry between bits 3 and 4
3 I	**Interrupt Mask Bit** — When the interrupt mask is set, all maskable CPU interrupts are disabled. CPU interrupts are enabled when the interrupt mask is cleared. When a CPU interrupt occurs, the interrupt mask is set automatically after the CPU registers are saved on the stack, but before the first instruction of the interrupt service routine is executed. Interrupts are not recognized at the instruction boundary after any instruction that clears I (CLI or TAP). This ensures that the next instruction after a CLI or TAP will always be executed without the possibility of an intervening interrupt, provided I was set. 0 Interrupts enabled

Table continues on the next page...

Table 10-1. CCR Register Field Descriptions (continued)

Field	Description
	1 Interrupts disabled
2 N	**Negative Flag** — The CPU sets the negative flag when an arithmetic operation, logic operation, or data manipulation produces a negative result, setting bit 7 of the result. Simply loading or storing an 8-bit or 16-bit value causes N to be set if the most significant bit of the loaded or stored value was 1. 0 Non-negative result 1 Negative result
1 Z	**Zero Flag** — The CPU sets the zero flag when an arithmetic operation, logic operation, or data manipulation produces a result of 0x00 or 0x0000. Simply loading or storing an 8-bit or 16-bit value causes Z to be set if the loaded or stored value was all 0s. 0 Non-zero result 1 Zero result
0 C	**Carry/Borrow Flag** — The CPU sets the carry/borrow flag when an addition operation produces a carry out of bit 7 of the accumulator or when a subtraction operation requires a borrow. Some instructions — such as bit test and branch, shift, and rotate — also clear or set the carry/borrow flag. 0 No carry out of bit 7 1 Carry out of bit 7

10.3 Addressing Modes

Addressing modes define the way the CPU accesses operands and data. In the HCS08 V6, memory, status and control registers, and input/output (I/O) ports share a single 64 KB CPU address space. This arrangement means that the same instructions that access variables in RAM can also be used to access I/O and control registers or nonvolatile program space.

Some instructions use more than one addressing mode. For instance, move instructions use one addressing mode to specify the source operand and a second addressing mode to specify the destination address. Instructions such as BRCLR, BRSET, CBEQ, and DBNZ use one addressing mode to specify the location of an operand for a test and then use relative addressing mode to specify the branch destination address when the tested condition is true. For BRCLR, BRSET, CBEQ, and DBNZ, the addressing mode listed in the instruction set tables is the addressing mode needed to access the operand to be tested, and relative addressing mode is implied for the branch destination.

Every addressing mode, except inherent, generates a 16-bit effective address. The effective address is the address of the memory location that the instruction acts on. Effective address computations do not require extra execution cycles. The HCS08 V6 CPU uses the 16 addressing modes described in the following sections.

10.3.1 Inherent Addressing Mode (INH)

In this addressing mode, instructions either have no operands or all operands are in internal CPU registers. In either case, the CPU does not need to access any memory locations to complete the instruction. Examples:

```
NOP          ;this instruction has no operands
CLRA         ;operand is a CPU register
```

10.3.2 Relative Addressing Mode (REL)

Relative addressing mode is used to specify the destination location for branch instructions. A signed two's complement byte offset value is located in the memory location immediately following the opcode. The offset gives a branching range of -128 to +127 bytes. In most assemblers, the programmer does not need to calculate the offset, because the assembler determines the proper offset and verifies that it is within the span of the branch.

During program execution, if a branch condition is true, the signed offset is sign-extended to a 16-bit value and is added to the current contents of the program counter, which causes program execution to continue at the branch destination address. If a branch condition is false, the CPU executes the next instruction.

10.3.3 Immediate Addressing Mode (IMM)

The operand for instructions with the immediate addressing mode is contained in the byte(s) immediately following the opcode. The byte or bytes that follow the opcode are the value of the statement rather than the address of the value. The pound symbol (#) is used to indicate an immediate addressing mode operand. One very common programming error is to accidentally omit the # symbol. This causes the assembler to misinterpret the following expression as an address rather than explicitly provided data. For example LDA #$55 means to load the immediate value $55 into the accumulator, while LDA $55 means to load the value from address $0055 into the accumulator. Without the # symbol, the instruction is erroneously interpreted as a direct addressing instruction.

Example:

```
LDA       #$55
CPHX      #$FFFF
LDHX      #$67
```

The size of the immediate operand is implied by the instruction context. In the third example, the instruction implies a 16-bit immediate value, but only an 8-bit value is supplied. In this case the assembler generates the 16-bit value $0067 because the CPU expects a 16-bit value in the instruction stream.

10.3.4 Direct Addressing Mode (DIR)

This addressing mode is sometimes called zero-page addressing because it accesses operands in the address range $0000 through $00FF. Since these addresses always begin with $00, only the low byte of the address needs to be included in the instruction, which saves program space and execution time. A system can be optimized by placing the most commonly accessed data in this area of memory. The low byte of the operand address is supplied with the instruction and the high byte of the address is assumed to be zero.

Examples:

```
LDA       $55
```

The value $55 is taken to be the low byte of an address in the range $0000 through $00FF. The high byte of the address is assumed to be zero. During execution, the CPU combines the value $55 from the instruction with the assumed value of $00 to form the address $0055, which is then used to access the data to be loaded into accumulator.

```
LDHX      $20
```

In this example, the value $20 is combined with the assumed value of $00 to form the address $0020. Since the LDHX instruction requires a 16-bit value, a 16-bit word of data is read from addresses $0020 and $0021. After execution, the H:X index register has the value from address $0020 in its high byte and the value from address $0021 in its low byte. The same happens for CPHX and STHX.

```
BRSET     0,$80,foo
```

In this example, direct addressing is used to access the operand and relative addressing is used to identify the destination address of a branch, in case the branch-taken conditions are met. This is also the case for BRCLR.

10.3.5 Extended Addressing Mode (EXT)

In extended addressing, the full 16-bit address of the memory location to be operated on is provided in the instruction. Extended addressing can access any location in the 64 KB memory map.

Example:

```
LDA     $F03B
```

This instruction uses extended addressing because $F03B is above the zero page. In most assemblers, the programmer does not need to specify whether an instruction is direct or extended. The assembler automatically selects the shortest form of the instruction.

10.3.6 Indexed Addressing Mode

Indexed addressing mode has seven variations, including five that use the 16-bit H:X index register pair and two that use the stack pointer as the base reference.

10.3.6.1 Indexed, No Offset (IX)

Instructions using the indexed, no offset addressing mode are one-byte instructions that can access data with variable addresses. The X (Index register low byte) register contains the low byte of the conditional address of the operand and the H (Index register high byte) register contains the high byte of the address.

Indexed, no offset instructions can move a pointer through a table or hold the address of a frequently used RAM or input/output (I/O) location.

10.3.6.2 Indexed, No Offset with Post Increment (IX+)

Instructions using the indexed, no offset with post increment addressing mode are two-byte instructions that address the operands and then increment the Index register (H:X). The X (Index register low byte) register contains the low byte of the conditional address of the operand and the H (Index register high byte) register contains the high byte of the address. This addressing mode is usually used for table searches. MOV and CBEQ instructions use this addressing mode as well.

10.3.6.3 Indexed, 8-Bit Offset (IX1)

Indexed with 8-bit offset instructions are two-byte instructions that can access data with a variable address. The CPU adds the unsigned bytes in the H:X register to the unsigned byte immediately following the opcode. The sum is the effective address.

Indexed, 8-bit offset instructions are useful in selecting the k-th element in an n-element table. The table can begin anywhere and can extend as far as the address map allows. The k value would typically be in H:X, and the address of the beginning of the table would be

in the byte following the opcode. Using H:X in this way, this addressing mode is limited to the first 256 addresses in memory. Tables can be located anywhere in the address map when H:X is used as the base address, and the byte following the opcode is the offset.

10.3.6.4 Indexed, 8-Bit Offset with Post Increment (IX1+)

Indexed, 8-bit offset with post-increment instructions are three-byte instructions that access the operands with variable addresses, then increment H:X. The CPU adds the unsigned bytes in the H:X register to the byte immediately following the opcode. The sum is the effective address. This addressing mode is generally used for table searches. This addressing mode is used for CBEQ instruction.

10.3.6.5 Indexed, 16-Bit Offset (IX2)

Indexed, 16-bit offset instructions are three-byte instructions that can access data with variable addresses at any location in memory. The CPU adds the unsigned contents of H:X to the 16-bit unsigned word formed by the two bytes following the opcode. The sum is the effective address of the operand. The first byte after the opcode is the most significant byte of the 16-bit offset; the second byte is the least significant byte of the 16-bit offset. As with direct and extended addressing, most assemblers determine the shortest form of indexed addressing.

Indexed, 16-bit offset instructions are useful in selecting the k-th element in an n-element table. The table can begin anywhere and can extend as far as the address map allows. The k value would typically be in H:X, and the address of the beginning of the table would be in the bytes following the opcode.

10.3.6.6 SP-Relative, 8-Bit Offset (SP1)

Stack pointer, 8-bit offset instructions are three-byte instructions that address operands in much the same way as indexed 8-bit offset instructions, except that the 8-bit offset is added to the value of the stack pointer instead of the index register.

The stack pointer, 8-bit offset addressing mode permits easy addressing of data on the stack. The CPU adds the unsigned byte in the 16-bit stack pointer (SP) register to the unsigned byte following the opcode. The sum is the effective address of the operand. If interrupts are disabled, this addressing mode allows the stack pointer to be used as a second "index" register.

Stack pointer relative instructions require a pre-byte for access. Consequently, all SP relative instructions take one cycle longer than their index relative counterparts.

10.3.6.7 SP-Relative, 16-Bit Offset (SP2)

Stack pointer, 16-bit offset instructions are four-byte instructions used to access data relative to the stack pointer with variable addresses at any location in memory. The CPU adds the unsigned contents of the 16-bit stack pointer to the 16-bit unsigned word formed by the two bytes following the opcode. The sum is the effective address of the operand.

As with direct and extended addressing, most assemblers determine the shortest form of stack pointer addressing. Due to the pre-byte, stack pointer relative instructions take one cycle longer than their index relative counterparts.

Stack pointer, 16-bit offset instructions are useful in selecting the k-th element a an n-element table. The table can begin anywhere and can extend anywhere in memory. The k value would typically be in the stack pointer register, and the address of the beginning of the table is located in the two bytes following the two-byte opcode.

10.3.7 Memory to memory Addressing Mode

Memory to memory addressing mode has the following four variations.

10.3.7.1 Direct to Direct

This addressing mode is used to move data within the direct page of memory. Both the source operand and the destination operand are in the direct page. The source data is addressed by the first byte immediately following the opcode, and the destination location is addressed by the second byte following the opcode.

10.3.7.2 Immediate to Direct

This addressing mode is used to move an 8-bit constant to any location in the direct page memory. The source data is the byte immediately following the opcode, and the destination is addressed by the second byte following the opcode.

10.3.7.3 Indexed to Direct, Post Increment

Used only by the MOV instruction, this addressing mode accesses a source operand addressed by the H:X register, and a destination location within the direct page addressed by the byte following the opcode. H:X is incremented after the source operand is accessed.

10.3.7.4 Direct to Indexed, Post-Increment

Used only with the MOV instruction, this addressing mode accesses a source operand addressed by the byte following the opcode, and a destination location addressed by the H:X register. H:X is incremented after the destination operand is written.

10.4 Operation modes

The CPU can be placed into the following operation modes: stop, wait, background and security.

10.4.1 Stop mode

Usually, all system clocks, including the crystal oscillator (when used), are halted during stop mode to minimize power consumption. In such systems, external circuitry is needed to control the time spent in stop mode and to issue a signal to wake up the target MCU when it is time to resume processing. Unlike the earlier M68HC05 and M68HC08 MCUs, the HCS08 V6 can be configured to keep a minimum set of clocks running in stop mode. This optionally allows an internal periodic signal to wake the target MCU from stop mode.

When a host debug system is connected to the background debug pin (BKGD) and the ENBDM control bit has been set by a serial command through the background interface (or because the MCU was reset into active background mode), the oscillator is forced to remain active when the MCU enters stop mode. In this case, if a serial BACKGROUND command is issued to the MCU through the background debug interface while the CPU is in stop mode, CPU clocks will resume and the CPU will enter active background mode where other serial background commands can be processed. This ensures that a host development system can still gain access to a target MCU even if it is in stop mode.

10.4.2 Wait mode

The WAIT instruction enables interrupts by clearing the I bit in the CCR. It then halts the clocks to the CPU to reduce overall power consumption while the CPU is waiting for the interrupt or reset event that will wake the CPU from wait mode. When an interrupt or reset event occurs, the CPU clocks will resume and the interrupt or reset event will be processed normally.

If a serial BACKGROUND command is issued to the MCU through the background debug interface while the CPU is in wait mode, CPU clocks will resume and the CPU will enter active background mode where other serial background commands can be processed. This ensures that a host development system can still gain access to a target MCU even if it is in wait mode.

While in wait mode, there are some restrictions on which background debug commands can be used. Only the BACKGROUND command and memory-access-with-status commands are available while in wait mode. The memory-access-with-status commands do not allow memory access, but they report an error indicating that the CPU is in either stop or wait mode. The BACKGROUND command can be used to wake the CPU from wait mode and enter active background mode.

10.4.3 Background mode

Background instruction (BGND) is not used in normal user programs because it forces the CPU to stop processing user instructions and enter the active background mode waiting for serial background commands. The only way to resume execution of the user program is through reset or by a host debug system issuing a GO, TRACE1, or TAGGO serial command through the background debug interface.

Software-based breakpoints can be set by replacing an opcode at the desired breakpoint address with the BGND opcode. When the program reaches this breakpoint address, the CPU is forced to active background mode rather than continuing the user program.

The active background mode functions are managed through the background debug controller (BDC) in the HCS08 V6 core. The BDC provides the means for analyzing MCU operation during software development. Active background mode is entered in any of the following ways:

- When the BKGD pin is low at the time the MCU exits reset.
- When a BACKGROUND command is received through the BKGD pin.

- When a BGND instruction is executed.
- When encountering a BDC breakpoint.

Background commands are of two types:

- Non-intrusive commands, defined as commands that can be issued while the user program is running. Non-intrusive commands can be issued through the BKGD pin while the MCU is in run mode; non-intrusive commands can also be executed when the MCU is in the active background mode. Non-intrusive commands include:
  - Memory access commands
  - Memory-access-with-status commands
  - BDC register access commands
  - The BACKGROUND command
- Active background commands, which can be executed only while the MCU is in active background mode. Active background commands include commands to:
  - Read or write CPU registers
  - Trace one user program instruction at a time
  - Leave active background mode to return to the user's application program (GO)

The active background mode is used to program a bootloader or user application program into the flash program memory before the MCU is operated in run mode for the first time. The active background mode can also be used to erase and reprogram the flash memory after it has been previously programmed.

10.4.4 Security mode

Usually HCS08 V6 MCUs are implemented with a secure operating mode. When in secure mode, external access to internal memory is restricted, so that only instructions fetched from secure memory can access secure memory.

The method by which the MCU is put into secure mode is not defined by the HCS08 V6 Core. The core receives an external input signal that, when asserted, informs to the core that the MCU is in secure mode.

While in secure mode, the core controls the following set of conditions:

1. The RAM, flash, and EEPROM arrays are considered secure memory. All registers in Direct Page or High Page are considered non-secure memory.
2. Read data is tagged as either secure or non-secure during a program read, depending on whether the read is from secure or non-secure memory.
3. A data read of secure memory returns a value of $00 when the current instruction is tagged as non-secure or the access is a BDC access.
4. A data write to secure memory is blocked and data at the target address does not change state when the current instruction is tagged as non-secure or the access is through BDC.
5. A data write to secure memory is never blocked during the stacking cycles of interrupt service routines.
6. Data accesses to either secure or non-secure memory are allowed when the current instruction is tagged as secure.
7. BDC accesses to non-secure memory are allowed.

When the device is in the non-secure mode, secure memory is treated the same as non-secure memory, and all accesses are allowed.

Table 10-2 details the security conditions for allowing or disabling a read access.

Table 10-2. Security conditions for read access

Inputs conditions					Read control
Security enabled	Ram, flash or EEPROM access	Program or vector read	Current CPU instruction from secure memory	Current access is via BDC	Read access allowed
0	x	x	x	x	1
1	0	x	x	x	1
1	1	1	x	x	1
1	1	0	1	0	1
1	1	0	1	1	0
1	1	0	0	0	0
1	1	0	0	1	0

10.5 HCS08 V6 Opcodes

The HCS08 V6 Core has 254 one-byte opcodes and 47 two-byte opcodes, totaling 301 opcodes. For a more detailed description of the HCS08 V6 instructions please refer to the Instruction Set Summary section.

10.6 Special Operations

The CPU performs a few special operations that are similar to instructions but do not have opcodes like other CPU instructions. This section provides additional information about these operations.

10.6.1 Reset Sequence

Reset can be caused by a power-on-reset (POR) event, internal conditions such as the COP (computer operating properly) watchdog, or by assertion of an external active-low reset pin. When a reset event occurs, the CPU immediately stops whatever it is doing (the MCU does not wait for an instruction boundary before responding to a reset event).

The reset event is considered concluded when the sequence to determine whether the reset came from an internal source is done and when the reset pin is no longer asserted. At the conclusion of a reset event, the CPU performs a 6-cycle sequence to fetch the reset vector from $FFFE and $FFFF and to fill the instruction queue in preparation for execution of the first program instruction.

10.6.2 Interrupt Sequence

When an interrupt is requested, the CPU completes the current instruction before responding to the interrupt. At this point, the program counter is pointing at the start of the next instruction, which is where the CPU should return after servicing the interrupt. The CPU responds to an interrupt by performing the same sequence of operations as for a software interrupt (SWI) instruction, except the address used for the vector fetch is determined by the highest priority interrupt that is pending when the interrupt sequence started.

The CPU sequence for an interrupt is:

1. Store the contents of PCL, PCH, X, A, and CCR on the stack, in that order.
2. Set the I bit in the CCR.
3. Fetch the high-order half of the interrupt vector.
4. Fetch the low-order half of the interrupt vector.
5. Delay for one free bus cycle.

Fetch three bytes of program information starting at the address indicated by the interrupt vector to fill the instruction queue in preparation for execution of the first instruction in the interrupt service routine.

After the CCR contents are pushed onto the stack, the I bit in the CCR is set to prevent other interrupts while in the interrupt service routine. Although it is possible to clear the I bit with an instruction in the interrupt service routine, this would allow nesting of interrupts (which is not recommended because it leads to programs that are difficult to debug and maintain).

For compatibility with the earlier M68HC05 MCUs, the high-order half of the H:X index register pair (H) is not saved on the stack as part of the interrupt sequence. The user must use a PSHH instruction at the beginning of the service routine to save H and then use a PULH instruction just before the RTI that ends the interrupt service routine. It is not necessary to save H if you are certain that the interrupt service routine does not use any instructions or auto-increment addressing modes that might change the value of H.

The software interrupt (SWI) instruction is like a hardware interrupt except that it is not masked by the global I bit in the CCR and it is associated with an instruction opcode within the program so it is not asynchronous to program execution.

10.7 Instruction Set Summary

Table 10-3. Instruction Set Summary

Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Bus Cycles
			V	H	I	N	Z	C				
ADC #opr8i	Add with Carry	A ← (A) + (M) + (C)	↕	↕	–	↕	↕	↕	IMM	A9	ii	2
ADC opr8a			↕	↕	–	↕	↕	↕	DIR	B9	dd	3
ADC opr16a			↕	↕	–	↕	↕	↕	EXT	C9	hh ll	4
ADC oprx16,X			↕	↕	–	↕	↕	↕	IX2	D9	ee ff	4
ADC oprx8,X			↕	↕	–	↕	↕	↕	IX1	E9	ff	3
ADC ,X			↕	↕	–	↕	↕	↕	IX	F9		3
ADC oprx16,SP			↕	↕	–	↕	↕	↕	SP2	9ED9	ee ff	5
ADC oprx8,SP			↕	↕	–	↕	↕	↕	SP1	9EE9	ff	4

Table continues on the next page...

Table 10-3. Instruction Set Summary (continued)

Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Bus Cycles
			V	H	I	N	Z	C				
ADD #opr8i	Add without Carry	A ← (A) + (M)	↕	↕	–	↕	↕	↕	IMM	AB	ii	2
ADD opr8a			↕	↕	–	↕	↕	↕	DIR	BB	dd	3
ADD opr16a			↕	↕	–	↕	↕	↕	EXT	CB	hh ll	4
ADD oprx16,X			↕	↕	–	↕	↕	↕	IX2	DB	ee ff	4
ADD oprx8,X			↕	↕	–	↕	↕	↕	IX1	EB	ff	3
ADD ,X			↕	↕	–	↕	↕	↕	IX	FB		3
ADD oprx16,SP			↕	↕	–	↕	↕	↕	SP2	9EDB	ee ff	5
ADD oprx8,SP			↕	↕	–	↕	↕	↕	SP1	9EEB	ff	4
AIS #opr8i	Add Immediate Value (Signed) to Stack Pointer	SP ← (SP) + (M) where M is sign extended to a 16-bit value	–	–	–	–	–	–	IMM	A7	ii	2
AIX #opr8i	Add Immediate Value (Signed) to Index Register (H:X)	H:X ← (H:X) + (M) where M is sign extended to a 16-bit value	–	–	–	–	–	–	IMM	AF	ii	2
AND #opr8i	Logical AND	A ← (A) & (M)	0	–	–	↕	↕	–	IMM	A4	ii	2
AND opr8a			0	–	–	↕	↕	–	DIR	B4	dd	3
AND opr16a			0	–	–	↕	↕	–	EXT	C4	hh ll	4
AND oprx16,X			0	–	–	↕	↕	–	IX2	D4	ee ff	4
AND oprx8,X			0	–	–	↕	↕	–	IX1	E4	ff	3
AND ,X			0	–	–	↕	↕	–	IX	F4		3
AND oprx16,SP			0	–	–	↕	↕	–	SP2	9ED4	ee ff	5
AND oprx8,SP			0	–	–	↕	↕	–	SP1	9EE4	ff	4
ASL opr8a	Arithmetic Shift Left (same as LSL)	C ← MSB, LSB ← 0	↕	–	–	↕	↕	↕	DIR	38	dd	5
ASLA			↕	–	–	↕	↕	↕	INH	48		1
ASLX			↕	–	–	↕	↕	↕	INH	58		1
ASL oprx8,X			↕	–	–	↕	↕	↕	IX1	68	ff	5
ASL ,X			↕	–	–	↕	↕	↕	IX	78		4
ASL oprx8,SP			↕	–	–	↕	↕	↕	SP1	9E68	ff	6
ASR opr8a	Arithmetic Shift Right	MSB → MSB, LSB → C	↕	–	–	↕	↕	↕	DIR	37	dd	5
ASRA			↕	–	–	↕	↕	↕	INH	47		1
ASRX			↕	–	–	↕	↕	↕	INH	57		1
ASR oprx8,X			↕	–	–	↕	↕	↕	IX1	67	ff	5
ASR ,X			↕	–	–	↕	↕	↕	IX	77		4
ASR oprx8,SP			↕	–	–	↕	↕	↕	SP1	9E67	ff	6
BCC rel	Branch if Carry Bit Clear	Branch if (C) = 0	–	–	–	–	–	–	REL	24	rr	3
			–	–	–	–	–	–	DIR (b0)	11	dd	5

Table continues on the next page...

Table 10-3. Instruction Set Summary (continued)

Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Bus Cycles
			V	H	I	N	Z	C				
BCLR n,opr8a	Clear Bit n in Memory	Mn ← 0	–	–	–	–	–	–	DIR (b1)	13	dd	5
			–	–	–	–	–	–	DIR (b2)	15	dd	5
			–	–	–	–	–	–	DIR (b3)	17	dd	5
			–	–	–	–	–	–	DIR (b4)	19	dd	
			–	–	–	–	–	–	DIR (b5)	1B	dd	5
			–	–	–	–	–	–	DIR (b6)	1D	dd	5
			–	–	–	–	–	–	DIR (b7)	1F	dd	5
BCS rel	Branch if Carry Bit Set (same as BLO)	Branch if (C) = 1	–	–	–	–	–	–	REL	25	rr	3
BEQ rel	Branch if Equal	Branch if (Z) = 1	–	–	–	–	–	–	REL	27	rr	3
BGE rel	Branch if Greater Than or Equal To (Signed Operands)	Branch if (N ⊕ V) = 0	–	–	–	–	–	–	REL	90	rr	3
BGND	Enter Active Background if ENBDM = 1	Waits For and Processes BDM Commands Until GO, TRACE1, or TAGGO	–	–	–	–	–	–	INH	82		5+
BGT rel	Branch if Greater Than (Signed Operands)	Branch if (Z) ǀ (N ⊕ V) = 0	–	–	–	–	–	–	REL	92	rr	3
BHCC rel	Branch if Half Carry Bit Clear	Branch if (H) = 0	–	–	–	–	–	–	REL	28	rr	3
BHCS rel	Branch if Half Carry Bit Set	Branch if (H) = 1	–	–	–	–	–	–	REL	29	rr	3
BHI rel	Branch if Higher	Branch if (C) ǀ (Z) = 0	–	–	–	–	–	–	REL	22	rr	3
BHS rel	Branch if Higher or Same (same as BCC)	Branch if (C) = 0	–	–	–	–	–	–	REL	24	rr	3
BIH rel	Branch if IRQ Pin High	Branch if IRQ pin = 1	–	–	–	–	–	–	REL	2F	rr	3
BIL rel	Branch if IRQ Pin Low	Branch if IRQ pin = 0	–	–	–	–	–	–	REL	2E	rr	3
BIT #opr8i	Bit Test	(A) & (M), (CCR Updated but Operands Not Changed)	0	–	–	↕	↕	–	IMM	A5	ii	2
BIT opr8a			0	–	–	↕	↕	–	DIR	B5	dd	3
BIT opr16a			0	–	–	↕	↕	–	EXT	C5	hh ll	4
BIT oprx16,X			0	–	–	↕	↕	–	IX2	D5	ee ff	4
BIT oprx8,X			0	–	–	↕	↕	–	IX1	E5	ff	3
BIT ,X			0	–	–	↕	↕	–	IX	F5		3
BIT oprx16,SP			0	–	–	↕	↕	–	SP2	9ED5	ee ff	5
BIT oprx8,SP			0	–	–	↕	↕	–	SP1	9EE5	ff	4

Table continues on the next page...

Table 10-3. Instruction Set Summary (continued)

Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Bus Cycles
			V	H	I	N	Z	C				
BLE rel	Branch if Less Than or Equal To (Signed Operands)	Branch if (Z) | (N ⊕ V) = 1	–	–	–	–	–	–	REL	93	rr	3
BLO rel	Branch if Lower (Same as BCS)	Branch if (C) = 1	–	–	–	–	–	–	REL	25	rr	3
BLS rel	Branch if Lower or Same	Branch if (C) | (Z) = 1	–	–	–	–	–	–	REL	23	rr	3
BLT rel	Branch if Less Than (Signed Operands)	Branch if (N ⊕ V) = 1	–	–	–	–	–	–	REL	91	rr	3
BMC rel	Branch if Interrupt Mask Clear	Branch if (I) = 0	–	–	–	–	–	–	REL	2C	rr	3
BMI rel	Branch if Minus	Branch if (N) = 1	–	–	–	–	–	–	REL	2B	rr	3
BMS rel	Branch if Interrupt Mask Set	Branch if (I) = 1	–	–	–	–	–	–	REL	2D	rr	3
BNE rel	Branch if Not Equal	Branch if (Z) = 0	–	–	–	–	–	–	REL	26	rr	3
BPL rel	Branch if Plus	Branch if (N) = 0	–	–	–	–	–	–	REL	2A	rr	3
BRA rel	Branch Always	No Test	–	–	–	–	–	–	REL	20	rr	3
BRCLR n,opr8a,rel	Branch if Bit n in Memory Clear	Branch if (Mn) = 0	–	–	–	–	–	↕	DIR (b0)	01	dd rr	5
			–	–	–	–	–	↕	DIR (b1)	03	dd rr	5
			–	–	–	–	–	↕	DIR (b2)	05	dd rr	5
			–	–	–	–	–	↕	DIR (b3)	07	dd rr	5
			–	–	–	–	–	↕	DIR (b4)	09	dd rr	5
			–	–	–	–	–	↕	DIR (b5)	0B	dd rr	5
			–	–	–	–	–	↕	DIR (b6)	0D	dd rr	5
			–	–	–	–	–	↕	DIR (b7)	0F	dd rr	5
BRN rel	Branch Never	Uses 3 Bus Cycles	–	–	–	–	–	–	REL	21	rr	3
BRSET n,opr8a,rel	Branch if Bit n in Memory Set	Branch if (Mn) = 1	–	–	–	–	–	↕	DIR (b0)	00	dd rr	5
			–	–	–	–	–	↕	DIR (b1)	02	dd rr	5
			–	–	–	–	–	↕	DIR (b2)	04	dd rr	5
			–	–	–	–	–	↕	DIR (b3)	06	dd rr	5
			–	–	–	–	–	↕	DIR (b4)	08	dd rr	5
			–	–	–	–	–	↕	DIR (b5)	0A	dd rr	5
			–	–	–	–	–	↕	DIR (b6)	0C	dd rr	5
			–	–	–	–	–	↕	DIR (b7)	0E	dd rr	5
			–	–	–	–	–	–	DIR (b0)	10	dd	5
			–	–	–	–	–	–	DIR (b1)	12	dd	5
			–	–	–	–	–	–	DIR (b2)	14	dd	5

Table continues on the next page...

Table 10-3. Instruction Set Summary (continued)

Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Bus Cycles
			V	H	I	N	Z	C				
BSET n,opr8a	Set Bit n in Memory	Mn ← 1	–	–	–	–	–	–	DIR (b3)	16	dd	5
			–	–	–	–	–	–	DIR (b4)	18	dd	5
			–	–	–	–	–	–	DIR (b5)	1A	dd	5
			–	–	–	–	–	–	DIR (b6)	1C	dd	5
			–	–	–	–	–	–	DIR (b7)	1E	dd	5
BSR rel	Branch to Subroutine	PC ← (PC) + 0x0002 push (PCL) SP ← (SP) – 0x0001 push (PCH) SP ← (SP) – 0x0001 PC ← (PC) + rel	–	–	–	–	–	–	REL	AD	rr	5
CBEQ opr8a,rel	Compare and Branch if Equal	Branch if (A) = (M)	–	–	–	–	–	–	DIR	31	dd rr	5
CBEQA #opr8i,rel		Branch if (A) = (M)	–	–	–	–	–	–	IMM	41	ii rr	4
CBEQX #opr8i,rel		Branch if (X) = (M)	–	–	–	–	–	–	IMM	51	ii rr	4
CBEQ oprx8,X+,rel		Branch if (A) = (M)	–	–	–	–	–	–	IX1+	61	ff rr	5
CBEQ ,X+,rel		Branch if (A) = (M)	–	–	–	–	–	–	IX+	71	rr	5
CBEQ oprx8,SP,rel		Branch if (A) = (M)	–	–	–	–	–	–	SP1	9E61	ff rr	6
CLC	Clear Carry Bit	C ← 0	–	–	–	–	–	0	INH	98		1
CLI	Clear Interrupt Mask Bit	I ← 0	–	–	0	–	–	–	INH	9A		1
CLR opr8a	Clear	M ← 0x00	0	–	–	0	1	–	DIR	3F	dd	5
CLRA		A ← 0x00	0	–	–	0	1	–	INH	4F		1
CLRX		X ← 0x00	0	–	–	0	1	–	INH	5F		1
CLRH		H ← 0x00	0	–	–	0	1	–	INH	8C		1
CLR oprx8,X		M ← 0x00	0	–	–	0	1	–	IX1	6F	ff	5
CLR ,X		M ← 0x00	0	–	–	0	1	–	IX	7F		4
CLR oprx8,SP		M ← 0x00	0	–	–	0	1	–	SP1	9E6F	ff	6
CMP #opr8i	Compare Accumulator with Memory	(A) – (M); (CCR Updated But Operands Not Changed)	↕	–	–	↕	↕	↕	IMM	A1	ii	2
CMP opr8a			↕	–	–	↕	↕	↕	DIR	B1	dd	3
CMP opr16a			↕	–	–	↕	↕	↕	EXT	C1	hh ll	4
CMP oprx16,X			↕	–	–	↕	↕	↕	IX2	D1	ee ff	4
CMP oprx8,X			↕	–	–	↕	↕	↕	IX1	E1	ff	3
CMP ,X			↕	–	–	↕	↕	↕	IX	F1		3
CMP oprx16,SP			↕	–	–	↕	↕	↕	SP2	9ED1	ee ff	5

Table continues on the next page...

Table 10-3. Instruction Set Summary (continued)

Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Bus Cycles
			V	H	I	N	Z	C				
CMP oprx8,SP			↕	–	–	↕	↕	↕	SP1	9EE1	ff	4
COM opr8a	One's Complement	M ← (M) = 0xFF – (M)	0	–	–	↕	↕	1	DIR	33	dd	5
COMA		A ← (A) = 0xFF – (A)	0	–	–	↕	↕	1	INH	43		1
COMX		X ← (X) = 0xFF – (X)	0	–	–	↕	↕	1	INH	53		1
COM oprx8,X		M ← (M) = 0xFF – (M)	0	–	–	↕	↕	1	IX1	63	ff	5
COM ,X		M ← (M) = 0xFF – (M)	0	–	–	↕	↕	1	IX	73		4
COM oprx8,SP		M ← (M) = 0xFF – (M)	0	–	–	↕	↕	1	SP1	9E63	ff	6
CPHX opr16a	Compare Index Register (H:X) with Memory	(H:X) – (M:M + 0x0001); (CCR Updated But Operands Not Changed)	↕	–	–	↕	↕	↕	EXT	3E	hh ll	6
CPHX #opr16i			↕	–	–	↕	↕	↕	IMM	65	jj kk	3
CPHX opr8a			↕	–	–	↕	↕	↕	DIR	75	dd	5
CPHX oprx8,SP			↕	–	–	↕	↕	↕	SP1	9EF3	ff	6
CPX #opr8i	Compare X (Index Register Low) with Memory	(X) – (M); (CCR Updated But Operands Not Changed)	↕	–	–	↕	↕	↕	IMM	A3	ii	2
CPX opr8a			↕	–	–	↕	↕	↕	DIR	B3	dd	3
CPX opr16a			↕	–	–	↕	↕	↕	EXT	C3	hh ll	4
CPX oprx16,X			↕	–	–	↕	↕	↕	IX2	D3	ee ff	4
CPX oprx8,X			↕	–	–	↕	↕	↕	IX1	E3	ff	3
CPX ,X			↕	–	–	↕	↕	↕	IX	F3		3
CPX oprx16,SP			↕	–	–	↕	↕	↕	SP2	9ED3	ee ff	5
CPX oprx8,SP			↕	–	–	↕	↕	↕	SP1	9EE3	ff	4
DAA	Decimal Adjust Accumulator After ADD or ADC of BCD Values	$(A)_{10}$	U	–	–	↕	↕	↕	INH	72		1
DBNZ opr8a,rel	Decrement and Branch if Not Zero	Decrement A, X, or M Branch if (result) ≠ 0 Affects X, Not H	–	–	–	–	–	–	DIR	3B	dd rr	7
DBNZA rel			–	–	–	–	–	–	INH	4B	rr	4
DBNZX rel			–	–	–	–	–	–	INH	5B	rr	4
DBNZ oprx8,X,rel			–	–	–	–	–	–	IX1	6B	ff rr	7
DBNZ ,X,rel			–	–	–	–	–	–	IX	7B	rr	6
DBNZ oprx8,SP,rel			–	–	–	–	–	–	SP1	9E6B	ff rr	8
DEC opr8a		M ← (M) – 0x01	↕	–	–	↕	↕	–	DIR	3A	dd	5
DECA		A ← (A) – 0x01	↕	–	–	↕	↕	–	INH	4A		1
DECX		X ← (X) – 0x01	↕	–	–	↕	↕	–	INH	5A		1

Table continues on the next page...

Table 10-3. Instruction Set Summary (continued)

Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Bus Cycles
			V	H	I	N	Z	C				
DEC oprx8,X	Decrement	M ← (M) – 0x01	↕	–	–	↕	↕	–	IX1	6A	ff	5
DEC ,X		M ← (M) – 0x01	↕	–	–	↕	↕	–	IX	7A		4
DEC oprx8,SP		M ← (M) – 0x01	↕	–	–	↕	↕	–	SP1	9E6A	ff	6
DIV	Divide	A ← (H:A)÷(X), H ← Remainder	–	–	–	–	↕	↕	INH	52		6
EOR #opr8i	Exclusive OR Memory with Accumulator	A ← (A ⊕ M)	0	–	–	↕	↕	–	IMM	A8	ii	2
EOR opr8a			0	–	–	↕	↕	–	DIR	B8	dd	3
EOR opr16a			0	–	–	↕	↕	–	EXT	C8	hh ll	4
EOR oprx16,X			0	–	–	↕	↕	–	IX2	D8	ee ff	4
EOR oprx8,X			0	–	–	↕	↕	–	IX1	E8	ff	3
EOR ,X			0	–	–	↕	↕	–	IX	F8		3
EOR oprx16,SP			0	–	–	↕	↕	–	SP2	9ED8	ee ff	5
EOR oprx8,SP			0	–	–	↕	↕	–	SP1	9EE8	ff	4
INC opr8a	Increment	M ← (M) + 0x01	↕	–	–	↕	↕	–	DIR	3C	dd	5
INCA		A ← (A) + 0x01	↕	–	–	↕	↕	–	INH	4C		1
INCX		X ← (X) + 0x01	↕	–	–	↕	↕	–	INH	5C		1
INC oprx8,X		M ← (M) + 0x01	↕	–	–	↕	↕	–	IX1	6C	ff	5
INC ,X		M ← (M) + 0x01	↕	–	–	↕	↕	–	IX	7C		4
INC oprx8,SP		M ← (M) + 0x01	↕	–	–	↕	↕	–	SP1	9E6C	ff	6
JMP opr8a	Jump	PC ← Jump Address							DIR	BC	dd	3
JMP opr16a									EXT	CC	hh ll	4
JMP oprx16,X			–	–	–	–	–	–	IX2	DC	ee ff	4
JMP oprx8,X									IX1	EC	ff	3
JMP ,X									IX	FC		3
JSR opr8a	Jump to Subroutine	PC ← (PC) + n (n = 1, 2, or 3) Push (PCL) SP ← (SP) – 0x0001 Push (PCH) SP ← (SP) – 0x0001 PC ← Unconditional Address	–	–	–	–	–	–	DIR	BD	dd	5
JSR opr16a			–	–	–	–	–	–	EXT	CD	hh ll	6
JSR oprx16,X			–	–	–	–	–	–	IX2	DD	ee ff	6
JSR oprx8,X			–	–	–	–	–	–	IX1	ED	ff	5
JSR ,X			–	–	–	–	–	–	IX	FD		5
LDA #opr8i	Load Accumulator from Memory	A ← (M)	0	–	–	↕	↕	–	IMM	A6	ii	2
LDA opr8a									DIR	B6	dd	3
LDA opr16a									EXT	C6	hh ll	4
LDA oprx16,X									IX2	D6	ee ff	4
LDA oprx8,X									IX1	E6	ff	3

Table continues on the next page...

Table 10-3. Instruction Set Summary (continued)

Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Bus Cycles
			V	H	I	N	Z	C				
LDA ,X									IX	F6		3
LDA oprx16,SP									SP2	9ED6	ee ff	5
LDA oprx8,SP									SP1	9EE6	ff	4
LDHX #opr16i	Load Index Register (H:X) from Memory	H:X ← (M:M + 0x0001)	0	–	–	↕	↕	–	IMM	45	jj kk	3
LDHX opr8a			0	–	–	↕	↕	–	DIR	55	dd	4
LDHX opr16a			0	–	–	↕	↕	–	EXT	32	hh ll	5
LDHX ,X			0	–	–	↕	↕	–	IX	9EAE		5
LDHX oprx16,X			0	–	–	↕	↕	–	IX2	9EBE	ee ff	6
LDHX oprx8,X			0	–	–	↕	↕	–	IX1	9ECE	ff	5
LDHX oprx8,SP			0	–	–	↕	↕	–	SP1	9EFE	ff	5
LDX #opr8i	Load X (Index Register Low) from Memory	X ← (M)	0	–	–	↕	↕	–	IMM	AE	ii	2
LDX opr8a			0	–	–	↕	↕	–	DIR	BE	dd	3
LDX opr16a			0	–	–	↕	↕	–	EXT	CE	hh ll	4
LDX oprx16,X			0	–	–	↕	↕	–	IX2	DE	ee ff	4
LDX oprx8,X			0	–	–	↕	↕	–	IX1	EE	ff	3
LDX ,X			0	–	–	↕	↕	–	IX	FE		3
LDX oprx16,SP			0	–	–	↕	↕	–	SP2	9EDE	ee ff	5
LDX oprx8,SP			0	–	–	↕	↕	–	SP1	9EEE	ff	4
LSL opr8a	Logical Shift Left (Same as ASL)	C ← MSB, LSB ← 0	↕	–	–	↕	↕	↕	DIR	38	dd	5
LSLA			↕	–	–	↕	↕	↕	INH	48		1
LSLX			↕	–	–	↕	↕	↕	INH	58		1
LSL oprx8,X			↕	–	–	↕	↕	↕	IX1	68	ff	5
LSL ,X			↕	–	–	↕	↕	↕	IX	78		4
LSL oprx8,SP			↕	–	–	↕	↕	↕	SP1	9E68	ff	6
LSR opr8a	Logical Shift Right	0 → MSB, LSB → C	↕	–	–	0	↕	↕	DIR	34	dd	5
LSRA			↕	–	–	0	↕	↕	INH	44		1
LSRX			↕	–	–	0	↕	↕	INH	54		1
LSR oprx8,X			↕	–	–	0	↕	↕	IX1	64	ff	5
LSR ,X			↕	–	–	0	↕	↕	IX	74		4
LSR oprx8,SP			↕	–	–	0	↕	↕	SP1	9E64	ff	6
MOV oprx8a,opr8a	Move	$(M)_{destination} \leftarrow (M)_{source}$	0	–	–	↕	↕	–	DIR/DIR	4E	dd	5
MOV opr8a,X+			0	–	–	↕	↕	–	DIR/IX+	5E	dd	5

Table continues on the next page...

Table 10-3. Instruction Set Summary (continued)

Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Bus Cycles
			V	H	I	N	Z	C				
MOV #opr8i,opr8a		H:X ← (H:X) + 0x0001 in IX+/DIR and DIR/IX+ Modes	0	–	–	↕	↕	–	IMM/DIR	6E	ii	4
MOV ,X+,opr8a			0	–	–	↕	↕	–	IX+/DIR	7E	dd	5
MUL	Unsigned multiply	X:A ← (X) × (A)	–	0	–	–	–	0	INH	42		5
NEG opr8a	Negate (Two's Complement)	M ← – (M) = 0x00 – (M)	↕	–	–	↕	↕	↕	DIR	30	dd	5
NEGA		A ← – (A) = 0x00 – (A)	↕	–	–	↕	↕	↕	INH	40		1
NEGX		X ← – (X) = 0x00 – (X)	↕	–	–	↕	↕	↕	INH	50		1
NEG oprx8,X		M ← – (M) = 0x00 – (M)	↕	–	–	↕	↕	↕	IX1	60	ff	5
NEG ,X		M ← – (M) = 0x00 – (M)	↕	–	–	↕	↕	↕	IX	70		4
NEG oprx8,SP		M ← – (M) = 0x00 – (M)	↕	–	–	↕	↕	↕	SP1	9E60	ff	6
NOP	No Operation	Uses 1 Bus Cycle	–	–	–	–	–	–	INH	9D		1
NSA	Nibble Swap Accumulator	A ← (A[3:0]:A[7:4])	–	–	–	–	–	–	INH	62		1
ORA #opr8i	Inclusive OR Accumulator and Memory	A ← (A) | (M)	0	–	–	↕	↕	–	IMM	AA	ii	2
ORA opr8a			0	–	–	↕	↕	–	DIR	BA	dd	3
ORA opr16a			0	–	–	↕	↕	–	EXT	CA	hh ll	4
ORA oprx16,X			0	–	–	↕	↕	–	IX2	DA	ee ff	4
ORA oprx8,X			0	–	–	↕	↕	–	IX1	EA	ff	3
ORA ,X			0	–	–	↕	↕	–	IX	FA		3
ORA oprx16,SP			0	–	–	↕	↕	–	SP2	9EDA	ee ff	5
ORA oprx8,SP			0	–	–	↕	↕	–	SP1	9EEA	ff	4
PSHA	Push Accumulator onto Stack	Push (A); SP ← (SP) – 0x0001	–	–	–	–	–	–	INH	87		2
PSHH	Push H (Index Register High) onto Stack	Push (H); SP ← (SP) – 0x0001	–	–	–	–	–	–	INH	8B		2
PSHX	Push X (Index Register Low) onto Stack	Push (X); SP ← (SP) – 0x0001	–	–	–	–	–	–	INH	89		2
PULA	Pull Accumulator from Stack	SP ← (SP + 0x0001); Pull (A)	–	–	–	–	–	–	INH	86		3
PULH	Pull H (Index Register High) from Stack	SP ← (SP + 0x0001); Pull (H)	–	–	–	–	–	–	INH	8A		3
PULX	Pull X (Index Register Low) from Stack	SP ← (SP + 0x0001); Pull (X)	–	–	–	–	–	–	INH	88		3
ROL opr8a			↕	–	–	↕	↕	↕	DIR	39	dd	5
ROLA			↕	–	–	↕	↕	↕	INH	49		1

Table continues on the next page...

Table 10-3. Instruction Set Summary (continued)

Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Bus Cycles
			V	H	I	N	Z	C				
ROLX	Rotate Left through Carry	C ← MSB, LSB ← C	↕	–	–	↕	↕	↕	INH	59		1
ROL oprx8,X			↕	–	–	↕	↕	↕	IX1	69	ff	5
ROL ,X			↕	–	–	↕	↕	↕	IX	79		4
ROL oprx8,SP			↕	–	–	↕	↕	↕	SP1	9E69	ff	6
ROR opr8a	Rotate Right through Carry	LSB → C, C → MSB	↕	–	–	↕	↕	↕	DIR	36	dd	5
RORA			↕	–	–	↕	↕	↕	INH	46		1
RORX			↕	–	–	↕	↕	↕	INH	56		1
ROR oprx8,X			↕	–	–	↕	↕	↕	IX1	66	ff	5
ROR ,X			↕	–	–	↕	↕	↕	IX	76		4
ROR oprx8,SP			↕	–	–	↕	↕	↕	SP1	9E66	ff	6
RSP	Reset Stack Pointer	SP ← 0xFF (High Byte Not Affected)	–	–	–	–	–	–	INH	9C		1
RTI	Return from Interrupt	SP ← (SP) + 0x0001, Pull (CCR) SP ← (SP) + 0x0001, Pull (A) SP ← (SP) + 0x0001, Pull (X) SP ← (SP) + 0x0001, Pull (PCH) SP ← (SP) + 0x0001, Pull (PCL)	↕	↕	↕	↕	↕	↕	INH	80		9
RTS	Return from Subroutine	SP ← SP + 0x0001, Pull (PCH) SP ← SP + 0x0001, Pull (PCL)	–	–	–	–	–	–	INH	81		6
SBC #opr8i	Subtract with Carry	A ← (A) – (M) – (C)	↕	–	–	↕	↕	↕	IMM	A2	ii	2
SBC opr8a			↕	–	–	↕	↕	↕	DIR	B2	dd	3
SBC opr16a			↕	–	–	↕	↕	↕	EXT	C2	hh ll	4
SBC oprx16,X			↕	–	–	↕	↕	↕	IX2	D2	ee ff	4
SBC oprx8,X			↕	–	–	↕	↕	↕	IX1	E2	ff	3
SBC ,X			↕	–	–	↕	↕	↕	IX	F2		3
SBC oprx16,SP			↕	–	–	↕	↕	↕	SP2	9ED2	ee ff	5
SBC oprx8,SP			↕	–	–	↕	↕	↕	SP1	9EE2	ff	4
SEC	Set Carry Bit	C ← 1	–	–	–	–	–	1	INH	99		1
SEI	Set Interrupt Mask Bit	I ← 1	–	–	1	–	–	–	INH	9B		1

Table continues on the next page...

Table 10-3. Instruction Set Summary (continued)

Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Bus Cycles
			V	H	I	N	Z	C				
STA opr8a	Store Accumulator in Memory	M ← (A)	0	–	–	↕	↕	–	DIR	B7	dd	3
STA opr16a			0	–	–	↕	↕	–	EXT	C7	hh ll	4
STA oprx16,X			0	–	–	↕	↕	–	IX2	D7	ee ff	4
STA oprx8,X			0	–	–	↕	↕	–	IX1	E7	ff	3
STA ,X			0	–	–	↕	↕	–	IX	F7		2
STA oprx16,SP			0	–	–	↕	↕	–	SP2	9ED7	ee ff	5
STA oprx8,SP			0	–	–	↕	↕	–	SP1	9EE7	ff	4
STHX opr8a	Store H:X (Index Reg.)	(M:M + 0x0001) ← (H:X)	0	–	–	↕	↕	–	DIR	35	dd	4
STHX opr16a			0	–	–	↕	↕	–	EXT	96	hh ll	5
STHX oprx8,SP			0	–	–	↕	↕	–	SP1	9EFF	ff	5
STOP	Enable Interrupts: Stop Processing. Refer to MCU Documentation.	I bit ← 0; Stop Processing	–	–	0	–	–	–	INH	8E		3+
STX opr8a	Store X (Low 8 Bits of Index Register) in Memory	M ← (X)	0	–	–	↕	↕	–	DIR	BF	dd	3
STX opr16a			0	–	–	↕	↕	–	EXT	CF	hh ll	4
STX oprx16,X			0	–	–	↕	↕	–	IX2	DF	ee ff	4
STX oprx8,X			0	–	–	↕	↕	–	IX1	EF	ff	3
STX ,X			0	–	–	↕	↕	–	IX	FF		2
STX oprx16,SP			0	–	–	↕	↕	–	SP2	9EDF	ee ff	5
STX oprx8,SP			0	–	–	↕	↕	–	SP1	9EEF	ff	4
SUB #opr8i	Subtract	A ← (A) – (M)	↕	–	–	↕	↕	↕	IMM	A0	ii	2
SUB opr8a			↕	–	–	↕	↕	↕	DIR	B0	dd	3
SUB opr16a			↕	–	–	↕	↕	↕	EXT	C0	hh ll	4
SUB oprx16,X			↕	–	–	↕	↕	↕	IX2	D0	ee ff	4
SUB oprx8,X			↕	–	–	↕	↕	↕	IX1	E0	ff	3
SUB ,X			↕	–	–	↕	↕	↕	IX	F0		3
SUB oprx16,SP			↕	–	–	↕	↕	↕	SP2	9ED0	ee ff	5
SUB oprx8,SP			↕	–	–	↕	↕	↕	SP1	9EE0	ff	4
		PC ← (PC) + 0x0001 Push (PCL) SP ← (SP) – 0x0001 Push (PCH) SP ← (SP) – 0x0001, Push (X) SP ← (SP) – 0x0001 Push (A)										

Table continues on the next page...

Table 10-3. Instruction Set Summary (continued)

Source Form	Operation	Description	Effect on CCR						Address Mode	Opcode	Operand	Bus Cycles
			V	H	I	N	Z	C				
SWI	Software Interrupt	SP ← (SP) – 0x0001 Push (CCR) SP ← (SP) – 0x0001 I ← 1 PCH ← Interrupt Vector High Byte PCL ← Interrupt Vector Low Byte	–	–	1	–	–	–	INH	83		11
TAP	Transfer Accumulator to CCR	CCR ← (A)	↕	↕	↕	↕	↕	↕	INH	84		1
TAX	Transfer Accumulator to X (Index Register Low)	X ← (A)	–	–	–	–	–	–	INH	97		1
TPA	Transfer CCR to Accumulator	A ← (CCR)	–	–	–	–	–	–	INH	85		1
TST opr8a	Test for Negative or Zero	(M) – 0x00	0	–	–	↕	↕	–	DIR	3D	dd	4
TSTA		(A) – 0x00	0	–	–	↕	↕	–	INH	4D		1
TSTX		(X) – 0x00	0	–	–	↕	↕	–	INH	5D		1
TST oprx8,X		(M) – 0x00	0	–	–	↕	↕	–	IX1	6D	ff	4
TST ,X		(M) – 0x00	0	–	–	↕	↕	–	IX	7D		3
TST oprx8,SP		(M) – 0x00	0	–	–	↕	↕	–	SP1	9E6D	ff	5
TSX	Transfer SP to Index Register	H:X ← (SP) + 0x0001	–	–	–	–	–	–	INH	95		2
TXA	Transfer X (Index Reg. Low) to Accumulator	A ← (X)	–	–	–	–	–	–	INH	9F		1
TXS	Transfer Index Register to SP	SP ← (H:X) – 0x0001	–	–	–	–	–	–	INH	94		2
WAIT	Enable Interrupts Wait for Interrupt	I bit ← 0, Halt CPU	–	–	0	–	–	–	INH	8F		3+

Chapter 11
Flash Memory Module (FTMRH)

11.1 Introduction

The FTMRH module implements the following:

- Program flash (flash) memory

The flash memory is ideal for single-supply applications allowing for field reprogramming without requiring external high voltage sources for program or erase operations. The flash module includes a memory controller that executes commands to modify flash memory contents. The user interface to the memory controller consists of the indexed Flash Common Command Object (FCCOB) register which is written to with the command, global address, data, and any required command parameters. The memory controller must complete the execution of a command before the FCCOB register can be written to with a new command.

CAUTION

A flash byte or longword must be in the erased state before being programmed. Cumulative programming of bits within a flash byte or longword is not allowed.

The flash memory is read as longwords. Read access time is one bus cycle for longwords. For flash memory, an erased bit reads 1 and a programmed bit reads 0.

11.2 Feature

11.2.1 Flash memory features

The flash memory has the following features:

- 16 KB of flash memory composed of one 16 KB flash block divided into 32 sectors of 512 bytes
- Automated program and erase algorithm with verify

- Fast sector erase and longword program operation
- Flexible protection scheme to prevent accidental programming or erasing of flash memory

11.2.2 Other flash module features

The flash memory module has the following other features:

- No external high-voltage power supply required for flash memory program and erase operations
- Interrupt generation on flash command completion and flash error detection
- Security mechanism to prevent unauthorized access to the flash memory

11.3 Functional description

11.3.1 Modes of operation

The flash memory module provides the normal user mode of operation. The operating mode is determined by module-level inputs and affects the FPROT, FCNFG, and FCLKDIV registers.

11.3.1.1 Wait mode

The flash memory module is not affected if the MCU enters Wait mode. The flash module can recover the MCU from Wait via the CCIF interrupt. See Flash interrupts.

11.3.1.2 Stop mode

If a flash command is active, that is, FSTAT[CCIF] = 0, when the MCU requests Stop mode, the current NVM operation will be completed before the MCU is allowed to enter Stop mode.

11.3.2 Flash block read access

If a flash block is read during execution of a command (while FSTAT[CCIF] = 0), the read operation will return invalid data and it will trigger a illegal access exception in the MCU.

11.3.3 Flash memory map

The MCU places the flash memory as shown in the following table.

Table 11-1. Flash memory addressing

Global address	Flash size	Description
0x0000_C000–0x0000_FFFF	16 KB	Flash block contains flash configuration field.

11.3.4 Flash initialization after system reset

On each system reset, the flash module executes an initialization sequence that establishes initial values for the flash block configuration parameters, the FPROT protection register, and the FOPT and FSEC registers. The initialization routine reverts to built-in default values that leave the module in a fully protected and secured state if errors are encountered during execution of the reset sequence. If an error is detected during the reset sequence, both FSTAT[MGSTAT] bits will be set.

FSTAT[CCIF] is cleared throughout the initialization sequence. The NVM module holds off all CPU access for a portion of the initialization sequence. Flash reads are allowed after the hold is removed. Completion of the initialization sequence is marked by setting FSTAT[CCIF] high, which enables user commands. While FSTAT[CCIF] remains cleared, it is not possible to write on registers FCCOBIX or FCCOB.

If a reset occurs while any flash command is in progress, that command will be immediately aborted. The state of the word being programmed or the sector/block being erased is not guaranteed.

11.3.5 Flash command operations

Flash command operations are used to modify flash memory contents.

The command operations contain three steps:

1. Configure the clock for flash program and erase command operations.
2. Use command write sequence to set flash command parameters and launch execution.
3. Execute valid flash commands according to MCU functional mode and MCU security state.

The figure below shows a general flowchart of the flash command write sequence.

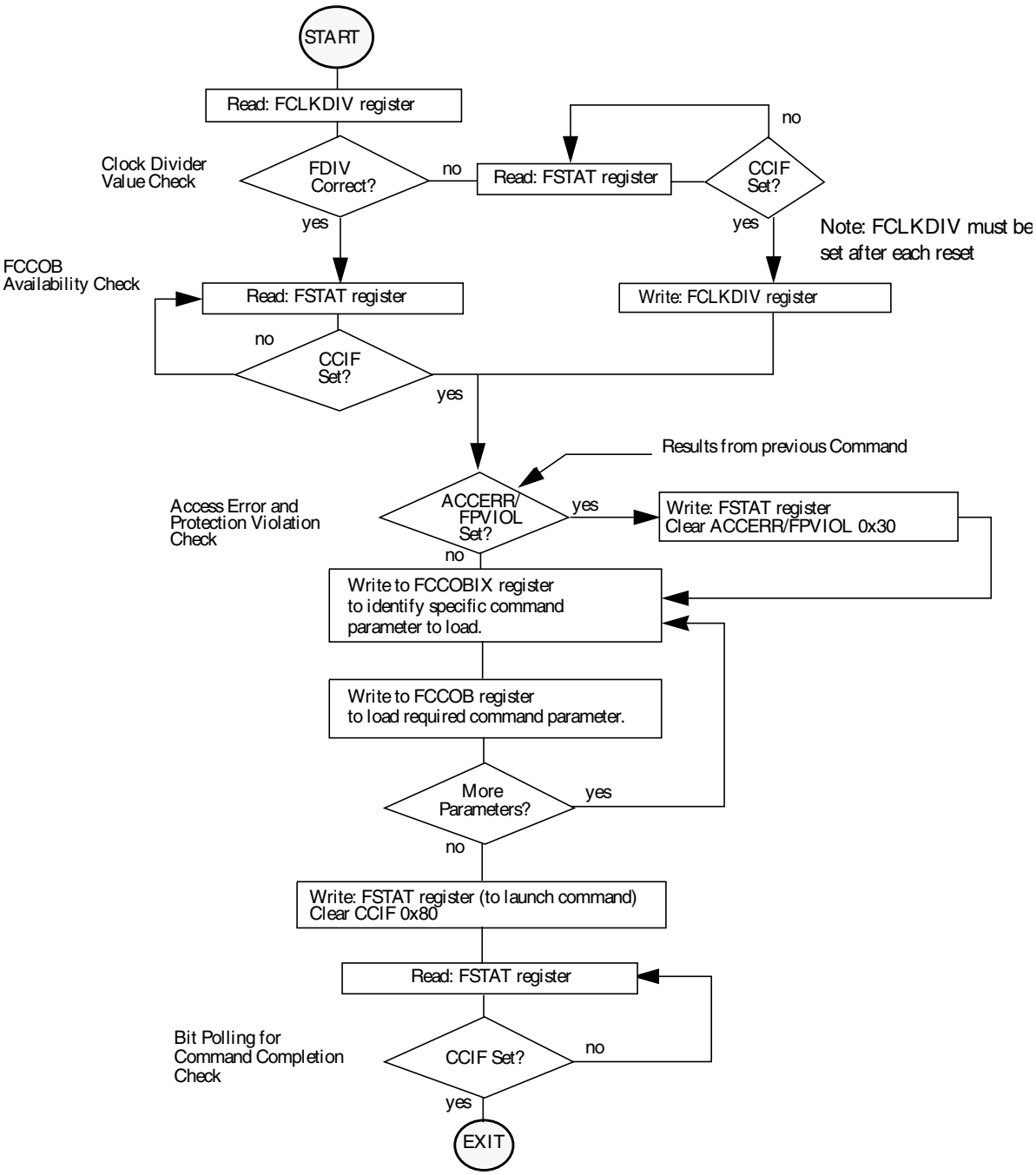


Figure 11-1. Generic flash command write sequence flowchart

11.3.5.1 Writing the FCLKDIV register

Prior to issuing any flash program or erase command after a reset, the user is required to write the FCLKDIV register to divide BUSCLK down to a target FCLK of 1 MHz. The following table shows recommended values for FCLKDIV[FDIV] based on BUSCLK frequency.

Table 11-2. FDIV values for various BUSCLK frequencies

BUSCLK frequency (MHz)		FDIV[5:0]
MIN[1]	MAX[2]	
1.0	1.6	0x00
1.6	2.6	0x01
2.6	3.6	0x02
3.6	4.6	0x03
4.6	5.6	0x04
5.6	6.6	0x05
6.6	7.6	0x06
7.6	8.6	0x07
8.6	9.6	0x08
9.6	10.6	0x09
10.6	11.6	0x0A
11.6	12.6	0x0B
12.6	13.6	0x0C
13.6	14.6	0x0D
14.6	15.6	0x0E
15.6	16.6	0x0F
16.6	17.6	0x10
17.6	18.6	0x11
18.6	19.6	0x12
19.6	20.6	0x13
20.6	21.6	0x14
21.6	22.6	0x15
22.6	23.6	0x16
23.6	24.6	0x17
24.6	25.6	0x18

1. BUSCLK is greater than this value.
2. BUSCLK is less than or equal to this value.

CAUTION

Programming or erasing the flash memory cannot be performed if the bus clock runs at less than 0.8 MHz. Setting FCLKDIV[FDIV] too high can destroy the flash memory due to overstress. Setting FCLKDIV[FDIV] too low can result in incomplete programming or erasure of the flash memory cells.

When the FCLKDIV register is written, FCLKDIV[FDIVLD] is set automatically. If FCLKDIV[FDIVLD] is 0, the FCLKDIV register has not been written since the last reset. If the FCLKDIV register has not been written, any flash program or erase command loaded during a command write sequence will not execute and FSTAT[ACCERR] will be set.

11.3.5.2 Command write sequence

The memory controller will launch all valid flash commands entered using a command write sequence.

Before launching a command, FSTAT[ACCERR] and FSTAT[FPVIOL] must be cleared and the FSTAT[CCIF] flag will be tested to determine the status of the current command write sequence. If FSTAT[CCIF] is 0, indicating that the previous command write sequence is still active, a new command write sequence cannot be started and all writes to the FCCOB register are ignored.

The FCCOB parameter fields must be loaded with all required parameters for the flash command being executed. Access to the FCCOB parameter fields is controlled via FCCOBIX[CCOBIX].

Flash command mode uses the indexed FCCOB register to provide a command code and its relevant parameters to the memory controller. First, the user must set up all required FCCOB fields. Then they can initiate the command's execution by writing a 1 to FSTAT[CCIF]. This action clears the CCIF command completion flag to 0. When the user clears FSTAT[CCIF], all FCCOB parameter fields are locked and cannot be changed by the user until the command completes (evidenced by the memory controller returning FSTAT[CCIF] to1). Some commands return information to the FCCOB register array.

The generic format for the FCCOB parameter fields in flash command mode is shown in the following table. The return values are available for reading after the FSTAT[CCIF] flag has been returned to 1 by the memory controller. Writes to the unimplemented parameter fields, FCCOBIX[CCOBIX] =110b and FCCOBIX[CCOBIX] = 111b, are ignored with read from these fields returning 0x0000.

Table 11-3 shows the generic flash command format. The high byte of the first word in the CCOB array contains the command code, followed by the parameters for this specific flash command. For details on the FCCOB settings required by each command, see the flash command descriptions in Flash command summary .

Table 11-3. FCCOB – flash command mode typical usage

CCOBIX[2:0]	Byte	FCCOB parameter fields in flash command mode
000	HI	FCMD[7:0] defining flash command
	LO	Global address [23:16]
001	HI	Global address [15:8]
	LO	Global address [7:0]
010	HI	Data 0 [15:8]
	LO	Data 0 [7:0]
011	HI	Data 1 [15:8]
	LO	Data 1 [7:0]
100	HI	Data 2 [15:8]
	LO	Data 2 [7:0]
101	HI	Data 3 [15:8]
	LO	Data 3 [7:0]

The contents of the FCCOB parameter fields are transferred to the memory controller when the user clears the FSTAT[CCIF] command completion flag by writing 1. The CCIF flag will remain clear until the flash command has completed. Upon completion, the memory controller will return FSTAT[CCIF] to 1 and the FCCOB register will be used to communicate any results.

The following table presents the valid flash commands, as enabled by the combination of the functional MCU mode with the MCU security state of unsecured or secured.

MCU secured state is selected by FSEC[SEC].

Table 11-4. Flash commands by mode and security state

FCMD	Command	Unsecured	Secured
		U[1]	U[2]
0x01	Erase verify all blocks	*	*
0x02	Erase verify block	*	*
0x03	Erase verify flash section	*	*
0x04	Read once	*	*
0x06	Program flash	*	*
0x07	Program once	*	*
0x08	Erase all block	*	*
0x09	Erase flash block	*	*

Table continues on the next page...

Table 11-4. Flash commands by mode and security state (continued)

FCMD	Command	Unsecured	Secured
		U[1]	U[2]
0x0A	Erase flash sector	*	*
0x0B	Unsecure flash	*	*
0x0C	Verify backdoor access key	*	*
0x0D	Set user margin level	*	*
0x0E	Set factory margin level	*	*

1. Unsecured User mode
2. Secured User mode

11.3.6 Flash interrupts

The flash module can generate an interrupt when a flash command operation has completed.

Table 11-5. Flash interrupt source

Interrupt source	Interrupt flag	Local enable	Global (CCR) mask
Flash command complete	CCIF (FSTAT register)	CCIE (FCNFG register)	I Bit

11.3.6.1 Description of flash interrupt operation

The flash module uses the FSTAT[CCIF] flag in combination with the FCNFG[CCIE] interrupt enable bit to generate the flash command interrupt request.

The logic used for generating the flash module interrupts is shown in the following figure.

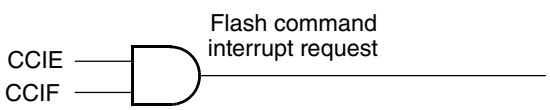


Figure 11-2. Flash module interrupts implementation

11.3.7 Protection

The FPROT register can be set to protect regions in the flash memory from accidental programing or erasing. Two separate memory regions, one growing downward from global address 0x0_FFFF in the flash memory, called the higher region, and the remaining addresses in the flash memory, can be activated for protection. The flash memory addresses covered by these protectable regions are shown in the flash memory map.

Default protection settings as well as security information that allows the MCU to restrict access to the flash module are stored in the flash configuration field as described in the table below.

Table 11-6. Flash configuration field

Global address	Size (Bytes)	Description
0xFF70–0xFF77	8	Backdoor comparison key. See Verify backdoor access key command and Unsecuring the MCU using backdoor key access.
0xFF78–0xFF7B [1]	4	Reserved
0xFF7C[1]	1	Flash protection byte
0xFF7D[1]	1	Reserved
0xFF7E[1]	1	Flash nonvolatile byte
0xFF7F[1]	1	Flash security byte

1. 0x0_FF78-0x0_FF7F form a Flash phrase and must be programmed in a single command write sequence. Each byte in the 0x0_FF78 - 0x0_FF7B reserved field should be programmed to 0xFF.

The flash module provides protection to the MCU. During the reset sequence, the FPROT register is loaded with the contents of the flash protection byte in the flash configuration field at global address FF7C in flash memory. The protection functions depend on the configuration of bit settings in FPROT register.

Table 11-7. Flash protection function

FPOPEN	FPHDIS	Function[1]
1	1	No flash protection
1	0	Protected high range
0	1	Full flash memory protected
0	0	Unprotected high range

1. For range sizes, see Table 4.

The flash protection scheme can be used by applications requiring reprogramming in single chip mode while providing as much protection as possible if reprogramming is not required.

All possible flash protection scenarios are shown in the following figure. Although the protection scheme is loaded from the flash memory at global address 0xFF7C during the reset sequence, it can be changed by the user.

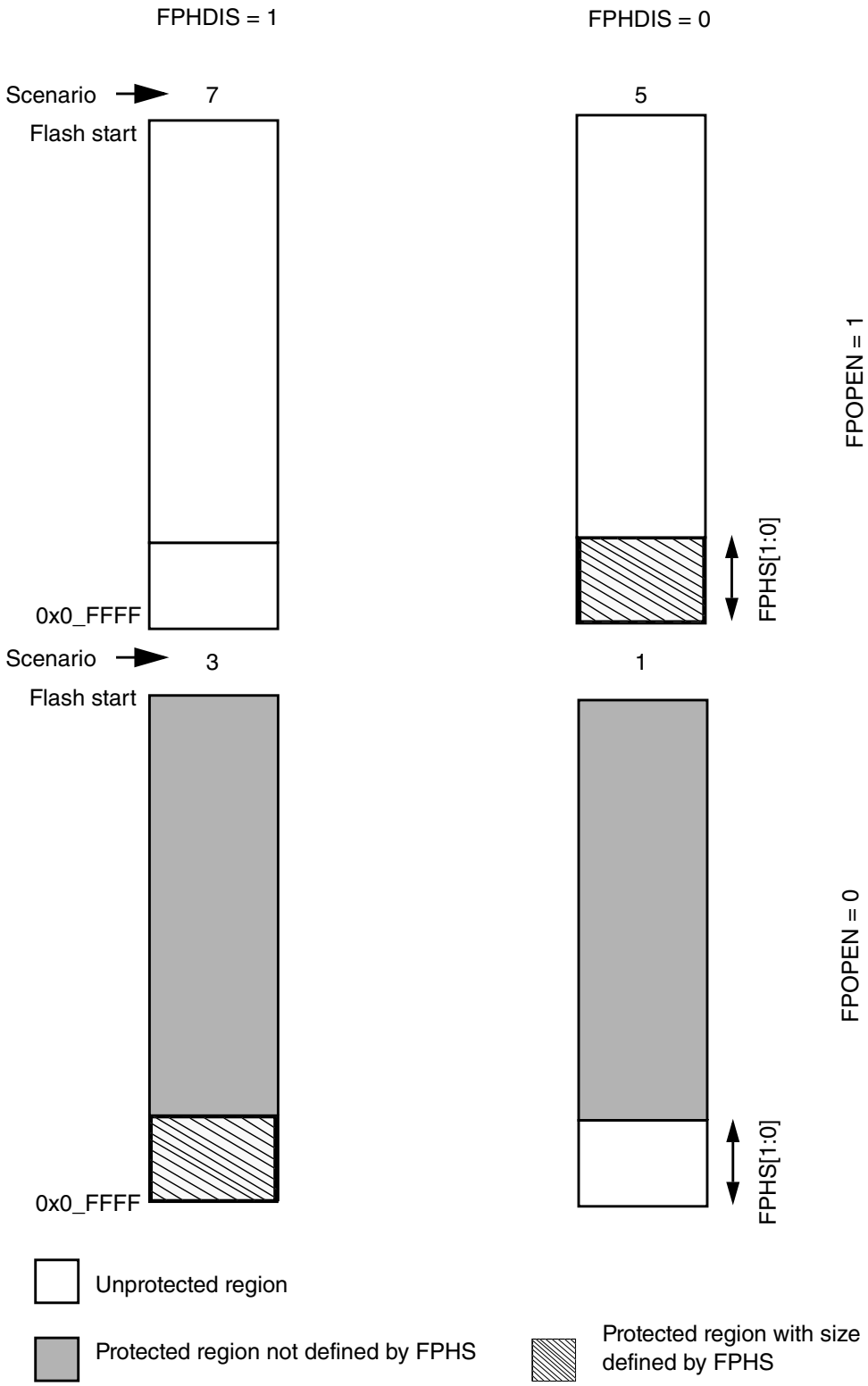


Figure 11-3. Flash protection scenarios

The general guideline is that flash protection can only be added and not removed. The following table specifies all valid transitions between flash protection scenarios. Any attempt to write an invalid scenario to the FPROT register will be ignored. The contents of the FPROT register reflect the active protection scenario. See the FPROT[FPHS] field descriptions for additional restrictions.

Table 11-8. Flash protection scenario transitions

From protection scenario	To protection scenario[1]			
	1	3	5	7
1	×	×		
3		×		
5		×	×	
7	×	×	×	×

1. Allowed transitions marked with X.

The flash protection address range is listed in the following two tables regarding the scenarios in the table above.

Table 11-9. Flash protection higher address range

FPHS[1:0]	Global address range	Protected size
00	0xF800–0xFFFF	2 KB
01	0xF000–0xFFFF	4 KB
10	0xE000–0xFFFF	8 KB
11	0xC000–0xFFFF	16 KB

11.3.8 Security

The flash module provides security information to the MCU. The flash security state is defined by FSEC[SEC]. During reset, the flash module initializes the FSEC register using data read from the security byte of the flash configuration field. The security state out of reset can be permanently changed by programming the security byte, assuming that the MCU is starting from a mode where the necessary flash erase and program commands are available and that the upper region of the flash is unprotected. If the flash security byte is successfully programmed, its new value will take effect after the next MCU reset.

The following subsections describe these security-related subjects:

- Unsecuring the MCU using backdoor key access

- Unsecuring the MCU using BDM
- Mode and security effects on flash command availability

11.3.8.1 Unsecuring the MCU using backdoor key access

The MCU may be unsecured by using the backdoor key access feature which requires knowledge of the contents of the backdoor keys, which are four 16-bit words programmed at addresses 0xFF70–0xFF77. If the KEYEN[1:0] bits are in the enabled state, the verify backdoor access key command – see Verify backdoor access key command, allows the user to present four prospective keys for comparison to the keys stored in the flash memory via the memory controller. If the keys presented in the verify backdoor access key command match the backdoor keys stored in the flash memory, FSEC[SEC] will be changed to unsecure the MCU. Key values of 0x0000 and 0xFFFF are not permitted as backdoor keys. While the Verify Backdoor Access Key command is active, flash memory will not be available for read access and will return invalid data.

The user code stored in the flash memory must have a method of receiving the backdoor keys from an external stimulus. This external stimulus would typically be through one of the on-chip serial ports.

If the KEYEN[1:0] bits are in the enabled state, the MCU can be unsecured by the backdoor key access sequence described below:

1. Follow the command sequence for the verify backdoor access key command as explained in Verify backdoor access key command.
2. If the verify backdoor access key command is successful, the MCU is unsecured and FSEC[SEC] is forced to the unsecure state of 10.

The verify backdoor access key command is monitored by the memory controller and an illegal key will prohibit future use of the verify backdoor access key command. A reset of the MCU is the only method to re-enable the verify backdoor access key command. The security as defined in the flash security byte is not changed by using the verify backdoor access key command sequence. The backdoor keys stored in addresses 0xFF70–0xFF77 are unaffected by the verify backdoor access key command sequence. The verify backdoor access key command sequence has no effect on the program and erase protections defined in the flash protection register, FPROT.

After the backdoor keys have been correctly matched, the MCU will be unsecured. After the MCU is unsecured, the sector containing the flash security byte can be erased and the flash security byte can be reprogrammed to the unsecure state, if desired. In the unsecure state, the user has full control of the contents of the backdoor keys by programming addresses 0xFF70–0xFF77 in the flash configuration field.

11.3.8.2 Unsecuring the MCU using BDM

A secured MCU can be unsecured by using the following method to erase the flash memory:

1. Reset the MCU.
2. Set FCDIV register as described in Writing the FCLKDIV register
3. Configure FPROT register to disable protection in the flash memory.
4. Execute the Erase All Blocks command write sequence to erase the flash memory. Alternatively the Unsecure Flash command can be executed.

   If theflash memory is verified as erased, the MCU will be unsecured. All BDM commands will now be enabled and the flash security byte may be programmed to the unsecure state by continuing with the following steps:
5. Execute the Program Flash command write sequence to program the flash security byte to the unsecured state
6. Reset the MCU

11.3.8.3 Mode and security effects on flash command availability

The availability of flash module commands depends on the MCU operating mode and security state as shown in Table 11-4.

11.3.9 Flash commands

11.3.9.1 Flash commands

The following table summarizes the valid flash commands as well as the effects of the commands on the flash block and other resources within the flash module.

NOTE

All commands in the following table, regardless of MCU mode or security state, cannot be launched while the flash array is

being read (i.e. the commands must not be executed if the core is fetching code from the flash). If the core attempts to read the flash while any command is running that may result in an illegal access. Refer to Flash block read access for details.

Table 11-10. Flash commands

FCMD	Command	Function on flash memory
0x01	Erase Verify All Blocks	Verifies that all flash blocks are erased
0x02	Erase Verify Block	Verifies that a flash block is erased
0x03	Erase Verify Flash Section	Verifies that a given number of words starting at the address provided are erased
0x04	Read Once	Reads a dedicated 64-byte field in the nonvolatile information register in flash block that was previously programmed using the program once command
0x06	Program Flash	Programs up to two longwords in a flash block
0x07	Program Once	Programs a dedicated 64 byte field in the nonvolatile information register in flash block that is allowed to be programmed only once
0x08	Erase All Block	Erases all flash blocks An erase of all flash blocks is possible only when the FPROT[FPHDIS] and FPROT[FPOEN] and the bit are set prior to launching the command
0x09	Erase Flash Block	Erases a flash block An erase of the full flash block is possible only when FPROT[FPHDIS] and FPROT[FPOEN] are set prior to launching the command.
0x0A	Erase Flash Sector	Erases all bytes in a flash sector
0x0B	Unsecure Flash	Supports a method of releasing MCU security by erasing all flash blocks and verifying that all flash blocks are erased
0x0C	Verify Backdoor Access key	Supports a method of releasing MCU security by verifying a set of security keys
0x0D	Set User Margin Level	Specifies a user margin read level for all flash blocks
0x0E	Set Factory Margin Level	Specifies a factory margin read level for all flash blocks

11.3.10 Flash command summary

This section provides details of all available flash commands launched by a command write sequence. The FSTAT[ACCERR] will be set during the command write sequence if any of the following illegal steps are performed, causing the command not to be processed by the memory controller:

- Starting any command write sequence that programs or erases flash memory before initializing the FLCKDIV register.
- Writing an invalid command as part of the command write sequence.
- For additional possible errors, refer to the error handling table provided for each command.

If a flash block is read during the execution of an algorithm (FSTAT[CCIF] = 0) on that same block, the read operation will return invalid data.

If FSTAT[ACCERR] or FSTAT[FPVIOL] are set, the user must clear these fields before starting any command write sequence.

CAUTION

An flash longword must be in the erased state before being programmed. Cumulative programming of bits within an flash longword is not allowed.

11.3.10.1 Erase Verify All Blocks command

The Erase Verify All Blocks command will verify that all flash blocks have been erased.

Table 11-11. Erase Verify All Blocks command FCCOB requirements

CCOBIX[2:0]	FCCOBHI parameters	FCCOBLO parameters
000	0x01	Not required

Upon clearing FSTAT[CCIF] to launch the Erase Verify All Blocks command, the memory controller will verify that the entire flash memory space is erased. The FSTAT[CCIF] flag will set after the erase verify all blocks operation has completed. If all blocks are not erased, it means blank check failed and both FSTAT[MGSTAT] bits will be set.

Table 11-12. Erase verify all blocks command error handling

Register	Error bit	Error condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 000 at command launch
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the read[1] or if blank check failed
	MGSTAT0	Set if any errors have been encountered during the read or if blank check failed

1. As found in the memory map for NVM

11.3.10.2 Erase Verify Block command

The Erase Verify Block command allows the user to verify that an entire flash block has been erased. The FCCOB global address [23:0] bits determine which block must be verified.

Table 11-13. Erase Verify Block Command FCCOB requirements

CCOBIX[2:0]	FCCOBHI parameters	FCCOBLO parameters
000	0x02	Global address [23:16] to identify flash block
001	Global address [15:0] in flash block to be verified	

Upon clearing FSTAT[CCIF] to launch the erase verify block command, the memory controller will verify that the selected flash block is erased. The FSTAT[CCIF] flag will set after the erase verify block operation has completed. If the block is not erased, it means blank check failed and both FSTAT[MGSTAT] bits will be set.

Table 11-14. Erase Verify Block command error handling

Register	Error bit	Error condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 001 at command launch
		Set if an invalid global address [23:0] is supplied[1]
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the read or if blank check failed
	MGSTAT0	Set if any errors have been encountered during the read or if blank check failed

1. As found in the memory map for NVM

11.3.10.3 Erase Verify Flash Section command

The Erase Verify Flash Section command will verify that a section of code in the flash memory is erased. The Erase Verify Flash Section command defines the starting point of the code to be verified and the number of longwords.

Table 11-15. Erase verify flash section command FCCOB requirements

CCOBIX[2:0]	FCCOBHI parameters	FCCOBLO parameters
000	0x03	Global address [23:16] of flash block
001	Global address [15:0] of the first longwords to be verified	
010	Number of long words to be verified	

Upon clearing FSTAT[CCIF] to launch the erase verify flash section command, the memory controller will verify that the selected section of flash memory is erased. The FSTAT[CCIF] flag will set after the erase verify flash section operation has completed. If the section is not erased, it means blank check failed and both FSTAT[MGSTAT] bits will be set.

Table 11-16. Erase Verify Flash Section command error handling

Register	Error bit	Error condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 010 at command launch
		Set if command not available in current mode (see Table 11-4)
		Set if an invalid global address [23:0] is supplied (see Table 11-1)[1]
		Set if a misaligned long words address is supplied (global address[1:0] != 00)
		Set if the requested section crosses flash address boundary
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the read[2] or if blank check failed
	MGSTAT0	Set if any errors have been encountered during the read[2] or if blank check failed

1. As defined by the memory map for NVM
2. As found in the memory map for NVM

11.3.10.4 Read once command

The read once command provides read access to a reserved 64-byte field (8 phrase) located in the nonvolatile information register of flash. The read once field can only be programmed once and can not be erased. It can be used to store the product ID or any other information that can be written only once. It is programmed using the program once command described in Program Once command. To avoid code runaway, the read once command must not be executed from the flash block containing the program once reserved field.

Table 11-17. Read Once command FCCOB requirements

CCOBIX[2:0]	FCCOB parameters	
000	0x04	Not required
001	Read once phrase index (0x0000 – 0x0007)	
010	Read once word 0 value	
011	Read once word 1 value	
100	Read once word 2 value	
101	Read once word 3 value	

Upon clearing FSTAT[CCIF] to launch the read once command, a read once phrase is fetched and stored in the FCCOB indexed register. The FSTAT[CCIF] flag will set after the read once operation has completed. Valid phrase index values for the read once command range from 0x0000 to 0x0007. During execution of the read once command, any attempt to read addresses within flash block will return invalid data.

Table 11-18. Read Once command error handling

Register	Error bit	Error condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 001 at command launch
		Set if command is not available in current mode (see Table 11-4)
		Set if an invalid phrase index is supplied
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the read
	MGSTAT0	Set if any errors have been encountered during the read

11.3.10.5 Program Flash command

The program flash operation will program up to two previously erased longwords in the flash memory using an embedded algorithm.

Note

A flash longword must be in the erased state before being programmed. Cumulative programming of bits within a flash longword is not allowed.

Table 11-19. Program Flash command FCCOB requirements

CCOBIX[2:0]	FCCOBHI parameters	FCCOBLO parameters
000	0x06	Global address [23:16] to identify flash block
001	Global address [15:0] of longwords location to be programmed[1]	
010	Word 0 (longword 0) program value	
011	Word 1 (longword 0) program value	
100	Word 2 (longword 1) program value	
101	Word 3 (longword 1) program value	

1. Global address [1:0] must be 00.

Upon clearing FSTAT[CCIF] to launch the Program Flash command, the memory controller will program the data longwords to the supplied global address and will then proceed to verify the data longwords read back as expected. The FSTAT[CCIF] flag will set after the program flash operation has completed.

Table 11-20. Program Flash command error handling

Register	Error bit	Error condition
FSTAT	ACCERR	Set if CCOBIX[2:0] ≠ 011 or 101 at command launch
		Set if command not available in current mode (see Table 11-4)
		Set if an invalid global address [23:0] is supplied (see Table 11-1.[1]

Table continues on the next page...

Table 11-20. Program Flash command error handling (continued)

Register	Error bit	Error condition
		Set if a misaligned longword address is supplied (global address [1:0] != 00)
		Set if the requested group of words breaches the end of the flash block.
	FPVIOL	Set if the global address [23:0] points to a protected data
	MGSTAT1	Set if any errors have been encountered during the verify operation
	MGSTAT0	Set if any errors have been encountered during the verify operation

1. As defined by the memory map of NVM.

11.3.10.6 Program Once command

The Program Once command restricts programming to a reserved 64-byte field (8 phrases) in the nonvolatile information register located in flash. The program once reserved field can be read using the read once command as described in Read once command. The program once command must be issued only once because the nonvolatile information register in flash cannot be erased. To avoid code runaway, the program once command must not be executed from the flash block containing the program once reserved field.

Table 11-21. Program Once command FCCOB requirements

CCOBIX[2:0]	FCCOB parameters	
000	0x07	Not required
001	Program Once phrase index (0x000 – 0x0007)	
010	Program once Word 0 value	
011	Program once Word 1value	
100	Program once Word 2 value	
101	Program once Word 3 value	

Upon clearing FSTAT[CCIF] to launch the program once command, the memory controller first verifies that the selected phrase is erased. If erased, then the selected phrase will be programmed and then verified with read back. The FSTAT[CCIF] flag will remain clear, setting only after the program once operation has completed.

The reserved nonvolatile information register accessed by the Program Once command cannot be erased, and any attempt to program one of these phrases a second time will not be allowed. Valid phrase index values for the program once command range from 0x0000 to 0x0007. During execution of the program once command, any attempt to read addresses within flash will return invalid data.

Table 11-22. Program Once ommand error handling

Register	Error bit	Error condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 101 at command launch
		Set if command not available in current mode (see Table 11-4)
		Set if an invalid phrase index is supplied
		Set if the requested phrase has already been programmed[1]
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the verify operation
	MGSTAT0	Set if any errors have been encountered during the verify operation

1. If a program once phrase is initially programmed to 0xFFFF_FFFF_FFFF_FFFF, the program once command will be allowed to execute again on that same phrase.

11.3.10.7 Erase All Blocks command

The Erase All Blocks operation will erase the entire flash memory space.

Table 11-23. Erase All Blocks command FCCOB requirements

CCOBIX[2:0]	FCCOBHI parameters	FCCOBLO parameters
000	0x08	Not required

Upon clearing FSTAT[CCIF] to launch the Erase All Blocks command, the memory controller will erase the entire NVM memory space and verify that it is erased. If the memory controller verifies that the entire NVM memory space was properly erased, security will be released. Therefore, the device is in unsecured state. During the execution of this command (FSTAT[CCIF] = 0) the user must not write to any NVM module register. The FSTAT[CCIF] flag will set after the erase all blocks operation has completed.

Table 11-24. Erase All Blocks command error handling

Register	Error bit	Error condition
FSTAT	ACCERR	Set if CCOBIX[2:0] ≠ 000 at command launch
		Set if command not available in current mode (see Table 11-4)
	FPVIOL	Set if any area of the flash memory is protected
	MGSTAT1	Set if any errors have been encountered during the verify operation[1]
	MGSTAT0	Set if any errors have been encountered during the verify operation[1]

1. As found in the memory map for NVM.

11.3.10.8 Erase flash block command

The erase flash block operation will erase all addresses in a flash block.

Table 11-25. Erase flash block command FCCOB requirements

CCOBIX[2:0]	FCCOB parameters	
000	0x09	Global address [23:16] to identify flash block
001	Global address[15:0] in flash block to be erased	

Upon clearing FSTAT[CCIF] to launch the erase flash block command, the memory controller will erase the selected flash block and verify that it is erased. The FSTAT[CCIF] flag will set after the erase flash block operation has completed.

Table 11-26. Erase flash block command error handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 001 at command launch
		Set if command not available in current mode (see Table 11-4)
		Set if an invalid global address [23:0] is supplied[1]
	FPVIOL	Set if an area of the selected flash block is protected
	MGSTAT1	Set if any errors have been encountered during the verify operation[2]
	MGSTAT0	Set if any errors have been encountered during the verify operation[2]

1. As defined by the memory map for NVM.
2. As found in the memory map for NVM.

11.3.10.9 Erase flash sector command

The erase flash sector operation will erase all addresses in a flash sector.

Table 11-27. Erase flash sector command FCCOB requirements

CCOBIX[2:0]	FCCOB parameters	
000	0x0A	Global address [23:16] to identify flash block to be erased
001	Global address [15:0] anywhere within the sector to be erased. Refer to Overview for the flash sector size	

Upon clearing FSTAT[CCIF] to launch the erase flash sector command, the memory controller will erase the selected flash sector and then verify that it is erased. The FSTAT[CCIF] flag will be set after the erase flash sector operation has completed.

Table 11-28. Erase flash sector command error handling

Register	Error bit	Error condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 001 at command launch
		Set if command not available in current mode (see Table 11-4)
		Set if an invalid global address [23:0] is supplied.[1] (see Table 11-1)
		Set if a misaligned longword address is supplied (global address [1:0] != 00)
	FPVIOL	Set if the selected flash sector is protected
	MGSTAT1	Set if any errors have been encountered during the verify operation
	MGSTAT0	Set if any errors have been encountered during the verify operation

1. As defined by the memory map for NVM

11.3.10.10 Unsecure flash command

The unsecure flash command will erase the entire flash memory space, and if the erase is successful, will release security.

Table 11-29. Unsecure flash command FCCOB requirements

CCOBIX[2:0]	FCCOB parameters	
000	0x0B	Not required

Upon clearing FSTAT[CCIF] to launch the unsecure flash command, the memory controller will erase the entire flash memory space and verify that it is erased. If the memory controller verifies that the entire flash memory space was properly erased, security will be released. If the erase verify is not successful, the unsecure flash operation sets FSTAT[MGSTAT1] and terminates without changing the security state. During the execution of this command (FSTAT[CCIF] = 0), the user must not write to any flash module register. The FSTAT[CCIF] flag is set after the unsecure flash operation has completed.

Table 11-30. Unsecure flash command error handling

Register	Error bit	Error condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 000 at command launch
		Set if command is not available in current mode (see Table 11-4)
	FPVIOL	Set if any area of the flash memory is protected
	MGSTAT1	Set if any errors have been encountered during the verify operation[1]
	MGSTAT0	Set if any errors have been encountered during the verify operation[1]

1. As found in the memory map for NVM

11.3.10.11 Verify backdoor access key command

The verify backdoor access key command will execute only if it is enabled by the FSEC[KEYEN] bits. The verify backdoor access key command releases security if user-supplied keys match those stored in the flash security bytes of the flash configuration field. See Table 11-1 for details. The code that performs verifying backdoor access command must be running from RAM.

Table 11-31. Verify backdoor access key command FCCOB requirements

CCOBIX[2:0]	FCCOBHI parameters	FCCOBLO parameters
000	0x0C	Not required
001	Key 0	
010	Key 1	
011	Key 2	
100	Key 3	

Upon clearing FSTAT[CCIF] to launch the verify backdoor access key command, the memory controller will check the FSEC[KEYEN] bits to verify that this command is enabled. If not enabled, the memory controller sets the FSTAT[ACCERR] bit. If the command is enabled, the memory controller compares the key provided in FCCOB to the backdoor comparison key in the flash configuration field with Key 0 compared to 0x0400, and so on. If the backdoor keys match, security will be released. If the backdoor keys do not match, security is not released and all future attempts to execute the verify backdoor access key command are aborted (set FSTAT[ACCERR]) until a reset occurs. The FSTAT[CCIF] flag is set after the verify backdoor access key operation has completed.

Table 11-32. Verify backdoor access key command error handling

Register	Error bit	Error condition
FSTAT	ACCERR	Set if CCOBIX[2:0] ≠ 100 at command launch
		Set if an incorrect backdoor key is supplied
		Set if backdoor key access has not been enabled (KEYEN[1:0] ≠ 10
		Set if the backdoor key has mismatched since the last reset
	FPVIOL	None
	MGSTAT1	None
	MGSTAT0	None

11.3.10.12 Set user margin level command

The user margin is a small delta to the normal read reference level and, in effect, is a minimum safety margin. That is, if the reads pass at the tighter tolerances of the user margins, the normal reads have at least that much safety margin before users experience data loss.

The set user margin level command causes the memory controller to set the margin level for future read operations of the flash block.

Table 11-33. Set user margin level command FCCOB requirements

CCOBIX[2:0]	FCCOBHI parameters	FCCOBLO parameters
000	0x0D	Global address [23:16] to identify flash block
001	Global address [15:0] to identify flash block	
010	Margin level setting	

Upon clearing FSTAT[CCIF] to launch the set user margin level command, the memory controller will set the user margin level for the targeted block and then set the FSTAT[CCIF] flag.

Note

Valid margin level settings for the set user margin level command are defined in the following tables.

Table 11-34. Valid set user margin level settings

CCOB (CCOBIX = 010)	Level description
0x0000	Return to normal level
0x0001	User margin-1 level[1]
0x0002	User margin-0 level[2]

1. Read margin to the erased state
2. Read margin to the programmed state

Table 11-35. Set user margin level command error handling

Register	Error bit	Error condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 010 at command launch
		Set if command is not available in current mode (see Table 11-4)
		Set if an invalid global address [23:0] is supplied
		Set if an invalid margin level setting is supplied
	FPVIOL	None
	MGSTAT1	None
	MGSTAT0	None

Note

User margin levels can be used to check that NVM memory contents have adequate margin for normal level read operations. If unexpected results are encountered when checking NVM memory contents at user margin levels, a potential loss of information has been detected.

11.3.10.13 Set factory margin level command

The set factory margin Level command causes the memory controller to set the margin level specified for future read operations of the flash block.

Table 11-36. Set factory margin level command FCCOB requirements

CCOBIX[2:0]	FCCOBHI parameters	FCCOBLO parameters
000	0x0E	Global address [23:16] to identify flash block
001	Global address [15:0] to identify flash block	
010	Margin level setting	

Upon clearing FSTAT[CCIF] to launch the set factory margin level command, the memory controller will set the factory margin level for the targeted block and then set the FSTAT[CCIF] flag.

Note

Valid margin level settings for the set factory margin level command are defined in the following tables.

Table 11-37. Valid set factory margin level settings

CCOB (CCOBIX = 010)	Level description
0x0000	Return to normal level
0x0001	User margin-1 level[1]
0x0002	User margin-0 level[2]
0x0003	Factory margin-1 level[1]
0x0004	Factory margin-0 level[2]

1. Read margin to the erased state
2. Read margin to the programmed state

Table 11-38. Set factory margin level command error handling

Register	Error bit	Error condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 010 at command launch
		Set if command is not available in current mode (see Table 11-4)
		Set if an invalid global address [23:0] is supplied
		Set if an invalid margin level setting is supplied
	FPVIOL	None
	MGSTAT1	None
	MGSTAT0	None

CAUTION

Factory margin levels must only be used during verify of the initial factory programming.

Note

Factory margin levels can be used to check that Flash memory contents have adequate margin for data retention at the normal level setting. If unexpected results are encountered when checking flash memory contents at factory margin levels, the fash memory contents must be erased and reprogrammed.

11.4 Memory map and register definition

This section presents a high-level summary of the registers and how they are mapped. The registers can be accessed in 32-bits, 16-bits (aligned on data[31:16] or on data[15:0]) or 8-bits. In the case of the writable registers, the write accesses are forbidden during flash command execution. For more details, see Caution note in Flash memory map.

FTMRH memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
1830	Flash Clock Divider Register (FTMRH_FCLKDIV)	8	R/W	00h	11.4.1/181
1831	Flash Security Register (FTMRH_FSEC)	8	R	Undefined	11.4.2/182
1832	Flash CCOB Index Register (FTMRH_FCCOBIX)	8	R/W	00h	11.4.3/183
1834	Flash Configuration Register (FTMRH_FCNFG)	8	R/W	00h	11.4.4/183
1836	Flash Status Register (FTMRH_FSTAT)	8	R/W	80h	11.4.5/184
1838	Flash Protection Register (FTMRH_FPROT)	8	R	See section	11.4.6/185
183A	Flash Common Command Object Register:High (FTMRH_FCCOBHI)	8	R/W	00h	11.4.7/186

Table continues on the next page...

FTMRH memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
183B	Flash Common Command Object Register: Low (FTMRH_FCCOBLO)	8	R/W	00h	11.4.8/187
183C	Flash Option Register (FTMRH_FOPT)	8	R	Undefined	11.4.9/187

11.4.1 Flash Clock Divider Register (FTMRH_FCLKDIV)

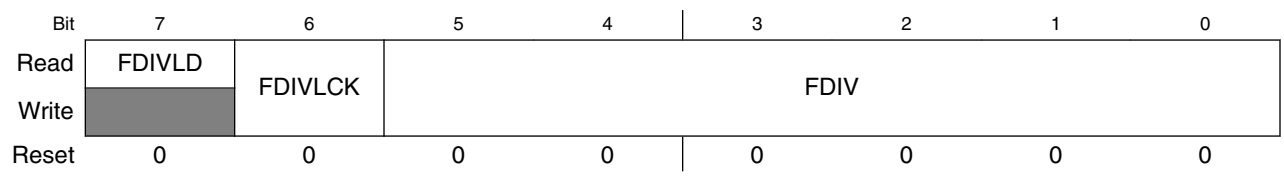
The FCLKDIV register is used to control timed events in program and erase algorithms.

All bits in the FCLKDIV register are readable, bit 7 is not writable, bit 6 is write-once-high and controls the writability of the FDIV field in user mode. In BDM mode, bits 6-0 are writable any number of times but bit 7 remains unwritable.

NOTE

The FCLKDIV register must not be written while a flash command is executing (FSTAT[CCIF] = 0)

Address: 1830h base + 0h offset = 1830h



FTMRH_FCLKDIV field descriptions

Field	Description
7 FDIVLD	Clock Divider Loaded 0 FCLKDIV register has not been written since the last reset. 1 FCLKDIV register has been written since the last reset.
6 FDIVLCK	Clock Divider Locked 0 FDIV field is open for writing. 1 FDIV value is locked and cannot be changed. After the lock bit is set high, only reset can clear this bit and restore writability to the FDIV field in user mode.
FDIV	Clock Divider Bits FDIV[5:0] must be set to effectively divide BUSCLK down to 1MHz to control timed events during flash program and erase algorithms. Refer to the table in the Writing the FCLKDIV register for the recommended values of FDIV based on the BUSCLK frequency.

11.4.2 Flash Security Register (FTMRH_FSEC)

The FSEC register holds all bits associated with the security of the MCU and NVM module. All fields in the FSEC register are readable but not writable. During the reset sequence, the FSEC register is loaded with the contents of the flash security byte in the flash configuration field located in flash memory.

See Security for security function.

Address: 1830h base + 1h offset = 1831h

Bit	7	6	5	4	3	2	1	0
Read	KEYEN		Reserved				SEC	
Write								
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

FTMRH_FSEC field descriptions

Field	Description
7–6 KEYEN	Backdoor Key Security Enable Bits The KEYEN[1:0] bits define the enabling of backdoor key access to the flash module. **NOTE:** 01 is the preferred KEYEN state to disable backdoor key access. 00 Disabled 01 Disabled 10 Enabled 11 Disabled
5–2 Reserved	This field is reserved.
SEC	Flash Security Bits Defines the security state of the MCU. If the flash module is unsecured using backdoor key access, the SEC field is forced to 10. **NOTE:** 01 is the preferred SEC state to set MCU to secured state. 00 Secured 01 Secured 10 Unsecured 11 Secured

11.4.3 Flash CCOB Index Register (FTMRH_FCCOBIX)

The FCCOBIX register is used to index the FCCOB register for NVM memory operations.

Address: 1830h base + 2h offset = 1832h

Bit	7	6	5	4	3	2	1	0
Read	0					CCOBIX		
Write								
Reset	0	0	0	0	0	0	0	0

FTMRH_FCCOBIX field descriptions

Field	Description
7–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CCOBIX	Common Command Register Index Selects which word of the FCCOB register array is being read or written to.

11.4.4 Flash Configuration Register (FTMRH_FCNFG)

The FCNFG register enables the flash command complete interrupt.

Address: 1830h base + 4h offset = 1834h

Bit	7	6	5	4	3	2	1	0
Read	CCIE	0			0			
Write								
Reset	0	0	0	0	0	0	0	0

FTMRH_FCNFG field descriptions

Field	Description
7 CCIE	Command Complete Interrupt Enable Controls interrupt generation when a flash command has completed. 0 Command complete interrupt is disabled. 1 An interrupt will be requested whenever the CCIF flag in the FSTAT register is set.
6–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

11.4.5 Flash Status Register (FTMRH_FSTAT)

The FSTAT register reports the operational status of the flash module.

Address: 1830h base + 6h offset = 1836h

Bit	7	6	5	4	3	2	1	0
Read	CCIF	0	ACCERR	FPVIOL	MGBUSY	0	MGSTAT	
Write								
Reset	1	0	0	0	0	0	0	0

FTMRH_FSTAT field descriptions

Field	Description
7 CCIF	Command Complete Interrupt Flag Indicates that a flash command has completed. The CCIF flag is cleared by writing a 1 to CCIF to launch a command and CCIF will stay low until command completion or command violation. 0 Flash command is in progress. 1 Flash command has completed.
6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5 ACCERR	Flash Access Error Flag Indicates an illegal access has occurred to the flash memory caused by either a violation of the command write sequence or issuing an illegal flash command. While ACCERR is set, the CCIF flag cannot be cleared to launch a command. Writing 1 to this field clears it while writing a 0 to this field has no effect. 0 No access error is detected. 1 Access error is detected.
4 FPVIOL	Flash Protection Violation Flag Indicates an attempt was made to program or erase an address in a protected area of flash memory during a command write sequence. Writing 1 to FPVIOL clears this field while writing 0 to this field has no effect. While FPIOL is set, it is not possible to launch a command or start a command write sequence. 0 No protection violation is detected. 1 Protection violation is detected.
3 MGBUSY	Memory Controller Busy Flag Reflects the active state of the memory controller. 0 Memory controller is idle. 1 Memory controller is busy executing a flash command (CCIF = 0).
2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
MGSTAT	Memory Controller Command Completion Status Flag

Table continues on the next page...

FTMRH_FSTAT field descriptions (continued)

Field	Description
	One or more MGSTAT flag bits are set if an error is detected during execution of a flash command or during the flash reset sequence. **NOTE:** Reset value can deviate from the value shown if a double bit fault is detected during the reset sequence.

11.4.6 Flash Protection Register (FTMRH_FPROT)

The FPROT register defines which flash sectors are protected against program and erase operations.

The unreserved bits of the FPROT register are writable with the restriction that the size of the protected region can only be increased (see Protection). All the unreserved bits of the FPROT register are writable without restriction in active Background Debug Mode (BDM).

During the reset sequence, the FPROT register is loaded with the contents of the flash protection byte in the flash configuration field at global address 0x40D located in flash memory. To change the flash protection that will be loaded during the reset sequence, the upper sector of the flash memory must be unprotected, then the flash protection byte must be reprogrammed.

Trying to alter data in any protected area in the flash memory will result in a protection violation error and the FPVIOL bit will be set in the FSTAT register. The block erase of a flash block is not possible if any of the flash sectors contained in the same flash block are protected.

Address: 1830h base + 8h offset = 1838h

Bit	7	6	5	4	3	2	1	0
Read	FPOPEN	RNV6	FPHDIS	FPHS		RNV		
Write								
Reset	x*	x*	x*	x*	x*	0	0	0

FTMRH_FPROT field descriptions

Field	Description
7 FPOPEN	Flash Protection Operation Enable The FPOPEN bit determines the protection function for program or erase operations. 0 When FPOPEN is clear, the FPHDIS fields defines unprotected address ranges as specified by the corresponding FPHS field. 1 When FPOPEN is set, the FPHDIS fields enables protection for the address range specified by the corresponding FPHS field.

Table continues on the next page...

FTMRH_FPROT field descriptions (continued)

Field	Description
6 RNV6	Reserved Nonvolatile Bit The RNV bit must remain in the erased state.
5 FPHDIS	Flash Protection Higher Address Range Disable The FPHDIS bit determines whether there is a protected/unprotected area in a specific region of the flash memory ending with global address 0x7FFF. 0 Protection/Unprotection enabled. 1 Protection/Unprotection disabled.
4–3 FPHS	Flash Protection Higher Address Size The FPHS bits determine the size of the protected/unprotected area in flash memory. The FPHS bits can be written to only while the FPHDIS bit is set.
RNV	Reserved Nonvolatile Bit The RNV bit must remain in the erased state.

11.4.7 Flash Common Command Object Register:High (FTMRH_FCCOBHI)

The FCCOB is an array of six words addressed via the CCOBIX index found in the FCCOBIX register. Byte-wide reads and writes are allowed to the FCCOB register.

Address: 1830h base + Ah offset = 183Ah

Bit	7	6	5	4	3	2	1	0
Read Write	CCOB							
Reset	0	0	0	0	0	0	0	0

FTMRH_FCCOBHI field descriptions

Field	Description
CCOB	Common Command Object Bit 15:8 High 8 bits of Common Command Object register

11.4.8 Flash Common Command Object Register: Low (FTMRH_FCCOBLO)

The FCCOB is an array of six words addressed via the CCOBIX index found in the FCCOBIX register. Byte-wide reads and writes are allowed to the FCCOB register.

Address: 1830h base + Bh offset = 183Bh

Bit	7	6	5	4	3	2	1	0
Read Write	CCOB							
Reset	0	0	0	0	0	0	0	0

FTMRH_FCCOBLO field descriptions

Field	Description
CCOB	Common Command Object Bit 7:0 Low 8 bits of Common Command Object register

11.4.9 Flash Option Register (FTMRH_FOPT)

The FOPT register is the flash option register.

All bits in the FOPT register are readable but are not writable. In active Background Debug Mode (BDM), all bits in the FOPT register are readable and writable .

During the reset sequence, the FOPT register is loaded from the flash nonvolatile byte in the flash configuration field at global address 0x040F located in flash memory as indicated by reset condition.

Address: 1830h base + Ch offset = 183Ch

Bit	7	6	5	4	3	2	1	0
Read	NV							
Write								
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- x = Undefined at reset.

FTMRH_FOPT field descriptions

Field	Description
NV	Nonvolatile Bits

FTMRH_FOPT field descriptions (continued)

Field	Description
	The NV[7:0] bits are available as nonvolatile bits. During the reset sequence, the FOPT register is loaded from the flash nonvolatile byte in the flash configuration field at global address 0x40F located in flash memory.

Chapter 12
Internal Clock Source (ICS)

12.1 Introduction

The internal clock source (ICS) module provides clock source choices for the MCU. The module contains a frequency-locked loop (FLL) as a clock source that is controllable by either an internal or an external reference clock. The module can provide this FLL clock or either of the internal or external reference clocks as a source for the MCU system clock. There are also signals provided to control a low-power oscillator (OSC) module. These signals configure and enable the OSC module to generate its external crystal/ resonator clock (OSC_OUT) used by peripheral modules and as the ICS external reference clock source. The ICS external reference clock can be the external crystal/ resonator (OSC_OUT) supplied by an OSC, or it can be another external clock source.

The ICS clock source chosen is passed through a reduced bus divider (BDIV) which allows a lower final output clock frequency to be derived.

12.1.1 Features

The key features of the ICS module are given below:

- Internal reference clock has 9 trim bits for accuracy
- Internal or external reference clocks can be used to control the FLL.
- Selectable dividers for external reference clock to ensure proper input frequency to FLL.
- Internal or external reference clocks can be selected as the clock source for the MCU.
- FLL Engaged Internal mode is automatically selected out of reset.
- FLL lock detector and external clock monitor
  - FLL lock detector with interrupt capability
  - External reference clock monitor with reset capability
- Digitally controlled oscillator optimized for 32-40 MHz frequency range

12.1.2 Block diagram

The following figure is the ICS block diagram.

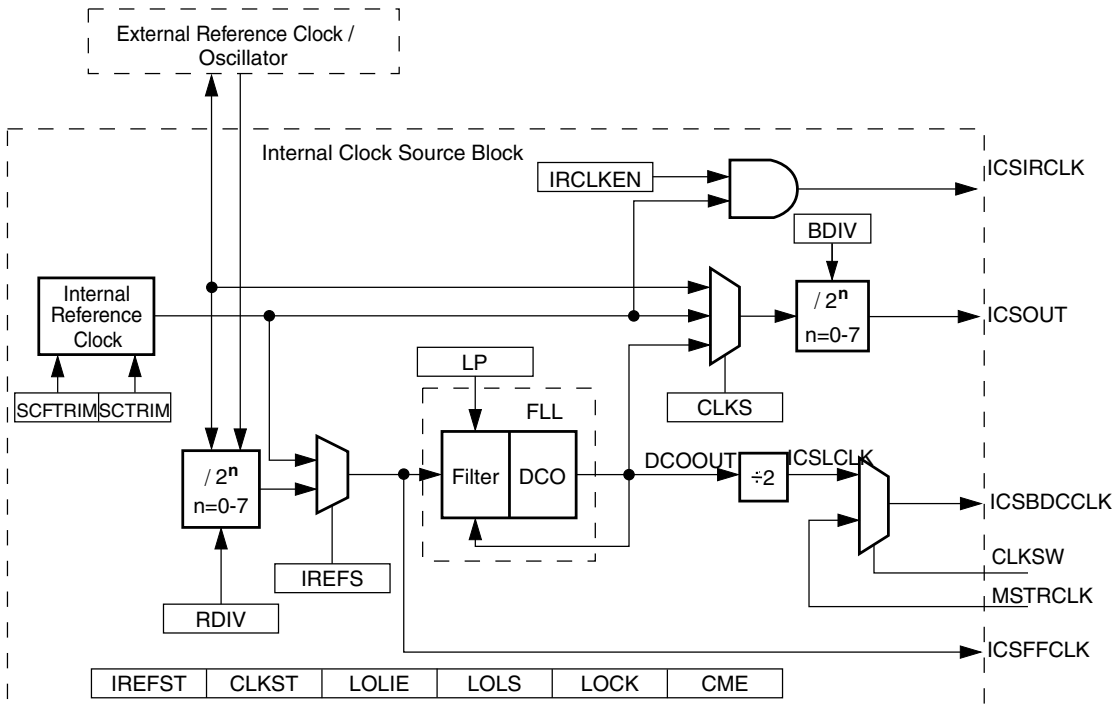


Figure 12-1. Internal clock source (ICS) block diagram

12.1.3 Modes of operation

There are seven modes of operation for the ICS: FEI, FEE, FBI, FBILP, FBE, FBELP, and STOP. Each of these modes is explained briefly in the following subsections.

12.1.3.1 FLL engaged internal (FEI)

In FLL engaged internal mode, which is the default mode, the ICS supplies a clock derived from the FLL which is controlled by the internal reference clock.

12.1.3.2 FLL engaged external (FEE)

In FLL engaged external mode, the ICS supplies a clock derived from the FLL which is controlled by an external reference clock source.

12.1.3.3 FLL bypassed internal (FBI)

In FLL bypassed internal mode, the FLL is enabled and controlled by the internal reference clock, but is bypassed. The ICS supplies a clock derived from the internal reference clock.

12.1.3.4 FLL bypassed internal low power (FBILP)

In FLL bypassed internal low power mode, the FLL is disabled and bypassed, and the ICS supplies a clock derived from the internal reference clock.

12.1.3.5 FLL bypassed external (FBE)

In FLL bypassed external mode, the FLL is enabled and controlled by an external reference clock, but is bypassed. The ICS supplies a clock derived from the external reference clock source.

12.1.3.6 FLL bypassed external low power (FBELP)

In FLL bypassed external low power mode, the FLL is disabled and bypassed, and the ICS supplies a clock derived from the external reference clock.

12.1.3.7 Stop (STOP)

In Stop mode, the FLL is disabled. The ICS does not provide any MCU clock sources.

NOTE

The DCO frequency changes from the pre-stop value to its reset value and the FLL needs to reacquire the lock before the frequency is stable. Timing sensitive operations must wait for the FLL acquisition time, $t_{Acquire}$, before executing.

12.2 External signal description

There are no ICS signals that connect off chip.

12.3 Register definition

ICS memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
1848	ICS Control Register 1 (ICS_C1)	8	R/W	04h	12.3.1/192
1849	ICS Control Register 2 (ICS_C2)	8	R/W	20h	12.3.2/193
184A	ICS Control Register 3 (ICS_C3)	8	R/W	See section	12.3.3/194
184B	ICS Control Register 4 (ICS_C4)	8	R/W	See section	12.3.4/195
184C	ICS Status Register (ICS_S)	8	R	10h	12.3.5/196

12.3.1 ICS Control Register 1 (ICS_C1)

Address: 1848h base + 0h offset = 1848h

Bit	7	6	5	4	3	2	1	0
Read Write	CLKS		RDIV			IREFS	IRCLKEN	Reserved
Reset	0	0	0	0	0	1	0	0

ICS_C1 field descriptions

Field	Description
7–6 CLKS	Clock Source Select Selects the clock source that controls the bus frequency. The actual bus frequency depends on the value of ICS_C2[BDIV]. 00 Output of FLL is selected. 01 Internal reference clock is selected. 10 External reference clock is selected. 11 Reserved, defaults to 00.
5–3 RDIV	Reference Divider Changing RDIV will cause the change of reference clock frequency of FLL, RDIV is not allowed to be changed in FEE/FBE mode. Selects the amount to divide down the FLL reference clock selected by the IREFS bits. Resulting frequency must be in the range 31.25 kHz to 39.0625 kHz.

RDIV	SIM_SOPT1[RANGE]= 0	SIM_SOPT1[RANGE]= 1
000	1[1]	32
001	2	64
010	4	128
011	8	256

Table continues on the next page...

ICS_C1 field descriptions (continued)

Field	Description		
	RDIV	**SIM_SOPT1[RANGE]= 0**	**SIM_SOPT1[RANGE]= 1**
	100	16	512
	101	32	1024
	110	64	Reserved
	111	128	Reserved
	1. Reset default		
2 IREFS	Internal Reference Select Selects the reference clock source for the FLL. 0 External reference clock is selected. 1 Internal reference clock is selected.		
1 IRCLKEN	Internal Reference Clock Enable Enables the internal reference clock for use as ICSIRCLK. 0 ICSIRCLK is inactive. 1 ICSIRCLK is active.		
0 Reserved	This field is reserved.		

1. Reset default

12.3.2 ICS Control Register 2 (ICS_C2)

Address: 1848h base + 1h offset = 1849h

Bit	7	6	5	4	3	2	1	0
Read Write	BDIV			LP	0			
Reset	0	0	1	0	0	0	0	0

ICS_C2 field descriptions

Field	Description
7–5 BDIV	Bus Frequency Divider Selects the amount to divide down the clock source selected by ICS_C1[CLKS]. This controls the bus frequency. 000 Encoding 0—Divides the selected clock by 1. 001 Encoding 1—Divides the selected clock by 2 (reset default). 010 Encoding 2—Divides the selected clock by 4. 011 Encoding 3—Divides the selected clock by 8. 100 Encoding 4—Divides the selected clock by 16. 101 Encoding 5—Divides the selected clock by 32.

Table continues on the next page...

ICS_C2 field descriptions (continued)

Field	Description
	110 Encoding 6—Divides the selected clock by 64. 111 Encoding 7—Divides the selected clock by 128.
4 LP	Low Power Select Controls whether the FLL is disabled in FLL bypassed modes. 0 FLL is not disabled in bypass mode. 1 FLL is disabled in bypass modes unless debug is active.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

12.3.3 ICS Control Register 3 (ICS_C3)

Address: 1848h base + 2h offset = 184Ah

Bit	7	6	5	4	3	2	1	0
Read Write	SCTRIM							
Reset	x*	x*	x*	x*	x*	x*	x*	x*

* Notes:
- SCTRIM is loaded during reset from a factory programmed location when not in BDM mode. If in a BDM mode, SCTRIM gets loaded with a value of 0x80. x = Undefined at reset.

ICS_C3 field descriptions

Field	Description
SCTRIM	Slow Internal Reference Clock Trim Setting Controls the slow internal reference clock frequency by controlling the internal reference clock period. The bits are binary weighted. In other words, bit 1 adjusts twice as much as bit 0. Increasing the binary value of SCTRIM will increase the period, and decreasing the value will decrease the period. An additional fine trim bit is available as the ICS_C4[SCFTRIM]. ICS_C3 is automatically loaded during reset from a factory programmed location when not in a debug mode. The factory programmed trim value adjusts the internal oscillator frequency to fint_ft as specified in the datasheet. The user can provide a custom trim value to attain other internal reference clock frequencies within the fint_t range. The custom trim value must be programmed into reserved flash location 0x0000_FF6F and copied to ICS_C3 during code initialization.

12.3.4 ICS Control Register 4 (ICS_C4)

Address: 1848h base + 3h offset = 184Bh

Bit	7	6	5	4	3	2	1	0
Read	LOLIE	0	CME	0				SCFTRIM
Write								
Reset	0	0	0	0	0	0	0	*

* Notes:
- SCFTRIM field: SCTRIM is loaded during reset from a factory programmed location when not in BDM mode. If in a BDM mode, SCTRIM gets loaded with a value of 1b.

ICS_C4 field descriptions

Field	Description
7 LOLIE	Loss of Lock Interrupt Determines if an interrupt request is made following a loss of lock indication. This field has an effect only when ICS_S[LOLS] is set. 0 No request on loss of lock. 1 Generates an interrupt request on loss of lock.
6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5 CME	Clock Monitor Enable Determines if a reset request is made following a loss of external clock indication. This field must be set to a logic 1 only when the ICS is in an operational mode that uses the external clock (FEE, FBE, or FBELP). 0 Clock monitor is disabled. 1 Generates a reset request on loss of external clock.
4–1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 SCFTRIM	Slow Internal Reference Clock Fine Trim Controls the smallest adjustment of the internal reference clock frequency. Setting SCFTRIM will increase the period and clearing SCFTRIM will decrease the period by the smallest amount possible. ICS_C4[SCFTRIM] is automatically loaded during reset from a factory programmed location when not in a debug mode. The factory programmed trim value adjusts the internal oscillator frequency to fint_ft as specified in the datasheet. The user can provide a custom trim value to attain other internal reference clock frequencies within the fint_t range. The custom fine trim bit value must be programmed into reserved flash location 0x0000_FF6E and copied to ICS_C4 during code initialization.

12.3.5 ICS Status Register (ICS_S)

Address: 1848h base + 4h offset = 184Ch

Bit	7	6	5	4	3	2	1	0
Read	LOLS	LOCK	0	IREFST	CLKST		0	
Write	w1c							
Reset	0	0	0	1	0	0	0	0

ICS_S field descriptions

Field	Description
7 LOLS	Loss of Lock Status Indicates the lock status for the FLL. LOLS is set when lock detection is enabled and after acquiring lock, the FLL output frequency has fallen outside the lock exit frequency tolerance, from ±4.7% to ±5.97%. ICS_C4[LOLIE] determines whether an interrupt request is made when set. LOLS is cleared by reset or by writing a logic 1 to LOLS when LOLS is set. Writing a logic 0 to LOLS has no effect. 0 FLL has not lost lock since LOLS was last cleared. 1 FLL has lost lock since LOLS was last cleared.
6 LOCK	Lock Status Indicates whether the FLL has acquired lock. Lock detection is disabled when FLL is disabled. If the lock status bit is set then changing the value of any of the following fields IREFS, RDIV[2:0], or, if in FEI or FBI modes, SCTRIM[7:0] will cause the lock status bit to clear and stay cleared until the FLL has reacquired lock. Stop mode entry will also cause the lock status bit to clear and stay cleared until the FLL has reacquired lock. 0 FLL is currently unlocked. 1 FLL is currently locked.
5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
4 IREFST	Internal Reference Status Indicates the current source for the reference clock. This field does not update immediately after a write to ICS_C1[IREFS] due to internal synchronization between clock domains. 0 Source of reference clock is external clock. 1 Source of reference clock is internal clock.
3–2 CLKST	Clock Mode Status Indicates the current clock mode. This field doesn't update immediately after a write to ICS_C1[CLKS] due to internal synchronization between clock domains. 00 Output of FLL is selected. 01 FLL Bypassed, internal reference clock is selected. 10 FLL Bypassed, external reference clock is selected. 11 Reserved.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

12.4 Functional description

12.4.1 Operational modes

The seven states of the ICS are shown as a state diagram and are described below. The arrows indicate the allowed movements among the states.

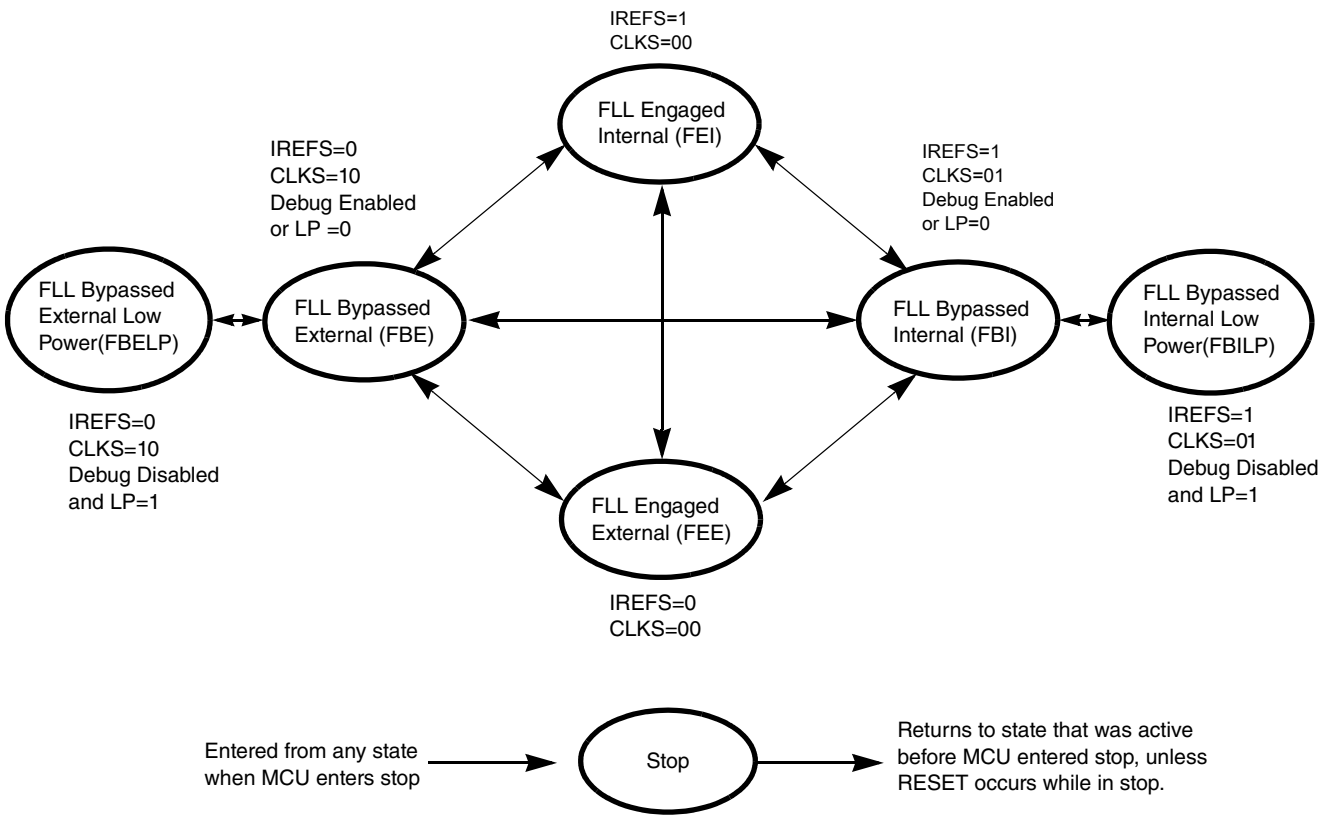


Figure 12-2. Clock switching modes

12.4.1.1 FLL engaged internal (FEI)

FLL engaged internal (FEI) is the default mode of operation and is entered when all the following conditions occur:

- 00b is written to ICS_C1[CLKS].
- 1b is written to ICS_C1[IREFS].

In FLL engaged internal mode, the ICSOUT clock is derived from the FLL clock, which is controlled by the internal reference clock. The FLL loop locks the frequency to 1024 times the internal reference frequency. The internal reference clock is enabled.

12.4.1.2 FLL engaged external (FEE)

The FLL engaged external (FEE) mode is entered when all the following conditions occur:

- 00b is written to ICS_C1[CLKS].
- 0b is written to ICS_C1[IREFS].
- ICS_C1[RDIV] and SIM_SOPT1[RANGE] are written to divide external reference clock to be within the range of 31.25 kHz to 39.0625 kHz.

In FLL engaged external mode, the ICSOUT clock is derived from the FLL clock which is controlled by the external reference clock source.The FLL loop locks the frequency to 1024 times the external reference frequency, as selected by ICS_C1[RDIV] and SIM_SOPT1[RANGE]. The external reference clock is enabled.

12.4.1.3 FLL bypassed internal (FBI)

The FLL bypassed internal (FBI) mode is entered when all the following conditions occur:

- 01b is written to ICS_C1[CLKS].
- 1b is written to ICS_C1[IREFS].
- BDM mode is active or ICS_C2[LP] bit is written to 0.

In FLL bypassed internal mode, the ICSOUT clock is derived from the internal reference clock. The FLL clock is controlled by the internal reference clock, and the FLL loop locks the FLL frequency to 1024 times the internal reference frequency. The internal reference clock is enabled.

12.4.1.4 FLL bypassed internal low power (FBILP)

The FLL bypassed internal low power (FBILP) mode is entered when all the following conditions occur:

- 01b is written to ICS_C1[CLKS].
- 1b is written to ICS_C1[IREFS].
- BDM mode is not active and ICS_C2[LP] bit is written to 1b.

In FLL bypassed internal low-power mode, the ICSOUT clock is derived from the internal reference clock and the FLL is disabled. The internal reference clock is enabled.

12.4.1.5 FLL bypassed external (FBE)

The FLL bypassed external (FBE) mode is entered when all the following conditions occur:

- 10b is written to ICS_C1[CLKS].
- 0b is written to ICS_C1[IREFS].
- ICS_C1[RDIV] and SIM_SOPT1[RANGE] fields are written to divide external reference clock to be within the range of 31.25 kHz to 39.0625 kHz.
- BDM mode is active or 0b is written to C2[LP].

In FLL bypassed external mode, the ICSOUT clock is derived from the external reference clock source. The FLL clock is controlled by the external reference clock, and the FLL loop locks the FLL frequency to 1024 times the external reference frequency, as selected by ICS_C1[RDIV] and SIM_SOPT1[RANGE], the external reference clock is enabled.

12.4.1.6 FLL bypassed external low power (FBELP)

The FLL bypassed external low-power (FBELP) mode is entered when all the following conditions occur:

- 10b is written to ICS_C1[CLKS].
- 0b is written to ICS_C1[IREFS].
- BDM mode is not active and ICS_C2[LP] bit is written to 1b.

In FLL bypassed external low-power mode, the ICSOUT clock is derived from the external reference clock source and the FLL is disabled. The external reference clock source is enabled.

12.4.1.7 Stop

NOTE

The DCO frequency changes from the pre-stop value to its reset value and the FLL need to re-acquire the lock before the frequency is stable. Timing sensitive operations must wait for the FLL acquisition time, $t_{Acquire}$, before executing.

Stop mode is entered whenever the MCU enters a STOP state.

12.4.2 Mode switching

ICS_C1[IREFS] can be changed at anytime, but the actual switch to the newly selected clock is shown by ICS_S[IREFST]. When switching between FLL engaged internal (FEI) and FLL engaged external (FEE) modes, the FLL begins locking again after the switch is completed.

ICS_C1[CLKS] can also be changed at anytime, but the actual switch to the newly selected clock is shown by ICS_S[CLKST]. If the newly selected clock is not available, the previous clock remains selected.

NOTE

When mode switching is from FEE, FBE or FBELP to FEI, it is suggested to wait IREFST switch completion, then change ICS_C1[CLKS].

12.4.3 Bus frequency divider

ICS_C2[BDIV] can be changed anytime and the actual switch to the new frequency occurs immediately.

12.4.4 Low-power field usage

The Low-Power (LP) field in the ICS_C2 register is provided to allow the FLL to be disabled and thus conserve power when it is not being used.

However, in some applications it may be desirable to allow the FLL to be enabled and to lock for maximum accuracy before switching to an FLL engaged mode. To do this, write 0b to ICS_C2[LP].

12.4.5 Internal reference clock

When ICS_C1[IRCLKEN] is set, the internal reference clock signal is presented as ICSIRCLK, which can be used as an additional clock source. To re-target the ICSIRCLK frequency, write a new value to the ICS_C3[SCTRIM] bits to trim the period of the internal reference clock:

- Writing a larger value slows down the ICSIRCLK frequency.
- Writing a smaller value to the ICS_C3 register speeds up the ICSIRCLK frequency.

The trim bits affect the ICSOUT frequency if the ICS is in FLL engaged internal (FEI), FLL bypassed internal (FBI), or FLL bypassed internal low power (FBILP) mode.

Until ICSIRCLK is trimmed, programming low bus divider (ICS_C2[BDIV]) factors may result in ICSOUT frequencies that exceed the maximum chip-level frequency and violate the chip-level clock timing specifications.

All MCU devices are factory programmed with a trim value in a reserved memory location. This value is uploaded to the ICS_C3 register and ICS_C4[SCFTRIM] during any reset initialization. For finer precision, trim the internal oscillator in the application and set ICS_C4[SCFTRIM] accordingly.

12.4.6 Fixed frequency clock

The ICS presents the divided FLL reference clock as ICSFFCLK for use as an additional clock source. ICSFFCLK frequency must be no more than 1/4 of the ICSOUT frequency to be valid. Because of this requirement, in bypass modes, the ICSFFCLK is valid only in bypass external modes (FBE and FBELP) for the following conditions of ICS_C2[BDIV], and divider factor of ICS_C1[RDIV] and SIM_SOPT1[RANGE] values:

if SIM_SOPT1[RANGE] is high,

- ICS_C2[BDIV] = 000, ICS_C2[RDIV] ≥ 010
- ICS_C2[BDIV] = 001 (divide by 2), ICS_C2[RDIV] ≥ 011
- ICS_C2[BDIV] = 010 (divide by 4), ICS_C2[RDIV] ≥ 100
- ICS_C2[BDIV] = 011 (divide by 8), ICS_C2[RDIV] ≥ 101

12.4.7 FLL lock and clock monitor

12.4.7.1 FLL clock lock

In FBE and FEE modes, the clock detector source uses the external reference as the reference. When FLL is detected from lock to unlock, ICS_S[LOLS] is set. An interrupt will be generated if ICS_C4[LOLIE] is set. ICS_S[LOLS] is cleared by reset or by writing a logic 1 to ICS_S[LOLS] when ICS_S[LOLS] is set. Writing a logic 0 to ICS_S[LOLS] has no effect.

In FBI and FEI modes, the lock detector source uses the internal reference as the reference. When FLL is detected from lock to unlock, ICS_S[LOLS] is set. An interrupt will be generated if ICS_C4[LOLIE] is set. ICS_S[LOLS] is cleared by reset or by writing a logic 1 to ICS_S[LOLS] when ICS_S[LOLS] is set. Writing a logic 0 to ICS_S[LOLS] has no effect.

In FBELP and FBILP modes, the FLL is not on so that lock detect function is not applicable.

12.4.7.2 External reference clock monitor

In FBE, FEE, or FBELP modes, if 1 is written to ICS_C4[CME], the clock monitor is enabled. If the external reference falls below a certain frequency, the MCU will reset. The SIM_SRS[LOC] will be set to indicate the error.

12.5 Initialization/application information

This section provides example code to give some basic direction to a user on how to initialize and configure the ICS module. The example software is implemented in C language.

12.5.1 Initializing FEI mode

The following code segment demonstrates setting ICS to FEI mode.

Example: 12.5.1.1 FEI mode initialization routine

```
/* the following code segment demonstrates setting the ICS to FEI mode using the factory
trim value. The resulting ICSOUT frequency is fint_ft*1024/BDIV. */
ICS_C2 = 0x20; // BDIV=divide by 2 - use default until clock dividers configured
ICS_C1 = 0x04; // internal reference clock as source to FLL
while ((ICS_S & ICS_S_LOCK_MASK) == 0); // wait for FLL to lock
ICS_C2 = 0x00; // BDIV=divide by 1 - allows max core and bus clock frequencies

/* the following code segment demonstrates setting the ICS to FEI mode using a custom trim
value provided by a programming tool. The resulting ICSOUT frequency is fint_t*1024/BDIV. */
ICS_C3 = *((uint8_t*) 0xFF6F); // trim internal reference clock
ICS_C4 = *((uint8_t*) 0xFF6E); // fine trim internal reference clock
ICS_C2 = 0x20; // BDIV=divide by 2 - use default until clock dividers configured
ICS_C1 = 0x04; // internal reference clock as source to FLL
while ((ICS_S & ICS_S_LOCK_MASK) == 0); // wait for FLL to lock
ICS_C2 = 0x00; // BDIV=divide by 1 - allows max core and bus clock frequencies
```

12.5.2 Initializing FBI mode

The following code segment demonstrates setting ICS to FBI mode.

Example: 12.5.2.1 FBI mode initialization routine

```
/* the following code segment demonstrates setting the ICS to FBI mode using the factory
trim value. The resulting ICSOUT frequency is fint_ft/BDIV. Note that the FLL will be
running at a frequency of fint_ft*1024/BDIV even though the FLL is bypassed. */
ICS_C2 = 0x20; // BDIV=divide by 2 - use default until clock dividers configured
ICS_C1 = 0x44; // internal reference clock as source for ICSOUT
while ((ICS_S & 0x0C) != 0x04); // wait until internal reference is selected
ICS_C2 = 0x00; // BDIV=divide by 1 - allows max core and bus clock frequencies
```

12.5.3 Initializing FEE mode

The following code segment demonstrates setting ICS to FEE mode.

Example: 12.5.3.1 FEE mode initialization routine

```
/* the following code segment demonstrates setting the ICS to FEE mode generating a 32MHZ
bus clock frequency using an external 8MHz crystal */
SIM_SOPT1[RANGE] = 1; // high range
ICS_C2 = 0x20; // BDIV=divide by 2 - use default until clock dividers configured
ICS_C1 = 0x18; // 8MHz external reference clock/256 as source to FLL
while ((ICS_S & ICS_S_IREFST_MASK) == 1); // wait for external source selected
while ((ICS_S & ICS_S_LOCK_MASK) == 0); // wait for FLL to lock
ICS_C2 = 0x00; // BDIV=divide by 1 - allows max core and bus clock frequencies
```

12.5.4 Initializing FBE mode

The following code segment demonstrates setting ICS to FBE mode.

Example: 12.5.4.1 FBE mode initialization routine

```
/* the following code segment demonstrates setting the ICS to FBE mode generating 20MHZ core
clock frequency using an external 20MHz crystal */
SIM_SOPT1[RANGE] = 1; // high range
ICS_C2 = 0x20; // BDIV=divide by 2 - use default until clock dividers configured
ICS_C1 = 0xA0; // 20MHz external clock as ICSOUT source; FLL source = 20MHz/512
while ((ICS_S & ICS_S_IREFST_MASK) == 1); // wait for external source selected
while ((ICS_S & 0x0C) != 0x08); // wait until FBE mode is selected
ICS_C2 = 0x00; // BDIV=divide by 1 - allows max core and bus clock frequencies
```

Chapter 13
Modulo Timer (MTIM)

13.1 Chip specific modulo timer

This device has one modulo timer (MTIM).

This MTIM is a 16-bit modulo timer module that provides a circuit of selectable clock sources and a programmable interrupt. The MTIM module contains a 16-bit modulo counter, which can operate as a free-running counter or a modulo counter. A timer overflow interrupt can be enabled to generate periodic interrupts for time-based software events. MTIM16 module may also use external clock source.

Customization:

- Primary clock: BUSCLK (20 MHz)
- Alternate clock: ICSFFCLK/2, TCLK
- The following table summarizes the signal connection of MTIM16 module.

Table 13-1. MTIM module signals connection

Module	Signal	Connect to
MTIM	Internal clock	BUSCLK, ICSFFCLK/2
	External clock	PTB6/TCLK

13.2 Introduction

NOTE

For the chip-specific implementation details of this module's instances, see the chip configuration information.

The MTIM (or MTIM16) module is a simple 16-bit timer with several software selectable clock sources and a programmable interrupt.

13.3 Features

Timer system features include:

- 16-bit up-counter
  - Free-running or 16-bit modulo limit
  - Software controllable interrupt on overflow
  - Counter reset bit (TRST)
  - Counter stop bit (TSTP)
- Four software selectable clock sources for input to prescaler:
  - System bus clock — rising edge
  - Fixed frequency clock (XCLK) — rising edge
  - External clock source on the TCLK pin — rising edge
  - External clock source on the TCLK pin — falling edge
- Nine selectable clock prescale values:
  - Clock source divide by 1, 2, 4, 8, 16, 32, 64, 128, or 256
- Modulo compare matched can be an output

13.3.1 Block Diagram

The following figure is a block diagram of the modulo timer module.

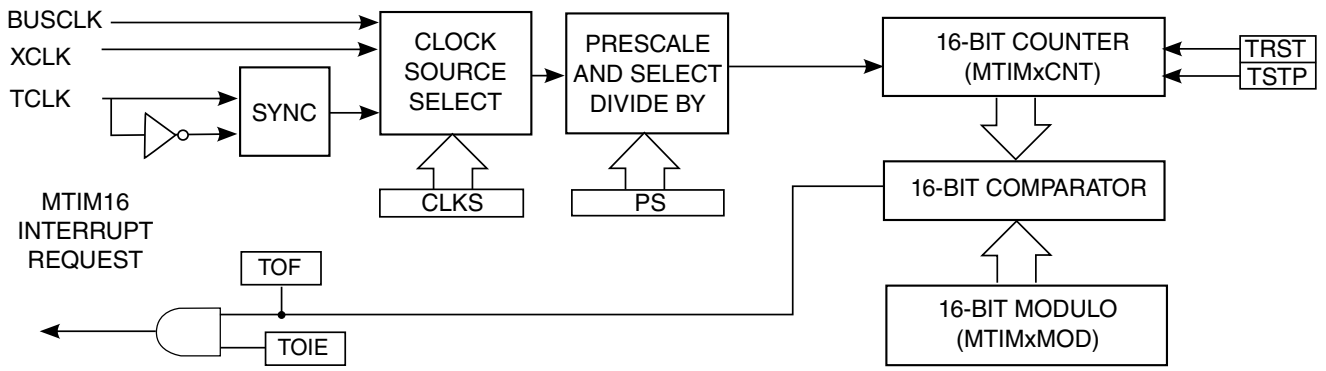


Figure 13-1. Modulo Timer (MTIM16) Block Diagram

13.3.2 Modes of Operation

This section defines MTIM16 operation in stop, wait, and background debug modes.

13.3.2.1 MTIM16 in Wait Mode

The MTIM16 continues to run in wait mode if enabled prior to the execution of the WAIT instruction. The timer overflow interrupt brings the MCU out of wait mode if it is enabled. For lowest possible current consumption, the MTIM16 should be stopped by software if it is not needed as an interrupt source during wait mode.

13.3.2.2 MTIM16 in Stop Modes

MTIM operation in stop modes is chip-specific. See details about MCU power modes and clocking.

- If the MTIM is unclocked in any MCU stop mode, it is disabled in that mode, regardless of the module settings before the STOP instruction was executed. It cannot be used as a wakeup source in that mode.
- If the MTIM is clocked and enabled in an MCU stop mode, it can be used as a wakeup source in that mode.

Upon waking from very low-power stop modes, the MTIM enters its reset state.

For low-power stop modes:

- If the device exits any of these modes with a reset, the MTIM module enters its reset state.

- If the device exits any of these modes with an interrupt, the MTIM module continues from its state in the low-power stop mode.
- If the counter was active upon entering any of these modes, the count resumes from the current value.

13.3.2.3 MTIM16 in Active Background Mode

The MTIM16 stops all counting until the microcontroller returns to normal user operating mode. Counting resumes from the suspended value as long as an MTIM16 reset did not occur (TRST written to a 1).

13.4 External Signal Description

This section describes the module external signals.

13.4.1 TCLK — External Clock Source Input into MTIM16

The MTIM16 includes one external signal, TCLK, used to input an external clock when selected as the MTIM16 clock source. The signal properties of TCLK are shown in the following table.

Table 13-2. Signal Properties

Signal	Function	I/O
TCLK	External clock source input into MTIM16	I

The TCLK input must be synchronized by the bus clock. Also, variations in duty cycle and clock jitter must be accommodated. As a result, the TCLK signal must be limited to one-fourth of the bus frequency.

The TCLK pin can be muxed with a general-purpose port pin. Refer to the chip-level signal multiplexing and pin assignment details for more information.

13.5 Memory Map and Register Descriptions

Each MTIM16 module includes six registers.

Refer to the direct-page register summary for the absolute address assignments for all MTIM16 registers. This section refers to registers and control bits by their names and relative address offsets.

If a chip has more than one MTIM16 module, register names include placeholder characters.

MTIM memory map

Address offset (hex)	Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
0	8	MTIM16 status and control register (MTIM_SC)	8	R/W	10h	13.5.1/209
1	9	MTIM16 clock configuration register (MTIM_CLK)	8	R/W	00h	13.5.2/210
2	A	MTIM16 counter register high (MTIM_CNTH)	8	R	00h	13.5.3/211
3	B	MTIM16 counter register low (MTIM_CNTL)	8	R	00h	13.5.4/212
4	C	MTIM16 modulo register high (MTIM_MODH)	8	R/W	00h	13.5.5/213
5	D	MTIM16 modulo register low (MTIM_MODL)	8	R/W	00h	13.5.6/214

13.5.1 MTIM16 status and control register (MTIM_SC)

This register contains the overflow status flag and control bits. Use them to configure the interrupt enable, reset the counter, and stop the counter.

Address: 8h base + 0h offset = 8h

Bit	7	6	5	4	3	2	1	0
Read	TOF	TOIE	0	TSTP	0			
Write	0		TRST					
Reset	0	0	0	1	0	0	0	0

MTIM_SC field descriptions

Field	Description
7 TOF	MTIM16 overflow flag

Table continues on the next page...

MTIM_SC field descriptions (continued)

Field	Description
	This bit is set when the MTIM16 counter register overflows to 0x0000 after reaching the value in the MTIM16 modulo register. Clear TOF by reading the SC register while TOF is set and then by writing 0 to TOF. Writing 1 has no effect. TOF is also cleared when 1 is written to TRST. 0 MTIM16 counter has not reached the overflow value in the MTIM16 modulo register. 1 MTIM16 counter has reached the overflow value in the MTIM16 modulo register.
6 TOIE	MTIM16 overflow interrupt enable This read/write bit enables MTIM16 overflow interrupts. If TOIE is set, then an interrupt is generated when TOF = 1. Reset clears TOIE. Do not set TOIE if TOF = 1; instead, clear TOF first, and then set TOIE. 0 TOF interrupts are disabled. Use software polling. 1 TOF interrupts are enabled.
5 TRST	MTIM16 counter reset When 1 is written to this write-only bit, the MTIM16 counter register resets to 0x0000 and TOF is cleared. Writing 1 to this bit also causes the modulo value to take effect at once. Reading this bit always returns 0. 0 No effect. MTIM16 counter remains in its current state. 1 MTIM16 counter is reset to 0x0000.
4 TSTP	MTIM16 counter stop When set, this read/write bit stops the MTIM16 counter at its current value. Counting resumes from the current value when TSTP is cleared. Reset sets TSTP to prevent the MTIM16 from counting. 0 MTIM16 counter is active. 1 MTIM16 counter is stopped.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

13.5.2 MTIM16 clock configuration register (MTIM_CLK)

This register contains the clock select bits (CLKS) and the prescaler select bits (PS).

Address: 8h base + 1h offset = 9h

Bit	7	6	5	4	3	2	1	0
Read	0		CLKS		PS			
Write								
Reset	0	0	0	0	0	0	0	0

MTIM_CLK field descriptions

Field	Description
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

Table continues on the next page...

MTIM_CLK field descriptions (continued)

Field	Description
5–4 CLKS	Clock source select These two read/write bits select one of four different clock sources as the input to the MTIM16 prescaler. Changing the clock source while the counter is active does not clear the counter. The count continues with the new clock source. Reset clears CLKS to 00. 00 Encoding 0. Bus clock (BUSCLK) 01 Encoding 1. Fixed-frequency clock (XCLK) 10 Encoding 3. External source (TCLK pin), falling edge 11 Encoding 4. External source (TCLK pin), rising edge
PS	Clock source prescaler These four read/write bits select one of nine outputs from the 8-bit prescaler. Changing the prescaler value while the counter is active does not clear the counter. The count continues with the new prescaler value. Reset clears PS to 0000. 0000 Encoding 0. MTIM16 clock source / 1 0001 Encoding 1. MTIM16 clock source / 2 0010 Encoding 2. MTIM16 clock source / 4 0011 Encoding 3. MTIM16 clock source / 8 0100 Encoding 4. MTIM16 clock source / 16 0101 Encoding 5. MTIM16 clock source / 32 0110 Encoding 6. MTIM16 clock source / 64 0111 Encoding 7. MTIM16 clock source / 128 1xxx Encoding 8+. MTIM16 clock source / 256

13.5.3 MTIM16 counter register high (MTIM_CNTH)

This register is the read-only value of the high byte of the current MTIM16 16-bit counter.

When either the CNTH or CNTL register is read, the content of the two registers is latched into a buffer where they remain latched until the other register is read. This allows the coherent 16-bit value to be read in both big-endian and little-endian compile environments and ensures the 16-bit counter is unaffected by the read operation. The coherency mechanism is automatically restarted by an MCU reset or by setting the TRST bit of the SC register (whether BDM mode is active or not).

When BDM is active, the coherency mechanism is frozen such that the buffer latches remain in the state they were in when BDM became active, even if one or both halves of the counter register are read while BDM is active. This assures that if the user was in the middle of reading a 16-bit register when BDM became active, the appropriate value from the other half of the 16-bit value is read after returning to normal execution. The value read from the CNTH and CNTL registers in BDM mode is the value of these registers and not the value of their read buffer.

Address: 8h base + 2h offset = Ah

Bit	7	6	5	4	3	2	1	0
Read	CNTH							
Write								
Reset	0	0	0	0	0	0	0	0

MTIM_CNTH field descriptions

Field	Description
CNTH	MTIM16 count (high byte) These 8 read-only bits contain the current high byte value of the 16-bit counter. Writing has no effect on this register. Reset clears the register to 0x00.

13.5.4 MTIM16 counter register low (MTIM_CNTL)

This register is the read-only value of the low byte of the current MTIM16 16-bit counter.

When either the CNTH or CNTL register is read, the content of the two registers is latched into a buffer where they remain latched until the other register is read. This allows the coherent 16-bit value to be read in both big-endian and little-endian compile environments and ensures the 16-bit counter is unaffected by the read operation. The coherency mechanism is automatically restarted by an MCU reset or by setting the TRST bit of the SC register (whether BDM mode is active or not).

When BDM is active, the coherency mechanism is frozen such that the buffer latches remain in the state they were in when BDM became active, even if one or both halves of the counter register are read while BDM is active. This assures that if the user was in the middle of reading a 16-bit register when BDM became active, the appropriate value from the other half of the 16-bit value is read after returning to normal execution. The value read from the CNTH and CNTL registers in BDM mode is the value of these registers and not the value of their read buffer.

Address: 8h base + 3h offset = Bh

Bit	7	6	5	4	3	2	1	0
Read	CNTL							
Write								
Reset	0	0	0	0	0	0	0	0

MTIM_CNTL field descriptions

Field	Description
CNTL	MTIM16 count (low byte)

MTIM_CNTL field descriptions (continued)

Field	Description
	These 8 read-only bits contain the current low byte value of the 16-bit counter. Writing has no effect on this register. Reset clears the register to 0x00.

13.5.5 MTIM16 modulo register high (MTIM_MODH)

A value of 0x0000 in MODH:MODL puts the MTIM16 in free-running mode. Writing to either MODH or MODL latches the value into a buffer and the latched buffers are updated after the second byte writing. The updated values take effect and reload to the MODH:MODL registers in the next MTIM16 counter cycle, except for the first writing of modulo after a chip reset or in BDM mode. However, after a software reset, the MODH:MODL takes effect at once even if it did not take effect before the reset. On the first writing of MODH:MODL after chip reset, the counter is reset and the modulo takes effect immediately. The latching mechanism may be manually reset by setting the TRST bit of the SC register (whether BDM is active or not).

When BDM is active, the coherency mechanism is frozen so that the buffer latches remain in the state they were in when the BDM became active, even if one or both halves of the modulo register are written while BDM is active. Any writing to the modulo registers bypasses the buffer latches and writes directly to the modulo register while BDM is active, and also the counter is cleared at the same time.

Reading MODH:MODL returns the modulo values that are taking effect whenever in normal run mode or in BDM mode.

Address: 8h base + 4h offset = Ch

Bit	7	6	5	4	3	2	1	0
Read Write	MODH							
Reset	0	0	0	0	0	0	0	0

MTIM_MODH field descriptions

Field	Description
MODH	MTIM16 modulo (high byte) These 8 read/write bits contain the modulo high byte value used to reset the counter and set TOF. Reset sets the register to 0x00.

13.5.6 MTIM16 modulo register low (MTIM_MODL)

A value of 0x0000 in MODH:MODL puts the MTIM16 in free-running mode. Writing to either MODH or MODL latches the value into a buffer and the latched buffers are updated after the second byte writing. The updated values take effect and reload to the MODH:MODL registers in the next MITIM16 counter cycle, except for the first writing of modulo after a chip reset or in BDM mode. However, after a software reset, the MODH:MODL takes effect at once even if it did not take effect before the reset. On the first writing of MODH:MODL after chip reset, the counter is reset and the modulo takes effect immediately. The latching mechanism may be manually reset by setting the TRST bit of the SC register (whether BDM is active or not).

When BDM is active, the coherency mechanism is frozen so that the buffer latches remain in the state they were in when the BDM became active, even if one or both halves of the modulo register are written while BDM is active. Any writing to the modulo registers bypasses the buffer latches and writes directly to the modulo register while BDM is active, and also the counter is cleared at the same time.

Reading MODH:MODL returns the modulo values that are taking effect whenever in normal run mode or in BDM mode.

Address: 8h base + 5h offset = Dh

Bit	7	6	5	4	3	2	1	0
Read Write	MODL							
Reset	0	0	0	0	0	0	0	0

MTIM_MODL field descriptions

Field	Description
MODL	MTIM16 modulo (low byte) These 8 read/write bits contain the modulo low byte value used to reset the counter and set TOF. Reset sets the register to 0x00.

13.6 Functional Description

The MTIM16 is composed of a main 16-bit up-counter with 16-bit modulo register, a clock source selector, and a prescaler block with nine selectable values. The module also contains software selectable interrupt logic.

The MTIM16 counter (CNTH:CNTL registers) has three modes of operation: stopped, free-running, and modulo. The counter is stopped out of reset. If the counter starts without writing a new value to the modulo registers, it will be in free-running mode. The counter is in modulo mode when a value other than 0x0000 is in the modulo registers.

After an MCU reset, the counter stops and resets to 0x0000, and the modulo also resets to 0x0000. The bus clock functions as the default clock source, and the prescale value is divided by 1. To start the MTIM16 in free-running mode, write to the MTIM16 status and control (SC) register and clear the MTIM16 stop (TSTP) bit.

Clock sources are software selectable:

- the internal bus clock
- the fixed frequency clock (XCLK)
- an external clock on the TCLK pin that is selectable as incrementing on either rising or falling edges

The MTIM16 clock select (CLKS) field in the CLK register selects the desired clock source. If the counter is active (the TSTP bit is 0) when a new clock source is selected, the counter continues counting from the previous value using the new clock source.

Nine prescale values are software selectable: clock source divided by 1, 2, 4, 8, 16, 32, 64, 128, or 256. The prescaler select bits (PS[3:0]) in the CLK register select the desired prescale value. If the counter is active (TSTP = 0) when a new prescaler value is selected, the counter continues counting from the previous value using the new prescaler value.

The MTIM16 modulo register (MODH:MODL) allows the overflow compare value to be set to any value from 0x0001 to 0xFFFF. Reset clears the modulo value to 0x0000, which results in a free-running counter.

When the counter is active (the TSTP bit is 0), it increases at the selected rate until the count matches the modulo value. When these values match, the counter overflows to 0x0000 and continues counting. The MTIM16 overflow flag (TOF) is set whenever the counter overflows. The flag sets on the transition from the modulo value to 0x0000.

Clearing TOF is a two-step process. The first step is to read the SC register while TOF is set. The second step is to write a 0 to TOF. If another overflow occurs between the first and second steps, the clearing process is reset and TOF stays set after the second step is performed. This will prevent the second occurrence from being missed. TOF is also cleared when a 1 is written to TRST.

The MTIM16 module allows for an optional interrupt to be generated whenever TOF is set. To enable the MTIM16 overflow interrupt, set the MTIM16 overflow interrupt enable (TOIE) bit in the SC register. The TOIE bit should never be written to be 1 while TOF is 1. Instead, TOF should be cleared first, and then the TOIE bit can be set to 1.

13.6.1 MTIM16 Operation Example

This section shows an example of the MTIM16 module's operation as the counter reaches a matching value from the modulo register.

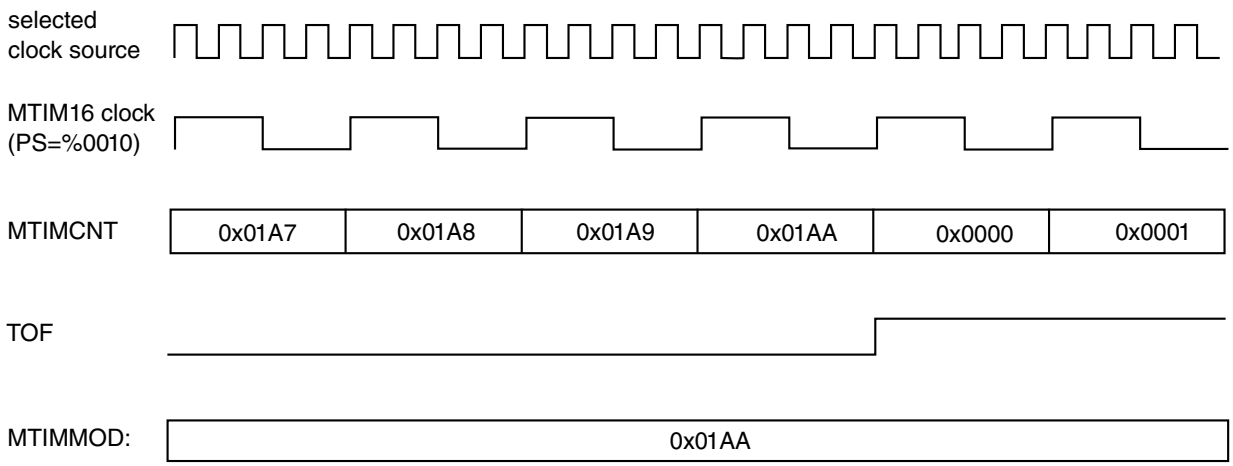


Figure 13-2. MTIM16 Counter Overflow Example

In this figure, the selected clock source could be any of the four possible choices. The prescaler is set to divide-by-4 (the PS field is 0010). The modulo value in the MODH:MODL register is set to 01AAh. When the counter (CNTH:CNTL registers) reaches the modulo value of 01AAh, the counter overflows to 0000h and continues counting. The timer overflow flag, TOF, sets when the counter value changes from 01AAh to 0000h. An MTIM16 overflow interrupt is generated when TOF is set, if the TOIE bit is 1.

Chapter 14
Power Management Controller (PMC)

14.1 Chip specific power management controller

This device includes a power management controller (PMC) which contains on-chip regulators for various operational power modes of run, wait, and stop modes. The low voltage detect (LVD) allows the system to protect against low voltage conditions to protect memory contents and control MCU system states during supply voltage variations. The on-chip voltage regulators supply the 1.8 V and 5 V (V_{DDX}) to internal circuitry and I/O logic. The on-chip voltage reference (VREFH ≈ 4.2V), which is internally connected to ADC reference and 6-bit DAC, provides independent accuracy reference which does not drop over the full operating voltage even when the operating voltage is falling.

NOTE

Over voltage protection(OVP), which allows the voltage over 18 V being detected, is in GDU module.

14.2 Introduction

The power management controller (PMC) module contains four main voltage regulators, which provide internal power to other analog modules and the MCU from an external DC source. The nominal output voltage remains steady over the input voltage range of 5.3 V to 20 V.

PMC also contains the power-on reset (POR), the low voltage detection system (low voltage reset and low voltage warning), a high-accuracy reference voltage output, a 20 kHz low-power oscillator and an internal temperature monitor.

14.3 Features

PMC includes the following features:

- Four integrated voltage regulators: $VREG_{VDDX}$, $VREG_{VDDF}$, $VREG_{VDD}$ and $VREG_{VREFH}$
  - 5 volt ($VREG_{VDDX}$ for analog modules)
  - 2.8 volt ($VREG_{VDDF}$ for flash memory and oscillator)
  - 1.8 volt ($VREG_{VDD}$ for digital logics)
  - 3.7~4.9 volt output with 6-bit trim ($VREG_{VREFH}$ for ADC voltage reference)
  - Output supply decoupling capacitors of 4.7~10 μF for $VREG_{VDDX}$ and 10 μF for $VREG_{VREFH}$ required
  - No output supply decoupling capacitors for $VREG_{VDDF}$ and $VREG_{VDD}$ required
  - Reduced performance mode (RPM), full performance mode (FPM), and wake-up from the RPM state via external input
- Integrated power-on reset (POR)
- Low voltage detection system
  - Integrated low voltage reset (LVR) with reset capability in $VREG_{VDDX}$, $VREG_{VDDF}$ and $VREG_{VDD}$
  - Integrated low voltage warning (LVW) indicator in $VREG_{VDDX}$
  - Programmable LVW indicator for VREFH in $VREG_{VREFH}$
- Buffered high-accuracy reference voltage output
  - Factory programmed trim for high-accuracy reference
- 20 kHz low-power oscillator (LPO) clock source
- Integrated temperature sensor allowing both internal and external monitoring

14.4 Overview

This section presents an overview of the PMC module. The following figure illustrates the simplified PMC block diagram.

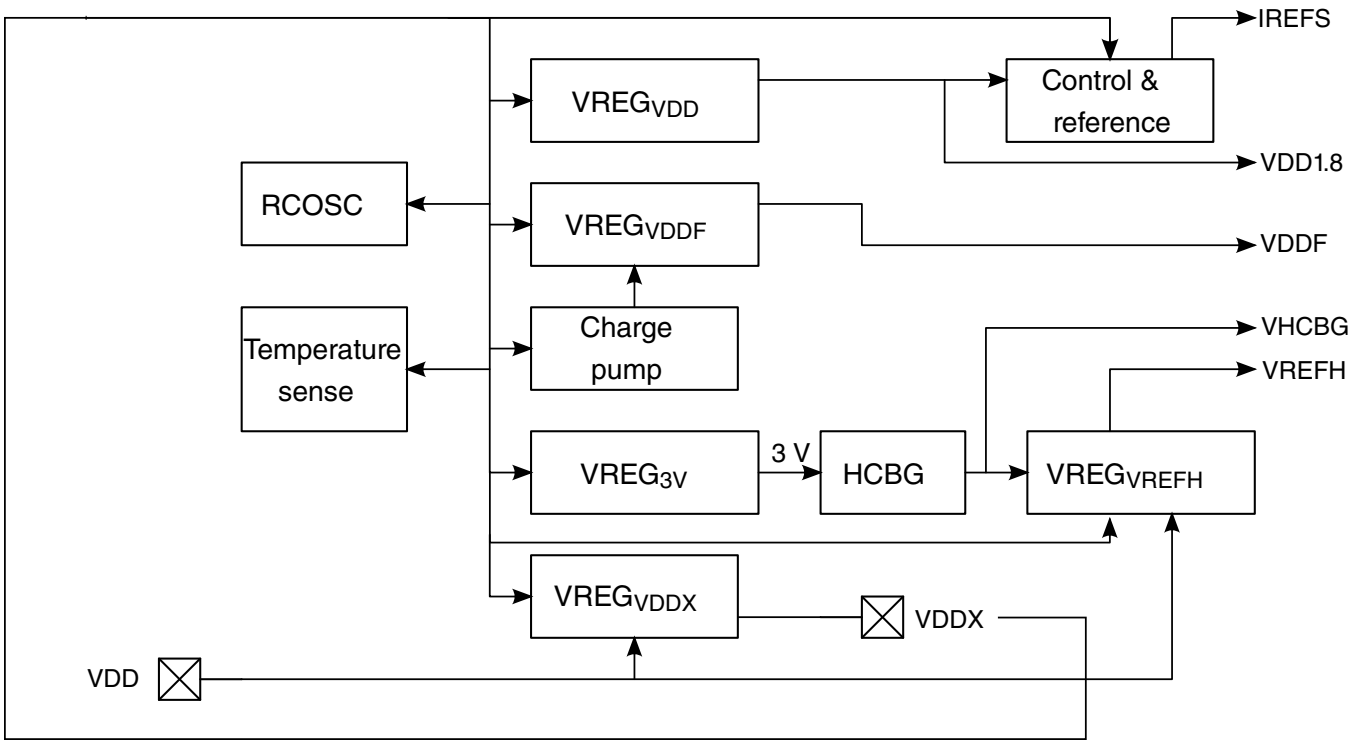


Figure 14-1. Simplified PMC block diagram

14.5 Modes of operation

PMC supports two operating modes: the reduced performance mode (RPM) and the full performance mode (FPM).

14.5.1 Reduced performance mode

When the MCU enters the Stop mode, PMC enters the reduced performance mode (RPM). In this state, the performance of four regulators is reduced but it allows an external logic input to wake up the PMC. When in RPM, the four internal regulators have some limitations as follows:

- $VREG_{VDDX}$ voltage accuracy is not guaranteed to be 5 V and its drive capability is reduced.
- $VREG_{VDDF}$ output voltage drops and its drive capability is very limited.
- $VREG_{VDD}$ output voltage drops and its drive capability is very limited.
- $VREG_{VREFH}$ is off.

When in RPM, the low voltage detection system, the high-accuracy reference voltage and the temperature sensor are off.

14.5.2 Full performance mode

In the full performance mode (FPM) state, the four regulators of PMC perform full regulation of the input supply, to produce the internal regulated supplies in the chip.

14.6 External signal description

PMC has one input signal channel. The following table itemizes all the PMC external pins.

Signal	I/O	Detailed description	Connect from/to
VDD	I	PMC voltage supply input	From pin VDD
VDDX	O	5 V power voltage output	To pin VDD1
VREFH	O	Accurate voltage reference for ADC	To pin VREFH
VDDF	O	2.8 V flash supply voltage	To flash module
VDD1.8	O	1.8 V core supply voltage	To MCU digital logics

14.6.1 VDD

VDD is the PMC integrated regulator power supply voltage pin. This is the voltage supply input from which the voltage regulator generates the on-chip voltage supplies. An external decoupling capacitor is required on this pin.

14.6.2 VDDX

VDDX is the 5 V voltage regulator ($VREG_{VDDX}$) output to supply power for the digital I/O drivers.

14.6.3 VREFH

VREFH is the accurate voltage reference ($VREG_{VREFH}$) output. It can be configured from 3.7 V to 4.9 V. An external decoupling capacitor is required on this pin.

14.6.4 VDDF

VDDF is the 2.8 V regulator ($VREG_{VDDF}$) output. It is the power supply of the on-chip NVM block. VDDF is not pinned out because no external decoupling capacitor is required.

14.6.5 VDD1.8

VDD1.8 is the 1.8 V regulator ($VREG_{VDD}$) output. It is the power supply of the on-chip digital logics including CPU and RAM. VDD1.8 is not pinned out because no external decoupling capacitor is required.

14.7 Memory map and register definition

This section includes the module memory map and detailed descriptions of all registers.

PMC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
1850	Control Register (PMC_CTRL)	8	R/W	See section	14.7.1/222
1851	Reset Flags Register (PMC_RST)	8	R/W	See section	14.7.2/223
1852	Temperature Control and Status Register (PMC_TPCTRLSTAT)	8	R/W	00h	14.7.3/223
1853	Temperature Offset Step Trim Register (PMC_TPTM)	8	R/W	See section	14.7.4/224
1854	RC Oscillator Offset Step Trim Register (PMC_RC20KTRM)	8	R/W	1Fh	14.7.5/225
1855	Low Voltage Control and Status Register 1 (system 5 V) (PMC_LVCTLSTAT1)	8	R/W	See section	14.7.6/226
1856	Low Voltage Control and Status Register 2 (VREFH) (PMC_LVCTLSTAT2)	8	R/W	See section	14.7.7/227
1857	VREFH Configuration Register (PMC_VREFHCFG)	8	R/W	See section	14.7.8/227
1858	VREFH Low Voltage Warning (LVW) Configuration Register (PMC_VREFHLVW)	8	R/W	02h	14.7.9/228
1859	Status Register (PMC_STAT)	8	R/W	04h	14.7.10/228

14.7.1 Control Register (PMC_CTRL)

Address: 1850h base + 0h offset = 1850h

Bit	7	6	5	4	3	2	1	0
Read	GWREN	0				VREFDN	RC20KENSTP	0
Write								
Reset	0	0	0	0	0	u*	u*	0
POR	0	0	0	0	0	1	1	0

* Notes:
- u = Unaffected by reset.

PMC_CTRL field descriptions

Field	Description
7 GWREN	General Write protection Enable General write protection enable by sharing with other trim write enable. This bit is the general write enable control for the following registers: • Temperature Offset Step Trim Register (PMC_TPTM) • RC Oscillator Offset Step Trim Register (PMC_RC20KTRM) • VREFH Configuration Register (PMC_VREFHCFG) 0 The general write protection is enabled. 1 The general write protection is disabled. Therefore, the related registers are writable.
6–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 VREFDN	VREFH Down This bit controls whether to disable the VREFH regulator. 0 Enables the VREFH regulator. 1 Disables the VREFH regulator.
1 RC20KENSTP	20 kHz RC oscillator Enable in Stop mode The default reset value is 1. In the Run mode this bit is always ON, and in the Stop mode it selects whether to enable the 20 kHz RC oscillator. 0 Disables 20 kHz RC oscillator in the Stop mode. 1 Enables 20 kHz RC oscillator in the Stop mode.
0 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

14.7.2 Reset Flags Register (PMC_RST)

Address: 1850h base + 1h offset = 1851h

Bit	7	6	5	4	3	2	1	0
Read	0	PORF	LVRF	0				
Write								
Reset	0	u*	u*	0	0	0	0	0
POR	0	1	1	0	0	0	0	0
LVR	0	1	1	0	0	0	0	0

* Notes:
- u = Unaffected by reset.

PMC_RST field descriptions

Field	Description
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6 PORF	Power-on Reset Flag This bit is set to 1 when a power-on reset (POR) occurs. This flag can only be cleared by writing a 1. Writing a 0 takes no effect. 0 POR does not occur. 1 POR occurs.
5 LVRF	Low Voltage Reset (LVR) Flag This bit is set to 1 when a low voltage reset occurs in the full performance mode (FPM). This flag can only be cleared by writing a 1. Writing a 0 takes no effect. 0 Low voltage reset does not occur. 1 Low voltage reset occurs.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

14.7.3 Temperature Control and Status Register (PMC_TPCTRLSTAT)

Address: 1850h base + 2h offset = 1852h

Bit	7	6	5	4	3	2	1	0
Read	0			SWON	TEMPEN	HTDS	HTIE	HTIF
Write								
Reset	0	0	0	0	0	0	0	0

PMC_TPCTRLSTAT field descriptions

Field	Description
7–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
4 SWON	Switch On This bit selects bandgap or temperature sensor output to the ADC channel, when the bit TEMPEN is enabled. 0 Selects the temperature sensor output. 1 Selects the bandgap output.
3 TEMPEN	Temperature sensor Enable This bit is the enable control to the temperature sensor. 0 Disables the temperature sensor. 1 Enables the temperature sensor.
2 HTDS	High Temperature Detection Status This read-only bit indicates the temperature status. Writing a value takes no effect. 0 Junction temperature is below the alert level. 1 Junction temperature is above the alert level.
1 HTIE	High Temperature Interrupt Enable 0 Disables the high temperature interrupt. 1 Enables the high temperature interrupt.
0 HTIF	High Temperature Interrupt Flag This bit is set to 1 when the HTDS status bit changes. This flag can only be cleared by writing a 1. Writing a 0 takes no effect. If enabled (HTIE = 1), HTIF causes an interrupt request. 0 No change in the HTDS bit. 1 The HTDS bit changes.

14.7.4 Temperature Offset Step Trim Register (PMC_TPTM)

Address: 1850h base + 3h offset = 1853h

Bit	7	6	5	4	3	2	1	0
Read	TRMTPEN	0			TOT[3:0]			
Write								
Reset	0	0	0	0	f	f	f	f

PMC_TPTM field descriptions

Field	Description
7 TRMTPEN	Temperature offset Trim Enable If the bit is set, the temperature sensor offset is enabled.

Table continues on the next page...

PMC_TPTM field descriptions (continued)

Field	Description
	0 The temperature sensor offset is disabled. TOT[3:0] takes no effect. 1 The temperature sense offset is enabled. TOT[3:0] selects the temperature offset value. See TOT[3:0] for more details.
6–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
TOT[3:0]	Temperature Offset step Trim These bits are used to trim the temperature offset. TOT[3:0] determines the trim value for assert or de-assert of system interrupt. Refer to the electrical specifications information in the chip datasheet to get the detailed values. **NOTE:** After de-assert of system reset, a trim value is automatically loaded from the flash memory. Normal IPS writable only after PMC_CTRL[GWREN] is set. **NOTE:** TOT[3:0] can be cleared only by POR.

14.7.5 RC Oscillator Offset Step Trim Register (PMC_RC20KTRM)

Address: 1850h base + 4h offset = 1854h

Bit	7	6	5	4	3	2	1	0
Read	0				OSCOT[5:0]			
Write								
Reset	0	0	0	1	1	1	1	1

PMC_RC20KTRM field descriptions

Field	Description
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
OSCOT[5:0]	RC Oscillator Offset step Trim These bits are defined as the step trim values for the RC oscillator offset. Refer to the electrical specifications information in the chip datasheet to get the detailed values. **NOTE:** After de-assert of system reset, a trim value is automatically loaded from the flash memory. Normal IPS writable only after PMC_CTRL[GWREN] is set.

14.7.6 Low Voltage Control and Status Register 1 (system 5 V) (PMC_LVCTLSTAT1)

Address: 1850h base + 5h offset = 1855h

Bit	7	6	5	4	3	2	1	0
Read	0			SLVWF		SLVWIE	SLVWSEL	0
Write					SLVWACK			
Reset	0	0	0	0	0	0	0	0
POR	0	0	0	1	0	0	0	0

PMC_LVCTLSTAT1 field descriptions

Field	Description
7–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
4 SLVWF	Low Voltage Warning (LVW) Flag for VDDX This read-only status bit indicates a low voltage warning event. SLVWF is set when V_{Supply} transitions below the trip point, or after reset and V_{Supply} is already below V_{LVW}. SLVWF is set to 1 after power-on reset. Therefore, to use the LVW interrupt function, before enabling SLVWIE, SLVWF must be cleared by writing SLVWACK first. 0 Low voltage warning event is not detected. 1 Low voltage warning event is detected.
3 SLVWACK	Low Voltage Warning Acknowledge This write-only bit is used to acknowledge low voltage warning errors. Write 1 to clear SLVWF. Reading always returns 0.
2 SLVWIE	Low Voltage Warning Interrupt Enable Enables hardware interrupt requests for SLVWF. 0 Hardware interrupt disabled (use polling). 1 Request a hardware interrupt when SLVWF = 1.
1 SLVWSEL	Low Voltage Warning Selection 0 4.2 V LVW threshold selected. 1 3.7 V LVW threshold selected.
0 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

14.7.7 Low Voltage Control and Status Register 2 (V_{REFH}) (PMC_LVCTLSTAT2)

Address: 1850h base + 6h offset = 1856h

Bit	7	6	5	4	3	2	1	0
Read	0			RLVWF		RLVWIE	0	
Write					RLVWACK			
Reset	0	0	0	0	0	0	0	0
POR	0	0	0	1	0	0	0	0

PMC_LVCTLSTAT2 field descriptions

Field	Description
7–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
4 RLVWF	Low Voltage Warning Flag for VREFH This read-only status bit indicates a low voltage warning event. RLVWF is set when V_{Supply} transitions below the trip point, or after reset and V_{Supply} is already below V_{LVW}. RLVWF is set to 1 after power-on reset. Therefore, to use LVW interrupt function, before enabling RLVWIE, RLVWF must be cleared by writing RLVWACK first. 0 Low voltage warning event is not detected. 1 Low voltage warning event is detected.
3 RLVWACK	Low Voltage Warning Acknowledge This write-only bit is used to acknowledge low voltage warning errors. Write 1 to clear RLVWF. Reading always returns 0.
2 RLVWIE	Low Voltage Warning Interrupt Enable Enables hardware interrupt requests for RLVWF. 0 Hardware interrupt disabled (use polling). 1 Request a hardware interrupt when RLVWF = 1.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

14.7.8 V_{REFH} Configuration Register (PMC_VREFHCFG)

Address: 1850h base + 7h offset = 1857h

Bit	7	6	5	4	3	2	1	0
Read	0		T5V					
Write								
Reset	0	0	f	f	f	f	f	f

PMC_VREFHCFG field descriptions

Field	Description
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
T5V	Trim 5 V reference voltage for ADC Auto-loaded from IFR. After de-assert of system reset, a trim value is automatically loaded from the flash memory. The default is 3.8 V. **NOTE:** Normal IPS writable only after PMC_CTRL[GWREN] is set.

14.7.9 VREFH Low Voltage Warning (LVW) Configuration Register (PMC_VREFHLVW)

Address: 1850h base + 8h offset = 1858h

Bit	7	6	5	4	3	2	1	0
Read	0						LVWCFG[1:0]	
Write								
Reset	0	0	0	0	0	0	1	0

PMC_VREFHLVW field descriptions

Field	Description
7–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
LVWCFG[1:0]	VREFH LVW reference voltage Configuration These bits are used to configure the VREFH low voltage warning threshold value. 00 3.6 V LVW threshold. 01 3.7 V LVW threshold. 10 4.1 V LVW threshold. 11 4.4 V LVW threshold.

14.7.10 Status Register (PMC_STAT)

Address: 1850h base + 9h offset = 1859h

Bit	7	6	5	4	3	2	1	0
Read	0					HBGRDY	0	VREFRDY
Write								
Reset	0	0	0	0	0	1	0	0

PMC_STAT field descriptions

Field	Description
7–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 HBGRDY	High-accuracy Bandgap Ready flag 0 Below 0.8 V, this bit is cleared. 1 Above 1 V, this bit is asserted.
1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 VREFRDY	VREFH Ready flag **NOTE:** Even after VREFRDY is set, it does not necessarily mean that VREF is okay for use instantly. It needs to wait about 3 ms, in the chip power-on or stop mode, to wake up before using the VREF for ADC. 0 Below 3.0 V, this bit is cleared. 1 Above 3.45 V, this bit is asserted.

14.8 Functional description

The following sections describe functional details of the module.

14.8.1 Voltage regulators

PMC integrates four voltage regulators: $VREG_{VDDX}$, $VREG_{VDDF}$, $VREG_{VDD}$ and $VREG_{VREFH}$. These four voltage regulators are the power supply for all on-chip modules.

14.8.1.1 $VREG_{VDDX}$

$VREG_{VDDX}$ is a 5 V output regulator. It is the power supply for the digital I/Os domain (VDDX) and the analog modules domain (VDDA).

14.8.1.2 $VREG_{VDDF}$

$VREG_{VDDF}$ is a 2.8 V output regulator. It is the power supply for the on-chip NVM module (VDDF).

14.8.1.3 $VREG_{VDD}$

$VREG_{VDD}$ is a 1.8 V output regulator. It is the power supply for the digital logics such as CPU and RAM.

14.8.1.4 $VREG_{VREFH}$

$VREG_{VREFH}$ is a 3.7~4.9 V configurable output regulator. It is the reference voltage output for on-chip analog modules such as ADC.

This regulator can be enabled or disabled by configuring PMC_CTRL[VREFDN]. After POR or enabled, the flag PMC_STAT[VREFRDY] is set, once the regulator output is ready.

The regulator output voltage can be trimmed from 3.7 V to 4.9 V through the PMC_VREFHCFG[T5V]. After reset, a factory trimmed value is automatically loaded to the PMC_VREFHCFG register so that $VREG_{VREFH}$ has a default output voltage. After the write protection enable bit PMC_CTRL[GWREN] is set to 1, user can configure this voltage to other levels according to the application need.

14.8.2 Power-on reset

The POR circuit monitors the $VREG_{VDD}$ outputs (VDD1.8), which represent the MCU VDD domain. When power is initially applied to the MCU, or when the supply voltage drops below the V_{POR} level, the POR circuit causes a reset condition.

When POR reset occurs, the flag bit PMC_RST[PORF] is set to 1. This flag is not affected by other system resets.

14.8.3 Low voltage reset (LVR)

The LVR circuit monitors the 5 V $VREG_{VDDX}$ output (VDDX), the 2.8 V $VREG_{VDDF}$ output (VDDF), and the 1.8 V $VREG_{VDD}$ output (VDD1.8). The LVR circuit can generate a reset when detecting a low voltage condition on VDDX, VDDF or VDD1.8. After an LVR reset occurs, the LVR system holds the MCU in reset status until the supply voltage rises above the low voltage level.

In the FPM mode, when LVR reset occurs, the flag bit PMC_RST[LVRF] is set to 1. The flag LVRF is also set to 1 when a POR occurs, but is not affected by other system resets.

14.8.3.1 LVR in low power mode

The LVR circuit is disabled when PMC enters the reduced performance mode (RPM).

14.8.4 Low voltage warning (LVW)

The LVW circuit monitors the 5 V $VREG_{VDDX}$ outputs (VDDX/VDDA) and the $VREG_{VREFH}$ output (VREFH). The LVW circuit can generate an interrupt when detecting a low voltage condition on VDDX/VDDA or VREFH.

14.8.4.1 LVW on VDDX/VDDA

Low voltage on VDDX/VDDA sets the flag bit PMC_LVCTLSTAT1[SLVWF] to 1. When PMC_LVCTLSTAT1[SLVWIE] is set, the low voltage condition incurs an interrupt.

To clear the SLVWF flag, user should write 1 to PMC_LVCTLSTAT1[SLVWACK].

14.8.4.2 LVW on VREFH

Low voltage on VREFH sets the flag PMC_LVCTLSTAT2[RLVWF] to 1. When PMC_LVCTLSTAT2[RLVWIE] is set, the low voltage condition incurs an interrupt.

To clear the RLVWF flag, user should write 1 to PMC_LVCTLSTAT2[RLVWACK].

The VREFH low voltage level can be configured through the LVWCFG[1:0] bits in PMC_VREFHLVW register.

14.8.4.3 LVW in low power mode

The LVW circuit is disabled when PMC enters the reduced performance mode (RPM).

14.8.5 High-accuracy reference voltage

PMC includes a high-accuracy reference voltage. This voltage is available for other on-chip modules.

After POR, the flag PMC_STAT[HBGRDY] is set to 1 when the high-accuracy output is ready.

The high-accurary reference voltage is always enabled except in RPM mode.

14.8.6 Temperature sensor

PMC includes an internal temperature sensing voltage generator and a high temperature warning comparator. The generator is always enabled in FPM mode and can provide the sensing voltage to ADC to measure the temperature.

To enable the PMC temperature sensor, PMC_TPCTRLSTAT[TEMPEN] should be set to 1.

After enabled, the temperature sensor voltage output can be provided to external modules such as ADC. Depending on the PMC_TPCTRLSTAT[SWON] bit value, the voltage output is selected as follows:

- If SWON = 0, the temperature sensor voltage is as the output.
- If SWON = 1, the bandgap is as the output. Refer to the chip datasheet for the detailed voltage value.

The temperature sensor is disabled in RPM mode.

14.8.6.1 High temperature warning

The temperature sensor integrates a high temperature warning circuit with configurable temperature threshold.

The flag PMC_TPCTRLSTAT[HTDS] shows the temperature status:

- HTDS bit is set when the temperature rises above the HTDS assert threshold.
- HTDS bit is cleared when the temperature falls below the HTDS de-assert threshold.

If PMC_TPCTRLSTAT[HTIE] is set to 1, PMC sets the HTIF flag in the same register and generates an interrupt when the HTDS status changes. Writing 1 to HTIF can clear this HTIF flag.

When PMC_TPTM[TRMTPEN] is set, the HTDS flag assert/de-assert threshold is defined by the TOT[3:0] bits in the same register. For the detailed HTDS flag assert/de-assert thresholds, see the reference in TOT[3:0].

Before configuring the TOT[3:0] bits, user should set PMC_CTRL[GWREN] to unlock the write protection.

14.8.7 Low-power RC oscillator

PMC integrates a low-power RC oscillator (LPO) which provides a typical 20 kHz output. This LPO can serve as an independent clock source for the MCU on-chip modules such as watchdog.

This RC oscillator is set to ON by default in the FPM mode, and can be controlled to OFF in the RPM mode by configuring PMC_CTRL[RC20KENSTP].

The RC oscillator out frequency can be configured by user through the OSCOT[5:0] bits in PMC_RC20KTRM register. Before writing to OSCOT[5:0], user should set PMC_CTRL[GWREN].

14.9 Application information

1. VREFH readiness

   VREFH is a high-accuracy voltage reference. It needs 3 ms to be stable after PMC_STAT[VREFRDY] is asserted. When entering the Stop mode, VREFH is disabled automatically. So after exiting from the Stop mode, it requires to wait enough settling time. After the PMC_VREFHCFG setting is changed, it takes some time to get it settled. ADC or other functions using VREFH cannot work correctly during this transition period.

2. 20 kHz LPO calibration

   LPO has to be calibrated after the PMC powers up, in order to get the ±5% precision. The LPO clock is connected to SBAR (in the SIM module), and the calibration can be achieved by using FTM1 with on-chip clock. Refer to the SIM chapter for more detailed setting information.

3. Special write enable register handling

PMC_TPTM, PMC_RC20KTRM and PMC_VREFHCFG are protected by a special write enable register handling. They cannot be written unless PMC_CTL[GWREN] is 1.

Chapter 15
Keyboard Interrupts (KBI)

15.1 Chip specific KBI information

This device has one KBI modules with up to 8 keyboard interrupt inputs grouped in a KBI modules available on PTA port.

15.2 Introduction

15.2.1 Features

The KBI features include:

- Up to eight keyboard interrupt pins with individual pin enable bits
- Each keyboard interrupt pin is programmable as:
  - falling-edge sensitivity only
  - rising-edge sensitivity only
  - both falling-edge and low-level sensitivity
  - both rising-edge and high-level sensitivity
- One software-enabled keyboard interrupt
- Exit from low-power modes

15.2.2 Modes of Operation

This section defines the KBI operation in:

- Wait mode
- Stop mode
- Background debug mode

15.2.2.1 KBI in Wait mode

Executing the Wait instruction places the MCU into Wait mode. The KBI interrupt should be enabled (KBI_SC[KBIE] = 1), if desired, before executing the Wait instruction, allowing the KBI to continue to operate while the MCU is in Wait mode. An enabled KBI pin (KBI_PE[KBIPEn] = 1) can be used to bring the MCU out of Wait mode if the KBI interrupt is enabled (KBI_SC[KBIE] = 1).

15.2.2.2 KBI in Stop modes

Executing the Stop instruction places the MCU into Stop mode (when Stop is selected), where the KBI can operate asynchronously. If this is the desired behavior, the KBI interrupt must be enabled (KBI_SC[KBIE] = 1) before executing the Stop instruction, allowing the KBI to continue to operate while the MCU is in Stop mode. An enabled KBI pin (KBI_PE[KBIPEn] = 1) can be used to bring the MCU out of Stop mode if the KBI interrupt is enabled (KBI_SC[KBIE] = 1).

15.2.3 Block Diagram

The block diagram for the keyboard interrupt module is shown below..

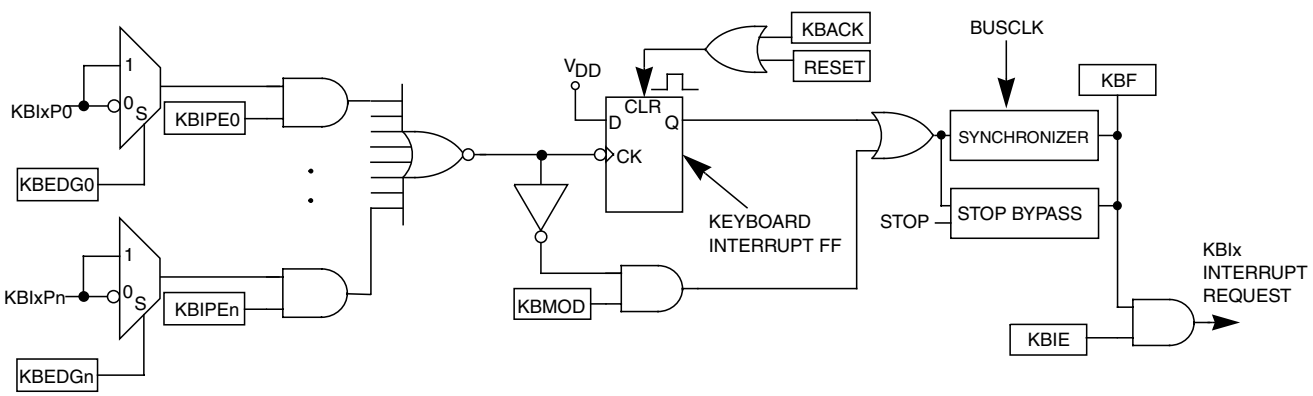


Figure 15-1. KBI block diagram

15.3 External signals description

The KBI input pins can be used to detect either falling edges, or both falling edge and low-level interrupt requests. The KBI input pins can also be used to detect either rising edges, or both rising edge and high-level interrupt requests.

The signal properties of KBI are shown in the following table:

Table 15-1. External signals description

Signal	Function	I/O
KBIPn	Keyboard interrupt pins	I

15.4 Register definition

The KBI includes following registers:

- A pin status and control register, KBI_SC
- A pin enable register, KBI_PE
- An edge select register, KBI_ES

See the direct-page register summary in the Memory chapter for the absolute address assignments for all KBI registers. This section refers to registers and control bits only by their names.

Some MCUs may have more than one KBI, so register names include placeholder characters to identify which KBI is being referenced.

15.5 Memory Map and Registers

KBI memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
7C	KBI Status and Control Register (KBI_SC)	8	R/W	00h	15.5.1/238
7D	KBI Pin Enable Register (KBI_PE)	8	R/W	00h	15.5.2/239
7E	KBI Edge Select Register (KBI_ES)	8	R/W	00h	15.5.3/239

15.5.1 KBI Status and Control Register (KBI_SC)

KBI_SC contains the status flag and control bits, which are used to configure the KBI.

Address: 7Ch base + 0h offset = 7Ch

Bit	7	6	5	4	3	2	1	0
Read	0				KBF		KBIE	KBMOD
Write						KBACK		
Reset	0	0	0	0	0	0	0	0

KBI_SC field descriptions

Field	Description
7–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 KBF	KBI Interrupt Flag KBF indicates when a KBI interrupt request is detected. Writes have no effect on KBF. 0 KBI interrupt request not detected. 1 KBI interrupt request detected.
2 KBACK	KBI Acknowledge Writing a 1 to KBACK is part of the flag clearing mechanism.
1 KBIE	KBI Interrupt Enable KBIE determines whether a KBI interrupt is enabled or not. 0 KBI interrupt not enabled. 1 KBI interrupt enabled.
0 KBMOD	KBI Detection Mode KBMOD (along with the KBEDG bits) controls the detection mode of the KBI interrupt pins. 0 Keyboard detects edges only. 1 Keyboard detects both edges and levels.

15.5.2 KBI Pin Enable Register (KBI_PE)

KBI_PE contains the pin enable control bits.

Address: 7Ch base + 1h offset = 7Dh

Bit	7	6	5	4	3	2	1	0
Read Write	KBIPE							
Reset	0	0	0	0	0	0	0	0

KBI_PE field descriptions

Field	Description
KBIPE	KBI Pin Enables Each of the KBIPEn bits enable the corresponding KBI interrupt pin. 0 Pin is not enabled as KBI interrupt. 1 Pin is enabled as KBI interrupt.

15.5.3 KBI Edge Select Register (KBI_ES)

KBI_ES contains the edge select control bits.

Address: 7Ch base + 2h offset = 7Eh

Bit	7	6	5	4	3	2	1	0
Read Write	KBEDG							
Reset	0	0	0	0	0	0	0	0

KBI_ES field descriptions

Field	Description
KBEDG	KBI Edge Selects Each of the KBEDGn bits selects the falling edge/low-level or rising edge/high-level function of the corresponding pin. 0 Falling edge/low level. 1 Rising edge/high level.

15.6 Functional Description

This on-chip peripheral module is called a keyboard interrupt module because originally it was designed to simplify the connection and use of row-column matrices of keyboard switches. However, these inputs are also useful as extra external interrupt inputs and as an external means of waking the MCU from stop or wait low-power modes.

The KBI module allows up to eight pins to act as additional interrupt sources. Writing to the KBI_PE[KBIPEn] bits independently enables or disables each KBI pin. Each KBI pin can be configured as edge sensitive or edge and level sensitive based on the KBI_SC[KBMOD] bit. Edge sensitive can be software programmed to be either falling or rising; the level can be either low or high. The polarity of the edge or edge and level sensitivity is selected using the KBI_ES[KBEDGn] bits.

15.6.1 Edge-only sensitivity

Synchronous logic is used to detect edges. A falling edge is detected when an enabled keyboard interrupt (KBI_PE[KBIPEn]=1) input signal is seen as a logic 1 (the deasserted level) during one bus cycle and then a logic 0 (the asserted level) during the next cycle. A rising edge is detected when the input signal is seen as a logic 0 (the deasserted level) during one bus cycle and then a logic 1 (the asserted level) during the next cycle.

Before the first edge is detected, all enabled keyboard interrupt input signals must be at the deasserted logic levels. After any edge is detected, all enabled keyboard interrupt input signals must return to the deasserted level before any new edge can be detected.

A valid edge on an enabled KBI pin will set KBI_SC[KBF]. If KBI_SC[KBIE] is set, an interrupt request will be presented to the MPU. Clearing of KBI_SC[KBF] is accomplished by writing a 1 to KBI_SC[KBACK].

15.6.2 Edge and level sensitivity

A valid edge or level on an enabled KBI pin will set KBI_SC[KBF]. If KBI_SC[KBIE] is set, an interrupt request will be presented to the MCU. Clearing of KBI_SC[KBF] is accomplished by writing a 1 to KBI_SC[KBACK], provided all enabled keyboard inputs are at their deasserted levels. KBI_SC[KBF] will remain set if any enabled KBI pin is asserted while attempting to clear KBI_SC[KBF] by writing a 1 to KBI_SC[KBACK].

15.6.3 KBI Pullup Resistor

Each KBI pin, if enabled by KBI_PE, can be configured via the associated I/O port pull enable register, see chapter, to use:

- an internal pullup resistor, or
- no resistor

If an internal pullup resistor is enabled for an enabled KBI pin, the associated I/O port pull select register (see I/O Port chapter) can be used to select an internal pullup resistor.

15.6.4 KBI initialization

When a keyboard interrupt pin is first enabled, it is possible to get a false keyboard interrupt flag. To prevent a false interrupt request during keyboard initialization, the user should do the following:

1. Mask keyboard interrupts by clearing KBI_SC[KBIE].
2. Enable the KBI polarity by setting the appropriate KBI_ES[KBEDGn] bits.
3. Before using internal pullup resistors, configure the associated bits in PORT_.
4. Enable the KBI pins by setting the appropriate KBI_PE[KBIPEn] bits.
5. Write to KBI_SC[KBACK] to clear any false interrupts.
6. Set KBI_SC[KBIE] to enable interrupts.

Chapter 16
Cyclic redundancy check (CRC)

16.1 Chip specific cyclic redundancy check (CRC)

The CRC generator module uses a programmable polynomial to generate CRC code for error detection. The CRC can be configured to work as a standard CRC. It provides the user with programmable polynomial, SEED and other parameters required to implement a 16-bit or 32-bit CRC standard. The 16-bit code is calculated for 8-bit of data at a time, and provides a simple check for all accessible memory locations in flash and RAM.

Features:

- Hardware 16/32-bit CRC generator
- Programmable initial seed value
- Programmable 16/32-bit polynomial
- Optional feature to reverse input and output data by bit
- Optional final complement output of result
- High-speed CRC calculation

Customization:

- Primary clock: BUSCLK (20Mhz)

Module Instances:

- One

16.2 Introduction

NOTE

For the chip-specific implementation details of this module's instances, see the chip configuration information.

The cyclic redundancy check (CRC) module generates 16/32-bit CRC code for error detection.

The CRC module provides a programmable polynomial and other parameters required to implement a 16-bit or 32-bit CRC standard.

The 16/32-bit code is calculated for 8 bits of data at a time.

16.2.1 Features

Features of the CRC module include:

- Hardware CRC generator circuit using a 16-bit or 32-bit programmable shift register
- Programmable initial seed value and polynomial
- Option to transpose input data or output data (the CRC result) bitwise or bytewise. This option is required for certain CRC standards. A bytewise transpose operation is not possible when accessing the CRC data register via 8-bit accesses. In this case, the user's software must perform the bytewise transpose function.
- Option for inversion of final CRC result
- 8-bit CPU register programming interface

16.2.2 Block diagram

The following is a block diagram of the CRC.

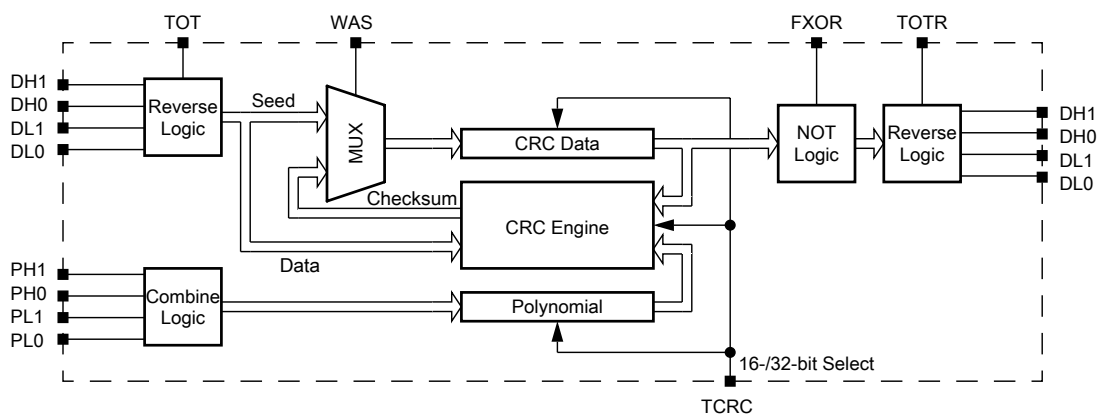


Figure 16-1. Programmable cyclic redundancy check (CRC) block diagram

16.2.3 Modes of operation

Various MCU modes affect the CRC module's functionality.

16.2.3.1 Run mode

This is the basic mode of operation.

16.2.3.2 Low-power modes (Wait or Stop)

Any CRC calculation in progress stops when the MCU enters a low-power mode that disables the module clock. It resumes after the clock is enabled or via the system reset for exiting the low-power mode. Clock gating for this module is dependent on the MCU.

16.3 Memory map and register descriptions

CRC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
1890	CRC Data register: High 1 (CRC_DH1)	8	R/W	FFh	16.3.1/245
1891	CRC Data register: High 0 (CRC_DH0)	8	R/W	FFh	16.3.2/246
1892	CRC Data register: Low 1 (CRC_DL1)	8	R/W	FFh	16.3.3/247
1893	CRC Data register: Low 0 (CRC_DL0)	8	R/W	FFh	16.3.4/247
1894	CRC Polynomial Register: High 1 (CRC_PH1)	8	R/W	00h	16.3.5/248
1895	CRC Polynomial Register: High 0 (CRC_PH0)	8	R/W	00h	16.3.6/249
1896	CRC Polynomial Register: Low 1 (CRC_PL1)	8	R/W	10h	16.3.7/249
1897	CRC Polynomial Register: Low 0 (CRC_PL0)	8	R/W	21h	16.3.8/250
1898	CRC Control register (CRC_CTRL)	8	R/W	00h	16.3.9/250

16.3.1 CRC Data register: High 1 (CRC_DH1)

This section describes the function of CRC data registers (DH1:DH0:DL1:DL0). The set of CRC data registers contains the value of seed, data, and checksum. When CTRL[WAS] is set, any write to the data registers will be regarded as seed for CRC module. When CTRL[WAS] is de-asserted, any write to the data registers will be regarded as data for CRC module for general CRC computation.

When programming the seed value in 16-bit CRC mode, the DH1:DH0 are not used and reads to these registers returns an indeterminate value. For 32-bit CRC, all registers are used.

When programming data values for CRC calculation, data must be provided in DL0 register only. Writes to other bytes of data regsiters are ignored.

When final data written, the CRC result can be read from this data register. In 16-bit CRC mode, the CRC result is avaiable in DL1:DL0. In 32-bit CRC mode, all registers are used. Reads to this register at any time returns the intermediate CRC value, provided CRC module is configured.

Address: 1890h base + 0h offset = 1890h

Bit	7	6	5	4	3	2	1	0
Read Write	DH1							
Reset	1	1	1	1	1	1	1	1

CRC_DH1 field descriptions

Field	Description
DH1	CRC Data Bits 31:24

16.3.2 CRC Data register: High 0 (CRC_DH0)

This section describes the function of CRC data registers (DH1:DH0:DL1:DL0). The set of CRC data registers contains the value of seed, data, and checksum. When CTRL[WAS] is set, any write to the data registers will be regarded as seed for CRC module. When CTRL[WAS] is de-asserted, any write to the data registers will be regarded as data for CRC module for general CRC computation.

When programming the seed value in 16-bit CRC mode, the DH1:DH0 are not used and reads to these registers returns an indeterminate value. For 32-bit CRC, all registers are used.

When programming data values for CRC calculation, data must be provided in DL0 register only. Writes to other bytes of data regsiters are ignored.

When final data written, the CRC result can be read from this data register. In 16-bit CRC mode, the CRC result is avaiable in DL1:DL0. In 32-bit CRC mode, all registers are used. Reads to this register at any time returns the intermediate CRC value, provided CRC module is configured.

Address: 1890h base + 1h offset = 1891h

Bit	7	6	5	4	3	2	1	0
Read Write	DH0							
Reset	1	1	1	1	1	1	1	1

CRC_DH0 field descriptions

Field	Description
DH0	CRC Data Bits 23:16

16.3.3 CRC Data register: Low 1 (CRC_DL1)

This section describes the function of CRC data registers (DH1:DH0:DL1:DL0). The set of CRC data registers contains the value of seed, data, and checksum. When CTRL[WAS] is set, any write to the data registers will be regarded as seed for CRC module. When CTRL[WAS] is de-asserted, any write to the data registers will be regarded as data for CRC module for general CRC computation.

When programming the seed value in 16-bit CRC mode, the DH1:DH0 are not used and reads to these registers returns an indeterminate value. For 32-bit CRC, all registers are used.

When programming data values for CRC calculation, data must be provided in DL0 register only. Writes to other bytes of data regsiters are ignored.

When final data written, the CRC result can be read from this data register. In 16-bit CRC mode, the CRC result is avaiable in DL1:DL0. In 32-bit CRC mode, all registers are used. Reads to this register at any time returns the intermediate CRC value, provided CRC module is configured.

Address: 1890h base + 2h offset = 1892h

Bit	7	6	5	4	3	2	1	0
Read Write	DL1							
Reset	1	1	1	1	1	1	1	1

CRC_DL1 field descriptions

Field	Description
DL1	CRC Data Bits 15:8

16.3.4 CRC Data register: Low 0 (CRC_DL0)

This section describes the function of CRC data registers (DH1:DH0:DL1:DL0). The set of CRC data registers contains the value of seed, data, and checksum. When CTRL[WAS] is set, any write to the data registers will be regarded as seed for CRC module. When CTRL[WAS] is de-asserted, any write to the data registers will be regarded as data for CRC module for general CRC computation.

When programming the seed value in 16-bit CRC mode, the DH1:DH0 are not used and reads to these registers returns an indeterminate value. For 32-bit CRC, all registers are used.

When programming data values for CRC calculation, data must be provided in DL0 register only. Writes to other bytes of data regsiters are ignored.

When final data written, the CRC result can be read from this data register. In 16-bit CRC mode, the CRC result is avaiable in DL1:DL0. In 32-bit CRC mode, all registers are used. Reads to this register at any time returns the intermediate CRC value, provided CRC module is configured.

Address: 1890h base + 3h offset = 1893h

Bit	7	6	5	4	3	2	1	0
Read Write	DL0							
Reset	1	1	1	1	1	1	1	1

CRC_DL0 field descriptions

Field	Description
DL0	CRC Data Bits 7:0

16.3.5 CRC Polynomial Register: High 1 (CRC_PH1)

This set of registers contains the value of polynomial for the CRC calculation. The registers of PH1:PH0 contain the upper 16 bits of CRC polynomial, which are only used in 32-bit CRC mode. Writes to PH1:PH0 are ignored in 16-bit CRC mode. The registers of PL1:PL0 contain the lower 16-bits of CRC polynomial, which are used in both 16- and 32-bit CRC modes.

Address: 1890h base + 4h offset = 1894h

Bit	7	6	5	4	3	2	1	0
Read Write	PH1							
Reset	0	0	0	0	0	0	0	0

CRC_PH1 field descriptions

Field	Description
PH1	CRC Polynomial Bits 31:24

16.3.6 CRC Polynomial Register: High 0 (CRC_PH0)

This set of registers contains the value of polynomial for the CRC calculation. The registers of PH1:PH0 contain the upper 16 bits of CRC polynomial, which are only used in 32-bit CRC mode. Writes to PH1:PH0 are ignored in 16-bit CRC mode. The registers of PL1:PL0 contain the lower 16-bits of CRC polynomial, which are used in both 16- and 32-bit CRC modes.

Address: 1890h base + 5h offset = 1895h

Bit	7	6	5	4	3	2	1	0
Read Write	PH0							
Reset	0	0	0	0	0	0	0	0

CRC_PH0 field descriptions

Field	Description
PH0	CRC Polynomial Bits 23:16

16.3.7 CRC Polynomial Register: Low 1 (CRC_PL1)

This set of registers contains the value of polynomial for the CRC calculation. The registers of PH1:PH0 contain the upper 16 bits of CRC polynomial, which are only used in 32-bit CRC mode. Writes to PH1:PH0 are ignored in 16-bit CRC mode. The registers of PL1:PL0 contain the lower 16-bits of CRC polynomial, which are used in both 16- and 32-bit CRC modes.

Address: 1890h base + 6h offset = 1896h

Bit	7	6	5	4	3	2	1	0
Read Write	PL1							
Reset	0	0	0	1	0	0	0	0

CRC_PL1 field descriptions

Field	Description
PL1	CRC Polynomial Bits 15:8

16.3.8 CRC Polynomial Register: Low 0 (CRC_PL0)

This set of registers contains the value of polynomial for the CRC calculation. The registers of PH1:PH0 contain the upper 16 bits of CRC polynomial, which are only used in 32-bit CRC mode. Writes to PH1:PH0 are ignored in 16-bit CRC mode. The registers of PL1:PL0 contain the lower 16-bits of CRC polynomial, which are used in both 16- and 32-bit CRC modes.

Address: 1890h base + 7h offset = 1897h

Bit	7	6	5	4	3	2	1	0
Read Write	PL0							
Reset	0	0	1	0	0	0	0	1

CRC_PL0 field descriptions

Field	Description
PL0	CRC Polynomial Bits 7:0

16.3.9 CRC Control register (CRC_CTRL)

This register controls the configuration and working of the CRC module. Appropriate bits must be set before starting a new CRC calculation. A new CRC calculation is initialized by asserting CTRL[WAS] and then writing the seed into the CRC data register.

Address: 1890h base + 8h offset = 1898h

Bit	7	6	5	4	3	2	1	0
Read	TOT		TOTR		0	FXOR	WAS	TCRC
Write								
Reset	0	0	0	0	0	0	0	0

CRC_CTRL field descriptions

Field	Description
7–6 TOT	Type Of Transpose For Writes Defines the transpose configuration of the data written to the CRC data register. See the description of the transpose feature for the available transpose options. 00 No transposition. 01 Bits in bytes are transposed; bytes are not transposed. 10 Both bits in bytes and bytes are transposed. 11 Only bytes are transposed; no bits in a byte are transposed.
5–4 TOTR	Type Of Transpose For Read

Table continues on the next page...

CRC_CTRL field descriptions (continued)

Field	Description
	Identifies the transpose configuration of the value read from the CRC Data register. See the description of the transpose feature for the available transpose options. 00 No transposition. 01 Bits in bytes are transposed; bytes are not transposed. 10 Both bits in bytes and bytes are transposed. 11 Only bytes are transposed; no bits in a byte are transposed.
3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 FXOR	Complement Read Of CRC Data Register Some CRC protocols require the final checksum to be XORed with 0xFFFFFFFF or 0xFFFF. Asserting this bit enables on the fly complementing of read data. 0 No XOR on reading. 1 Invert or complement the read value of the CRC Data register.
1 WAS	Write CRC Data Register As Seed When asserted, a value written to the CRC data register is considered a seed value. When deasserted, a value written to the CRC data register is taken as data for CRC computation. 0 Writes to the CRC data register are data values. 1 Writes to the CRC data register are seed values.
0 TCRC	Width of CRC protocol. 0 16-bit CRC protocol. 1 32-bit CRC protocol.

16.4 Functional description

16.4.1 CRC initialization/reinitialization

To enable the CRC calculation, the user must program CRC_CTRL[WAS], CRC_PH1, CRC_PH0, CRC_PL1 and CRC_PL0,necessary parameters for transposition and CRC result inversion in the applicable registers. Asserting CRC_CTRL[WAS] enables the programming of the seed value into the CRC_DH1, CRC_DH0, CRC_DL1 and CRC_DL0 register.

After a completed CRC calculation, the module can be reinitialized for a new CRC computation by reasserting CRC_CTRL[WAS] and programming a new, or previously used, seed value. All other parameters must be set before programming the seed value and subsequent data values.

16.4.2 CRC calculations

16.4.2.1 16-bit CRC

To compute a 16-bit CRC:

1. Clear CRC_CTRL[TCRC] to enable 16-bit CRC mode.
2. Program the transpose and complement options in the CTRL register as required for the CRC calculation. See Transpose feature and CRC result complement for details.
3. Write a 16-bit polynomial to the CRC_PL1 and CRC_PL0. The CRC_PH1 and CRC_PH2 are not usable in 16-bit CRC mode.
4. Set CRC_CTRL[WAS] to program the seed value.
5. Write a 16-bit seed to CRC_DL1 and CRC_DL0. CRC_DH1 and CRC_DH0 are not used.
6. Clear CRC_CTRL[WAS] to start writing data values.
7. Write data values into CRC_DL0. A CRC is computed on every data value write, and the intermediate CRC result is stored back into CRC_DL1 and CRC_DL0.
8. When all values have been written, read the final CRC result from CRC_DL1 and CRC_DL0.

Transpose and complement operations are performed on the fly while reading or writing values. See Transpose feature and CRC result complement for details.

16.4.2.2 32-bit CRC

To compute a 32-bit CRC:

1. Set CRC_CTRL[TCRC] to enable 32-bit CRC mode.
2. Program the transpose and complement options in the CTRL register as required for the CRC calculation. See Transpose feature and CRC result complement for details.
3. Write a 32-bit polynomial to CRC_PH1, CRC_PH0, CRC_PL1 and CRC_PL0.
4. Set CRC_CTRL[WAS] to program the seed value.
5. Write a 32-bit seed to CRC_DH1, CRC_DH0, CRC_DL1 and CRC_DL0.
6. Clear CRC_CTRL[WAS] to start writing data values.
7. Write data values into CRC_DL0. A CRC is computed on every data value write, and the intermediate CRC result is stored back into CRC_DH1, CRC_DH0, CRC_DL1 and CRC_DL0.

8. When all values have been written, read the final CRC result from CRC_DH1, CRC_DH0, CRC_DL1 and CRC_DL0. The CRC is calculated bytewise, and two clocks are required to complete one CRC calculation.

Transpose and complement operations are performed on the fly while reading or writing values. See Transpose feature and CRC result complement for details.

16.4.3 Transpose feature

By default, the transpose feature is not enabled. However, some CRC standards require the input data and/or the final checksum to be transposed. The user software has the option to configure each transpose operation separately, as desired by the CRC standard. The data is transposed on the fly while being read or written.

Some protocols use little endian format for the data stream to calculate a CRC. In this case, the transpose feature usefully flips the bits. This transpose option is one of the types supported by the CRC module.

16.4.3.1 Types of transpose

The CRC module provides several types of transpose functions to flip the bits and/or bytes, for both writing input data and reading the CRC result, separately using the CTRL[TOT] or CTRL[TOTR] fields, according to the CRC calculation being used.

The following types of transpose functions are available for writing to and reading from the CRC data register:

1. CTRL[TOT] or CTRL[TOTR] is 00.

   No transposition occurs.

2. CTRL[TOT] or CTRL[TOTR] is 01.

   Bits in a byte are transposed, while bytes are not transposed.

   reg[7:0] becomes reg[0:7]

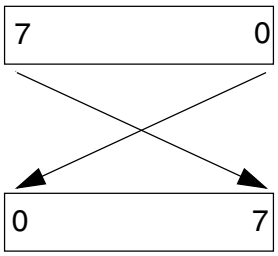


Figure 16-2. Transpose type 01

3. CTRL[TOT] or CTRL[TOTR] is 10 or 11.

   These invalid values must not be used.

NOTE

Bytewise transposition cannot be performed with 8-bit accesses and hence must be done in software by the CPU.

16.4.4 CRC result complement

When CTRL[FXOR] is set, the checksum is complemented. The CRC result complement function outputs the complement of the checksum value stored in the CRC data register every time the CRC data register is read. When CTRL[FXOR] is cleared, reading the CRC data register accesses the raw checksum value.

Chapter 17
Analog-to-digital converter (ADC)

17.1 Chip-specific ADC information

17.1.1 Analog-to-digital converter (ADC)

This device contains two analog-to-digital converters (ADC) of 12-bit, a successive approximation ADC for operation within an integrated microcontroller system-on-chip. The resolution can be configured to 8-, 10-, 12-bit. The fastest conversion rate at 12-bit is 2.5 µs. Single and continuous conversion with both software and hardware trigger are supported. Each ADC supports2 result FIFO with selectable FIFO depth.

ADC clock can be selected for either bus clock, exteranl clock, or its own asynchronous clock.

Additional information, such as ADC channel assignments, alternate clock function, and hardware trigger function, are configured as described following sections.

Customization:

- Primary clock: BUSCLK (20 MHz)
- Alternate clock: CLKIN , ADACK
- FIFO Enabled
- The following table summarizes the signal connection of ADC module.

Table 17-1. ADC module signals connection

Module	Signal	Connect to
ADC0 ADC1	ADO~AD30	Analog signal (details shouwn in below section)
	HW_TRIG	XBAR_OUT2, XBAR_OUT3
	Internal Clock	BUSCLK, ADACK
	External Clock	CLKIN

17.1.2 ADC channel assignments

The ADC channel assignments for the device are shown in the following table. Reserved channels convert to an unknown value.

Table 17-2. ADC0 channel assignments

ADCH	Channel	Input
00000	AD0	PTA0/ADCA0
00001	AD1	PTA1/ADCA1
00010	AD2	PTB0/ADCA2
00011	AD3	PTB1/ADCA3
00100	AD4	PTB2/ADCA4
00101	AD5	GDU_AMP0
00110	AD6	GDU_AMP1
00111	AD7	GDU_Vbus_sense_out
10110	AD22	Temperature sensor from ADC
10111	AD23	Bandgap[1]
11101	AD29	V_{REFH}
11110	AD30	V_{REFL}
11111	Module disabled	None

1. This is 1.15 V, high-precision bandgap from PMC.

Table 17-3. ADC1 channel assignments

ADCH	Channel	Input
00000	AD0	PTA2/ADCB0
00001	AD1	PTA3/ADCB1
00010	AD2	PTB0/ADCB2
00011	AD3	PTB1/ADCB3
00100	AD4	PTB2/ADCB4
00101	AD5	GDU_AMP0
00110	AD6	GDU_AMP1
00111	AD7	below a_ip_pmc_outa[1]
10110	AD22	Temperature sensor from ADC
10111	AD23	Bandgap[2]
11101	AD29	V_{REFH}
11110	AD30	V_{REFL}
11111	Module disabled	None

1. AD7 can select PMC temperature or bandgap by configuring PMC_TPCTRLSTAT.
2. This is 1.15 V, high-precision bandgap from PMC.

17.1.3 ADC analog supply and reference connections

This device does not have dedicated VDDA and VSSA pins. PMC provides a dedicated 4.2V voltage reference for the msot on-chip analog circuits. ADC analog supply and reference connections are below:

- Analog Power (V_{DDAD}) - The ADC analog portion uses V_{DDX} from PMC as its power connection. It is star connection to V_{DDX} pad.
- Analog Ground(V_{SSAD}) - The ADC analog portion uses V_{SS} as its groudn connection. It is star connection to V_{SS} pad.
- Voltage Reference High (V_{REFH}) - The high reference voltage for the converter. It is connected internally to V_{REFH} pad which is the output of a high accuracy 4.2 V voltage reference for PMC. PMC V_{REFH} is disabled by default. Customer must connect external V_{REFH} or enable internal V_{REFH} regulator.
- Voltage Reference Low (V_{REFL}) - the low-reference voltage for the converter. It is star connection to V_{SS} pin on package.

NOTE

Each V_{DDX} and V_{refh} pins must be connected to an output load capacitor respectively

17.1.4 Alternate clock

The ADC module is capable of performing conversions using the MCU bus clock, the bus clock divided by two, the local asynchronous clock (ADACK) within the module, or the alternate clock (ALTCLK). The alternate clock for the devices is the external clock input (CLKIN).

The selected clock source must run at a frequency such that the ADC conversion clock (ADCK) runs within its specified range (f_{ADCK}) after being divided down from one of clock sources as determined by the ADIV bits.

ALTCLK is active while the MCU is in wait mode provided the conditions described above are met. This allows ALTCLK to be used as the conversion clock source for the ADC while the MCU is in wait mode.

ALTCLK cannot be used as the ADC conversion clock source while the MCU is in stop mode.

17.1.5 Hardware trigger

The ADC hardware trigger is selectable from intermodule crossbar, which source can be from any input of intermodule crossbar, such as PDBs, FTM, PWM, analog comparators, package pins. The MCU can be configured to use any of those hardware trigger sources in run and wait modes. The stand alone analog comparator can be used as ADC hardware trigger in STOP mode.

17.1.6 Temperature sensor

The ADC module integrates an on-chip temperature sensor. Following actions must be performed to use this temperature sensor.

- Configure ADC for long sample with a maximum of 1 MHz clock
- Convert the bandgap voltage reference channel (AD23)
  - By converting the digital value of the bandgap voltage reference channel using the value of V_{BG} the user can determine V_{DD}.
- Convert the temperature sensor channel (AD22)
  - By using the calculated value of V_{DD}, convert the digital value of AD22 into a voltage, V_{TEMP}

The following equation provides an approximate transfer function of the on-chip temperature sensor for V_{DD} = 5.0V, Temp = 25°C, using the ADC at f_{ADCK} = 1.0 MHz and configured for long sample.

$$\text{Temp} = 25 - \left(\left(V_{TEMP} - V_{TEMP25}\right) \div m\right)$$

where:

- V_{TEMP} is the voltage of the temperature sensor channel at the ambient temperature
- V_{TEMP25} is the voltage of the temperature sensor channel at 25 °C
- m is the hot or cold voltage versus temperature slope in V/°C

For temperature calculations, use the V_{TEMP25} and m values in the data sheet.

In application code, you read the temperature sensor channel, calculate V_{TEMP}, and compare it to V_{TEMP25}. If V_{TEMP} is greater than V_{TEMP25}, the cold slope value is applied in the above equation. If V_{TEMP} is less than V_{TEMP25} the hot slope value is applied.

Calibrating at 25°C will improve accuracy to ±4.5°C.

Calibration at three points -40°C, 25°C, and 105°C will improve accuracy to ±2.5°C. After calibration has been completed, you will need to calculate the slope for both hot and cold. In application code, you can calculate the temperature as detailed above and determine if it is above or below 25°C. After you have determined whether the temperature is above or below 25 °C. you can recalculate the temperature using the hot or cold slope value obtained during calibration.

17.2 Introduction

The 12-bit analog-to-digital converter (ADC) is a successive approximation ADC designed for operation within an integrated microcontroller system-on-chip.

17.2.1 Features

Features of the ADC module include:

- Linear Successive Approximation algorithm with 8-, 10-, or 12-bit resolution
- Up to 8 external analog inputs, external pin inputs, and 5 internal analog inputs including internal bandgap, temperature sensor, and references
- Output formatted in 8-, 10-, or 12-bit right-justified unsigned format
- Single or Continuous Conversion (automatic return to idle after single conversion)
- Support up to eight result FIFO with selectable FIFO depth
- Configurable sample time and conversion speed/power
- Conversion complete flag and interrupt
- Input clock selectable from up to four sources
- Operation in Wait or Stop modes for lower noise operation
- Asynchronous clock source for lower noise operation
- Selectable asynchronous hardware conversion trigger
- Automatic compare with interrupt for less-than, or greater-than or equal-to, programmable value

17.2.2 Block Diagram

This figure provides a block diagram of the ADC module.

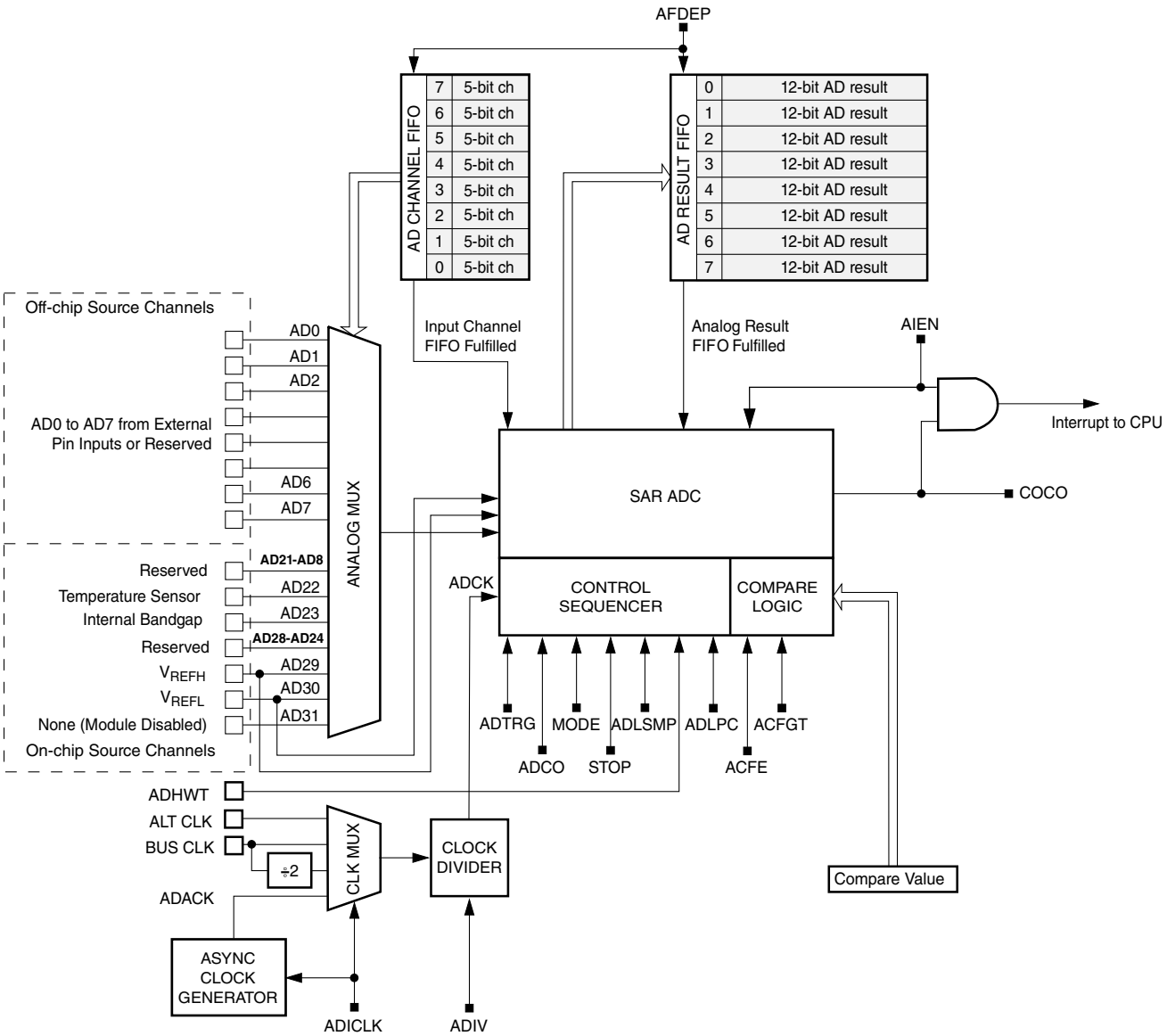


Figure 17-1. ADC Block Diagram

17.3 External Signal Description

The ADC module supports up to 24 separate analog inputs. It also requires four supply/ reference/ground connections.

Table 17-4. Signal Properties

Name	Function
AD23–AD0	Analog Channel inputs
V_{REFH}	High reference voltage
V_{REFL}	Low reference voltage
V_{DDA}	Analog power supply
V_{SSA}	Analog ground

17.3.1 Analog Power (V_{DDA})

The ADC analog portion uses V_{DDA} as its power connection. In some packages, V_{DDA} is connected internally to V_{DDX}. If externally available, connect the V_{DDA} pin to the same voltage potential as V_{DDX}. External filtering may be necessary to ensure clean V_{DDA} for good results.

17.3.2 Analog Ground (V_{SSA})

The ADC analog portion uses V_{SSA} as its ground connection. In some packages, V_{SSA} is connected internally to V_{SS}. If externally available, connect the V_{SSA} pin to the same voltage potential as V_{SS}.

17.3.3 Voltage Reference High (V_{REFH})

V_{REFH} is the high reference voltage for the converter. In some packages, V_{REFH} is connected internally to V_{DDA}. If externally available, V_{REFH} may be connected to the same potential as V_{DDA} or may be driven by an external source between the minimum V_{DDA} specified in the data sheet and the V_{DDA} potential (V_{REFH} must never exceed V_{DDA}).

17.3.4 Voltage Reference Low (V_{REFL})

V_{REFL} is the low-reference voltage for the converter. In some packages, V_{REFL} is connected internally to V_{SSA}. If externally available, connect the V_{REFL} pin to the same voltage potential as V_{SSA}.

17.3.5 Analog Channel Inputs (ADx)

The ADC module supports up to 24 separate analog inputs. An input is selected for conversion through the ADCH channel select bits.

17.4 ADC Control Registers

ADC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
10	Status and Control Register 1 (ADC0_SC1)	8	R/W	1Fh	17.4.1/262
11	Status and Control Register 2 (ADC0_SC2)	8	R/W	08h	17.4.2/264
12	Status and Control Register 3 (ADC0_SC3)	8	R/W	00h	17.4.3/265
13	Status and Control Register 4 (ADC0_SC4)	8	R/W	00h	17.4.4/266
14	Conversion Result High Register (ADC0_RH)	8	R	00h	17.4.5/267
15	Conversion Result Low Register (ADC0_RL)	8	R	00h	17.4.6/268
16	Compare Value High Register (ADC0_CVH)	8	R/W	00h	17.4.7/269
17	Compare Value Low Register (ADC0_CVL)	8	R/W	00h	17.4.8/269
18	Status and Control Register 1 (ADC1_SC1)	8	R/W	1Fh	17.4.1/262
19	Status and Control Register 2 (ADC1_SC2)	8	R/W	08h	17.4.2/264
1A	Status and Control Register 3 (ADC1_SC3)	8	R/W	00h	17.4.3/265
1B	Status and Control Register 4 (ADC1_SC4)	8	R/W	00h	17.4.4/266
1C	Conversion Result High Register (ADC1_RH)	8	R	00h	17.4.5/267
1D	Conversion Result Low Register (ADC1_RL)	8	R	00h	17.4.6/268
1E	Compare Value High Register (ADC1_CVH)	8	R/W	00h	17.4.7/269
1F	Compare Value Low Register (ADC1_CVL)	8	R/W	00h	17.4.8/269

17.4.1 Status and Control Register 1 (ADCx_SC1)

This section describes the function of the ADC status and control register (ADC_SC1). Writing ADC_SC1 aborts the current conversion and initiates a new conversion (if the ADCH bits are equal to a value other than all 1s).

When FIFO is enabled, the analog input channel FIFO is written via ADCH. The analog input channel queue must be written to ADCH continuously. The resulting FIFO follows the order in which the analog input channel is written. The ADC will start conversion when the input channel FIFO is fulfilled at the depth indicated by the ADC_SC4[AFDEP]. Any write 0x1F to these bits will reset the FIFO and stop the conversion if it is active.

Address: Base address + 0h offset

Bit	7	6	5	4	3	2	1	0
Read	COCO	AIEN	ADCO	ADCH				
Write								
Reset	0	0	0	1	1	1	1	1

ADCx_SC1 field descriptions

Field	Description
7 COCO	Conversion Complete Flag Conversion Complete Flag. The COCO flag is a read-only bit set each time a conversion is completed when the compare function is disabled (ADC_SC2[ACFE] = 0). When the compare function is enabled (ADC_SC2[ACFE] = 1), the COCO flag is set upon completion of a conversion only if the compare result is true. When the FIFO function is enabled (ADC_SC4[AFDEP] > 0), the COCO flag is set upon completion of the set of FIFO conversion. This bit is cleared when ADC_SC1 is written or when ADC_RL is read. 0 Conversion not completed. 1 Conversion completed.
6 AIEN	Interrupt Enable AIEN enables conversion complete interrupts. When COCO becomes set while AIEN is high, an interrupt is asserted. 0 Conversion complete interrupt disabled. 1 Conversion complete interrupt enabled.
5 ADCO	Continuous Conversion Enable ADCO enables continuous conversions. 0 One conversion following a write to the ADC_SC1 when software triggered operation is selected, or one conversion following assertion of ADHWT when hardware triggered operation is selected. When the FIFO function is enabled (AFDEP > 0), a set of conversions are triggered. 1 Continuous conversions are initiated following a write to ADC_SC1 when software triggered operation is selected. Continuous conversions are initiated by an ADHWT event when hardware triggered operation is selected. When the FIFO function is enabled (AFDEP > 0), a set of conversions are loop triggered.
ADCH	Input Channel Select The ADCH bits form a 5-bit field that selects one of the input channels. 00000-00111 AD0-AD7 01000-10011 V_{SS} 10100-10101 Reserved 10110 Temperature Sensor 10111 Bandgap 11000-11100 Reserved 11101 V_{REFH} 11110 V_{REFL} 11111 Module disabled **NOTE:** Reset FIFO in FIFO mode.

17.4.2 Status and Control Register 2 (ADCx_SC2)

The ADC_SC2 register controls the compare function, conversion trigger, and conversion active of the ADC module.

Address: Base address + 1h offset

Bit	7	6	5	4	3	2	1	0
Read	ADACT	ADTRG	ACFE	ACFGT	FEMPTY	FFULL	REFSEL	
Write								
Reset	0	0	0	0	1	0	0	0

ADCx_SC2 field descriptions

Field	Description
7 ADACT	Conversion Active Indicates that a conversion is in progress. ADACT is set when a conversion is initiated and cleared when a conversion is completed or aborted. 0 Conversion not in progress. 1 Conversion in progress.
6 ADTRG	Conversion Trigger Select Selects the type of trigger used for initiating a conversion. Two types of trigger are selectable: software trigger and hardware trigger. When software trigger is selected, a conversion is initiated following a write to ADC_SC1. When hardware trigger is selected, a conversion is initiated following the assertion of the ADHWT input. 0 Software trigger selected. 1 Hardware trigger selected.
5 ACFE	Compare Function Enable Enables the compare function. 0 Compare function disabled. 1 Compare function enabled.
4 ACFGT	Compare Function Greater Than Enable Configures the compare function to trigger when the result of the conversion of the input being monitored is greater than or equal to the compare value. The compare function defaults to triggering when the result of the compare of the input being monitored is less than the compare value. 0 Compare triggers when input is less than compare level. 1 Compare triggers when input is greater than or equal to compare level.
3 FEMPTY	Result FIFO empty 0 Indicates that ADC result FIFO have at least one valid new data. 1 Indicates that ADC result FIFO have no valid new data.
2 FFULL	Result FIFO full

Table continues on the next page...

ADCx_SC2 field descriptions (continued)

Field	Description
	0 Indicates that ADC result FIFO is not full and next conversion data still can be stored into FIFO. 1 Indicates that ADC result FIFO is full and next conversion will override old data in case of no read action.
REFSEL	Voltage Reference Selection Selects the voltage reference source used for conversions. 00 Default voltage reference pin pair (V_{REFH}/V_{REFL}). 01 Analog supply pin pair (V_{DDA}/V_{SSA}). 10 Reserved. 11 Reserved - Selects default voltage reference (V_{REFH}/V_{REFL}) pin pair.

17.4.3 Status and Control Register 3 (ADCx_SC3)

ADC_SC3 selects the mode of operation, clock source, clock divide, and configure for low power or long sample time.

Address: Base address + 2h offset

Bit	7	6	5	4	3	2	1	0
Read Write	ADLPC	ADIV		ADLSMP	MODE		ADICLK	
Reset	0	0	0	0	0	0	0	0

ADCx_SC3 field descriptions

Field	Description
7 ADLPC	Low-Power Configuration ADLPC controls the speed and power configuration of the successive approximation converter. This optimizes power consumption when higher sample rates are not required. 0 High speed configuration. 1 Low power configuration:The power is reduced at the expense of maximum clock speed.
6–5 ADIV	Clock Divide Select ADIV selects the divide ratio used by the ADC to generate the internal clock ADCK. 00 Divide ration = 1, and clock rate = Input clock. 01 Divide ration = 2, and clock rate = Input clock ÷ 2. 10 Divide ration = 3, and clock rate = Input clock ÷ 4. 11 Divide ration = 4, and clock rate = Input clock ÷ 8.
4 ADLSMP	Long Sample Time Configuration ADLSMP selects between long and short sample time. This adjusts the sample period to allow higher impedance inputs to be accurately sampled or to maximize conversion speed for lower impedance inputs. Longer sample times can also be used to lower overall power consumption when continuous conversions are enabled if high conversion rates are not required.

Table continues on the next page...

ADCx_SC3 field descriptions (continued)

Field	Description
	0 Short sample time. 1 Long sample time.
3–2 MODE	Conversion Mode Selection MODE bits are used to select between 12-, 10-, or 8-bit operation. 00 8-bit conversion (N=8) 01 10-bit conversion (N=10) 10 12-bit conversion (N=12) 11 Reserved
ADICLK	Input Clock Select ADICLK bits select the input clock source to generate the internal clock ADCK. 00 Bus clock 01 Bus clock divided by 2 10 Alternate clock (ALTCLK) 11 Asynchronous clock (ADACK)

17.4.4 Status and Control Register 4 (ADCx_SC4)

This register controls the FIFO scan mode, FIFO compare function and FIFO depth selection of the ADC module.

Address: Base address + 3h offset

Bit	7	6	5	4
Read Write	HTRGME	ASCANE	ACFSEL	HTRGMASKE
Reset	0	0	0	0

Bit	3	2	1	0
Read Write	HTRGMASKSEL	AFDEP		
Reset	0	0	0	0

ADCx_SC4 field descriptions

Field	Description
7 HTRGME	Hardware Trigger Multiple Conversion Enable This field enables hardware trigger multiple conversion. 0 One hardware trigger pulse triggers one conversion. 1 One hardware trigger pulse triggers multiple conversions in fifo mode.
6 ASCANE	FIFO Scan Mode Enable The FIFO always use the first dummied FIFO channels when it is enabled. When this bit is set and FIFO function is enabled, ADC will repeat using the first FIFO channel as the conversion channel until the result

Table continues on the next page...

ADCx_SC4 field descriptions (continued)

Field	Description
	FIFO is fulfilled. In continuous mode (ADCO = 1), ADC will start next conversion with the same channel when COCO is set. 0 FIFO scan mode disabled. 1 FIFO scan mode enabled.
5 ACFSEL	Compare function select OR/AND when the FIFO function is enabled (AFDEP > 0). When this field is cleared, ADC will OR all of compare triggers and set COCO after at least one of compare trigger occurs. When this field is set, ADC will AND all of compare triggers and set COCO after all of compare triggers occur. 0 OR all of compare trigger. 1 AND all of compare trigger.
4 HTRGMASKE	Hardware Trigger Mask Enable This field enables hardware trigger mask when HTRGMASKSEL is low. 0 Hardware trigger mask disable. 1 Hardware trigger mask enable and hardware trigger cannot trigger ADC conversion..
3 HTRGMASKSEL	Hardware Trigger Mask Mode Select This field selects hardware trigger mask mode. 0 Hardware trigger mask with HTRGMASKE. 1 Hardware trigger mask automatically when data fifo is not empty.
AFDEP	FIFO Depth enables the FIFO function and sets the depth of FIFO. When AFDEP is cleared, the FIFO is disabled. When AFDEP is set to nonzero, the FIFO function is enabled and the depth is indicated by the AFDEP bits. The ADCH in ADC_SC1 and ADC_RH:ADC_RL must be accessed by FIFO mode when FIFO function is enabled. ADC starts conversion when the analog channel FIFO is upon the level indicated by AFDEP bits. The COCO bit is set when the set of conversions are completed and the result FIFO is upon the level indicated by AFDEP bits. **NOTE:** The bus clock frequency must be at least double the ADC clock when FIFO mode is enabled. It means, if ICS FBE mode is used, the ADC clock can not be ADACK. 000 FIFO is disabled. 001 2-level FIFO is enabled. 010 3-level FIFO is enabled.. 011 4-level FIFO is enabled. 100 5-level FIFO is enabled. 101 6-level FIFO is enabled. 110 7-level FIFO is enabled. 111 8-level FIFO is enabled.

17.4.5 Conversion Result High Register (ADCx_RH)

In 12-bit operation, ADC_RH contains the upper four bits of the result of a 12-bit conversion.

ADC_RH is updated each time a conversion completes except when automatic compare is enabled and the compare condition is not met. Reading ADC_RH prevents the ADC from transferring subsequent conversion results into the result registers until ADC_RL is read. If ADC_RL is not read until after the next conversion is completed, the intermediate conversion result is lost. In 8-bit mode, there is no interlocking with ADC_RL.

When FIFO is enabled, the result FIFO is read via ADC_RH:ADC_RL. The ADC conversion completes when the input channel FIFO is fulfilled at the depth indicated by the AFDEP. The AD result FIFO can be read via ADC_RH:ADC_RL continuously by the order set in analog input channel ADCH.

If the MODE bits are changed, any data in ADC_RH becomes invalid.

Address: Base address + 4h offset

Bit	7	6	5	4	3	2	1	0
Read	0				ADR			
Write								
Reset	0	0	0	0	0	0	0	0

ADCx_RH field descriptions

Field	Description
7–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
ADR	Conversion Result[12:8]

17.4.6 Conversion Result Low Register (ADCx_RL)

ADC_RL contains the lower eight bits of the result of a 12-bit conversion. This register is updated each time a conversion completes except when automatic compare is enabled and the compare condition is not met. In 12-bit mode, reading ADC_RH prevents the ADC from transferring subsequent conversion results into the result registers until ADC_RL is read. If ADC_RL is not read until the next conversion is completed, the intermediate conversion results are lost. In 8-bit mode, there is no interlocking with ADC_RH. If the MODE bits are changed, any data in ADC_RL becomes invalid.

When FIFO is enabled, the result FIFO is read via ADC_RH:ADC_RL. The ADC conversion completes when the input channel FIFO is fulfilled at the depth indicated by the AFDEP. The AD result FIFO can be read via ADC_RH:ADC_RL continuously by the order set in analog input channel FIFO.

Address: Base address + 5h offset

Bit	7	6	5	4	3	2	1	0
Read	ADR							
Write								
Reset	0	0	0	0	0	0	0	0

ADCx_RL field descriptions

Field	Description
ADR	Conversion Result[7:0]

17.4.7 Compare Value High Register (ADCx_CVH)

In 12-bit mode, this register holds the upper four bits of the 12-bit compare value. These bits are compared to the upper four bits of the result following a conversion in 12-bit mode when the compare function is enabled.

Address: Base address + 6h offset

Bit	7	6	5	4	3	2	1	0
Read	0				CV			
Write								
Reset	0	0	0	0	0	0	0	0

ADCx_CVH field descriptions

Field	Description
7–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CV	Conversion Result[15:8]

17.4.8 Compare Value Low Register (ADCx_CVL)

This register holds the lower 8 bits of the 12-bit compare value. Bits CV7:CV0 are compared to the lower 8 bits of the result following a conversion in 12-bit mode.

Address: Base address + 7h offset

Bit	7	6	5	4	3	2	1	0
Read Write	CV							
Reset	0	0	0	0	0	0	0	0

ADCx_CVL field descriptions

Field	Description
CV	Conversion Result[7:0]

17.5 Functional description

The ADC module is disabled during reset or when the ADC_SC1[ADCH] bits are all high. The module is idle when a conversion has completed and another conversion has not been initiated. When idle, the module is in its lowest power state.

The ADC can perform an analog-to-digital conversion on any of the software selectable channels. In 12-bit mode, the selected channel voltage is converted by a successive approximation algorithm into a 12-bit digital result. In 10-bit mode, the selected channel voltage is converted by a successive approximation algorithm into a 10-bit digital result. In 8-bit mode, the selected channel voltage is converted by a successive approximation algorithm into a 8-bit digital result.

When the conversion is completed, the result is placed in the data registers (ADC_R). In 10-bit mode, the result is rounded to 10 bits and placed in the data registers (ADC_R). In 8-bit mode, the result is rounded to 8 bits and placed in ADC_R. The conversion complete flag (ADC_SC1[COCO]) is then set and an interrupt is generated if the conversion complete interrupt has been enabled (ADC_SC1[AIEN] = 1).

The ADC module has the capability of automatically comparing the result of a conversion with the contents of its compare registers. The compare function is enabled by setting the ADC_SC2[ACFE] bit and operates with any of the conversion modes and configurations.

17.5.1 Clock select and divide control

One of four clock sources can be selected as the clock source for the ADC module. This clock source is then divided by a configurable value to generate the input clock to the converter (ADCK). The clock is selected from one of the following sources by means of the ADC_SC3[ADICLK] bits.

- The bus clock, which is equal to the frequency at which software is executed. This is the default selection following reset.
- The bus clock divided by 2: For higher bus clock rates, this allows a maximum divide by 16 of the bus clock.

- ALTCLK, that is, alternate clock which is OSCOUT
- The asynchronous clock (ADACK): This clock is generated from a clock source within the ADC module. When selected as the clock source, this clock remains active while the MCU is in Wait or Stop mode and allows conversions in these modes for lower noise operation.

Whichever clock is selected, its frequency must fall within the specified frequency range for ADCK. If the available clocks are too slow, the ADC does not perform according to specifications. If the available clocks are too fast, the clock must be divided to the appropriate frequency. This divider is specified by the ADC_SC3[ADIV] bits and can be divide-by 1, 2, 4, or 8.

17.5.2 Hardware trigger

The ADC module has a selectable asynchronous hardware conversion trigger, ADHWT, that is enabled when the ADC_SC2[ADTRG] bit is set. This source is not available on all MCUs. See the module introduction for information on the ADHWT source specific to this MCU.

When ADHWT source is available and hardware trigger is enabled (ADC_SC2[ADTRG] = 1), a conversion is initiated on the rising edge of ADHWT. If a conversion is in progress when a rising edge occurs, the rising edge is ignored. In continuous convert configuration, only the initial rising edge to launch continuous conversions is observed. The hardware trigger function operates in conjunction with any of the conversion modes and configurations.

17.5.3 Conversion control

Conversions can be performed in 12-bit mode, 10-bit mode, or 8-bit mode as determined by the ADC_SC3[MODE] bits. Conversions can be initiated by a software or hardware trigger. In addition, the ADC module can be configured for low power operation, long sample time, continuous conversion, and an automatic compare of the conversion result to a software determined compare value.

17.5.3.1 Initiating conversions

A conversion initiates under the following conditions:

- A write to ADC_SC1 or a set of write to ADC_SC1 in FIFO mode (with ADCH bits not all 1s) if software triggered operation is selected.
- A hardware trigger (ADHWT) event if hardware triggered operation is selected.
- The transfer of the result to the data registers when continuous conversion is enabled.

If continuous conversions are enabled, a new conversion is automatically initiated after the completion of the current conversion. In software triggered operation, continuous conversions begin after ADC_SC1 is written and continue until aborted. In hardware triggered operation, continuous conversions begin after a hardware trigger event and continue until aborted.

17.5.3.2 Completing conversions

A conversion is completed when the result of the conversion is transferred into the data result register, ADC_R. This is indicated by the setting of ADC_SC1[COCO]. An interrupt is generated if ADC_SC1[AIEN] is high at the time that ADC_SC1[COCO] is set.

17.5.3.3 Aborting conversions

Any conversion in progress is aborted in the following cases:

- A write to ADC_SC1 occurs.
  - The current conversion will be aborted and a new conversion will be initiated, if ADC_SC1[ADCH] are not all 1s and ADC_SC4[AFDEP] are all 0s.
  - The current conversion and the rest of conversions will be aborted and no new conversion will be initialed, if ADC_SC4[AFDEP] are not all 0s.
  - A new conversion will be initiated when the FIFO is re-fulfilled upon the levels indicated by the ADC_SC4[AFDEP] bits).
- A write to ADC_SC2, ADC_SC3, ADC_SC4, ADC_CV occurs. This indicates a mode of operation change has occurred and the current and rest of conversions (when ADC_SC4[AFDEP] are not all 0s) are therefore invalid.
- The MCU is reset.
- The MCU enters Stop mode with ADACK not enabled.

When a conversion is aborted, the contents of the data register, ADC_R, are not altered. However, they continue to be the values transferred after the completion of the last successful conversion. If the conversion was aborted by a reset, ADC_R return to their reset states.

17.5.3.4 Power control

The ADC module remains in its idle state until a conversion is initiated. If ADACK is selected as the conversion clock source, the ADACK clock generator is also enabled.

Power consumption when active can be reduced by setting ADC_SC3[ADLPC]. This results in a lower maximum value for f_{ADCK} (see the data sheet).

17.5.3.5 Sample time and total conversion time

The total conversion time depends on the sample time (as determined by ADC_SC3[ADLSMP]), the MCU bus frequency, the conversion mode (8-bit, 10-bit or 12-bit), and the frequency of the conversion clock (f_{ADCK}). After the module becomes active, sampling of the input begins.ADC_SC3[ADLSMP] selects between short (3.5 ADCK cycles) and long (23.5 ADCK cycles) sample times. When sampling is complete, the converter is isolated from the input channel and a successive approximation algorithm is performed to determine the digital value of the analog signal. The result of the conversion is transferred to ADC_R upon completion of the conversion algorithm.

If the bus frequency is less than the f_{ADCK} frequency, precise sample time for continuous conversions cannot be guaranteed when short sample is enabled (ADC_SC3[ADLSMP] = 0). If the bus frequency is less than 1/11th of the f_{ADCK} frequency, precise sample time for continuous conversions cannot be guaranteed when long sample is enabled (ADC_SC3[ADLSMP] = 1).

The maximum total conversion time for different conditions is summarized in the table below.

Table 17-5. Total conversion time vs. control conditions

Conversion type	ADICLK	ADLSMP	Max total conversion time
Single or first continuous 8-bit	0x, 10	0	20 ADCK cycles + 5 bus clock cycles
Single or first continuous 10-bit or 12-bit	0x, 10	0	23 ADCK cycles + 5 bus clock cycles
Single or first continuous 8-bit	0x, 10	1	40 ADCK cycles + 5 bus clock cycles
Single or first continuous 10-bit or 12-bit	0x, 10	1	43 ADCK cycles + 5 bus clock cycles
Single or first continuous 8-bit	11	0	5 µs + 20 ADCK + 5 bus clock cycles
Single or first continuous 10-bit or 12-bit	11	0	5 µs + 23 ADCK + 5 bus clock cycles

Table continues on the next page...

Table 17-5. Total conversion time vs. control conditions (continued)

Conversion type	ADICLK	ADLSMP	Max total conversion time
Single or first continuous 8-bit	11	1	5 µs + 40 ADCK + 5 bus clock cycles
Single or first continuous 10-bit or 12-bit	11	1	5 µs + 43 ADCK + 5 bus clock cycles
Subsequent continuous 8-bit; $f_{BUS} > f_{ADCK}$	xx	0	17 ADCK cycles
Subsequent continuous 10-bit or 12-bit; $f_{BUS} > f_{ADCK}$	xx	0	20 ADCK cycles
Subsequent continuous 8-bit; $f_{BUS} > f_{ADCK}/11$	xx	1	37 ADCK cycles
Subsequent continuous 10-bit or 12-bit; $f_{BUS} > f_{ADCK}/11$	xx	1	40 ADCK cycles

The maximum total conversion time is determined by the selected clock source and the divide ratio. The clock source is selectable by the ADC_SC3[ADICLK] bits, and the divide ratio is specified by the ADC_SC3[ADIV] bits. For example, in 10-bit mode, with the bus clock selected as the input clock source, the input clock divide-by-1 ratio selected, and a bus frequency of 8 MHz, then the conversion time for a single conversion as given below:

$$\text{Conversion time} = \frac{23\,\text{ADCK Cyc}}{8\,\text{MHz}/1} + \frac{5\,\text{bus Cyc}}{8\,\text{MHz}} = 3.5\,\mu\text{s}$$

The number of bus cycles at 8 MHz is:

$$\text{Bus cycles} = 3.5\mu\text{s} \times 8\text{MHz} = 28$$

Note

The ADCK frequency must be between f_{ADCK} minimum and f_{ADCK} maximum to meet ADC specifications.

17.5.4 Automatic compare function

The compare function can be configured to check for an upper or lower limit. After the input is sampled and converted, the result is added to the complement of the compare value (ADC_CV). When comparing to an upper limit (ADC_SC2[ACFGT] = 1), if the result is greater-than or equal-to the compare value, ADC_SC1[COCO] is set. When comparing to a lower limit (ADC_SC2[ACFGT] = 0), if the result is less than the compare value, ADC_SC1[COCO] is set. The value generated by the addition of the conversion result and the complement of the compare value is transferred to ADC_R.

On completion of a conversion while the compare function is enabled, if the compare condition is not true, ADC_SC1[COCO] is not set and no data is transferred to the result registers. An ADC interrupt is generated on the setting of ADC_SC1[COCO] if the ADC interrupt is enabled (ADC_SC1[AIEN] = 1).

On completion of all conversions while the compare function is enabled and FIFO enabled, if none of the compare conditions are true when ADC_SC4[ACFSEL] is low or if not all of compare conditions are true when ADC_SC4[ACFSEL] is high, ADC_SC1[COCO] is not set. The compare data are transferred to the result registers regardless of compare condition true or false when FIFO enabled.

Note

The compare function can monitor the voltage on a channel while the MCU is in Wait or Stop mode. The ADC interrupt wakes the MCU when the compare condition is met.

Note

The compare function can not work in continuous conversion mode when FIFO enabled.

17.5.5 FIFO operation

The ADC module supports FIFO operation to minimize the interrupts to CPU in order to reduce CPU loading in ADC interrupt service routines. This module contains two FIFOs to buffer analog input channels and analog results respectively.

The FIFO function is enabled when the ADC_SC4[AFDEP] bits are set non-zero. The FIFO depth is indicated by these bits. The FIFO supports up to eight level buffer.

The analog input channel FIFO is accessed by ADC_SC1[ADCH] bits, when FIFO function is enabled. The analog channel must be written to this FIFO in order. The ADC will not start the conversion if the channel FIFO is fulfilled below the level indicated by the ADC_SC4[AFDEP] bits, no matter whether software or hardware trigger is set. Read ADC_SC1[ADCH] will read the current active channel value. Write to ADC_SC1[ADCH] will re-fill channel FIFO to initial new conversion. It will abort current conversion and any other conversions that did not start. Write to the ADC_SC1 after all the conversions are completed or ADC is in idle state.

The result of the FIFO is accessed by ADC_R register, when FIFO function is enabled. The result must be read via these two registers by the same order of analog input channel FIFO to get the proper results. Don't read ADC_R until all of the conversions are completed in FIFO mode. The ADC_SC1[COCO] bit will be set only when all conversions indicated by the analog input channel FIFO complete whatever software or

hardware trigger is set. An interrupt request will be submitted to CPU if the ADC_SC1[AIEN] is set when the FIFO conversion completes and the ADC_SC1[COCO] bit is set.

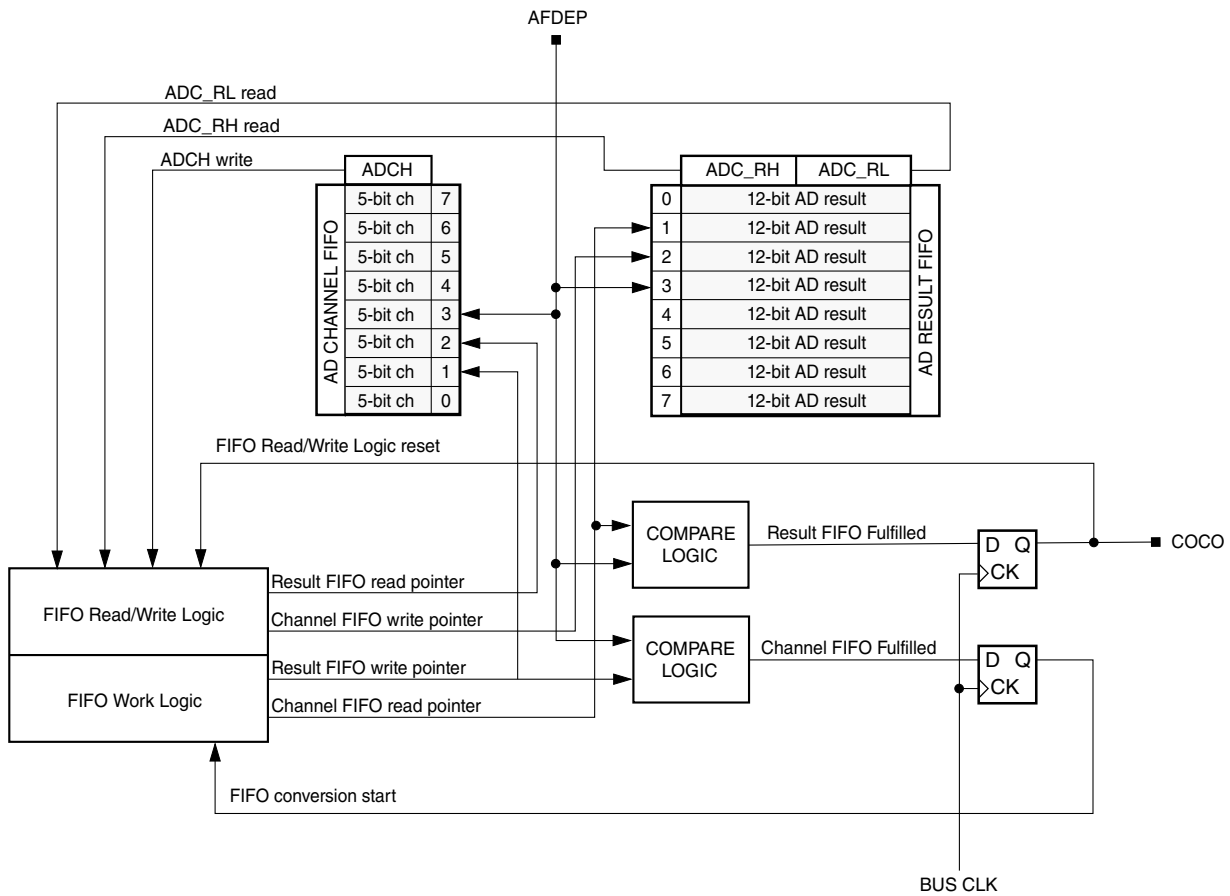


Figure 17-2. FADC FIFO structure

If software trigger is enabled, the next analog channel is fetched from analog input channel FIFO as soon as a conversion completes and its result is stored in the result FIFO. When all conversions set in the analog input channel FIFO completes, the ADC_SC1[COCO] bit is set and an interrupt request will be submitted to CPU if the ADC_SC1[AIEN] bit is set.

If single hardware trigger mode is enabled(ADC_SC2[ADTRG]= 1 and ADC_SC4[HTRGME]=0), the next analog is fetched from analog input channel FIFO only when this conversion completes, its result is stored in the result FIFO, and the next hardware trigger is fed to ADC module. If multi hardware tigger mode is enabled(ADC_SC2[ADTRG]=1 and ADC_SC4[HTRGME]=1), the next analog is fetched from analog input channel FIFO only when this conversion completes, its result is stored in the result FIFO, and next conversion will start without waiting for next

hardware trigger. When all conversions set in the analog input channel FIFO completes, the ADC_SC1[COCO] bit is set and an interrupt request will be submitted to CPU if the ADC_SC1[AIEN] bit is set.

In single conversion in which ADC_SC1[ADCO] bit is clear, the ADC stops conversions when ADC_SC1[COCO] bit is set until the channel FIFO is fulfilled again or new hardware trigger occur.

The FIFO also provides scan mode to simplify the dummy work of input channel FIFO. When the ADC_SC4[ASCANE] bit is set in FIFO mode, the FIFO will always use the first dummied channel in spite of the value in the input channel FIFO. The ADC conversion start to work in FIFO mode as soon as the first channel is dummied. The following write operation to the input channel FIFO will cover the first channel element in this FIFO. In scan FIFO mode, the ADC_SC1[COCO] bit is set when the result FIFO is fulfilled according to the depth indicated by the ADC_SC4[AFDEP] bits.

In continuous conversion in which the ADC_SC1[ADCO] bit is set, the ADC starts next conversion immediately when all conversions are completed. ADC module will fetch the analog input channel from the beginning of analog input channel FIFO.

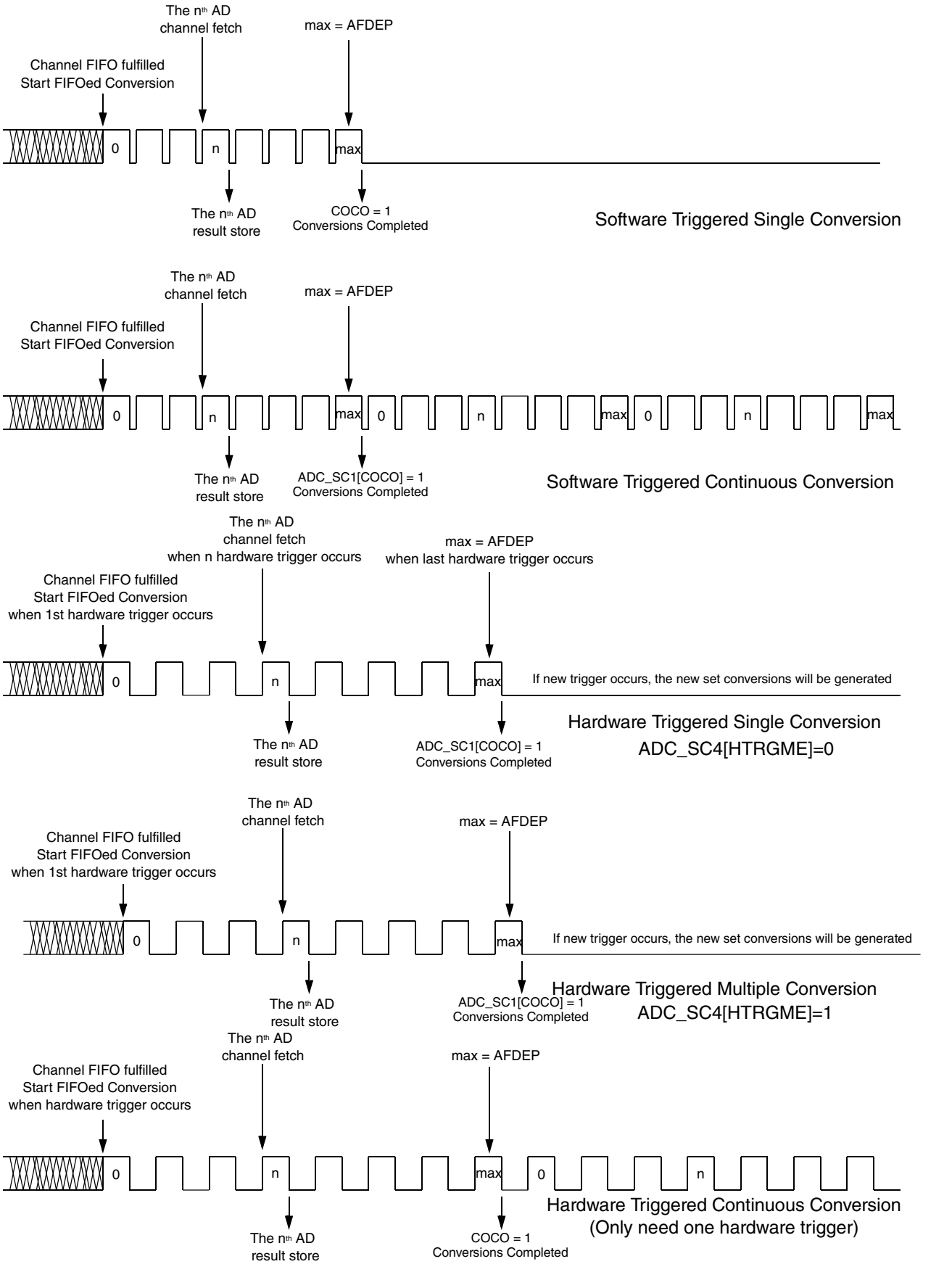


Figure 17-3. ADC FIFO conversion sequence

17.5.6 MCU wait mode operation

Wait mode is a low-power consumption standby mode from which recovery is fast because the clock sources remain active. If a conversion is in progress when the MCU enters wait mode, it continues until completion. Conversions can be initiated while the MCU is in wait mode by means of the hardware trigger or if continuous conversions are enabled.

The bus clock, bus clock divided by two, ALTCLK and ADACK are available as conversion clock sources while in wait mode.

ADC_SC1[COCO] is set by a conversion complete event that generates an ADC interrupt to wake the MCU from wait mode if the ADC interrupt is enabled (ADC_SC1[AIEN] = 1).

17.5.7 MCU Stop mode operation

Stop mode is a low-power consumption standby mode during which most or all clock sources on the MCU are disabled.

17.5.7.1 Stop mode with ADACK disabled

If the asynchronous clock, ADACK, is not selected as the conversion clock, executing a STOP instruction aborts the current conversion and places the ADC in its idle state. The contents of ADC_R are unaffected by Stop mode. After exiting from Stop mode, a software or hardware trigger is required to resume conversions.

17.5.7.2 Stop mode with ADACK enabled

If ADACK is selected as the conversion clock, the ADC continues operation during Stop mode. For guaranteed ADC operation, the MCU's voltage regulator must remain active during Stop mode. See the module introduction for configuration information for this MCU.

If a conversion is in progress when the MCU enters Stop mode, it continues until completion. Conversions can be initiated while the MCU is in Stop mode by means of the hardware trigger or if continuous conversions are enabled.

A conversion complete event sets the ADC_SC1[COCO] and generates an ADC interrupt to wake the MCU from Stop mode if the ADC interrupt is enabled (ADC_SC1[AIEN] = 1). In fifo mode, ADC cannot complete the conversion operation fully or wake the MCU from Stop mode.

Note

The ADC module can wake the system from low-power stop and cause the MCU to begin consuming run-level currents without generating a system level interrupt. To prevent this scenario, the data transfer blocking mechanism must be cleared when entering Stop and continuing ADC conversions.

17.6 Initialization information

This section gives an example that provides some basic direction on how to initialize and configure the ADC module. You can configure the module for 8-, 10-, or 12-bit resolution, single or continuous conversion, and a polled or interrupt approach, among many other options. Refer to ADC_SC3 register for information used in this example.

Note

Hexadecimal values prefixed by a 0x, binary values prefixed by a %, and decimal values have no preceding character.

17.6.1 ADC module initialization example

Before the ADC module can be used to complete conversions, it must be initialized. Given below is a method to initialize ADC module.

17.6.1.1 Initialization sequence

A typical initialization sequence is as follows:

1. Update the configuration register (ADC_SC3) to select the input clock source and the divide ratio used to generate the internal clock, ADCK. This register is also used for selecting sample time and low-power configuration.
2. Update status and control register 2 (ADC_SC2) to select the hardware or software conversion trigger and compare function options, if enabled.

3. Update status and control register 1 (ADC_SC1) to select whether conversions will be continuous or completed only once, and to enable or disable conversion complete interrupts. The input channel on which conversions will be performed is also selected here.

17.6.1.2 Pseudo-code example

In this example, the ADC module is set up with interrupts enabled to perform a single 10-bit conversion at low power with a long sample time on input channel 1, where the internal ADCK clock is derived from the bus clock divided by 1.

Example: 17.6.1.2.1 General ADC initialization routine

```
void ADC_init(void)
{
      /* The following code segment demonstrates how to initialize ADC by low-power mode,
long
      sample time, bus frequency, software triggered from AD1 external pin without FIFO
enabled
      */

      ADC_SC3 = ADC_SC3_ADLPC_MASK | ADC_SC3_ADLSMP_MASK | ADC_SC3_MODE0_MASK;
      ADC_SC2 = 0x00;
      ADC_SC1 = ADC_SC1_AIEN_MASK | ADC_SC1_ADCH0_MASK;
}
```

17.6.2 ADC FIFO module initialization example

Before the ADC module can be used to start FIFOed conversions, an initialization procedure must be performed. A typical sequence is as follows:

1. Update the configuration register (ADC_SC3) to select the input clock source and the divide ratio used to generate the internal clock, ADCK. This register is also used to select sample time and low-power configuration.
2. Update the configuration register (ADC_SC4) to select the FIFO scan mode, FIFO compare function selection (OR or AND function) and FIFO depth.
3. Update status and control register 2 (ADC_SC2) to select the hardware or software conversion trigger, compare function options if enabled.
4. Update status and control register 1 (ADC_SC1) to select whether conversions will be continuous or completed only once, and to enable or disable conversion complete interrupts. The input channel on which conversions will be performed is also selected here.

17.6.2.1 Pseudo-code example

In this example, the ADC module is set up with interrupts enabled to perform a single hardware triggered 10-bit 4-level-FIFO conversion at low power with a long sample time on input channels of 1, 3, 5, and 7. Here the internal ADCK clock is derived from the bus clock divided by 1.

Example: 17.6.2.1.1 FIFO ADC initialization routine

```
void ADC_init(void)
{
/* The following code segment demonstrates how to initialize ADC by low-power mode, long
sample time, bus frequency, hardware triggered from AD1, AD3, AD5, and AD7 external pins
with 4-level FIFO enabled */

ADC_SC3 = ADC_SC3_ADLPC_MASK | ADC_SC3_ADLSMP_MASK | ADC_SC3_MODE1_MASK;


// setting hardware trigger
ADC_SC2 = ADC_SC2_ADTRG_MASK ;

//4-Level FIFO
ADC_SC4 = ADC_SC4_AFDEP1_MASK | ADC_SC4_AFDEP0_MASK;

// dummy the 1st channel
ADC_SC1 = ADC_SC1_ADCH0_MASK;

// dummy the 2nd channel
ADC_SC1 = ADC_SC1_ADCH1_MASK | ADC_SC1_ADCH0_MASK;

// dummy the 3rd channel
ADC_SC1 = ADC_SC1_ADCH2_MASK | ADC_SC1_ADCH0_MASK;

// dummy the 4th channel and ADC starts conversion
ADC_SC1 = ADC_SC1_AIEN_MASK | ADC_SC1_ADCH2_MASK | ADC_SC1_ADCH1_MASK | ADC_SC1_ADCH0_MASK;

}
```

Example: 17.6.2.1.2 FIFO ADC interrupt service routine

```
unsigned short buffer[4];
interrupt VectorNumber_Vadc void ADC_isr(void)
{
      /* The following code segment demonstrates read AD result FIFO */
      // read conversion result of channel 1 and COCO bit is cleared
      buffer[0] = ADC_R;
      // read conversion result of channel 3
      buffer[1] = ADC_R;
      // read conversion result of channel 5
      buffer[2] = ADC_R;
      // read conversion result of channel 7
      buffer[3] = ADC_R;
}
```

NOTE

ADC_R is 16-bit ADC result register, combined from ADC_RH and ADC_RL

17.7 Application information

This section contains information for using the ADC module in applications. The ADC has been designed to be integrated into a microcontroller for use in embedded control applications requiring an A/D converter.

17.7.1 External pins and routing

The following sections discuss the external pins associated with the ADC module and how they are used for best results.

17.7.1.1 Analog supply pins

The ADC module has analog power and ground supplies (V_{DDA} and V_{SSA}) available as separate pins on some devices. V_{SSA} is shared on the same pin as the MCU digital V_{SS} on some devices. On other devices, V_{SSA} and V_{DDA} are shared with the MCU digital supply pins. In these cases, there are separate pads for the analog supplies bonded to the same pin as the corresponding digital supply so that some degree of isolation between the supplies is maintained.

When available on a separate pin, both V_{DDA} and V_{SSA} must be connected to the same voltage potential as their corresponding MCU digital supply (V_{DD} and V_{SS}) and must be routed carefully for maximum noise immunity and bypass capacitors placed as near as possible to the package.

If separate power supplies are used for analog and digital power, the ground connection between these supplies must be at the V_{SSA} pin. This should be the only ground connection between these supplies if possible. The V_{SSA} pin makes a good single point ground location.

17.7.1.2 Analog reference pins

In addition to the analog supplies, the ADC module has connections for two reference voltage inputs. The high reference is V_{REFH}, which may be shared on the same pin as V_{DDA} on some devices. The low reference is V_{REFL}, which may be shared on the same pin as V_{SSA} on some devices.

When available on a separate pin, V_{REFH} may be connected to the same potential as V_{DDA}, or may be driven by an external source between the minimum V_{DDA} spec and the V_{DDA} potential (V_{REFH} must never exceed V_{DDA}). When available on a separate pin, V_{REFL} must be connected to the same voltage potential as V_{SSA}. V_{REFH} and V_{REFL} must be routed carefully for maximum noise immunity and bypass capacitors placed as near as possible to the package.

AC current in the form of current spikes required to supply charge to the capacitor array at each successive approximation step is drawn through the V_{REFH} and V_{REFL} loop. The best external component to meet this current demand is a 0.1 μF capacitor with good high frequency characteristics. This capacitor is connected between V_{REFH} and V_{REFL} and must be placed as near as possible to the package pins. Resistance in the path is not recommended because the current causes a voltage drop that could result in conversion errors. Inductance in this path must be minimum (parasitic only).

17.7.1.3 Analog input pins

The external analog inputs are typically shared with digital I/O pins on MCU devices. The pin I/O control is disabled by setting the appropriate control bit in one of the pin control registers. Conversions can be performed on inputs without the associated pin control register bit set. It is recommended that the pin control register bit always be set when using a pin as an analog input. This avoids problems with contention because the output buffer is in its high impedance state and the pullup is disabled. Also, the input buffer draws DC current when its input is not at V_{DD} or V_{SS}. Setting the pin control register bits for all pins used as analog inputs should be done to achieve lowest operating current.

Empirical data shows that capacitors on the analog inputs improve performance in the presence of noise or when the source impedance is high. Use of 0.01 μF capacitors with good high-frequency characteristics is sufficient. These capacitors are not necessary in all cases, but when used they must be placed as near as possible to the package pins and be referenced to V_{SSA}.

For proper conversion, the input voltage must fall between V_{REFH} and V_{REFL}. If the input is equal to or exceeds V_{REFH}, the converter circuit converts the signal to 0xFFF (full scale 12-bit representation), 0x3FF (full scale 10-bit representation) or 0xFF (full scale 8-bit representation). If the input is equal to or less than V_{REFL}, the converter circuit converts it to 0x000. Input voltages between V_{REFH} and V_{REFL} are straight-line linear conversions. There is a brief current associated with V_{REFL} when the sampling capacitor is charging. The input is sampled for 3.5 cycles of the ADCK source when ADC_SC3[ADLSMP] is low, or 23.5 cycles when ADC_SC3[ADLSMP] is high.

For minimal loss of accuracy due to current injection, pins adjacent to the analog input pins should not be transitioning during conversions.

17.7.2 Sources of error

Several sources of error exist for A/D conversions. These are discussed in the following sections.

17.7.2.1 Sampling error

For proper conversions, the input must be sampled long enough to achieve the proper accuracy. Given the maximum input resistance of approximately 7 kΩ and input capacitance of approximately 5.5 pF, sampling to within 1/4 LSB (at 12-bit resolution) can be achieved within the minimum sample window (3.5 cycles at 8 MHz maximum ADCK frequency) provided the resistance of the external analog source (R_{AS}) is kept below 2 kΩ.

Higher source resistances or higher-accuracy sampling is possible by setting ADC_SC3[ADLSMP] (to increase the sample window to 23.5 cycles) or decreasing ADCK frequency to increase sample time.

17.7.2.2 Pin leakage error

Leakage on the I/O pins can cause conversion error if the external analog source resistance (R_{AS}) is high. If this error cannot be tolerated by the application, keep R_{AS} lower than $V_{DDA} / (2^N * I_{LEAK})$ for less than 1/4 LSB leakage error (N = 8 in 8-bit, 10 in 10-bit or 12 in 12-bit mode).

17.7.2.3 Noise-induced errors

System noise that occurs during the sample or conversion process can affect the accuracy of the conversion. The ADC accuracy numbers are guaranteed as specified only if the following conditions are met:

- There is a 0.1 µF low-ESR capacitor from V_{REFH} to V_{REFL}.
- There is a 0.1 µF low-ESR capacitor from V_{DDA} to V_{SSA}.

- If inductive isolation is used from the primary supply, an additional 1 µF capacitor is placed from V_{DDA} to V_{SSA}.
- V_{SSA} (and V_{REFL}, if connected) is connected to V_{SS} at a quiet point in the ground plane.
- Operate the MCU in wait or Stop mode before initiating (hardware triggered conversions) or immediately after initiating (hardware or software triggered conversions) the ADC conversion.
  - For software triggered conversions, immediately follow the write to ADC_SC1 with a wait instruction or stop instruction.
  - For Stop mode operation, select ADACK as the clock source. Operation in Stop reduces V_{DD} noise but increases effective conversion time due to stop recovery.
- There is no I/O switching, input or output, on the MCU during the conversion.

There are some situations where external system activity causes radiated or conducted noise emissions or excessive V_{DD} noise is coupled into the ADC. In these situations, or when the MCU cannot be placed in wait or Stop or I/O activity cannot be halted, these recommended actions may reduce the effect of noise on the accuracy:

- Place a 0.01 µF capacitor (C_{AS}) on the selected input channel to V_{REFL} or V_{SSA} (this improves noise issues, but affects the sample rate based on the external analog source resistance).
- Average the result by converting the analog input many times in succession and dividing the sum of the results. Four samples are required to eliminate the effect of a 1LSB, one-time error.
- Reduce the effect of synchronous noise by operating off the asynchronous clock (ADACK) and averaging. Noise that is synchronous to ADCK cannot be averaged out.

17.7.2.4 Code width and quantization error

The ADC quantizes the ideal straight-line transfer function into 4096 steps (in 12-bit mode). Each step ideally has the same height (1 code) and width. The width is defined as the delta between the transition points to one code and the next. The ideal code width for an N bit converter (in this case N can be 8, 10 or 12), defined as 1LSB, is:

$$1\text{lsb} = \left(V_{REFH} - V_{REFL}\right) / \ 2^N$$

There is an inherent quantization error due to the digitization of the result. For 8-bit or 10-bit conversions the code transitions when the voltage is at the midpoint between the points where the straight line transfer function is exactly represented by the actual transfer function. Therefore, the quantization error will be ± 1/2 lsb in 8- or 10-bit mode. As a consequence, however, the code width of the first (0x000) conversion is only 1/2 lsb and the code width of the last (0xFF or 0x3FF) is 1.5 lsb.

For 12-bit conversions the code transitions only after the full code width is present, so the quantization error is -1 lsb to 0 lsb and the code width of each step is 1 lsb.

17.7.2.5 Linearity errors

The ADC may also exhibit non-linearity of several forms. Every effort has been made to reduce these errors but the system must be aware of them because they affect overall accuracy. These errors are:

- Zero-scale error (E_{ZS}) (sometimes called offset) — This error is defined as the difference between the actual code width of the first conversion and the ideal code width (1/2 lsb in 8-bit or 10-bit modes and 1 lsb in 12-bit mode). If the first conversion is 0x001, the difference between the actual 0x001 code width and its ideal (1 lsb) is used.
- Full-scale error (E_{FS}) — This error is defined as the difference between the actual code width of the last conversion and the ideal code width (1.5 lsb in 8-bit or 10-bit modes and 1LSB in 12-bit mode). If the last conversion is 0x3FE, the difference between the actual 0x3FE code width and its ideal (1 lsb) is used.
- Differential non-linearity (DNL) — This error is defined as the worst-case difference between the actual code width and the ideal code width for all conversions.
- Integral non-linearity (INL) — This error is defined as the highest-value that the absolute value of the running sum of DNL achieves. More simply, this is the worst-case difference of the actual transition voltage to a given code and its corresponding ideal transition voltage, for all codes.
- Total unadjusted error (TUE) — This error is defined as the difference between the actual transfer function and the ideal straight-line transfer function and includes all forms of error.

17.7.2.6 Code jitter, non-monotonicity, and missing codes

Analog-to-digital converters are susceptible to three special forms of error. These are code jitter, non-monotonicity, and missing codes.

Code jitter occurs when, at certain points, a given input voltage converts to one of two values when sampled repeatedly. Ideally, when the input voltage is infinitesimally smaller than the transition voltage, the converter yields the lower code (and vice-versa). However, even small amounts of system noise can cause the converter to be indeterminate, between two codes, for a range of input voltages around the transition voltage. This range is normally around ±1/2 lsb in 8-bit or 10-bit mode, or around 2 lsb in 12-bit mode, and increases with noise.

This error may be reduced by repeatedly sampling the input and averaging the result. Additionally the techniques discussed in Noise-induced errors reduces this error.

Non-monotonicity is defined when, except for code jitter, the converter converts to a lower code for a higher input voltage. Missing codes are those values that are never converted for any input value.

In 8-bit or 10-bit mode, the ADC is guaranteed to be monotonic and have no missing codes.

Chapter 18
Chip-specific ACMP information

Chip-specific ACMP information

18.1 CMP configuration information

The CMP features eight different inputs muxed with both positive and negative inputs to the CMP. One is fixed connected to built-in 6-bit DAC output. Other inputs are internally and externally mapped on GDU outputs and pinouts. Built-in 6-bit DAC output to GDU as possible the reference of GDU phase detection comparator. The following table shows the connection of CMP input assignments.

The CMP continues to operate in wait and stop modes if enabled and its interrupt can wake the MCU.

Customization:

- Primary clock: BUSCLK (20 MHz)
- Alternate clock: None
- ACMP supply uses VDDX and VSS
- 6-bit DAC reference are Vin1 to VREFH (a PMC high accuracy reference) and Vin2 to VDDX
- The following table summarizes the signal connection of CMP module.

Table 18-1. CMP module signals Connection

Module	Signal	Connect to
CMP	CH0	PTA0/CMP0
	CH1	PTA1/CMP1
	CH2	PTA2/CMP2
	CH3	6-bit DAC
	6-bit DAC	To GDU

Table continues on the next page...

Table 18-1. CMP module signals Connection (continued)

Module	Signal	Connect to
	Window/sample	XBAR_OUT13
	6-bit DAC Vin1	VREFH (a PMC high accuracy reference)
	6-bit DAC Vin2	VDDX
	Internal clock	BUSCLK

Module Instances:

- One

18.2 ACMP in stop mode

ACMP continues to operate in stop mode if enabled. If ACMP_CS[ACOPE] is enabled, comparator output will operate as in the normal operating mode. The MCU is brought out of stop when a compare event occurs and ACMP_CS[ACIE] is enabled; ACF flag sets accordingly.

18.4 Introduction

The Comparator module (CMP) provides a circuit for comparing two analog input voltages. The comparator circuit is designed to operate across the full range of the supply voltage (rail to rail operation).

The Analog MUX (ANMUX) provides a circuit for selecting an analog input signal from four channels. One signal provided by the 6-bit DAC. The mux circuit is designed to operate across the full range of the supply voltage.

The 6-bit DAC is 64-tap resistor ladder network which provides a selectable voltage reference for applications where voltage reference is needed. The 64-tap resistor ladder network divides the supply reference Vin into 64 voltage level. A 6-bit digital signal input selects output voltage level, which varies from Vin to Vin/64. Vin can be selected from two voltage sources, V_{in1} and V_{in2}. The 6-bit DAC from a comparator is available as an on-chip internal signal only and is not available externally to a pin.

18.5 CMP Features

The CMP has the following features:

- Operates over the entire supply range
- Inputs may range from rail to rail
- Programmable hysteresis control
- Selectable interrupt on rising edge, falling edge, or both rising or falling edges of comparator output
- Selectable inversion on comparator output
- Comparator output may be:
  - Sampled
  - Windowed (ideal for certain PWM zero-crossing-detection applications)
  - Digitally Filtered
    - Filter can be bypassed
    - Can be clocked via external SAMPLE signal or scaled bus clock
- External hysteresis can be used at the same time that the output filter is used for internal functions.
- Two software selectable performance levels:
  - Shorter propagation delay at the expense of higher power
  - Low power, with longer propagation delay
- Functional in all modes of operation.
- The window and filter functions are not available in Stop mode.

18.6 6-bit DAC Key Features

- 6-bit resolution
- Selectable supply reference source
- Power down mode to conserve power when it is not being used
- Output can be routed to internal comparator input

18.7 ANMUX Key Features

- Two 4 to 1 channel mux
- Operates the entire supply range

18.8 CMP, DAC, and ANMUX Diagram

The following figure shows the block diagram for the High Speed Comparator, Digital to Analog Converter, and Analog MUX modules.

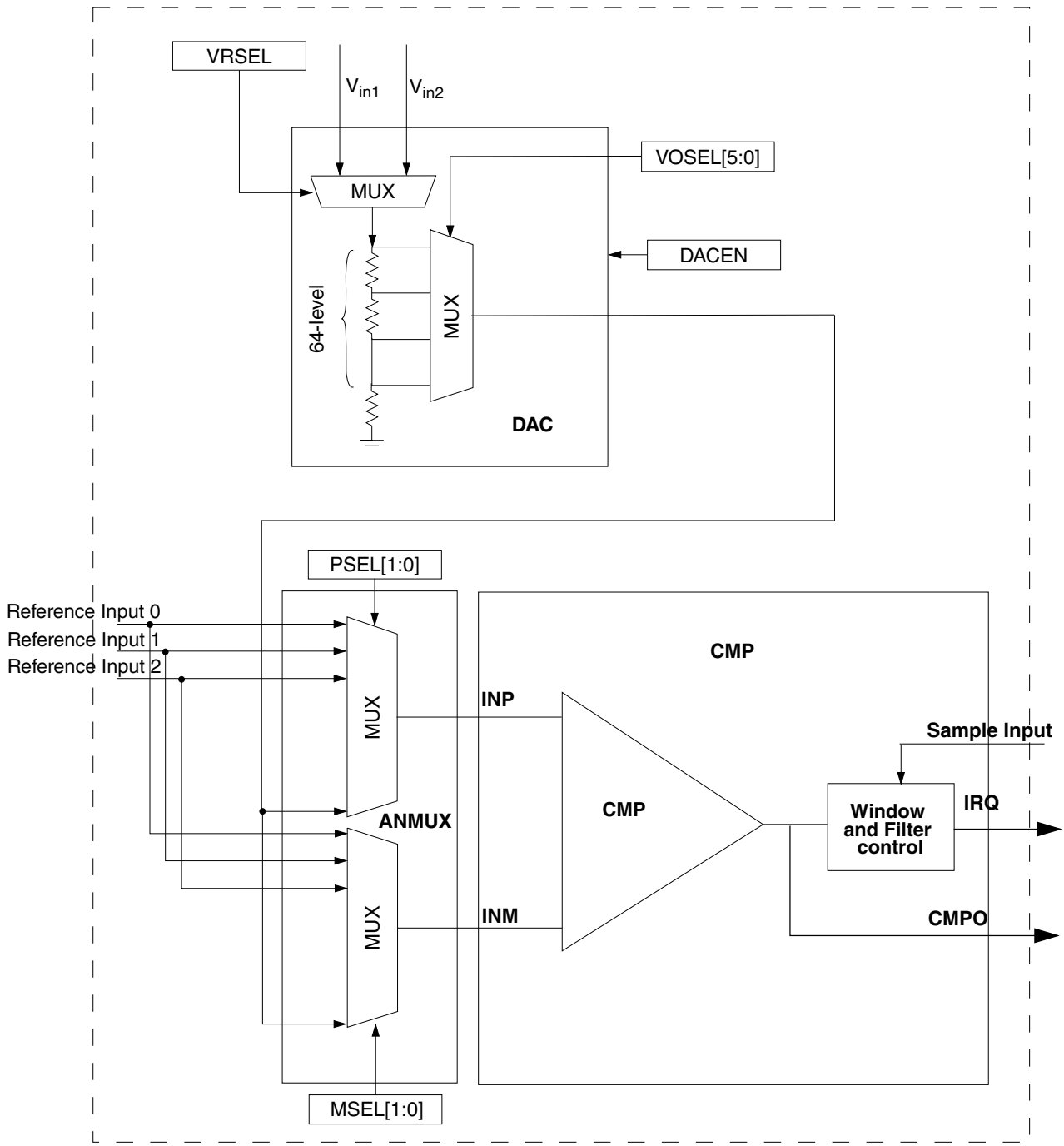


Figure 18-1. CMP, DAC and ANMUX Blocks Diagram

18.9 CMP Block Diagram

The following figure shows the block diagram for the Comparator module.

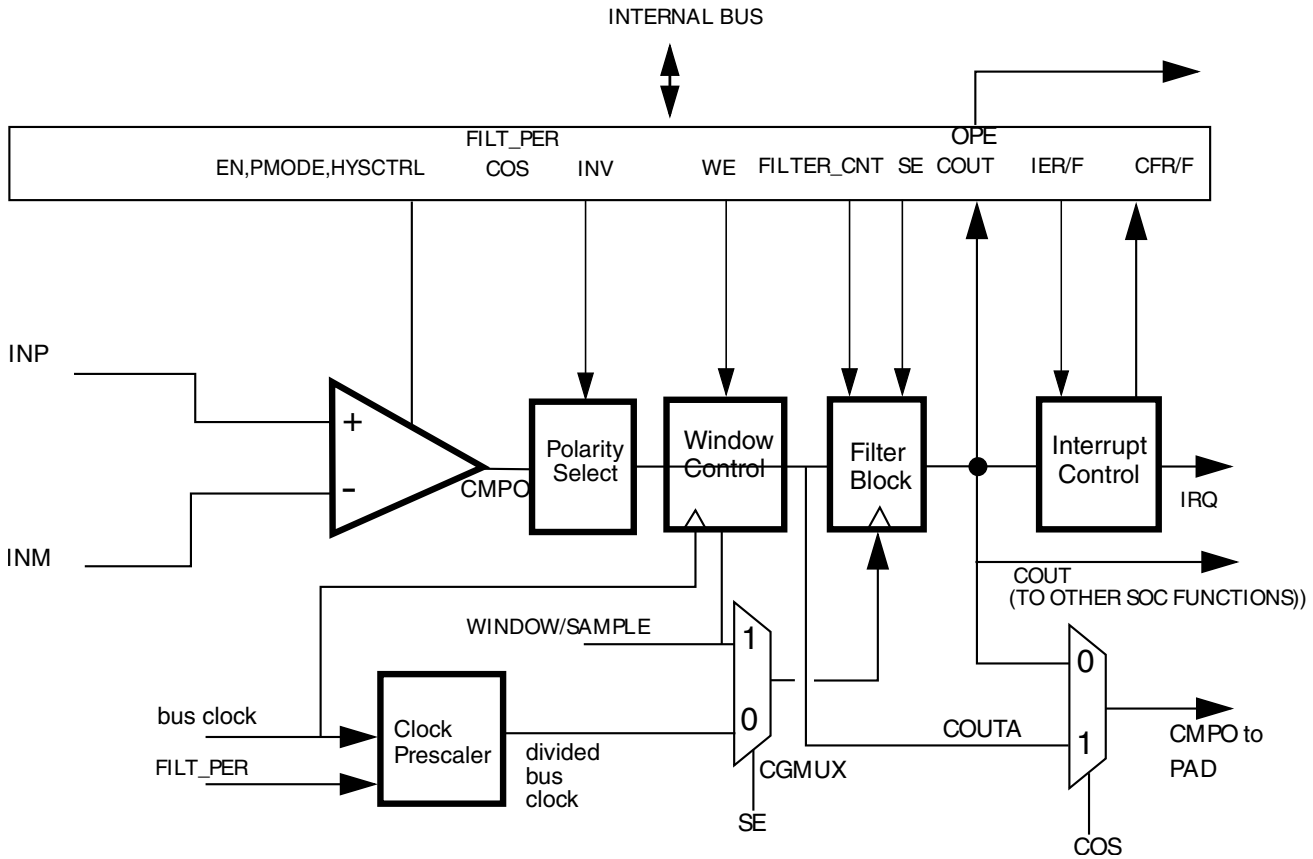


Figure 18-2. Comparator Module Block Diagram

In the CMP block diagram:

- The Window Control block is bypassed when CR1[WE] = 0
- If CR1[WE] = 1, the comparator output will be sampled on every bus clock when WINDOW=1 to generate COUTA. Sampling does NOT occur when WINDOW = 0.
- The Filter Block is bypassed when not in use.
- The Filter Block acts as a simple sampler if the filter is bypassed and CR0[FILTER_CNT] is set to 0x01.
- The Filter Block filters based on multiple samples when the filter is bypassed and CR0[FILTER_CNT] is set greater than 0x01.
  - If CR1[SE] = 1, the external SAMPLE input is used as sampling clock
  - IF CR1[SE] = 0, the divided bus clock is used as sampling clock

- If enabled, the Filter Block will incur up to 1 bus clock additional latency penalty on COUT due to the fact that COUT (which is crossing clock domain boundaries) must be resynchronized to the bus clock.
- CR1[WE] and CR1[SE] are mutually exclusive.

18.10 Memory Map/Register Definitions

Address offsets are in terms of bytes for the S08 architecture.

CMP memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
68	CMP Control Register 0 (CMP_CR0)	8	R/W	00h	18.10.1/295
69	CMP Control Register 1 (CMP_CR1)	8	R/W	00h	18.10.2/296
6A	CMP Filter Period Register (CMP_FPR)	8	R/W	00h	18.10.3/297
6B	CMP Status and Control Register (CMP_SCR)	8	R/W	00h	18.10.4/298
6C	DAC Control Register (CMP_DACCR)	8	R/W	00h	18.10.5/299
6D	MUX Control Register (CMP_MUXCR)	8	R/W	00h	18.10.6/300
6E	MUX Pin Enable Register (CMP_MUXPE)	8	R/W	00h	18.10.7/301

18.10.1 CMP Control Register 0 (CMP_CR0)

Address: 68h base + 0h offset = 68h

Bit	7	6	5	4	3	2	1	0
Read	0	FILTER_CNT			0			HYSTCTR
Write								
Reset	0	0	0	0	0	0	0	0

CMP_CR0 field descriptions

Field	Description
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6–4 FILTER_CNT	Filter Sample Count These bits represent the number of consecutive samples that must agree prior to the comparator ouput filter accepting a new output state. For information regarding filter programming and latency reference the Functional Description.

Table continues on the next page...

CMP_CR0 field descriptions (continued)

Field	Description
	000 Filter is disabled. If SE = 1, then COUT is a logic zero (this is not a legal state, and is not recommended). If SE = 0, COUT = COUTA. 001 1 consecutive sample must agree (comparator output is simply sampled). 010 2 consecutive samples must agree. 011 3 consecutive samples must agree. 100 4 consecutive samples must agree. 101 5 consecutive samples must agree. 110 6 consecutive samples must agree. 111 7 consecutive samples must agree.
3–1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 HYSTCTR	Comparator hard block hysteresis control Defines the programmable hysteresis level. The hysteresis values associated with each level is device-specific. See the device's data sheet for the exact values. 0 Level 0 1 Level 1

18.10.2 CMP Control Register 1 (CMP_CR1)

Address: 68h base + 1h offset = 69h

Bit	7	6	5	4	3	2	1	0
Read Write	SE	WE	0	PMODE	INV	COS	OPE	EN
Reset	0	0	0	0	0	0	0	0

CMP_CR1 field descriptions

Field	Description
7 SE	Sample Enable At any given time, either SE or WE can be set. If a write to this register attempts to set both, then SE is set and WE is cleared. However, avoid writing ones to both bit locations because this "11" case is reserved and may change in future implementations. 0 Sampling mode not selected. 1 Sampling mode selected.
6 WE	Windowing Enable At any given time, either SE or WE can be set. If a write to this register attempts to set both, then SE is set and WE is cleared. However, avoid writing ones to both bit locations because this "11" case is reserved and may change in future implementations. 0 Windowing mode not selected. 1 Windowing mode selected.

Table continues on the next page...

CMP_CR1 field descriptions (continued)

Field	Description
5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
4 PMODE	Power Mode Select 0 Low Speed (LS) comparison mode selected. 1 High Speed (HS) comparison mode selected.
3 INV	Comparator INVERT This bit allows you to select the polarity of the analog comparator function. It is also driven to the COUT output (on both the device pin and as SCR[COUT]) when CR1[OPE]=0. 0 Does not invert the comparator output. 1 Inverts the comparator output.
2 COS	Comparator Output Select 0 Set CMPO to equal COUT (filtered comparator output). 1 Set CMPO to equal COUTA (unfiltered comparator output).
1 OPE	Comparator Output Pin Enable **NOTE:** This bit is always disabled in the device, writing 1 to this bit does not take effect. 0 The comparator output (CMPO) is not available on the associated CMPO output pin. Instead, the INV bit is driven if the comparator owns the pin (usually a result of properly setting pin mux controls at the SoC level). If the comparator does not own the pin, this bit has no effect. The pin is available for use by other on-chip functions. 1 The comparator output (CMPO) is available on the associated CMPO output pin. The comparator output (CMPO) is driven out on the associated CMPO output pin if the comparator owns the pin. If the comparator does not own the pin, this bit has no effect. The comparator output (CMPO) is driven out on the associated CMPO output pin.
0 EN	Comparator Module Enable The EN bit enables the Analog Comparator Module. When the module is not enabled, it remains in the off state, and consumes no power. When you select the same input from analog mux to the positive and negative port, the comparator is disabled automatically. 0 Analog Comparator disabled. 1 Analog Comparator enabled.

18.10.3 CMP Filter Period Register (CMP_FPR)

Address: 68h base + 2h offset = 6Ah

Bit	7	6	5	4	3	2	1	0
Read Write	FILT_PER							
Reset	0	0	0	0	0	0	0	0

CMP_FPR field descriptions

Field	Description
FILT_PER	Filter Sample Period When CR1[SE] is equal to zero, this field specifies the sampling period, in bus clock cycles, of the comparator output filter. Setting FILT_PER to 0x0 disables the filter. Filter programming and latency details appear in the Functional Description. This field has no effect when CR1[SE] is equal to one. In that case, the external SAMPLE signal is used to determine the sampling period.

18.10.4 CMP Status and Control Register (CMP_SCR)

Address: 68h base + 3h offset = 6Bh

Bit	7	6	5	4	3	2	1	0
Read	0		0	IER	IEF	CFR	CFF	COUT
Write						w1c	w1c	
Reset	0	0	0	0	0	0	0	0

CMP_SCR field descriptions

Field	Description
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
4 IER	Comparator Interrupt Enable Rising The IER bit enables the CFR interrupt from the CMP. When this bit is set, an interrupt will be asserted when the CFR bit is set. 0 Interrupt disabled. 1 Interrupt enabled.
3 IEF	Comparator Interrupt Enable Falling The IEF bit enables the CFF interrupt from the CMP. When this bit is set, an interrupt will be asserted when the CFF bit is set. 0 Interrupt disabled. 1 Interrupt enabled.
2 CFR	Analog Comparator Flag Rising During normal operation, the CFR bit is set when a rising edge on COUT has been detected. In STOP mode, CFR is level sensitive and as long as COUT is high CFR will be set. The CFR bit is cleared by writing a logic one to the bit. 0 Rising edge on COUT has not been detected. 1 Rising edge on COUT has occurred.
1 CFF	Analog Comparator Flag Falling

Table continues on the next page...

CMP_SCR field descriptions (continued)

Field	Description
	During normal operation, the CFF bit is set when a falling edge on COUT has been detected. In STOP mode, CFF is level senstive and as long as the COUT is low CFF will be set in STOP mode. The CFF bit is cleared by writing a logic one to the bit. 0 Falling edge on COUT has not been detected. 1 Falling edge on COUT has occurred.
0 COUT	Analog Comparator Output Reading the COUT bit will return the current value of the analog comparator output. The register bit is reset to zero and will read as CR1[INV] when the Analog Comparator module is disabled (CR1[EN] = 0). Writes to this bit are ignored.

18.10.5 DAC Control Register (CMP_DACCR)

Address: 68h base + 4h offset = 6Ch

Bit	7	6	5	4	3	2	1	0
Read Write	DACEN	VRSEL	VOSEL					
Reset	0	0	0	0	0	0	0	0

CMP_DACCR field descriptions

Field	Description
7 DACEN	DAC Enable This bit is used to enable the DAC. When the DAC is disabled, it is powered down to conserve power. 0 DAC is disabled. 1 DAC is enabled.
6 VRSEL	Supply Voltage Reference Source Select 0 V_{in1} is selected as resistor ladder network supply reference Vin. 1 V_{in2} is selected as resistor ladder network supply reference Vin.
VOSEL	DAC Output Voltage Select This bit selects an output voltage from one of 64 distinct levels. `DACO = (Vin/64) * (VOSEL[5:0] + 1)`, so the DACO range is from Vin/64 to Vin.

18.10.6 MUX Control Register (CMP_MUXCR)

NOTE

PEN and MEN bits should be enabled or disabled together with CR1[EN] bit.

Address: 68h base + 5h offset = 6Dh

Bit	7	6	5	4	3	2	1	0
Read	0		PSEL		0		MSEL	
Write								
Reset	0	0	0	0	0	0	0	0

CMP_MUXCR field descriptions

Field	Description
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5–4 PSEL	Plus Input MUX Control Determines which input is selected for the plus input of the comparator. For INx inputs, refer to CMP, DAC and ANMUX Blocks Diagram. **NOTE:** When an inappropriate operation selects the same input for both MUXes, the comparator automatically shuts down to prevent itself from becoming a noise generator. 00 IN0 01 IN1 10 IN2 11 6-bit DAC output is selected
3–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
MSEL	Minus Input MUX Control Determines which input is selected for the minus input of the comparator. For INx inputs, refer to CMP, DAC and ANMUX Blocks Diagram. **NOTE:** When an inappropriate operation selects the same input for both MUXes, the comparator automatically shuts down to prevent itself from becoming a noise generator. 00 IN0 01 IN1 10 IN2 11 6-bit DAC output is selected

18.10.7 MUX Pin Enable Register (CMP_MUXPE)

This register requests static ownership of a given package pin by the ANMUX. The MUXPE must be programmed to enable ANMUX ownership of all input pins that may be required by an application. These fields are in addition to MUXCR[PSEL] and MUXCR[MSEL], which control "on the fly" switching between inputs.

Address: 68h base + 6h offset = 6Eh

Bit	7	6	5	4	3	2	1	0
Read	0					INPE		
Write								
Reset	0	0	0	0	0	0	0	0

CMP_MUXPE field descriptions

Field	Description
7–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
INPE	Positive Input Pin Enable XX1 Input pin P0 is required by the ANMUX. X1X Input pin P1 is required by the ANMUX. 1XX Input pin P2 is required by the ANMUX.

18.11 CMP Functional Description

The Comparator can be used to compare two analog input voltages applied to INP and INM. The analog comparator output (CMPO) is high when the non-inverting input is greater than the inverting input, and is low when the non-inverting input is less than the inverting input. This signal can be selectively inverted by setting CR1[INV] = 1.

The SCR[IER], SCR[IEF] bits are used to select the condition which will cause the comparator module to assert an interrupt to the processor. SCR[CFF] is set on a falling edge and SCR[CFR] is set on rising edge of the comparator output. The (optionally filtered) comparator output can be read directly through the SCR[COUT] bit.

18.11.1 CMP Functional Modes

There are three main sub-blocks to the comparator module: the comparator itself, the window function and the filter function. The filter, CR0[FILTER_CNT] can be clocked from an internally or external clock source. Additionally, the filter is programmable with respect to how many samples must agree before a change on the output is registered. In the simplest case, only 1 sample must agree. In this case, the filter acts as a simple sampler.

The external sample input is enabled using CR1[SE]. When set, the output of the comparator is sampled only on rising edges of the sample input.

The "windowing mode" is enabled by setting CR1[WE]. When set, the comparator output is sampled only when the WINDOW input signal is equal to one. This feature can be used to ignore the comparator output during time periods in which the input voltages are not valid. This is especially useful when implementing zero-crossing-detection for certain PWM applications.

The comparator filter and sampling features can be combined as shown in the following table. Individual modes are discussed below.

Table 18-2. Comparator Sample/Filter Controls

Mode #	CR1[EN]	CR1[WE]	CR1[SE]	CR0[FILTER_CNT]	FPR[FILT_PER]	Operation
1	0	X	X	X	X	**Disabled** Refer to the Disabled Mode (# 1).
2A	1	0	0	0x00	X	**Continuous Mode** Refer to the Continuous Mode (#s 2A & 2B).
2B	1	0	0	X	0x00	
3A	1	0	1	0x01	X	**Sampled, Non-Filtered mode** Refer to the Sampled, Non-Filtered Mode (#s 3A & 3B).
3B	1	0	0	0x01	> 0x00	
4A	1	0	1	> 0x01	X	**Sampled, Filtered mode** Refer to the Sampled, Filtered Mode (#s 4A & 4B).
4B	1	0	0	> 0x01	> 0x00	
5A	1	1	0	0x00	X	**Windowed mode** Comparator output is sampled on every rising bus clock edge when SAMPLE=1 to generate COUTA Refer to the Windowed Mode (#s 5A & 5B).
5B	1	1	0	X	0x00	
6	1	1	0	0x01	0x01 - 0xFF	**Windowed/Resampled mode** Comparator output is sampled on every rising bus clock edge when SAMPLE=1 to generate COUTA,

Table continues on the next page...

Table 18-2. Comparator Sample/Filter Controls (continued)

Mode #	CR1[EN]	CR1[WE]	CR1[SE]	CR0[FILTER_CNT]	FPR[FILT_PER]	Operation
						which is then resampled on an interval determined by FILT_PER to generate COUT. Refer to the Windowed/Resampled Mode (# 6).
7	1	1	0	> 0x01	0x01 - 0xFF	**Windowed/Filtered mode** Comparator output is sampled on every rising bus clock edge when SAMPLE=1 to generate COUTA, which is then resampled and filtered to generate COUT. Refer to the Windowed/Filtered Mode (#7).
All other combinations of CR1[EN], CR1[WE], CR1[SE], CR0[FILTER_CNT], and FPR[FILT_PER] are illegal.						

For cases where a comparator is used to drive a fault input (for example, for a motor-control module such as FTM), it should generally be configured to operate in continuous mode, so that an external fault can immediately pass through the comparator to the target fault circuitry.

Note

Filtering and sampling settings should be changed only after setting CR1[SE]=0 and CR0[FILTER_CNT]=0x00. This has the effect of resetting the filter to a known state.

18.11.1.1 Disabled Mode (# 1)

In disabled mode, the analog comparator is non-functional and consumes no power. The output of the analog comparator block (CMPO) is zero in this mode.

18.11.1.2 Continuous Mode (#s 2A & 2B)

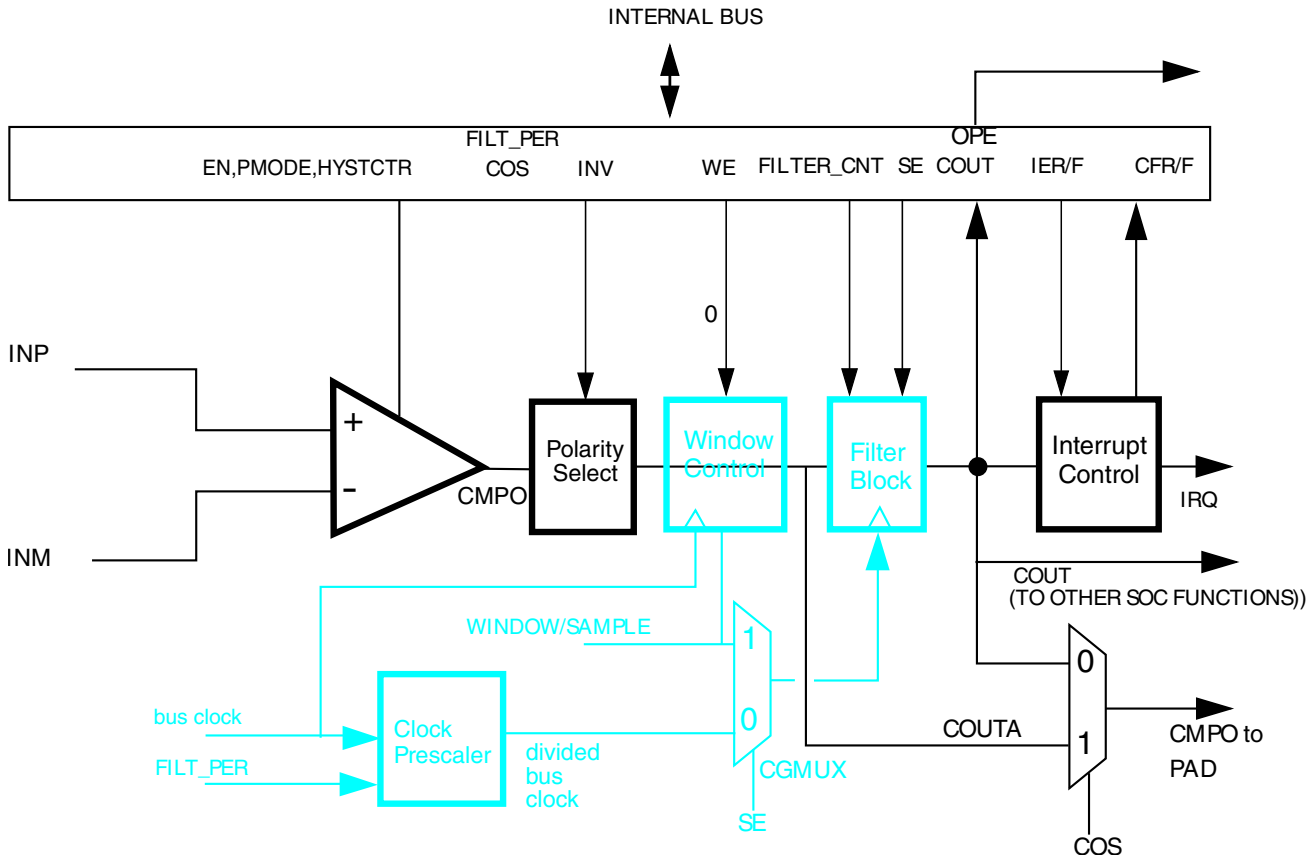


Figure 18-3. Comparator Operation in Continuous Mode

NOTE

Refer to the chip configuration section for the source of sample/ window input.

The analog comparator block is powered and active. CMPO may be optionally inverted, but is not subject to external sampling or filtering. Both Window Control and Filter Blocks are completely bypassed. SCR[COUT] is updated continuously. The path from comparator inputs pins to output pin is operating in combinational (unclocked) mode. COUT and COUTA are identical.

For control configurations which result in disabling the Filter Block, refer to Filter Block Bypass Logic diagram.

18.11.1.3 Sampled, Non-Filtered Mode (#s 3A & 3B)

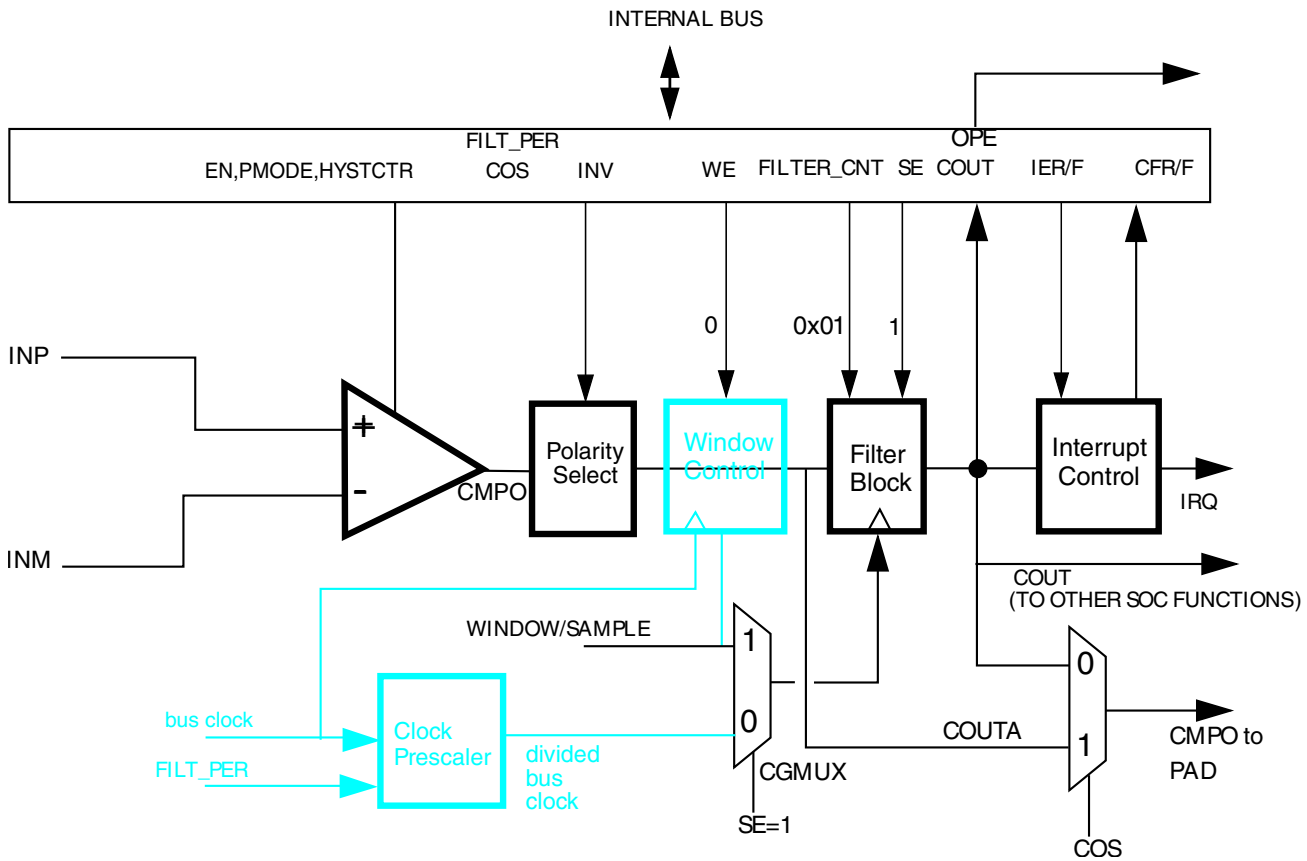


Figure 18-4. Sampled, Non-Filtered (# 3A): Sampling point externally driven

In Sampled, Non-Filtered mode, the analog comparator block is powered and active. The path from analog inputs to COUTA is combinational (unclocked). Windowing Control is completely bypassed. COUTA is sampled whenever a rising edge is detected on the Filter Block clock input.

The only difference in operation between Sampled, Non-Filtered (# 3A) and Sampled, Non-Filtered (# 3B) is in how the clock to the Filter Block is derived. In #3A, the clock to filter block is externally derived while in #3B, the clock to filter block is internally derived.

The comparator filter has no other function than sample/hold of the comparator output in this mode (# 3B).

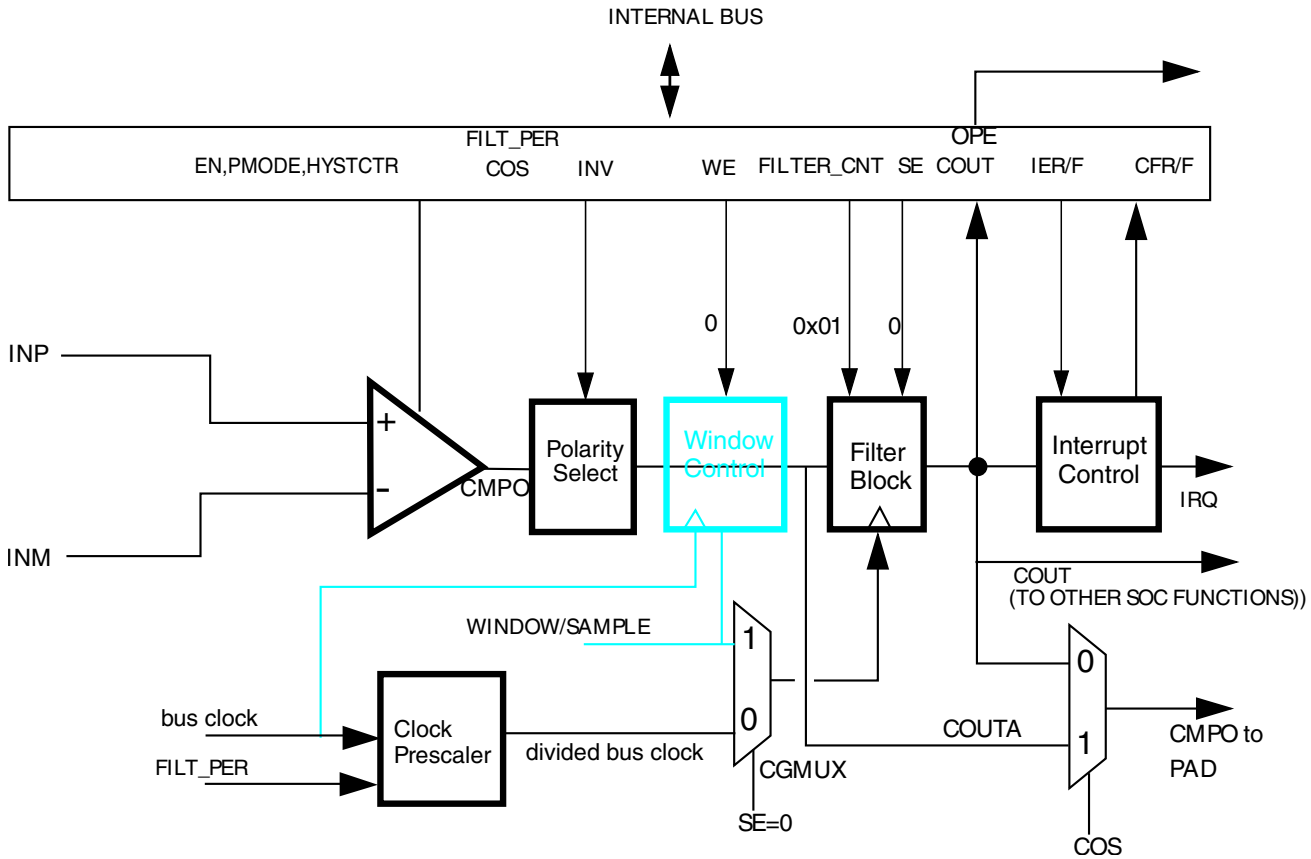


Figure 18-5. Sampled, Non-Filtered (# 3B): Sampling interval internally derived

18.11.1.4 Sampled, Filtered Mode (#s 4A & 4B)

In Sampled, Filtered mode, the analog comparator block is powered and active. The path from analog inputs to COUTA is combinational (unclocked). Windowing Control is completely bypassed. COUTA is sampled whenever a rising edge is detected on the Filter Block clock input.

The only difference in operation between Sampled, Non-Filtered (# 3A) and Sampled, Filtered (# 4A) is that CR0[FILTER_CNT] is now greater than 1, which activates filter operation.

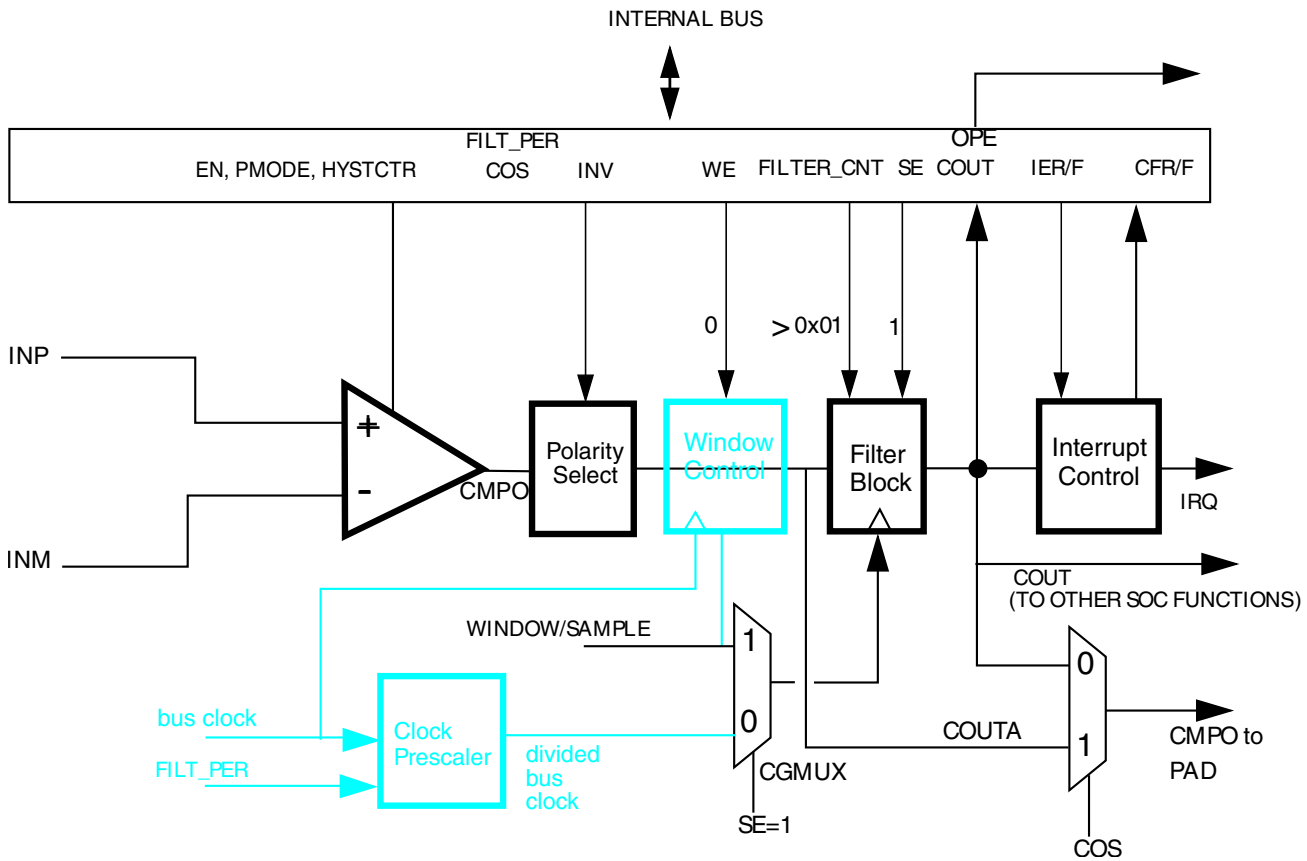


Figure 18-6. Sampled, Filtered (# 4A): Sampling point externally driven

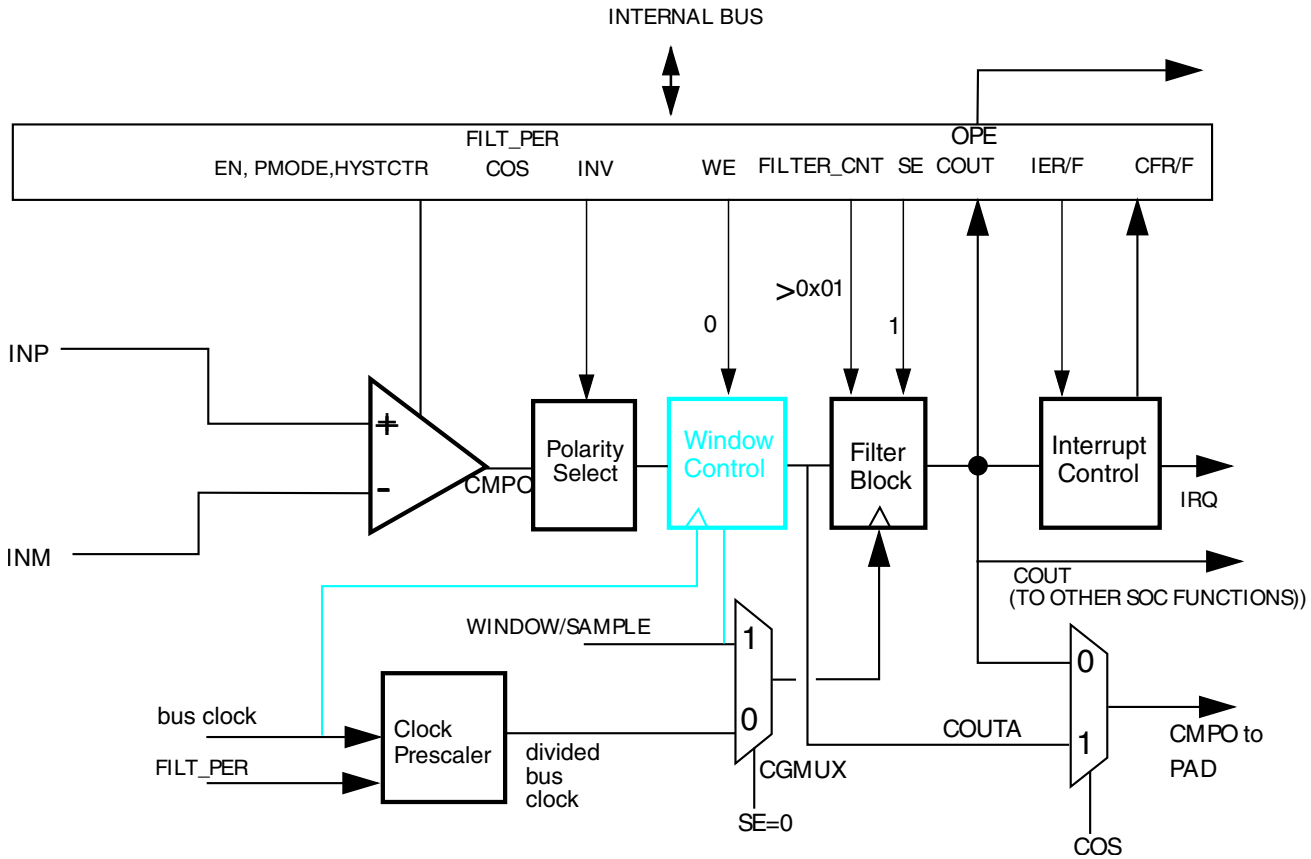


Figure 18-7. Sampled, Filtered (# 4B): Sampling point internally derived

The only difference in operation between Sampled, Non-Filtered (# 3B) and Sampled, Filtered (# 4B) is that CR0[FILTER_CNT] is now greater than 1, which activates filter operation.

18.11.1.5 Windowed Mode (#s 5A & 5B)

The following figure illustrates comparator operation in the windowed mode, ignoring latency of the analog comparator, polarity select and Window Control block. It also assumes that the Polarity Select is set to "non-inverting". Note that the analog comparator output is passed to COUTA only when the WINDOW signal is high.

In actual operation, COUTA may lag the analog inputs by up to one bus clock cycle plus the combinational path delay through the comparator and polarity select logic.

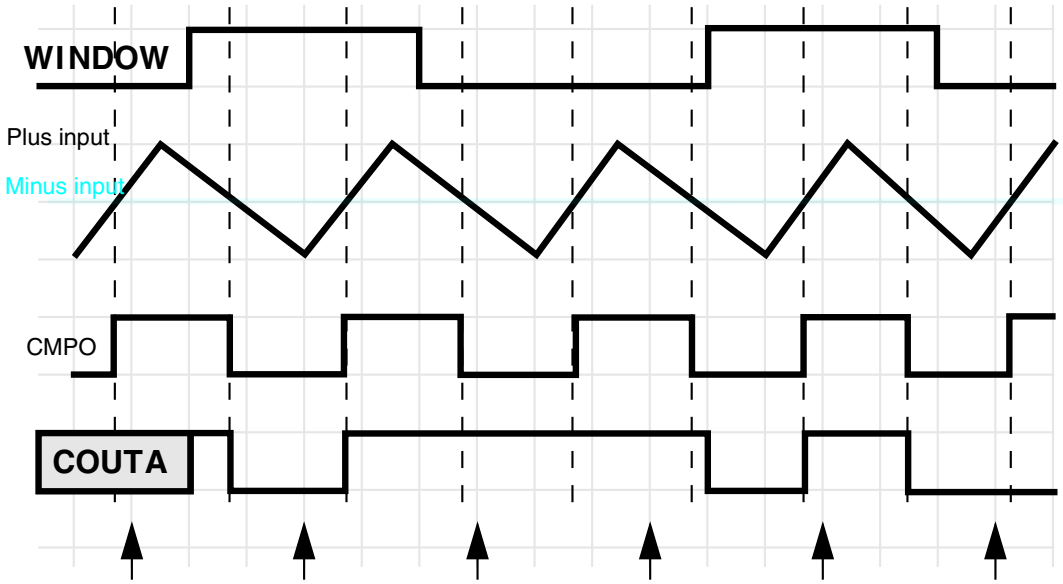


Figure 18-8. Windowed Mode Operation

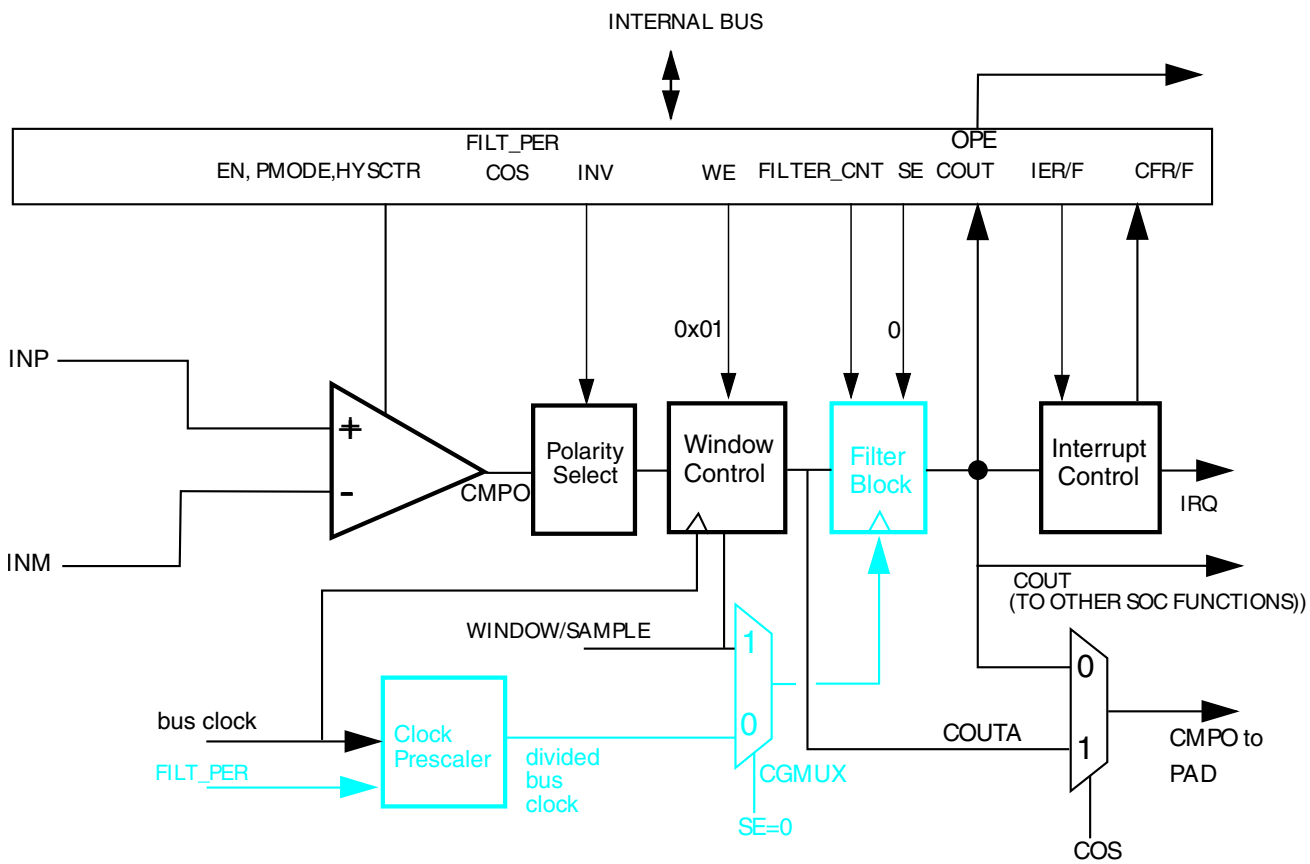


Figure 18-9. Windowed Mode

For control configurations which result in disabling the Filter Block, refer to Filter Block Bypass Logic diagram.

When any windowed mode is active, COUTA is clocked by the bus clock whenever WINDOW = 1. The last latched value is held when WINDOW = 0.

18.11.1.6 Windowed/Resampled Mode (# 6)

The following figure uses the same input stimulus shown in Figure 18-8, and adds resampling of COUTA to generate COUT. Samples are taken at the time points indicated by the arrows. Again, prop delays and latency is ignored for clarity's sake. This example was generated solely to demonstrate operation of the comparator in windowing / resampled mode, and does not reflect any specific application. Depending upon the sampling rate and window placement, COUT may not see zero-crossing events detected by the analog comparator. Sampling period and/or window placement must be carefully considered for a given application.

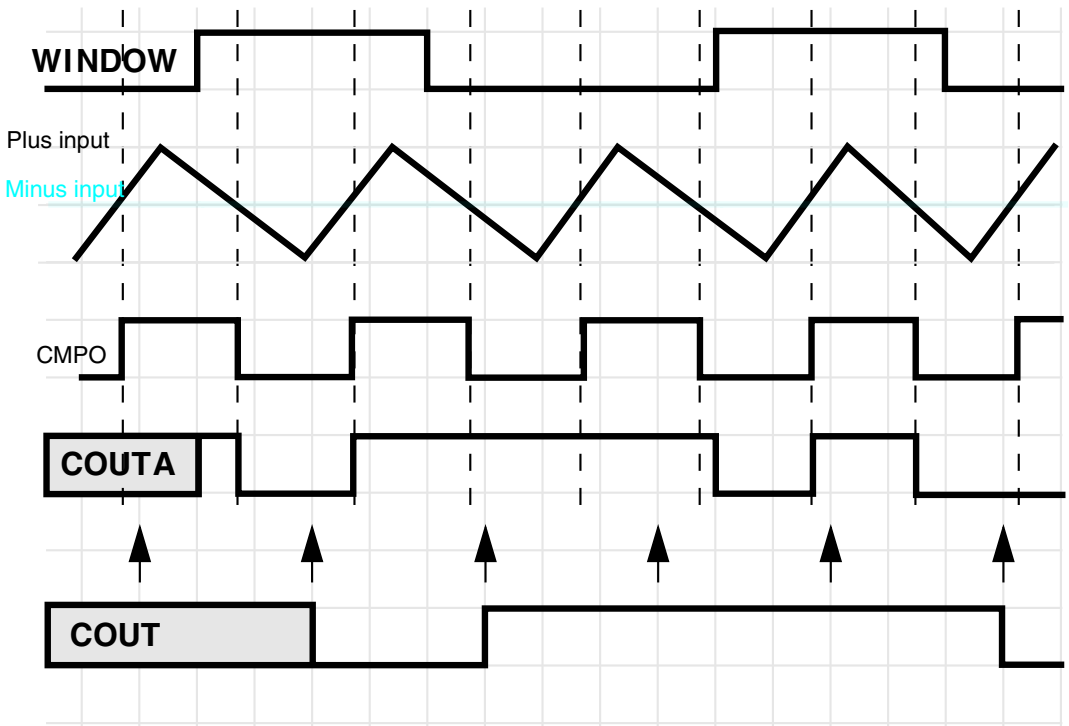


Figure 18-10. Windowed / Resampled Mode Operation

This mode of operation results in an unfiltered string of comparator samples where the interval between the samples is determined by FPR[FILT_PER] and the bus clock rate. Configuration for this mode is virtually identical to that for the Windowed/Filtered Mode shown in the next section. The only difference is that the value of CR0[FILTER_CNT] must be exactly one.

18.11.1.7 Windowed/Filtered Mode (#7)

This is the most complex mode of operation for the comparator block, as it utilizes both windowing and filtering features. It also has the highest latency of any of the modes. This can be approximated: up to 1 bus clock synchronization in the window function + ((CR0[FILTER_CNT] X FPR[FILT_PER]) + 1) X bus clock for the filter function.

When any windowed mode is active, COUTA is clocked by the bus clock whenever WINDOW = 1. The last latched value is held when WINDOW = 0.

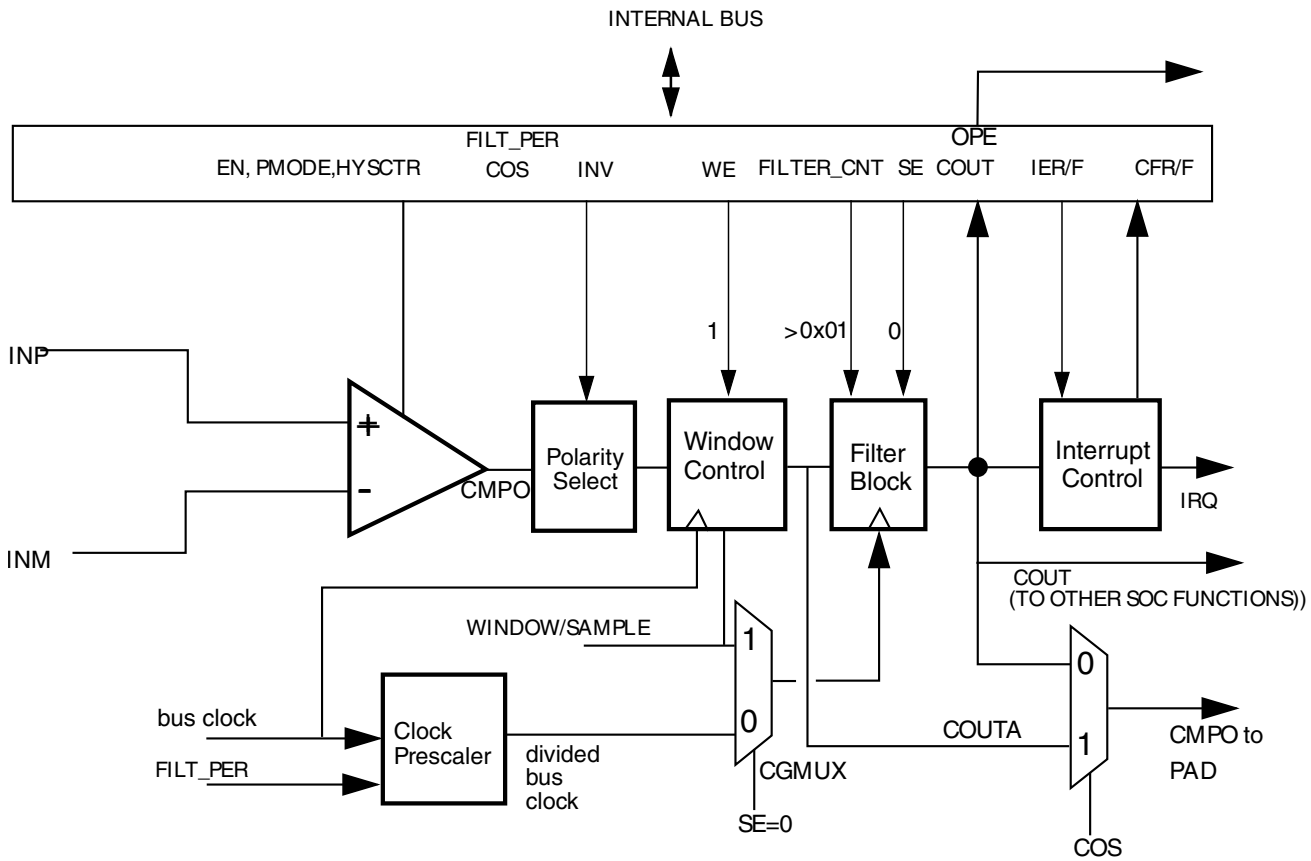


Figure 18-11. Windowed/Filtered Mode

18.11.2 Power Modes

18.11.2.1 Wait Mode Operation

During Wait mode and if enabled, the CMP continues to operate normally. Also, if enabled, a CMP interrupt can wake the MCU.

18.11.2.2 Stop Mode Operation

Subject to platform-specific clock restrictions, the comparator acts with two different behaviors. For platforms which shut off peripheral's clock, none of the comparator window, sample, and filter is available during stop modes. For platforms that can leave the peripherals' clock optional, windowed, sampled, and filtered modes of operation continue to operate in Stop4 and Stop3 modes if the clock to the peripheral is enabled for stop. The MCU is brought out of stop when a compare event occurs and the corresponding interrupt is enabled. Similarly, if CR1[OPE] is enabled, the comparator output operates as in the normal operating mode and comparator output is placed onto the external pin. In stop modes, the comparator can be operational in both high speed (HS) comparison mode (CR1[PMODE] = 1) and low speed (LS) comparison mode (CR1[PMODE] = 0), but it is recommended to use the low speed mode to minimize power consumption.

If stop is exited with a reset, all comparator registers are put into their reset state.

18.11.2.3 Background Debug Mode Operation

When the microcontroller is in active background debug mode, the CMP continues to operate normally.

18.11.3 Startup and Operation

A typical startup sequence is as follows.

The time required to stabilize COUT will be the power-on delay of the comparators plus the largest propagation delay from a selected analog source through the analog comparator, windowing function and filter. Power on delay of the comparators are available from data sheets. The windowing function has a maximum of 1 bus clock period delay. Filter delay is specified in Low Pass Filter.

During operation, the propagation delay of the selected data paths must always be considered. It can take many bus clock cycles for COUT and the CFR/CFF status bits to reflect an input change or a configuration change to one of the components involved in the data path.

When programmed for filtering modes, COUT will initially be equal to zero until sufficient clock cycles have elapsed to fill all stages of the filter. This occurs even if COUTA is at a logic one.

18.11.4 Low Pass Filter

The low-pass filter operates on the unfiltered and unsynchronized and optionally inverted comparator output COUTA and generates the filtered and synchronized output COUT. Both COUTA and COUT can be configured as module outputs and are used for different purposes within the system.

Synchronization and edge detection are always used to determine status register bit values. They also apply to COUT for all sampling and windowed modes. Filtering can be performed using an internal timebase defined by FPR[FILT_PER], or using an external SAMPLE input to determine sample time.

The need for digital filtering and the amount of filtering is dependent on user requirements. Filtering can become more useful in the absence of an external hysteresis circuit. Without external hysteresis, high frequency oscillations can be generated at COUTA when the selected INM and INP input voltages differ by less than the offset voltage of the differential comparator.

18.11.4.1 Enabling Filter Modes

Filter Modes are enabled by setting CR0[FILTER_CNT] greater than 0x01 and (setting FPR[FILT_PER] to a non-zero value OR setting CR1[SE]=1). If using the divided bus clock to drive the filter, it will take samples of COUTA every FPR[FILT_PER] bus clock cycles.

The filter output will be at logic zero when first initalized, and will subsequently change when CR0[FILTER_CNT] consecutive samples all agree that the output value has changed. Said another way, SCR[COUT] will be zero for some initial period, even when COUTA is at logic one.

Setting both CR1[SE] and FPR[FILT_PER] to 0 disables the filter and eliminates switching current associated with the filtering process.

Note

Always switch to this setting prior to making any changes in filter parameters. This resets the filter to a known state. Switching CR0[FILTER_CNT] on the fly without this intermediate step can result in unexpected behavior.

If CR1[SE]=1, the filter takes samples of COUTA on each positive transition of the sample input. The output state of the filter changes when CR0[FILTER_CNT] consecutive samples all agree that the output value has changed.

18.11.4.2 Latency Issues

The FPR[FILT_PER] value (or SAMPLE period) should be set such that the sampling period is just larger than the period of the expected noise. This way a noise spike will only corrupt one sample. The CR0[FILTER_CNT] value should be chosen to reduce the probability of noisy samples causing an incorrect transition to be recognized. The probability of an incorrect transition is defined as the probability of an incorrect sample raised to the CR0[FILTER_CNT] power.

The following table summarizes maximum latency values for the various modes of operation *in the absence of noise*. Filtering latency is restarted each time an actual output transition is masked by noise.

The values of FPR[FILT_PER] (or SAMPLE period) and CR0[FILTER_CNT] must also be traded off against the desire for minimal latency in recognizing actual comparator output transitions. The probability of detecting an actual output change within the nominal latency is the probability of a correct sample raised to the CR0[FILTER_CNT] power.

Table 18-3. Comparator Sample/Filter Maximum Latencies

Mode #	CR1[EN]	CR1[WE]	CR1[SE]	CR0[FILTER_CNT]	FPR[FILT_PER]	Operation	Maximum Latency[1]
1	0	X	X	X	X	Disabled	N/A
2A	1	0	0	0x00	X	Continuous Mode	T_{PD}
2B	1	0	0	X	0x00		
3A	1	0	1	0x01	X	Sampled, Non-Filtered mode	$T_{PD} + T_{SAMPLE} + T_{per}$
3B	1	0	0	0x01	> 0x00		T_{PD} + (FPR[FILT_PER] x T_{per}) + T_{per}
4A	1	0	1	> 0x01	X	Sampled, Filtered mode	T_{PD} + (CR0[FILTER_CNT] x T_{SAMPLE}) + T_{per}
4B	1	0	0	> 0x01	> 0x00		T_{PD} + (CR0[FILTER_CNT] x FPR[FILT_PER] x T_{per}) + T_{per}
5A	1	1	0	0x00	X	Windowed mode	$T_{PD} + T_{per}$
5B	1	1	0	X	0x00		$T_{PD} + T_{per}$
6	1	1	0	0x01	0x01 - 0xFF	Windowed / Resampled mode	T_{PD} + (FPR[FILT_PER] x T_{per}) + $2T_{per}$
7	1	1	0	> 0x01	0x01 - 0xFF	Windowed / Filtered mode	T_{PD} + (CR0[FILTER_CNT] x FPR[FILT_PER] x T_{per}) + $2T_{per}$

1. T_{PD} represents the intrinsic delay of the analog component plus the polarity select logic. T_{SAMPLE} is the clock period of the external sample clock. T_{per} is the period of the bus clock.

18.12 CMP Interrupts

The CMP module is capable of generating an interrupt on either the rising or falling edge of the comparator output (or both). The interrupt request is asserted when both SCR[IER] bit and SCR[CFR] are set. It is also asserted when both SCR[IEF] bit and SCR[CFF] are set. The interrupt is de-asserted by clearing either SCR[IER] or SCR[CFR] for a rising edge interrupt, or SCR[IEF] and SCR[CFF] for a falling edge interrupt.

18.13 Digital to Analog Converter Block Diagram

The following figure shows the block diagram of the DAC module. It contains a 64-tap resistor ladder network and a 64-to-1 multiplexer, which selects an output voltage from one of 64 distinct levels that outputs from DACO. It is controlled through DAC Control register (DACCR). Its supply reference source can be selected from two sources V_{in1} and V_{in2}. The module can be powered down (disabled) when it is not used. When in disable mode, DACO is connected to the analog ground.

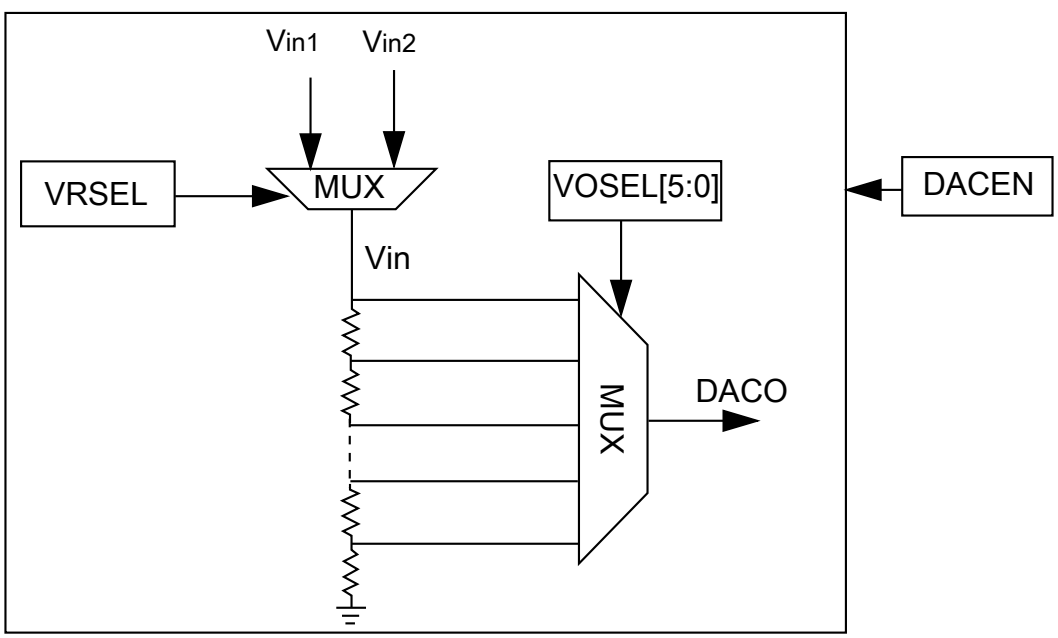


Figure 18-12. 6-bit DAC Block Diagram

18.14 DAC Functional Description

This section provides DAC functional description.

18.14.1 Voltage Reference Source Select

- V_{in1} should be used to connect to the primary voltage source as supply reference of 64 tap resistor ladder
- V_{in2} should be used to connect to alternate voltage source (or primary source if alternate voltage source is not available)

18.15 DAC Resets

This module has a single reset input, corresponding to the chip-wide peripheral reset.

18.16 DAC Clocks

This module has a single clock input, the bus clock.

18.17 DAC Interrupts

This module has no interrupts.

Chapter 19
FlexTimer Module (FTM)

19.1 Chip specific FlexTimer module

This device has one Flextimer (FTM) module.

The FlexTimer module is a two-channel timer that supports input capture, output compare, and the generation of PWM signals. FTM time reference is a 16-bit counter that can be used as an unsigned or signed counter. It is fully backward compatible to the TPM.

FTM module has independent external clock input. The following table summarizes the signals of FTM module.

Customization:

- Primary clock: BUSCLK (20 MHz)
- Alternate clock: ICSFFCLK/2, TCLK
- FTM module has independent external clock input. The following table summarizes thel signals of FTM module.

Table 19-1. FTM module signals Connection

Module	Signal	Connect to
FTM	channel 0 output	XBAR_IN10
	channel 0 input	XBAR_OUT4
	channel 1 output	XBAR_IN11
	channel 1 input	XBAR_OUT5
	Internal Clock	BUSCLK, ICSFFCLK/2
	External Clock	PTB6/TCLK

19.2 Introduction

NOTE

For the chip-specific implementation details of this module's instances, see the chip configuration information.

The FlexTimer module is a two to eight channel timer which supports input capture, output compare, and the generation of PWM signals to control electric motor and power management applications. The FTM time reference is a 16-bit unsigned counter.

19.2.1 FlexTimer philosophy

The FlexTimer is built upon a very simple timer used for many years on NXP's 8-bit microcontrollers, the HCS08 Timer PWM Module – TPM. The FlexTimer extends the functionality to meet the demands of motor control, digital lighting solutions, and power conversion, while providing low cost and backwards compatibility with the TPM module.

All of the features common with the TPM module have fully backwards compatible register assignments and the FlexTimer can use code on the same core platform without change to perform the same functions.

19.2.2 Features

The FTM features include:

- Selectable FTM source clock:
  - Source clock can be the system clock, the fixed frequency clock, or an external clock
  - Fixed frequency clock is an additional clock input to allow the selection of an on chip clock source other than the system clock
  - Selecting external clock connects FTM clock to a chip level input pin therefore allowing to synchronize the FTM counter with an off chip clock source
- Prescaler divide-by 1, 2, 4, 8, 16, 32, 64, or 128
- FTM has a 16-bit counter

    - It can be a free-running counter or a counter with selectable final value
    - The counting can be up or up-down
- Each channel can be configured for input capture, output compare, or edge-aligned PWM mode
- In input capture mode:
    - The capture can occur on rising edges, falling edges or both edges
- In output compare mode the output signal can be set, cleared, or toggled on match
- All channels can be configured for center-aligned PWM mode
- The generation of an interrupt per channel
- The generation of an interrupt when the counter overflows
- Backwards compatible with TPM

19.2.3 Modes of operation

When the MCU is in active BDM background or BDM foreground mode, the FTM temporarily suspends all counting until the MCU returns to normal user operating mode. During stop mode, all FTM input clocks are stopped, so the FTM is effectively disabled until clocks resume. During wait mode, the FTM continues to operate normally. If the FTM does not need to produce a real time reference or provide the interrupt sources needed to wake the MCU from wait mode, the power can then be saved by disabling FTM functions before entering wait mode.

19.2.4 Block diagram

The FTM uses one input/output (I/O) pin per channel, CHn (FTM channel (n)) where n is the channel number (0–7).

The following figure shows the FTM structure. The central component of the FTM is the 16-bit counter with programmable final value and its counting can be up or up-down.

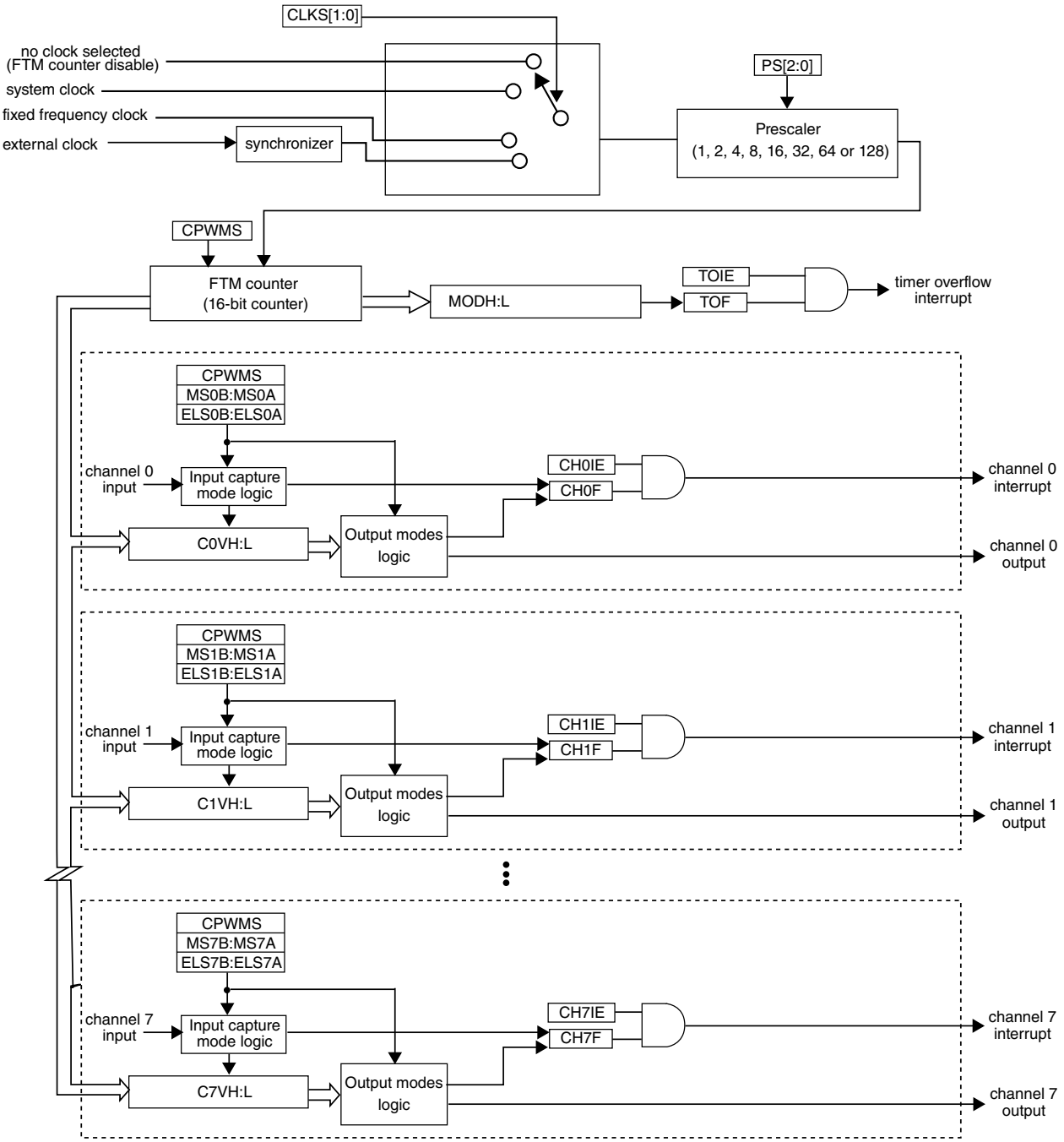


Figure 19-1. FTM block diagram

19.3 Signal description

The following table shows the user-accessible signals for the FTM.

Table 19-2. Signal properties

Name	Function
EXTCLK	External clock – FTM external clock can be selected to drive the FTM counter.
CHn[1]	Channel (n) – I/O pin associated with FTM channel (n).

1. n = channel number (0 to 7)

19.3.1 EXTCLK — FTM external clock

The external clock input signal is used as the FTM counter clock if selected by CLKS[1:0] bits in the SC register. This clock signal must not exceed 1/4 of system clock frequency. The FTM counter prescaler selection and settings are also used when an external clock is selected.

19.3.2 CHn — FTM channel (n) I/O pin

Each FTM channel can be configured to operate either as input or output. The direction associated with each channel, input or output, is selected according to the mode assigned for that channel.

19.4 Memory map and register definition

This section provides a detailed description of all FTM registers.

19.4.1 Module memory map

This section presents a high-level summary of the FTM registers and how they are mapped.

19.4.2 Register descriptions

This section consists of register descriptions in address order.

FTM memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
70	Status and Control (FTM0_SC)	8	R/W	00h	19.4.3/322
71	Counter High (FTM0_CNTH)	8	R/W	00h	19.4.4/323
72	Counter Low (FTM0_CNTL)	8	R/W	00h	19.4.5/324
73	Modulo High (FTM0_MODH)	8	R/W	00h	19.4.6/324
74	Modulo Low (FTM0_MODL)	8	R/W	00h	19.4.7/325
75	Channel Status and Control (FTM0_C0SC)	8	R/W	00h	19.4.8/326
76	Channel Value High (FTM0_C0VH)	8	R/W	00h	19.4.9/327
77	Channel Value Low (FTM0_C0VL)	8	R/W	00h	19.4.10/328
78	Channel Status and Control (FTM0_C1SC)	8	R/W	00h	19.4.8/326
79	Channel Value High (FTM0_C1VH)	8	R/W	00h	19.4.9/327
7A	Channel Value Low (FTM0_C1VL)	8	R/W	00h	19.4.10/328

19.4.3 Status and Control (FTMx_SC)

SC contains the overflow status flag and control bits used to configure the interrupt enable, FTM configuration, clock source, and prescaler factor. These controls relate to all channels within this module.

Address: 70h base + 0h offset = 70h

Bit	7	6	5	4	3	2	1	0
Read	TOF	TOIE	CPWMS	CLKS		PS		
Write	0							
Reset	0	0	0	0	0	0	0	0

FTMx_SC field descriptions

Field	Description
7 TOF	Timer Overflow Flag Set by hardware when the FTM counter passes the value in the Counter Modulo registers. The TOF bit is cleared by reading the SC register while TOF is set and then writing a 0 to TOF bit. Writing a 1 to TOF has no effect. If another FTM overflow occurs between the read and write operations, the write operation has no effect; therefore, TOF remains set indicating an overflow has occurred. In this case a TOF interrupt request is not lost due to the clearing sequence for a previous TOF. 0 FTM counter has not overflowed. 1 FTM counter has overflowed.
6 TOIE	Timer Overflow Interrupt Enable Enables FTM overflow interrupts.

Table continues on the next page...

FTMx_SC field descriptions (continued)

Field	Description
	0 Disable TOF interrupts. Use software polling. 1 Enable TOF interrupts. An interrupt is generated when TOF equals one.
5 CPWMS	Center-aligned PWM Select Selects CPWM mode. This mode configures the FTM to operate in up-down counting mode. 0 FTM counter operates in up counting mode. 1 FTM counter operates in up-down counting mode.
4–3 CLKS	Clock Source Selection Selects one of the three FTM counter clock sources. 00 No clock selected (this in effect disables the FTM counter). 01 System clock 10 Fixed frequency clock 11 External clock
PS	Prescale Factor Selection Selects one of 8 division factors for the clock source selected by CLKS. The new prescaler factor affects the clock source on the next system clock cycle after the new value is updated into the register bits. 000 Divide by 1 001 Divide by 2 010 Divide by 4 011 Divide by 8 100 Divide by 16 101 Divide by 32 110 Divide by 64 111 Divide by 128

19.4.4 Counter High (FTMx_CNTH)

The Counter registers contain the high and low bytes of the counter value. Reading either byte latches the contents of both bytes into a buffer where they remain latched until the other half is read. This allows coherent 16-bit reads in either big-endian or little-endian order which makes this more friendly to various compiler implementations. The coherency mechanism is automatically restarted by an MCU reset or any write to the Status and Control register.

Writing any value to COUNT_H or COUNT_L updates the FTM counter with its initial 16-bit value (all zeroes) and resets the read coherency mechanism, regardless of the data involved in the write.

When BDM is active, the FTM counter is frozen (this is the value that you may read); the read coherency mechanism is frozen such that the buffer latches remain in the state they were in when the BDM became active, even if one or both counter bytes are read while

BDM is active. This assures that if you were in the middle of reading a 16-bit register when BDM became active, it reads the appropriate value from the other half of the 16-bit value after returning to normal execution.

Address: 70h base + 1h offset = 71h

Bit	7	6	5	4	3	2	1	0
Read Write	COUNT_H							
Reset	0	0	0	0	0	0	0	0

FTMx_CNTH field descriptions

Field	Description
COUNT_H	Counter value high byte

19.4.5 Counter Low (FTMx_CNTL)

See the description for the Counter High register.

Address: 70h base + 2h offset = 72h

Bit	7	6	5	4	3	2	1	0
Read Write	COUNT_L							
Reset	0	0	0	0	0	0	0	0

FTMx_CNTL field descriptions

Field	Description
COUNT_L	Counter value low byte

19.4.6 Modulo High (FTMx_MODH)

The Modulo registers contain the high and low bytes of the modulo value for the FTM counter. After the FTM counter reaches the modulo value, the overflow flag (TOF) becomes set at the next clock, and the next value of FTM counter depends on the selected counting method (Counter).

Writing to either byte latches the value into a buffer. The register is updated with the value of their write buffer according to Update of the registers with write buffers.

This write coherency mechanism may be manually reset by writing to the SC register whether BDM is active or not.

When BDM is active, this write coherency mechanism is frozen such that the buffer latches remain in the state they were in when the BDM became active, even if one or both bytes of the modulo register are written while BDM is active. Any write to the modulo register bypasses the buffer latches and directly writes to the modulo register while BDM is active.

It is recommended to initialize the FTM counter, by writing to CNTH or CNTL, before writing to the FTM modulo register to avoid confusion about when the first counter overflow will occur.

Address: 70h base + 3h offset = 73h

Bit	7	6	5	4	3	2	1	0
Read Write	MOD_H							
Reset	0	0	0	0	0	0	0	0

FTMx_MODH field descriptions

Field	Description
MOD_H	High byte of the modulo value

19.4.7 Modulo Low (FTMx_MODL)

See the description for the Modulo High register.

Address: 70h base + 4h offset = 74h

Bit	7	6	5	4	3	2	1	0
Read Write	MOD_L							
Reset	0	0	0	0	0	0	0	0

FTMx_MODL field descriptions

Field	Description
MOD_L	Low byte of the modulo value

19.4.8 Channel Status and Control (FTM*x*_C*n*SC)

CnSC contains the channel-interrupt-status flag and control bits used to configure the interrupt enable, channel configuration, and pin function.

Table 19-3. Mode, edge, and level selection

CPWMS	MSnB:MSnA	ELSnB:ELSnA	Mode	Configuration
X	XX	00	None	Pin not used for FTM
0	00	01	Input capture	Capture on Rising Edge Only
		10		Capture on Falling Edge Only
		11		Capture on Rising or Falling Edge
	01	01	Output compare	Toggle Output on match
		10		Clear Output on match
		11		Set Output on match
	1X	10	Edge-aligned PWM	High-true pulses (clear Output on match)
		X1		Low-true pulses (set Output on match)
1	XX	10	Center-aligned PWM	High-true pulses (clear Output on match-up)
		X1		Low-true pulses (set Output on match-up)

Address: 70h base + 5h offset + (3d × i), where i=0d to 1d

Bit	7	6	5	4	3	2	1	0
Read	CHF	CHIE	MSB	MSA	ELSB	ELSA	0	0
Write	0							
Reset	0	0	0	0	0	0	0	0

FTM*x*_C*n*SC field descriptions

Field	Description
7 CHF	Channel Flag Set by hardware when an event occurs on the channel. CHF is cleared by reading the CnSC register while CHnF is set and then writing a 0 to the CHF bit. Writing a 1 to CHF has no effect. If another event occurs between the read and write operations, the write operation has no effect; therefore, CHF remains set indicating an event has occurred. In this case a CHF interrupt request is not lost due to the clearing sequence for a previous CHF.

Table continues on the next page...

FTMx_C*n*SC field descriptions (continued)

Field	Description
	0 No channel event has occurred. 1 A channel event has occurred.
6 CHIE	Channel Interrupt Enable Enables channel interrupts. 0 Disable channel interrupts. Use software polling. 1 Enable channel interrupts.
5 MSB	Channel Mode Select Used for further selections in the channel logic. Its functionality is dependent on the channel mode. See the table in the register description.
4 MSA	Channel Mode Select Used for further selections in the channel logic. Its functionality is dependent on the channel mode. See the table in the register description.
3 ELSB	Edge or Level Select The functionality of ELSB and ELSA depends on the channel mode. See the table in the register description.
2 ELSA	Edge or Level Select The functionality of ELSB and ELSA depends on the channel mode. See the table in the register description.
1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

19.4.9 Channel Value High (FTMx_C*n*VH)

These registers contain the captured FTM counter value of the input capture function or the match value for the output modes.

In input capture mode, reading a single byte in CnV latches the contents into a buffer where they remain latched until the other byte is read. This latching mechanism also resets, or becomes unlatched, when the CnSC register is written whether BDM mode is active or not. Any write to the channel registers is ignored during this mode.

When BDM is active, the read coherency mechanism is frozen such that the buffer latches remain in the state they were in when the BDM became active, even if one or both bytes of the channel value register are read while BDM is active. This ensures that if you were in the middle of reading a 16-bit register when BDM became active, it reads the appropriate value from the other half of the 16-bit value after returning to normal execution. Any read of the CnV registers in BDM mode bypasses the buffer latches and returns the value of these registers and not the value of their read buffer.

In output modes, writing to CnV latches the value into a buffer. The registers are updated with the value of their write buffer according to Update of the registers with write buffers.

This write coherency mechanism may be manually reset by writing to the CnSC register whether BDM mode is active or not.

When BDM is active, the write coherency mechanism is frozen such that the buffer latches remain in the state they were in when the BDM became active even if one or both bytes of the channel value register are written while BDM is active. Any write to the CnV registers bypasses the buffer latches and writes directly to the register while BDM is active. The values written to the channel value registers while BDM is active are used in output modes operation after normal execution resumes. Writes to the channel value registers while BDM is active do not interfere with the partial completion of a coherency sequence. After the write coherency mechanism has been fully exercised, the channel value registers are updated using the buffered values while BDM was not active.

Address: 70h base + 6h offset + (3d × i), where i=0d to 1d

Bit	7	6	5	4	3	2	1	0
Read Write	VAL_H							
Reset	0	0	0	0	0	0	0	0

FTMx_C*n*VH field descriptions

Field	Description
VAL_H	Channel Value High Byte Captured FTM counter value of the input capture function or the match value for the output modes

19.4.10 Channel Value Low (FTMx_C*n*VL)

See the description for the Channel Value High register.

Address: 70h base + 7h offset + (3d × i), where i=0d to 1d

Bit	7	6	5	4	3	2	1	0
Read Write	VAL_L							
Reset	0	0	0	0	0	0	0	0

FTMx_C*n*VL field descriptions

Field	Description
VAL_L	Channel Value Low Byte Captured FTM counter value of the input capture function or the match value for the output modes

FTMx_C*n*VL field descriptions (continued)

Field	Description

19.5 Functional Description

The following sections describe the FTM features.

The notation used in this document to represent the counters and the generation of the signals is shown in the following figure.

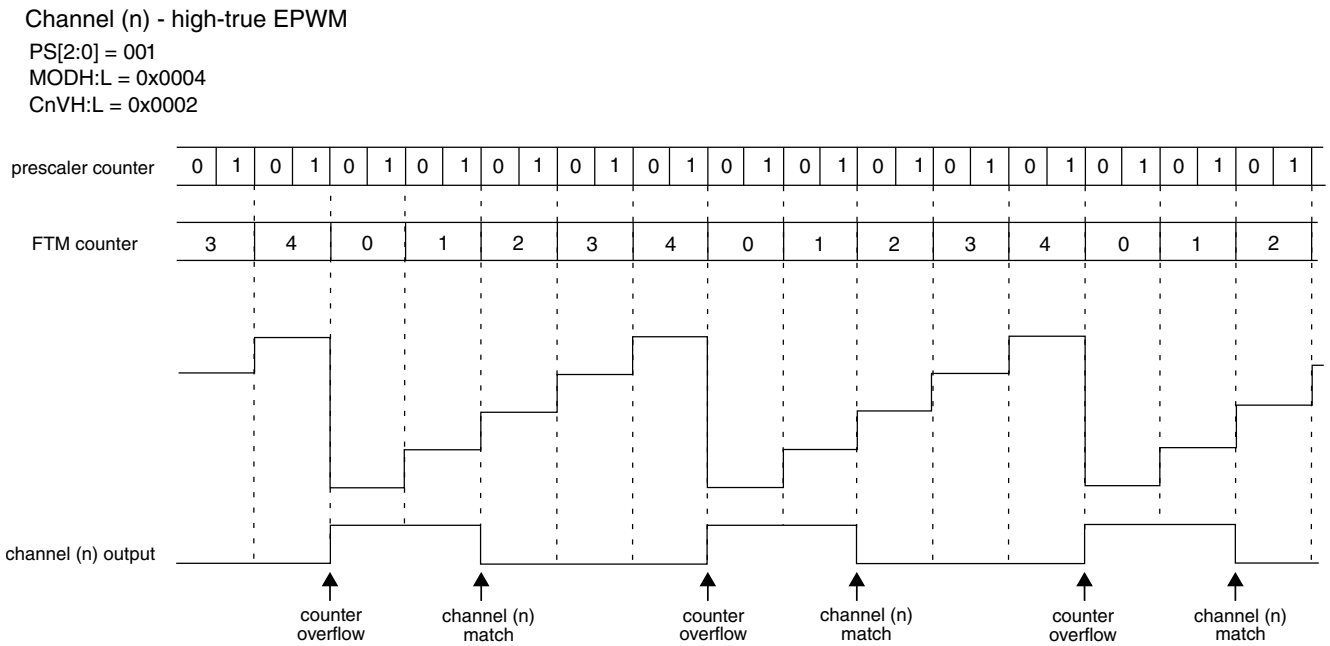


Figure 19-2. Notation used

19.5.1 Clock Source

FTM module has only one clock domain that is the system clock.

19.5.1.1 Counter Clock Source

The CLKS[1:0] bits in the SC register select one of three possible clock sources for the FTM counter or disable the FTM counter. After any MCU reset, CLKS[1:0] = 0:0 so no clock source is selected.

The CLKS[1:0] bits may be read or written at any time. Disabling the FTM counter by writing 0:0 to the CLKS[1:0] bits does not affect the FTM counter value or other registers.

The fixed frequency clock is an alternative clock source for the FTM counter that allows the selection of a clock other than the system clock or an external clock. This clock input is defined by chip integration. Refer to chip specific documentation for further information. Due to FTM hardware implementation limitations, the frequency of the fixed frequency clock must not exceed the system clock frequency.

The external clock passes through a synchronizer clocked by the system clock to ensure that counter transitions are properly aligned to system clock transitions. Therefore, to meet the Nyquist criteria and account for jitter, the frequency of the external clock source must not exceed 1/4 of the system clock frequency.

19.5.2 Prescaler

The selected counter clock source passes through a prescaler that is a 7-bit counter. The value of the prescaler is selected by the PS[2:0] bits. The following figure shows an example of the prescaler counter and FTM counter.

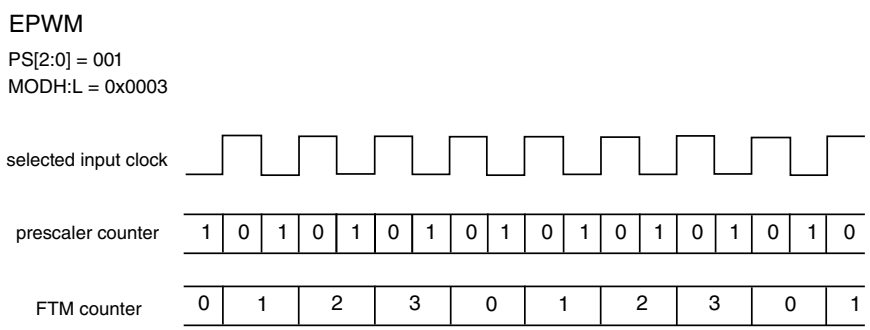


Figure 19-3. Example of the prescaler counter

19.5.3 Counter

The FTM has a 16-bit counter that is used by the channels either for input or output modes. The FTM counter clock is the selected clock divided by the prescaler (see Prescaler).

The FTM counter has these modes of operation:

- up counting (see Up counting)
- up-down counting (see Up-down counting)

19.5.3.1 Up counting

Up counting is selected when (CPWMS = 0).

The starting value of the count is 0x0000 and MODH:L defines the final value of the count; see the following figure. The value of 0x0000 is loaded into the FTM counter, and the counter increments until the value of MODH:L is reached, at which point the counter is reloaded with 0x0000.

The FTM period when using up counting is (MODH:L + 0x0001) × period of the FTM counter clock.

The TOF bit is set when the FTM counter changes from MODH:L to 0x0000.

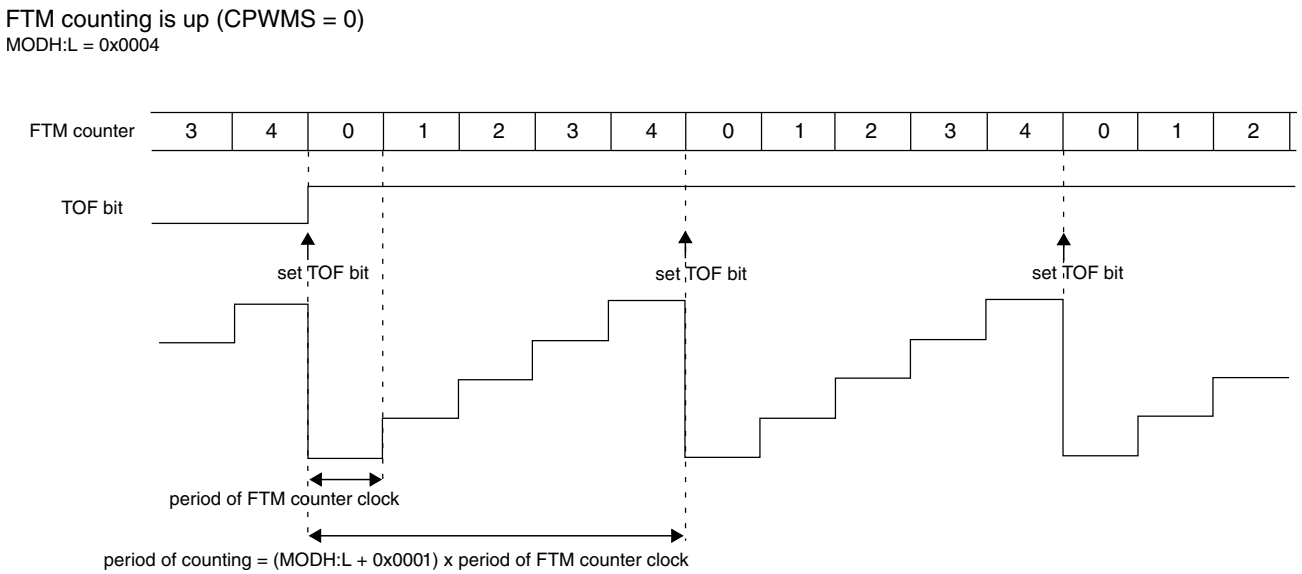


Figure 19-4. Example of FTM up counting

19.5.3.2 Up-down counting

Up-down counting is selected when (CPWMS = 1).

The starting value of the count is 0x0000 and MODH:L defines the final value of the count. The value of 0x0000 is loaded into the FTM counter, and the counter increments until the value of MODH:L is reached, at which point the counter is decremented until it returns to the value of 0x0000 and the up-down counting restarts.

The FTM period when using up-down counting is 2 × (MODH:L) × period of the FTM counter clock.

The TOF bit is set when the FTM counter changes from MODH:L to (MODH:L – 1).

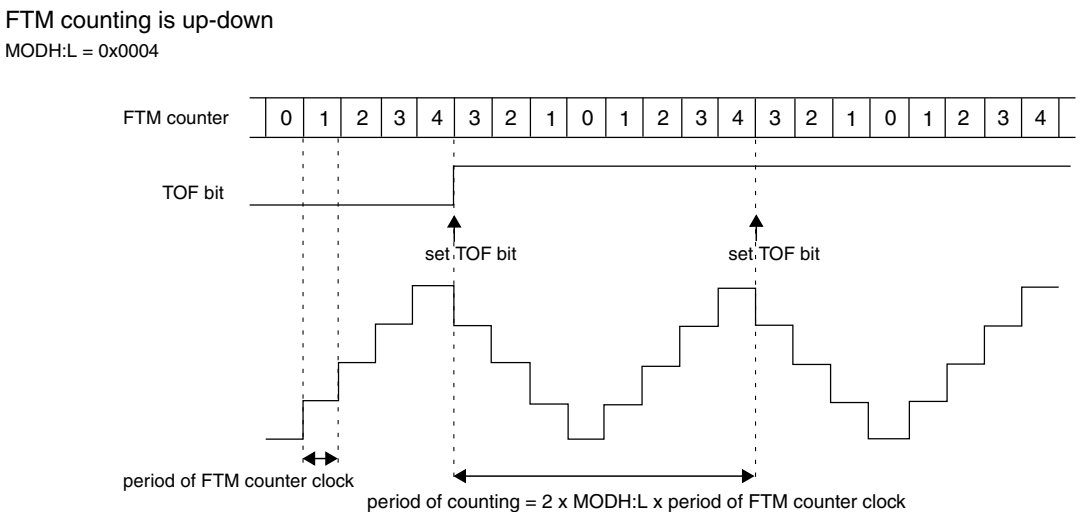


Figure 19-5. Example of up-down counting

19.5.3.3 Free running counter

If (MODH:L = 0x0000 or MODH:L = 0xFFFF), the FTM counter is a free running counter. In this case, the FTM counter runs free from 0x0000 through 0xFFFF and the TOF bit is set when the FTM counter changes from 0xFFFF to 0x0000 See the following figure.

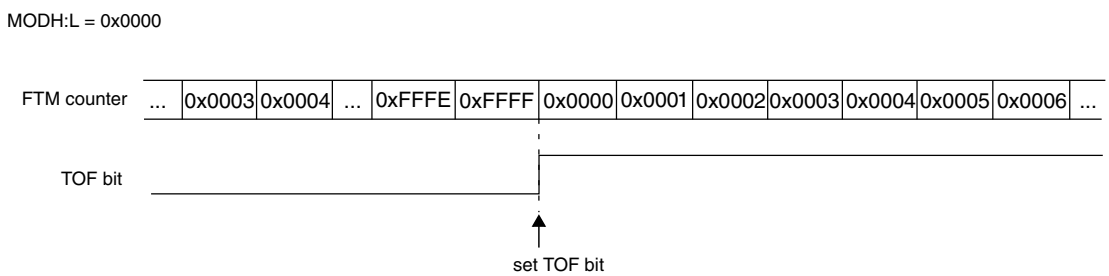


Figure 19-6. Example when the FTM counter is a free running

19.5.3.4 Counter reset

Any write to CNTH or CNTL register resets the FTM counter to the value of 0x0000 and the channels output to its initial value, except for channels in output compare mode.

19.5.4 Input capture mode

The input capture mode is selected when (CPWMS = 0), (MSnB:MSnA = 0:0), and (ELSnB:ELSnA ≠ 0:0).

When a selected edge occurs on the channel input, the current value of the FTM counter is captured into the CnVH:L registers. At the same time, the CHnF bit is set and the channel interrupt is generated if enabled by CHnIE = 1. See the following figure.

When a channel is configured for input capture, the CHn pin is an edge-sensitive input. ELSnB:ELSnA control bits determine which edge, falling or rising, triggers input-capture event. Note that the maximum frequency for the channel input signal to be detected correctly is system clock divided by four, which is required to meet Nyquist criteria for signal sampling.

When either half of the 16-bit capture register (CnVH:L) is read, the other half is latched into a buffer to support coherent 16-bit access in big-endian or little-endian order. This read coherency mechanism can be manually reset by writing to CnSC register.

Writes to the CnVH:L registers are ignored in input capture mode.

While in BDM, the input capture function works as configured. When a selected edge event occurs, the FTM counter value, which is frozen because of BDM, is captured into the CnVH:L registers and the CHnF bit is set.

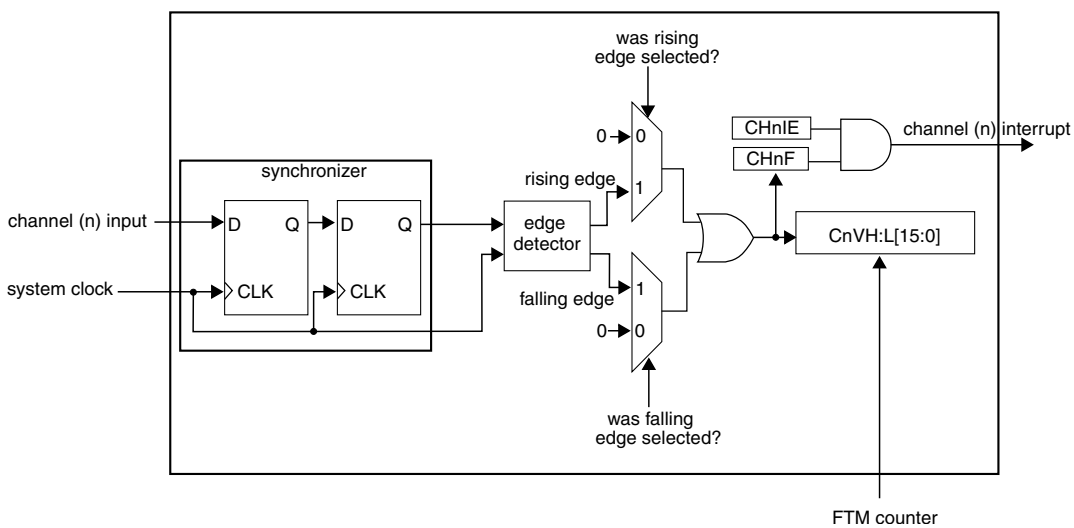


Figure 19-7. Input capture mode

The input signal is always delayed three rising edges of the system clock; that is, two rising edges to the synchronizer plus one more rising edge to the edge detector. In other words, the CHnF bit is set on the third rising edge of the system clock after a valid edge occurs on the channel input.

19.5.5 Output compare mode

The output compare mode is selected when (CPWMS = 0) and (MSnB:MSnA = 0:1).

In output compare mode, the FTM can generate timed pulses with programmable position, polarity, duration, and frequency. When the counter matches the value in the CnVH:CnVL registers of an output compare channel, the channel (n) output can be set, cleared, or toggled.

When a channel is initially configured to toggle mode, the previous value of the channel output is held until the first output compare event occurs.

The CHnF bit is set and the channel (n) interrupt is generated (if CHnIE = 1) at the channel (n) match (FTM counter = CnVH:CnVL).

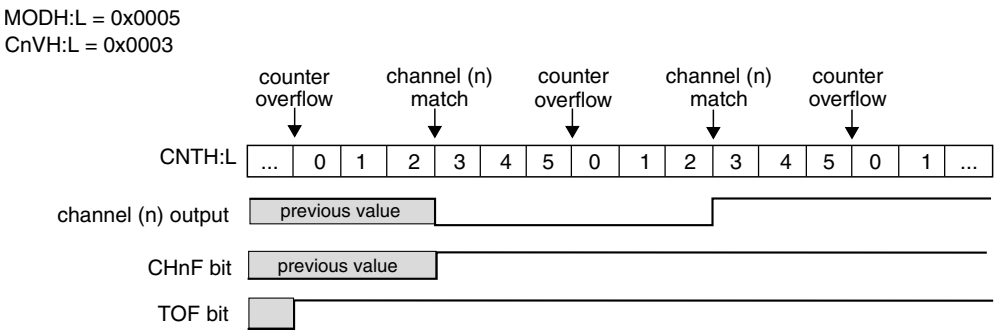


Figure 19-8. Example of the output compare mode when the match toggles the channel output

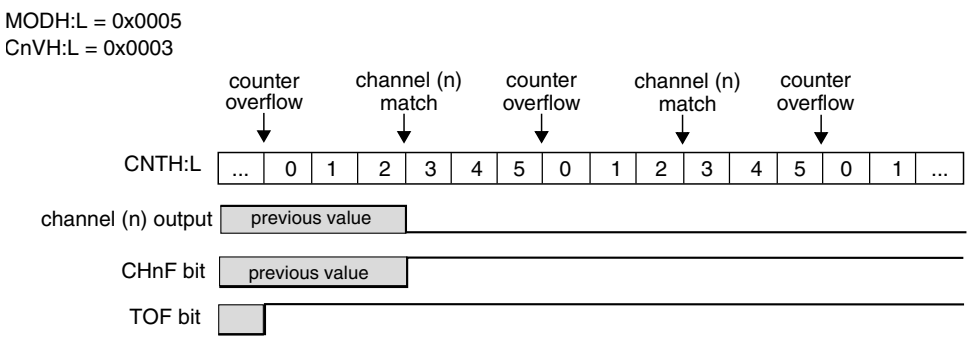


Figure 19-9. Example of the output compare mode when the match clears the channel output

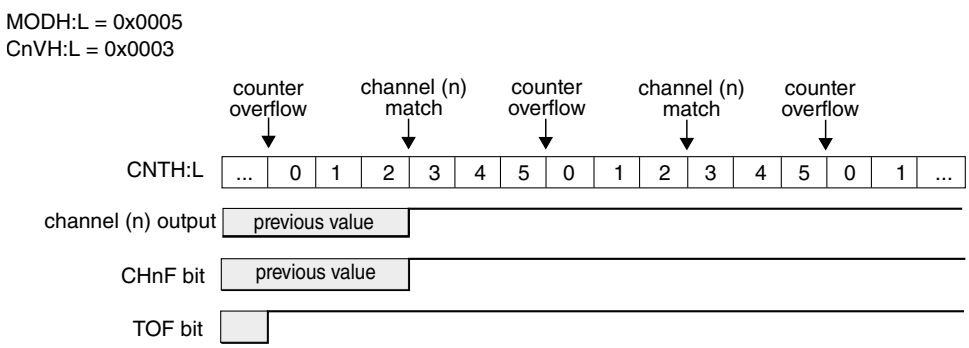


Figure 19-10. Example of the output compare mode when the match sets the channel output

It is possible to use the output compare mode with (ELSnB:ELSnA = 0:0). In this case, when the counter reaches the value in the CnVH:CnVL registers, the CHnF bit is set and the channel (n) interrupt is generated, if CHnIE = 1. However, the channel (n) output is not modified and controlled by FTM.

19.5.6 Edge-aligned PWM (EPWM) mode

The edge-aligned mode is selected when all of the following apply:

- (CPWMS = 0)
- (MSnB = 1)

The EPWM period is determined by (MODH:L + 0x0001) and the pulse width (duty cycle) is determined by (CnVH:L).

The CHnF bit is set and the channel (n) interrupt is generated (if CHnIE = 1) at the channel (n) match (FTM counter = CnVH:L), that is, at the end of the pulse width.

This type of PWM signal is called edge-aligned because the leading edges of all PWM signals are aligned with the beginning of the period, which is the same for all channels within an FTM.

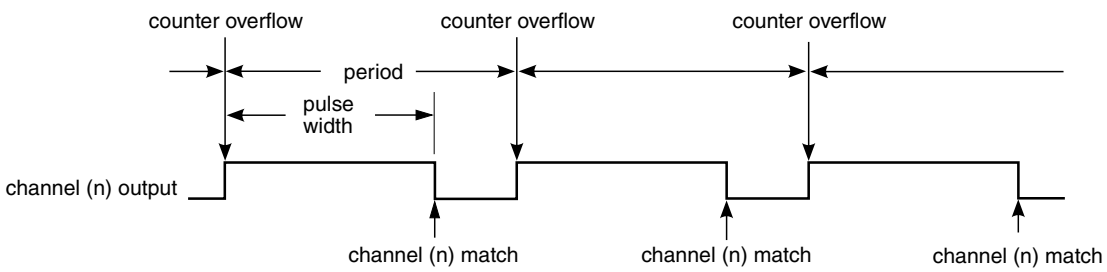


Figure 19-11. EPWM period and pulse width with ELSnB:ELSnA = 1:0

If (ELSnB:ELSnA = 0:0) when the counter reaches the value in the CnVH:L registers, the CHnF bit is set and the channel (n) interrupt is generated (if CHnIE = 1), however, the channel (n) output is not controlled by FTM.

If (ELSnB:ELSnA = 1:0), then the channel (n) output is forced high at the counter overflow, when the value of 0x0000 is loaded into the FTM counter. Additionally, it is forced low at the channel (n) match, when the FTM counter = CnVH:L. See the following figure.

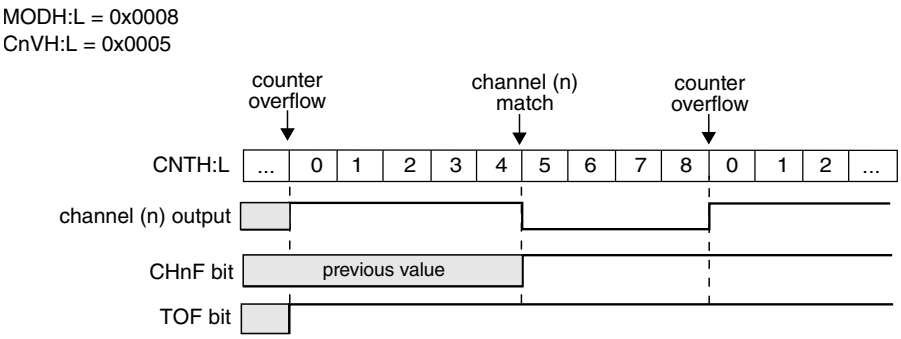


Figure 19-12. EPWM signal with ELSnB:ELSnA = 1:0

If (ELSnB:ELSnA = X:1), then the channel (n) output is forced low at the counter overflow, when the value of 0x0000 is loaded into the FTM counter. Additionally, it is forced high at the channel (n) match, when the FTM counter = CnVH:L. See the following figure.

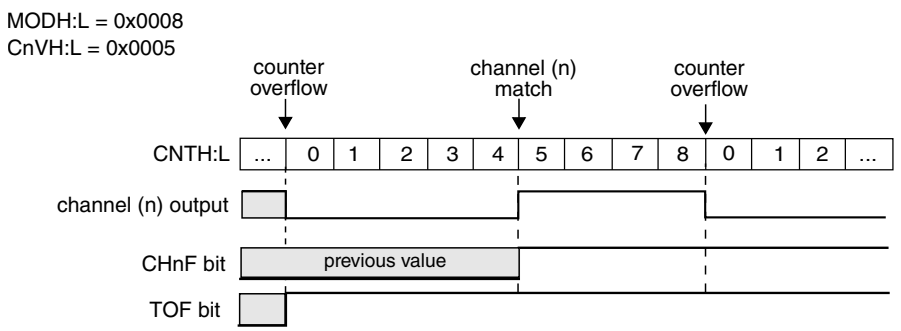


Figure 19-13. EPWM signal with ELSnB:ELSnA = X:1

If (CnVH:L = 0x0000), then the channel (n) output is a 0% duty cycle EPWM signal and CHnF bit is not set, even when there is the channel (n) match. If (CnVH:L > MODH:L), then the channel (n) output is a 100% duty cycle EPWM signal and CHnF bit is not set, even when there is the channel (n) match. Therefore, MODH:MODL must be less than 0xFFFF in order to get a 100% duty cycle EPWM signal.

19.5.7 Center-aligned PWM (CPWM) mode

The center-aligned mode is selected when:

- (CPWMS = 1)

The CPWM pulse width (duty cycle) is determined by 2 × (CnVH:L). The period is determined by 2 × (MODH:L). See the following figure. MODH:L must be kept in the range of 0x0001 to 0x7FFF because values outside this range can produce ambiguous results.

In the CPWM mode, the FTM counter counts up until it reaches MODH:L and then counts down until it reaches the value of 0x0000.

The CHnF bit is set and channel (n) interrupt is generated (if CHnIE = 1) at the channel (n) match (FTM counter = CnVH:L) when the FTM counting is down, at the begin of the pulse width, and when the FTM counting is up, at the end of the pulse width.

This type of PWM signal is called center-aligned because the pulse width centers for all channels are aligned with the value of 0x0000.

The other channel modes are not compatible with the up-down counter (CPWMS = 1). Therefore, all FTM channels must be used in CPWM mode when (CPWMS = 1).

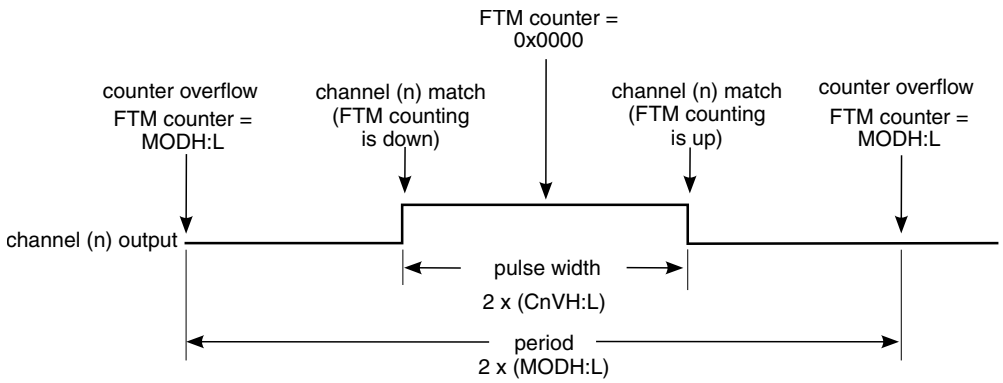


Figure 19-14. CPWM period and pulse width with ELSnB:ELSnA = 1:0

If (ELSnB:ELSnA = 0:0) when the counter reaches the value in the CnVH:L registers, the CHnF bit is set and the channel (n) interrupt is generated (if CHnIE = 1), however the channel (n) output is not controlled by FTM.

If (ELSnB:ELSnA = 1:0), then the channel (n) output is forced high at the channel (n) match (FTM counter = CnVH:L) when counting down, and it is forced low at the channel (n) match when counting up; see the following figure.

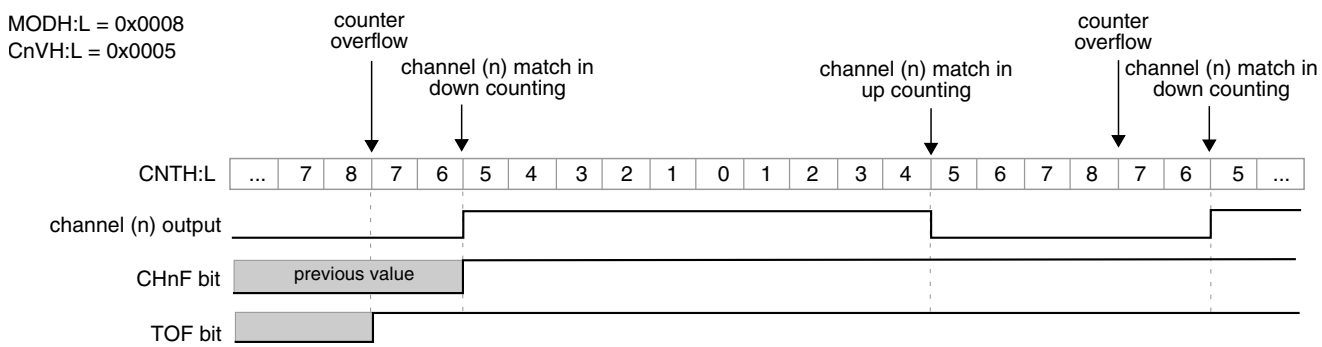


Figure 19-15. CPWM signal with ELSnB:ELSnA = 1:0

If (ELSnB:ELSnA = X:1), then the channel (n) output is forced low at the channel (n) match (FTM counter = CnVH:L) when counting down, and it is forced high at the channel (n) match when counting up; see the following figure.

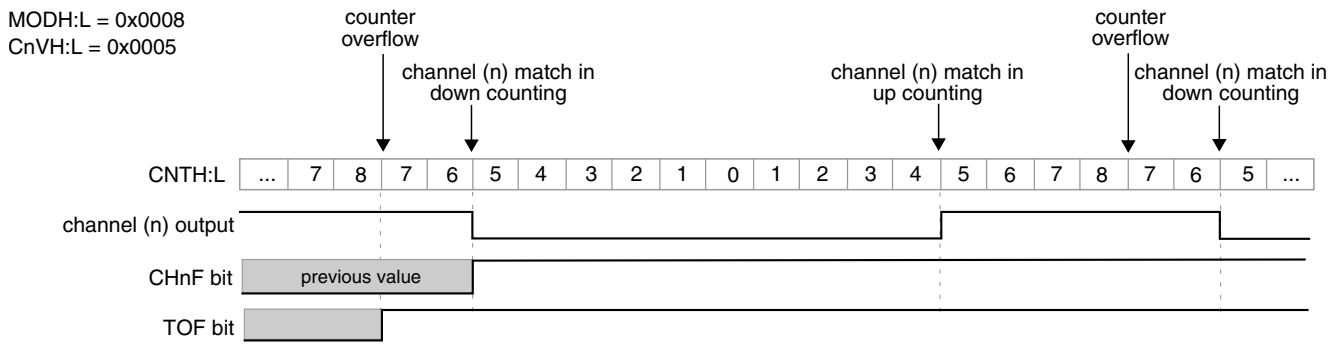


Figure 19-16. CPWM signal with ELSnB:ELSnA = X:1

If (CnVH:L = 0x0000) or (CnVH:L is a negative value, that is, CnVH[7] = 1) then the channel (n) output is a 0% duty cycle CPWM signal and CHnF bit is not set even when there is the channel (n) match.

If (CnVH:L is a positive value, that is, CnVH[7] = 0), (CnVH:L ≥ MODH:L), and (MODH:L ≠ 0x0000), then the channel (n) output is a 100% duty cycle CPWM signal and CHnF bit is not set even when there is the channel (n) match. This implies that the usable range of periods set by MODH:L is 0x0001 through 0x7FFE, or 0x7FFF if you do not need to generate a 100% duty cycle CPWM signal. This is not a significant limitation because the resulting period is much longer than required for normal applications.

The CPWM mode must not be used when the FTM counter is a free running counter.

19.5.8 Update of the registers with write buffers

This section describes the updating of registers that have write buffers.

19.5.8.1 MODH:L registers

If (CLKS[1:0] = 0:0), then MODH:L registers are updated when their second byte is written.

If (CLKS[1:0] ≠ 0:0), then MODH:L registers are updated according to the CPWMS bit:

- If the selected mode is not CPWM mode, then MODH:L registers are updated after both bytes have been written and the FTM counter changes from (MODH:L) to (all zeroes). If the FTM counter is a free-running counter, then this update is made when the FTM counter changes from 0xFFFF to 0x0000.
- If the selected mode is CPWM mode, then MODH:L registers are updated after both bytes have been written and the FTM counter changes from MODH:L to (MODH:L – 0x0001).

19.5.8.2 CnVH:L registers

If (CLKS[1:0] = 0:0), then CnVH:L registers are updated when their second byte is written.

If (CLKS[1:0] ≠ 0:0), then CnVH:L registers are updated according to the selected mode:

- If the selected mode is output compare mode, then CnVH:L registers are updated after their second byte is written and on the next change of the FTM counter.
- If the selected mode is EPWM mode, the CnVH:L registers are updated after both bytes have been written and the FTM counter changes from MODH:L to all zeroes. If the FTM counter is a free running counter, then this update is made when the FTM counter changes from 0xFFFF to 0x0000.
- If the selected mode is CPWM mode, then CnVH:L registers are updated after both bytes have been written and the FTM counter changes from MODH:L to (MODH:L – 0x0001).

19.5.9 BDM mode

When BDM mode is active, the FlexTimer counter and the channels output are frozen.

However, the value of FlexTimer counter or the channels output are modified in BDM mode when:

- A write of any value to the CNTH or CNTL registers (Counter reset) resets the FTM counter to the value of 0x0000 and the channels output to their initial value, except for channels in output compare mode.

19.6 Reset overview

The FTM is reset whenever any chip reset occurs.

When the FTM exits from reset:

- The FTM counter and the prescaler counter are zero and are stopped (CLKS[1:0] = 0b00)
- The timer overflow interrupt is zero (Timer overflow interrupt)
- The channels interrupts are zero (Channel (n) interrupt)

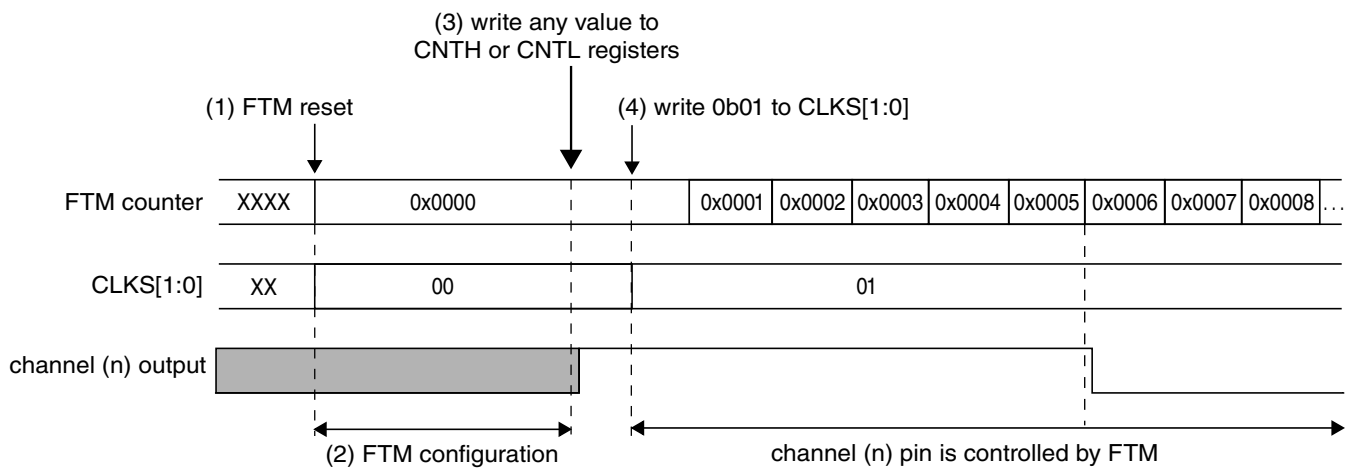
- The channels are in input capture mode (Input capture mode)
- The channels outputs are zero
- The channels pins are not controlled by FTM (ELS(n)B:ELS(n)A = 0b00). See table "Mode, Edge, and Level Selection"

The following figure shows the FTM behavior after the reset. At the reset (item 1), the FTM counter is disabled (see table "FTM Clock Source Selection"), its value is updated to zero and the pins are not controlled by FTM (table "Mode, Edge, and Level Selection").

After the reset, the FTM should be configured (item 2). It is necessary to define the FTM counter mode, the FTM counting limit (MODH:L registers value), the channels mode and CnVH:L registers value according to the channels mode.

Because of this, you should write any value to CNTH or CNTL registers (item 3). This write updates the FTM counter with the value of 0x0000 and the channels output with its initial value (except for channels in output compare mode) (Counter reset).

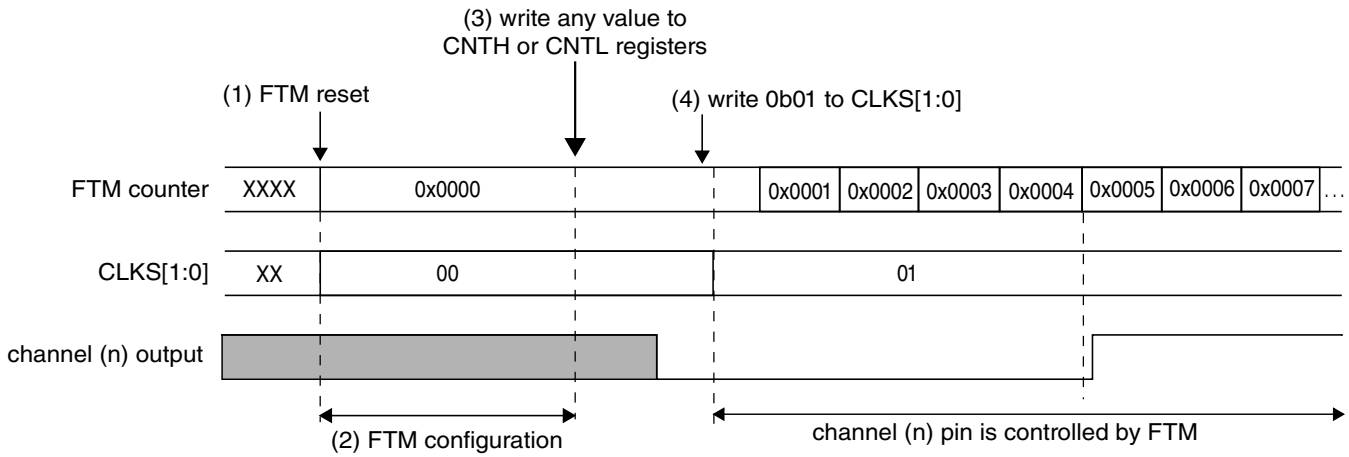
The next step is to select the FTM counter clock by the CLKS[1:0] bits (item 4). It is important to highlight that the pins are controlled only by FTM when CLKS[1:0] bits are different from zero (table "Mode, Edge, and Level Selection").



Note
– Channel (n) is in high-true EPWM mode with 0 < C(n)VH:L < MODH:L
– C(n)VH:L = 0x0005

Figure 19-17. FTM behavior after the reset when the channel (n) is in EPWM mode

The following figure shows an example when the channel (n) is in output compare mode and the channel (n) output is toggled when there is a match. In the output compare mode, the channel output is not updated to its initial value when there is a write to CNTH or CNTL registers (item 3).



Note
– Channel (n) is in output compare and the channel (n) output is toggled when there is a match
– C(n)VH:L = 0x0004

Figure 19-18. FTM behavior after the reset when the channel (n) is in output compare mode

19.7 FTM Interrupts

19.7.1 Timer overflow interrupt

The timer overflow interrupt is generated when (TOIE = 1) and (TOF = 1).

19.7.2 Channel (n) interrupt

The channel (n) interrupt is generated when (CHnIE = 1) and (CHnF = 1).

Chapter 20
Pules Width Timer (PWT)

20.1 Chip specific pules width timer

This device has two pules width timer (PWT).

The PWT is used to captures a pulse width and pulse period.

Customization:

- Primary clock: BUSCLK (20 MHz)
- Alternate clock: TCLK
- The following table summarizes the signal connection of PWT module.

Table 20-1. PWT module signals Connection

Module	Signal	Connect to
PWT0	PWT0_IN0	PTA6/PWT0
	PWT0_IN1	XBAR_OUT6
	PWT0_IN2, PWT0_IN3,	tie to ground
	Internal Clock	BUSCLK
	External Clock	PTB6/TCLK
PWT1	PWT1_IN0	PTA7/PWT1
	PWT1_IN1	XBAR_OUT7
	PWT1_IN2, PWT1_IN3	tie to ground
	Internal Clock	BUSCLK
	External Clock	PTB6/TCLK

20.2 Introduction

20.2.1 Features

The pulse width timer (PWT) includes the following features:

- Automatic measurement of pulse width with 16 bit resolution
- Separate positive and negative pulse width measurements
- Programmable measuring time between successive alternating edges, rising edges or falling edges
- Programmable pre-scaler from clock input as 16-bit counter time base
- Two selectable clock sources — bus clock and alternative clock
- Four selectable pulse inputs
- Programmable interrupt generation upon pulse width value updated and counter overflows

20.2.2 Modes of operation

This module supports the following mode:

- Run Mode

  When enabled, the pulse width timer module is active.

- Wait Mode

  When enabled, the pulse width timer module is active and can perform the waking up function if the corresponding interrupt is enabled.

- Stop Mode

  The pulse width timer module is halted when entering stop and the register contents and operating status is preserved. If stop exits with reset then the module resets. If stop exits with another source, the module resumes operation based on module status upon exit.

- Active Background Mode

  Upon entering BDM mode, the PWT suspends all counting and pulse edge detection until the microcontroller returns to normal user operating mode. Counting and edge detection resume from the suspended value when normal user operating mode returns as long as the PWTSR bit (PWT software reset) is not written to 1 and the PWT module is still enabled.

20.2.3 Block diagram

The following figure show the block diagram of the PWT.

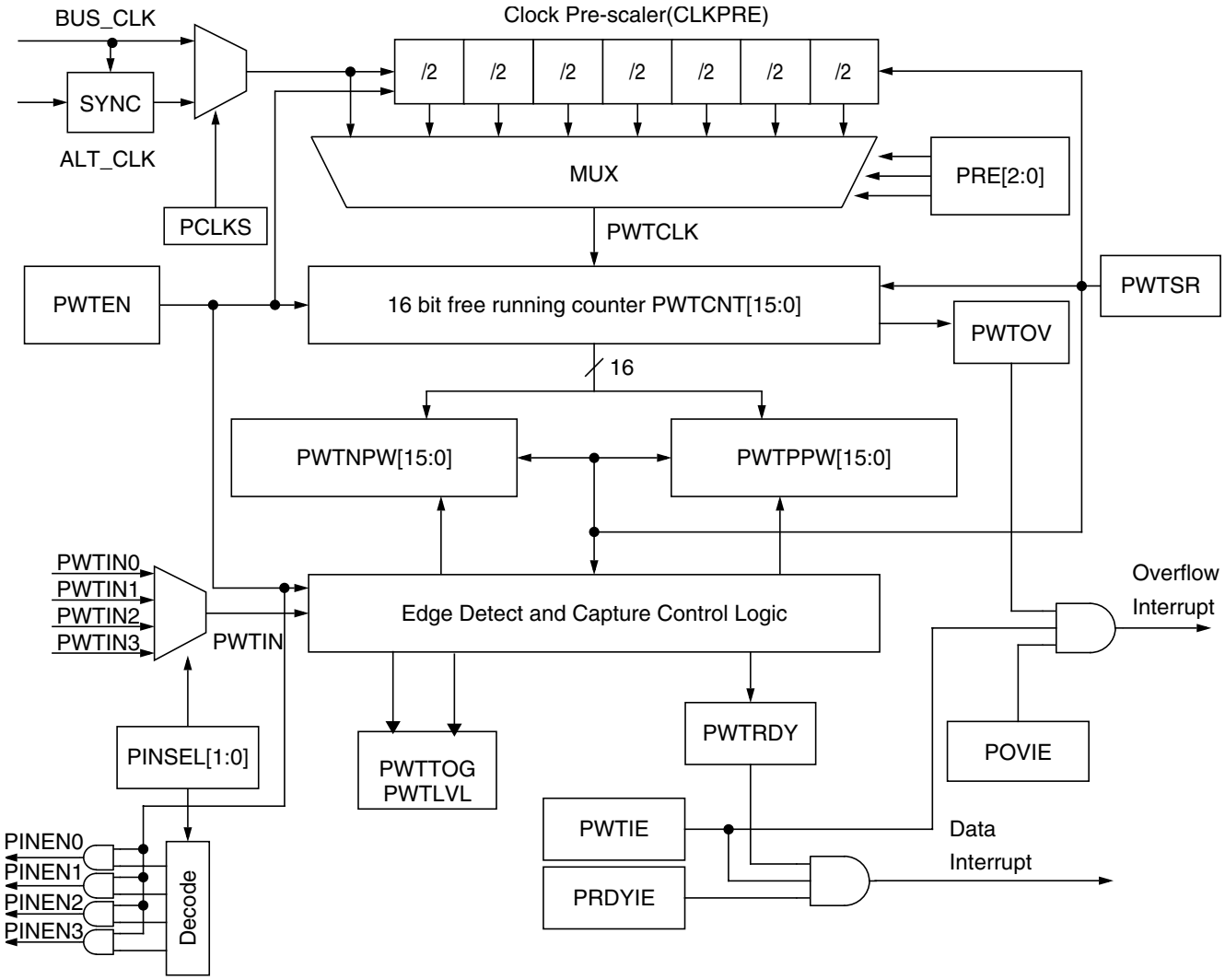


Figure 20-1. Pulse width timer (PWT) block diagram

20.3 External signal description

20.3.1 Overview

PWT has the following signal.

Table 20-2. PWT signal properties

Signal	Pullup	Description	I/O
PWTIN[3:0]	No	Pulse inputs	I
ALTCLK	No	Alternative clock source for the counter	I

20.3.2 PWTIN[3:0] — pulse width timer capture inputs

The input signals are pulse capture inputs which can come from internal or external. The PWT input is selected by PINSEL[1:0] to be routed to the pulse width timer. If the input comes from external and is selected as the PWT input, the input port is enabled for PWT function by PINSEL[1:0] automatically. The minimum pulse width to be measured is 1 PWTCLK cycle, any pulse narrower than this value is ignored by PWT module. The PWTCLK cycle time depends on the PWT clock source selection and pre-scaler rate setting.

20.3.3 ALTCLK— alternative clock source for counter

The PWT has an alternative clock input ALTCLK which can be selected as the clock source of the counter when the PCLKS bit is set. The ALTCLK input must be synchronized by the bus clock. Variations in duty cycle and clock jitter must also be accommodated so that the ALTCLK signal must not exceed one-fourth of the bus frequency. The ALTCLK pin can be shared with a general-purpose port pin. See the Pins and Connections chapter for the pin location and priority of this function.

20.4 Memory Map and Register Descriptions

PWT memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
30	Pulse Width Timer Control and Status Register (PWT0_CS)	8	R/W	00h	20.4.1/347
31	Pulse Width Timer Control Register (PWT0_CR)	8	R/W	00h	20.4.2/348
32	Pulse Width Timer Positive Pulse Width Register: High (PWT0_PPH)	8	R	00h	20.4.3/349
33	Pulse Width Timer Positive Pulse Width Register: Loq (PWT0_PPL)	8	R	00h	20.4.4/350

Table continues on the next page...

PWT memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
34	Pulse Width Timer Negative Pulse Width Register: High (PWT0_NPH)	8	R	00h	20.4.5/350
35	Pulse Width Timer Negative Pulse Width Register: Low (PWT0_NPL)	8	R	00h	20.4.6/351
36	Pulse Width Timer Counter Register: High (PWT0_CNTH)	8	R	00h	20.4.7/351
37	Pulse Width Timer Counter Register: Low (PWT0_CNTL)	8	R	00h	20.4.8/351
38	Pulse Width Timer Control and Status Register (PWT1_CS)	8	R/W	00h	20.4.1/347
39	Pulse Width Timer Control Register (PWT1_CR)	8	R/W	00h	20.4.2/348
3A	Pulse Width Timer Positive Pulse Width Register: High (PWT1_PPH)	8	R	00h	20.4.3/349
3B	Pulse Width Timer Positive Pulse Width Register: Loq (PWT1_PPL)	8	R	00h	20.4.4/350
3C	Pulse Width Timer Negative Pulse Width Register: High (PWT1_NPH)	8	R	00h	20.4.5/350
3D	Pulse Width Timer Negative Pulse Width Register: Low (PWT1_NPL)	8	R	00h	20.4.6/351
3E	Pulse Width Timer Counter Register: High (PWT1_CNTH)	8	R	00h	20.4.7/351
3F	Pulse Width Timer Counter Register: Low (PWT1_CNTL)	8	R	00h	20.4.8/351

20.4.1 Pulse Width Timer Control and Status Register (PWTx_CS)

Address: Base address + 0h offset

Bit	7	6	5	4	3	2	1	0
Read	PWTEN	PWTIE	PRDYIE	POVIE	0	FCTLE	PWTRDY	PWTOV
Write					PWTSR			
Reset	0	0	0	0	0	0	0	0

PWTx_CS field descriptions

Field	Description
7 PWTEN	PWT Module Enable Enables/disables the PWT module. To avoid unexpected behavior, do not change any PWT configurations as long as PWTEN is set. 0 The PWT is disabled. 1 The PWT is enabled.
6 PWTIE	PWT Module Interrupt Enable Enables the PWT module to generate an interrupt. 0 Disables the PWT to generate interrupt. 1 Enables the PWT to generate interrupt.

Table continues on the next page...

PWTx_CS field descriptions (continued)

Field	Description
5 PRDYIE	PWT Pulse Width Data Ready Interrupt Enable Enables/disables the PWT to generate an interrupt when PWTRDY is set as long as PWTIE is set. 0 Disable PWT to generate interrupt when PWTRDY is set. 1 Enable PWT to generate interrupt when PWTRDY is set.
4 POVIE	PWT Counter Overflow Interrupt Enable Enables/disables the PWT to generate an interrupt when PWTOV is set due to PWT counter overflow. 0 Disable PWT to generate interrupt when PWTOV is set. 1 Enable PWT to generate interrupt when PWTOV is set.
3 PWTSR	PWT Soft Reset Performs a soft reset to the PWT. This field always reads as 0. 0 No action taken. 1 Writing 1 to this field will perform soft reset to PWT.
2 FCTLE	First counter load enable after enable This bit determines if the counter value should be loaded to the corresponding PWTx_PPW{H,L}, PWTx_NPW{H,L} after first enable. 0 Do not load the first counter values to corresponding registers 1 Load the first coutner value to corresponding registers depended by the PWTIN level
1 PWTRDY	PWT Pulse Width Valid Indicates that the PWT Pulse Width register(s) has been updated and is ready to be read. This field is cleared by reading PWTRDY and then writing 0 to PWTRDY bit when PWTRDY is set.Writing 1 to this field has no effect. 0 PWT pulse width register(s) is not up-to-date. 1 PWT pulse width register(s) has been updated.
0 PWTOV	PWT Counter Overflow Indicates that the PWT counter has run from 0x0000_0xFFFF to 0x0000_0x0000. This field is cleared by writing 0 to PWTOV when PWTOV is set. Writing 1 to this field has no effect. If another overflow occurs when this field is being cleared, the clearing fails. 0 PWT counter no overflow. 1 PWT counter runs from 0xFFFF to 0x0000.

20.4.2 Pulse Width Timer Control Register (PWTx_CR)

Address: Base address + 1h offset

Bit	7	6	5	4	3	2	1	0
Read	PCLKS	PINSEL		TGL	LVL	PRE		
Write				w1c				
Reset	0	0	0	0	0	0	0	0

PWTx_CR field descriptions

Field	Description
7 PCLKS	PWT Clock Source Selection Controls the selection of clock source for the PWT counter. 0 BUS_CLK is selected as the clock source of PWT counter. 1 Alternative clock is selected as the clock source of PWT counter.
6–5 PINSEL	PWT Pulse Inputs Selection Enables the corresponding PWT input port, if this PWT input comes from an external source. 00 PWTIN[0] is enabled. 01 PWTIN[1] is enabled. 10 PWTIN[2] enabled. 11 PWTIN[3] enabled.
4 TGL	PWTIN states Toggled from last state This flag indicates if the selected PWTIN has toggled its state since last time this bit has cleared to 0. 0 The selected PWTIN hasn't changed its original states from last time. 1 The selected PWTIN has toggled its states.
3 LVL	PWTIN Level when Overflows This Read Only bit signalizes the selected PWTIN states when the coutner overflows to read out.
PRE	PWT Clock Prescaler (CLKPRE) Setting Selects the value by which the clock is divided to clock the PWT counter. 000 Clock divided by 1. 001 Clock divided by 2. 010 Clock divided by 4. 011 Clock divided by 8. 100 Clock divided by 16. 101 Clock divided by 32. 110 Clock divided by 64. 111 Clock divided by 128.

20.4.3 Pulse Width Timer Positive Pulse Width Register: High (PWTx_PPH)

Address: Base address + 2h offset

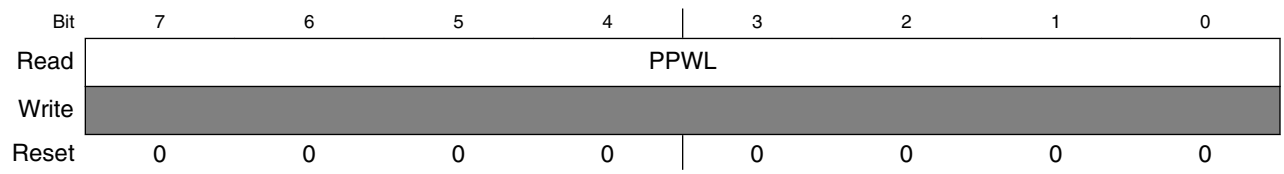
Bit	7	6	5	4	3	2	1	0
Read	PPWH							
Write								
Reset	0	0	0	0	0	0	0	0

PWTx_PPH field descriptions

Field	Description
PPWH	Positive Pulse Width[15:8] High byte of captured positive pulse width value.

20.4.4 Pulse Width Timer Positive Pulse Width Register: Loq (PWTx_PPL)

Address: Base address + 3h offset

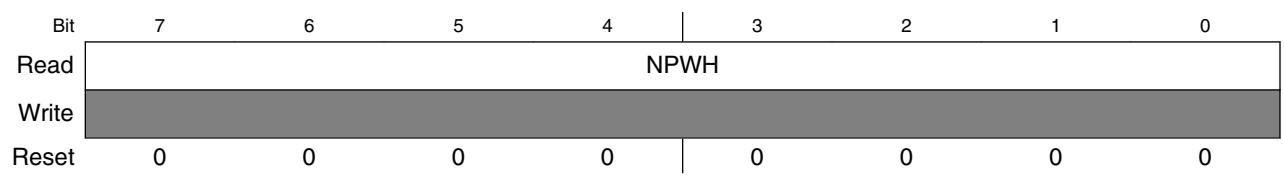


PWTx_PPL field descriptions

Field	Description
PPWL	Positive Pulse Width[7:0] Low byte of captured positive pulse width value.

20.4.5 Pulse Width Timer Negative Pulse Width Register: High (PWTx_NPH)

Address: Base address + 4h offset

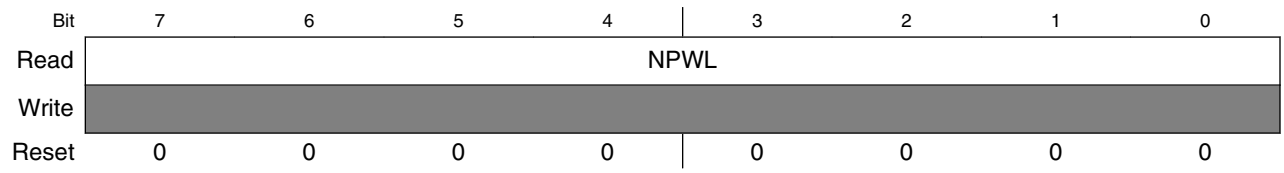


PWTx_NPH field descriptions

Field	Description
NPWH	Negative Pulse Width[15:8] High byte of captured negative pulse width value.

20.4.6 Pulse Width Timer Negative Pulse Width Register: Low (PWTx_NPL)

Address: Base address + 5h offset

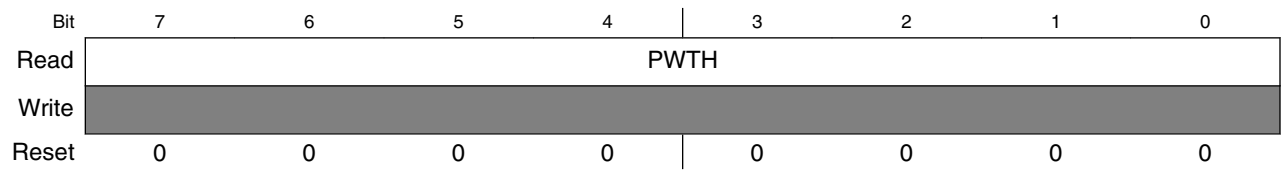


PWTx_NPL field descriptions

Field	Description
NPWL	Negative Pulse Width[7:0] Low byte of captured negative pulse width value.

20.4.7 Pulse Width Timer Counter Register: High (PWTx_CNTH)

Address: Base address + 6h offset

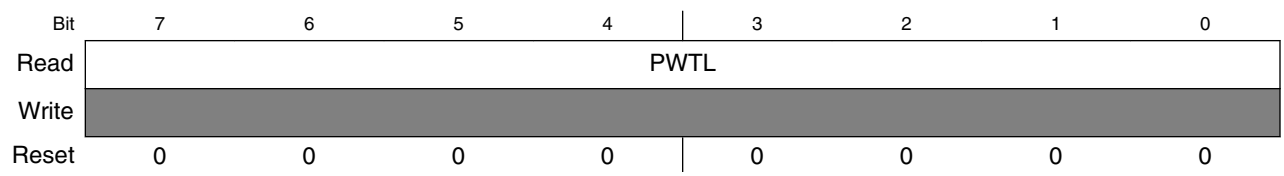


PWTx_CNTH field descriptions

Field	Description
PWTH	PWT counter[15:8] High byte of PWT counter register.

20.4.8 Pulse Width Timer Counter Register: Low (PWTx_CNTL)

Address: Base address + 7h offset



PWTx_CNTL field descriptions

Field	Description
PWTL	PWT counter[7:0] Low byte of PWT counter register.

20.5 Functional description

20.5.1 PWT counter and PWT clock pre-scaler

The pulse width timer (PWT) measures duration of a pulse or the period of a signal input to the PWTIN by a 16-bit free running counter (PWT_CNTH:L). There is a clock pre-scaler (CLKPRE) in PWT module that provides the frequency divided clock to the PWT_CNTH:L.. The clock pre-scaler can select clock input from bus clock and alternative clock by PWT_CR[PCLKS].

The PWT counter uses the frequency divided clock from CLKPRE for counter advancing. The frequency of pre-scaler is programmable as the clock frequency divided by 1, 2, 4, 8, 16, 32, 64, 128 (depending on the setting of PRE[2:0]).

When PWT_CNT is running, any edge to be measured after the trigger edge causes the value of the PWT_CNT to be uploaded to the appropriate pulse width registers. At the same time, PWT_CNT will be reset to $0000 and the clock pre-scaler output will also be reset together. PWT_CNT will then start advancing again with the input clock. If the PWTxCNT runs from 0xFFFF to 0x0000, the PWTOV bit is set.

20.5.2 Edge detection and capture control

The edge detection and capture control part detects measurement trigger edges and controls when and which pulse width register(s) will be updated.

The edge detection logic determines from which edge appeared on PWTIN the pulse width starts to be measured, when and which pulse width registers should be updated.

The PWTIN can be selected from one of four sources by configuring PINSEL[1:0].

As soon as the PWT is enabled, the 16-bit free counter will begin to count up until a edge transistion on the selected PWTIN. Determined by PWT_CS[FCTLE] and PWTIN state, the counter contents can be uploaded to the corresponding registers.

If PWT_CS[FCTLE] is cleared to 0, the first 16-bit free counter content will just be ignored and not uploaded to neither PWT_PPH:L nor PWT_NPH:L. Otherwise, detemined by current PWTIN state(as signalized by PWT_CR[LVL]), the counter content will be uploaded to PWT_PPH:L if PWT_CR[LVL] is 1 and PWT_NPH:L if PWT_CR[LVL] is 0.

In normal measurement, when the PWT_CS[PWTRDY] is set, software can then read out the positive pulse width and negative pulse width values from PWT_PPH:L and PWT_NPH:L respectively and the selected PWTIN duty ratio can then be calculated. The exception is when overflow happens, software need to check PWT_CR[TGL] and PWT_CR[LVL] to determine if it is low overflow(0 duty ratio) ,high overflow(100% duty ratio), toggled low overflow or toggled high overflow. Below table 1 shows the meaning:

Table 20-3. Abnormal PWTIN duty ratio

Flag	PWT_CR[TGL]	PWT_CR[LVL]	Description
PWT_CS[PWTOV]	0	0	Low overflow
	0	1	High overflow
	1	0	Toggled low overflow
	1	1	Toggled high overflow

The following figure illustrates the trigger edge detection and pulse width registers update of PWT.

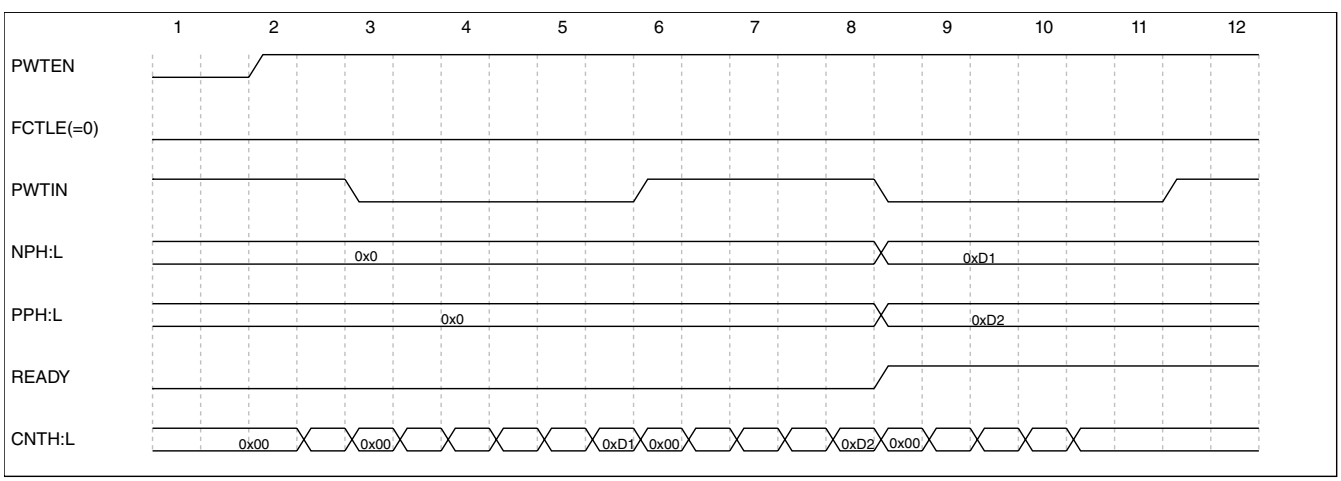


Figure 20-2. PWT normal measurement with FCTLE = 0

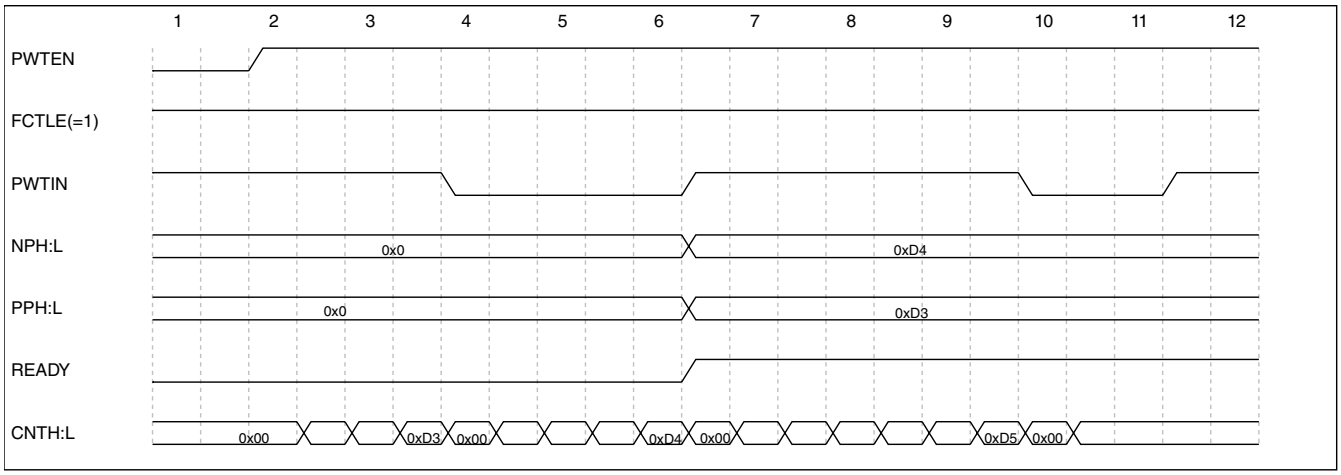


Figure 20-3. PWT normal measurement with FCTLE = 1

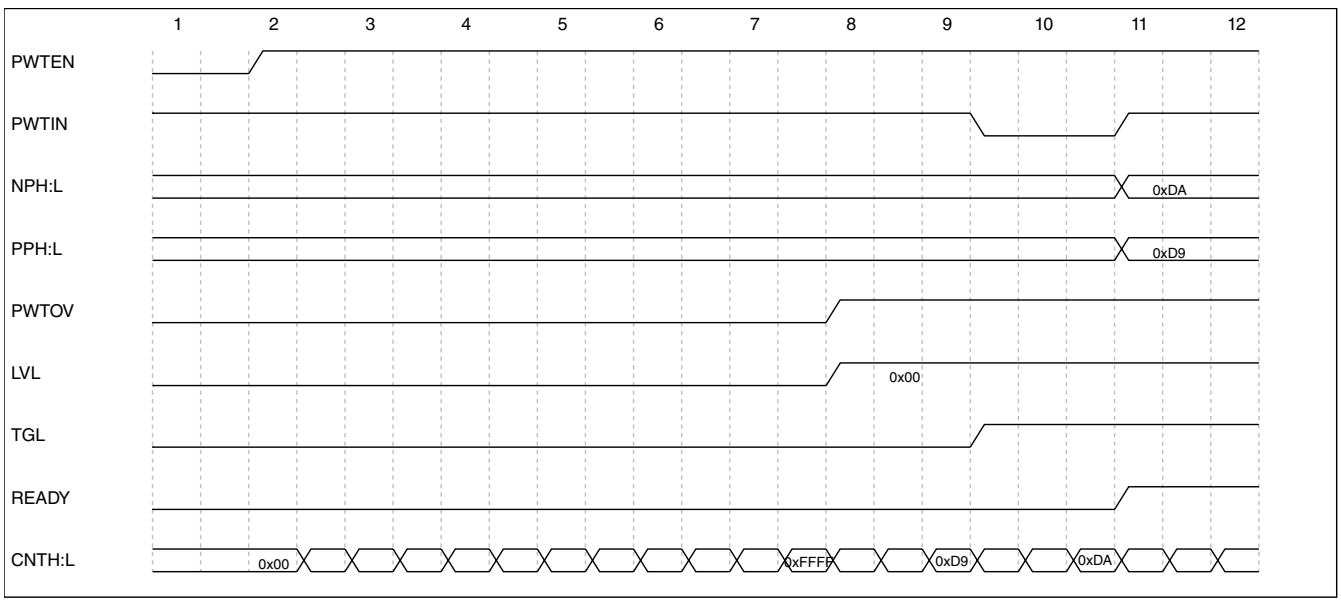


Figure 20-4. PWT measurement overflows at high level with FCTLE = 1

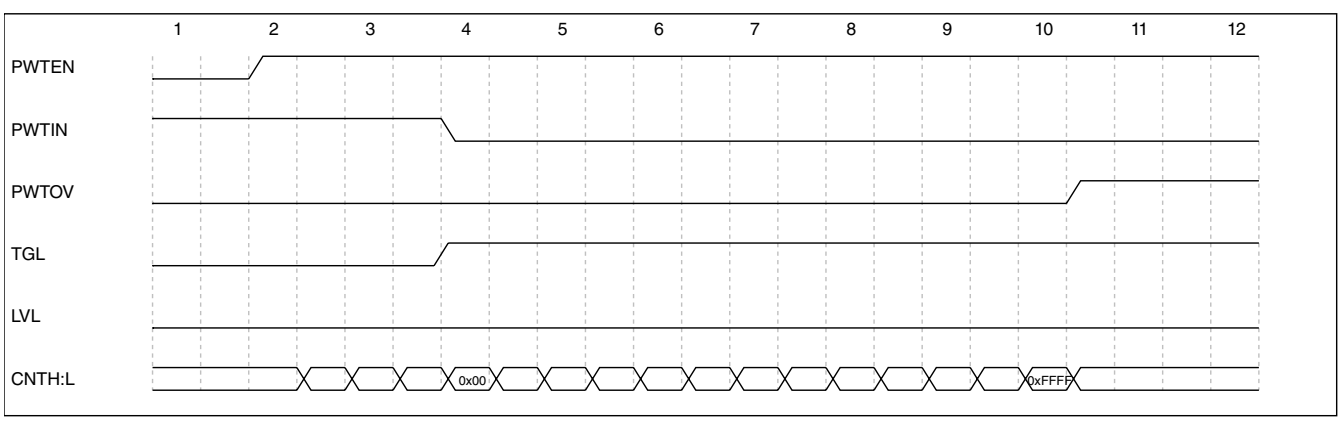


Figure 20-5. PWT measurement overflows with PWTIN toggles

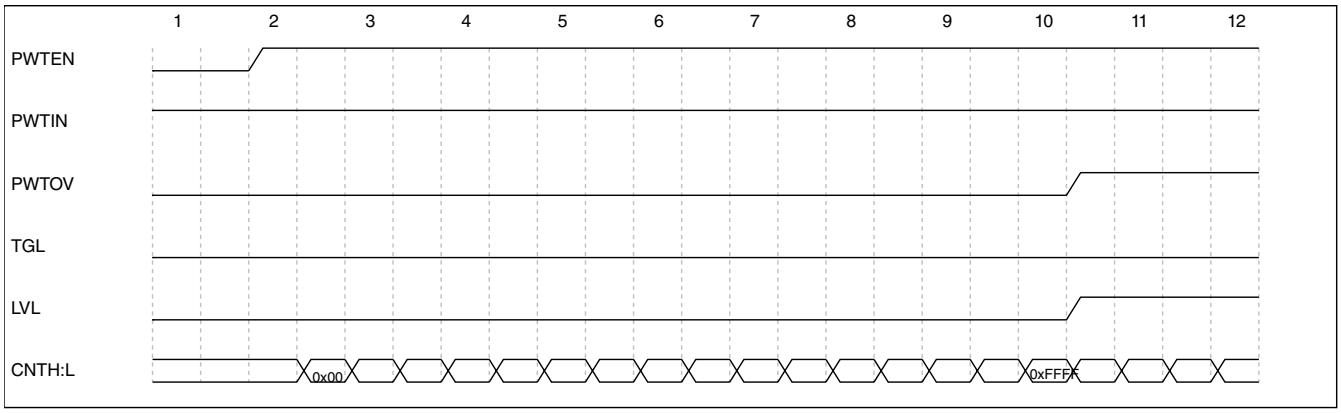


Figure 20-6. PWT measurement overflows without PWTIN toggles

The PWTRDY flag bit indicates that the data can be read in PWTxPPH:L and/or PWTxNPH:L, whenever there is a valid edge transition happened on the selected PWTIN.

When PWTRDY bit is set, the updated pulse width register(s) transfers the data to corresponding 16-bit read buffer(s). The read value of pulse width registers actually comes from the corresponding read buffers, whenever the chip is in normal run mode or BDM mode. Reading followed by writing 0 to the PWTRDY flag clears this bit. Until the PWTRDY bit is cleared, the 16-bit read buffer(s) cannot be updated. But this does not affect the upload of pulse width registers from the PWT counter.

If another pulse measurement is completed and the pulse width registers are updated, the clearing of the PWTRDY flag fails, i.e., the PWTRDY will still be set, but the 16-bit read buffer(s) will be updated again as long as the action is cleared. The user should complete the pulse width data reading before clearing the PWTRDY flag to avoid missing data. This mechanism assures that the second pulse measurement will not be lost in case the MCU does not have enough time to read the first one ready for read. The mechanism is automatically restarted by an MCU reset , writing 1 to PWTSR bit or writing a 0 to PWTEN bit followed by writing a 1 to it.

The following figure illustrates the buffering mechanism of pulse width register:

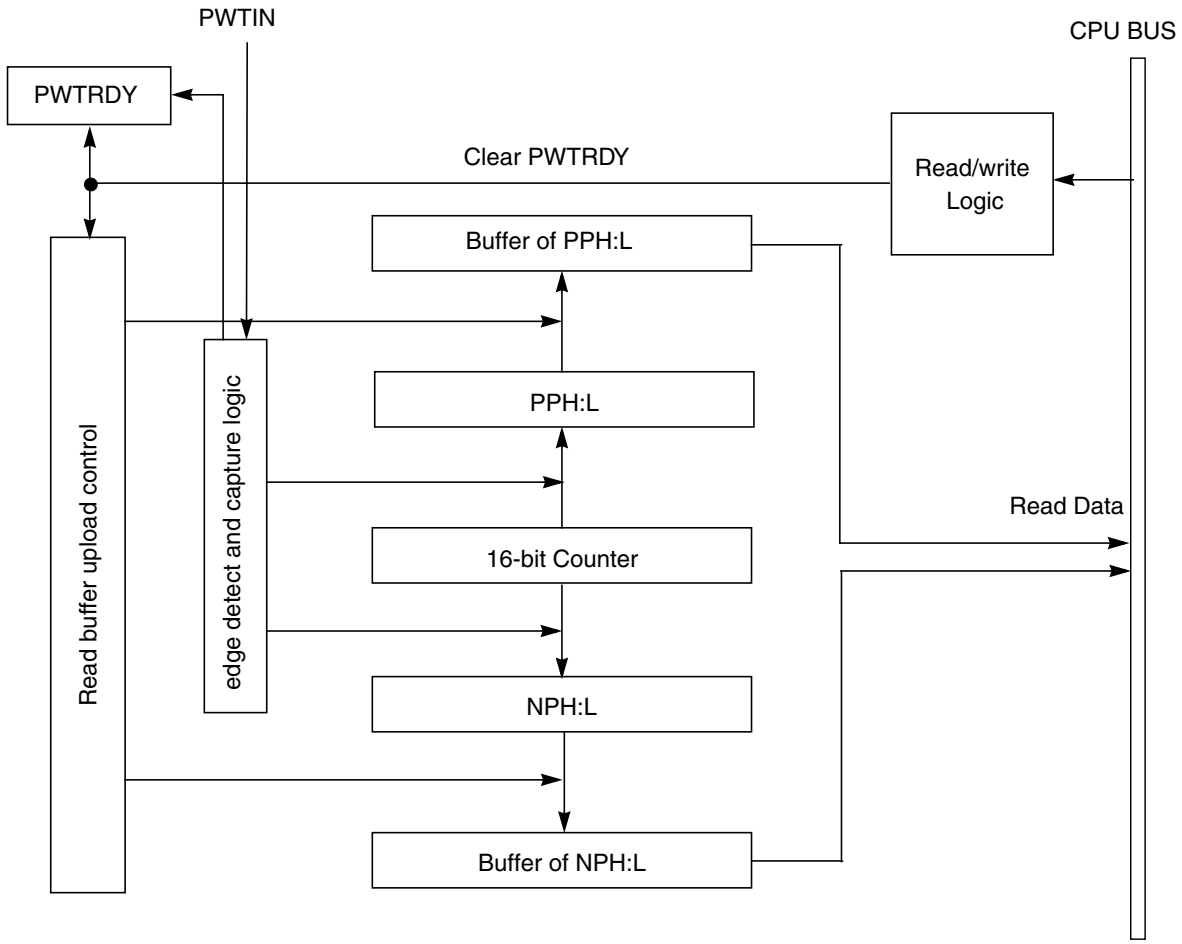


Figure 20-7. Buffering mechanism of pulse width register

When PWT completes any pulse width measurement, a signal is generated to reset PWTxCNTH:L and the clock pre-scaler output after the data has been uploaded to the pulse width registers.. To assure that there is no missing count, the PWTxCNTH:L and the clock pre-scaler output are reset in a bus clock cycle after the completion of a pulse width measurement.

20.6 Reset overview

20.6.1 Description of reset operation

PWT soft reset is built into PWT as a mechanism used to reset/restart the pulse width timer. The PWT soft reset is triggered by writing 1 to the PWTSR bit. (This bit always reads 0). Unlike reset by the CPU, the PWT reset does not restore everything in the PWT to its reset state. The following occurs

1. The PWT counter is set to 0x0000

2. The 16-bit buffer of PWT counter is reset and the reading coherency mechanism is restarted
3. The PWT clock pre-scaler output is reset
4. The edge detection logic is reset
5. The capture logic is reset and the latching mechanism of pulse width registers is also restarted.
6. PWTxPPH, PWTxPPL, PWTxNPH, PWTxNPL are set to 0x00
7. PWTOV, PWTRDY, TGL and LVL status are set to 0
8. All other PWT register settings are not changed

Writing a 0 to PWTEN bit also has the above effects except that the reset state will be held until the PWTEN bit is set to 1.

20.7 Interrupts

20.7.1 Description of interrupt operation

The other major component of the PWT is the interrupts control logic. When the PWTOV bit and POVIE bit of PWTxCS are set, a PWT overflow interrupt can be generated. When PWTRDY bit and PRDYIE bit of PWTxCS are set, a pulse width data ready interrupt can be generated. The PWTIE bit of PWTxCS controls the interrupt generation of the PWT module. The functionality of the PWT is not affected while the interrupt is being generated.

20.7.2 Application examples

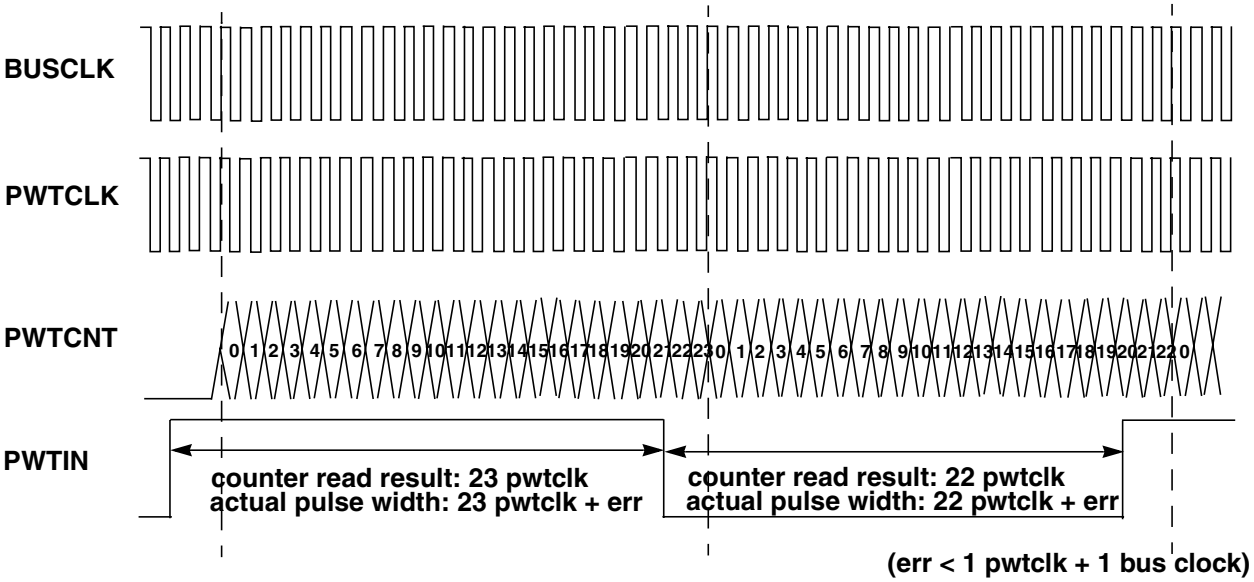


Figure 20-8. Example at PWTCLK is bus clock divided by 1

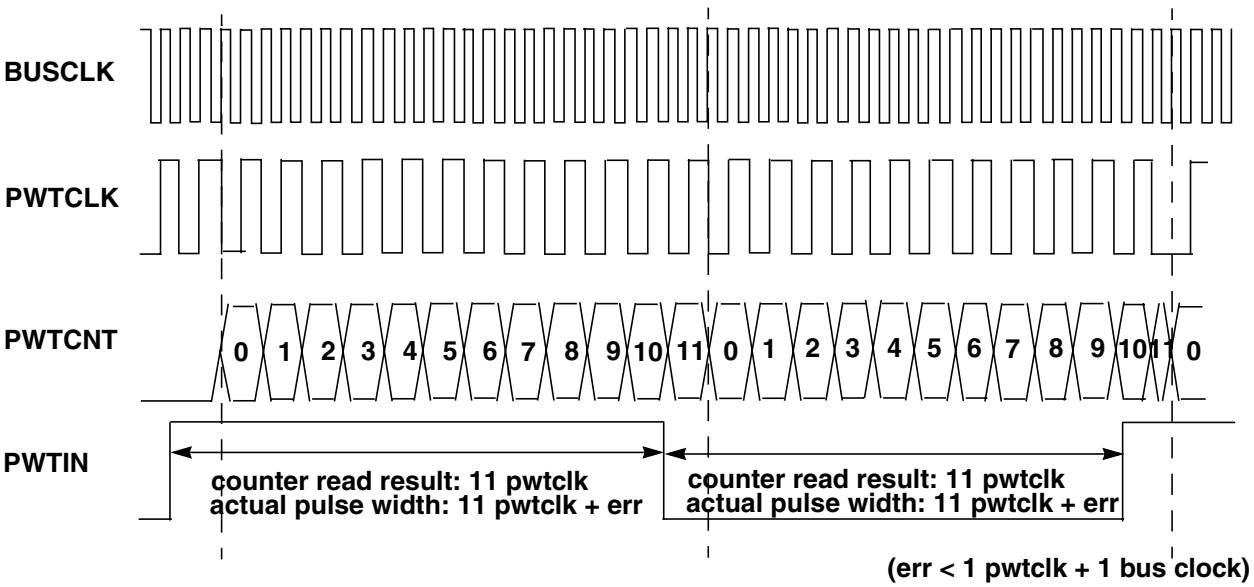


Figure 20-9. Example at PWTCLK is Bus Clock divided by 2

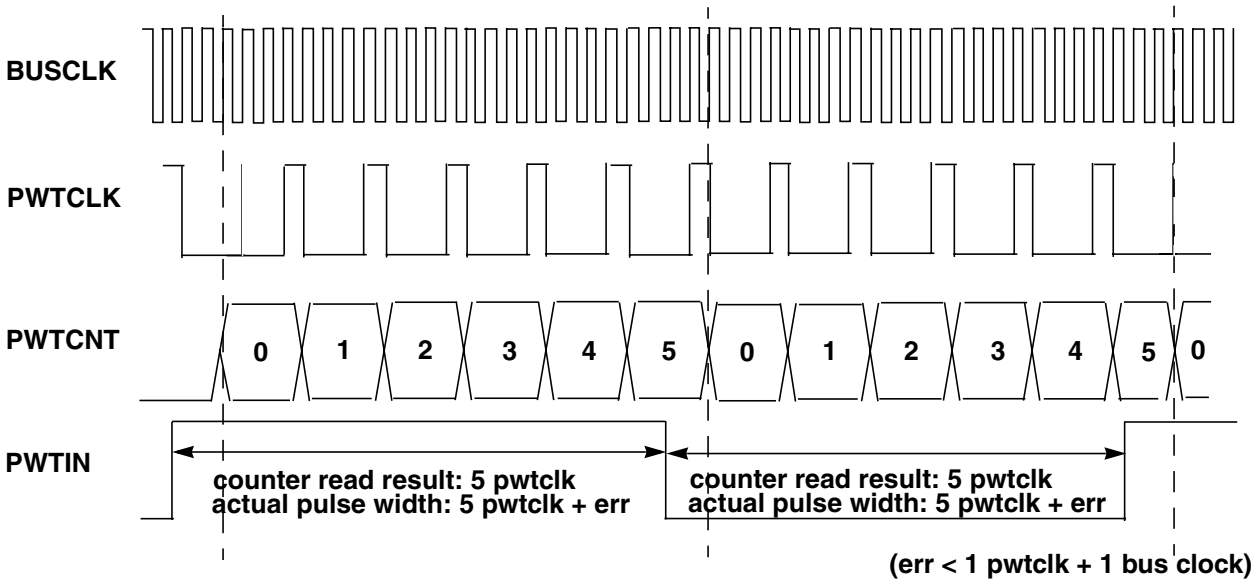


Figure 20-10. Example at PWTCLK is bus clock divided by 4

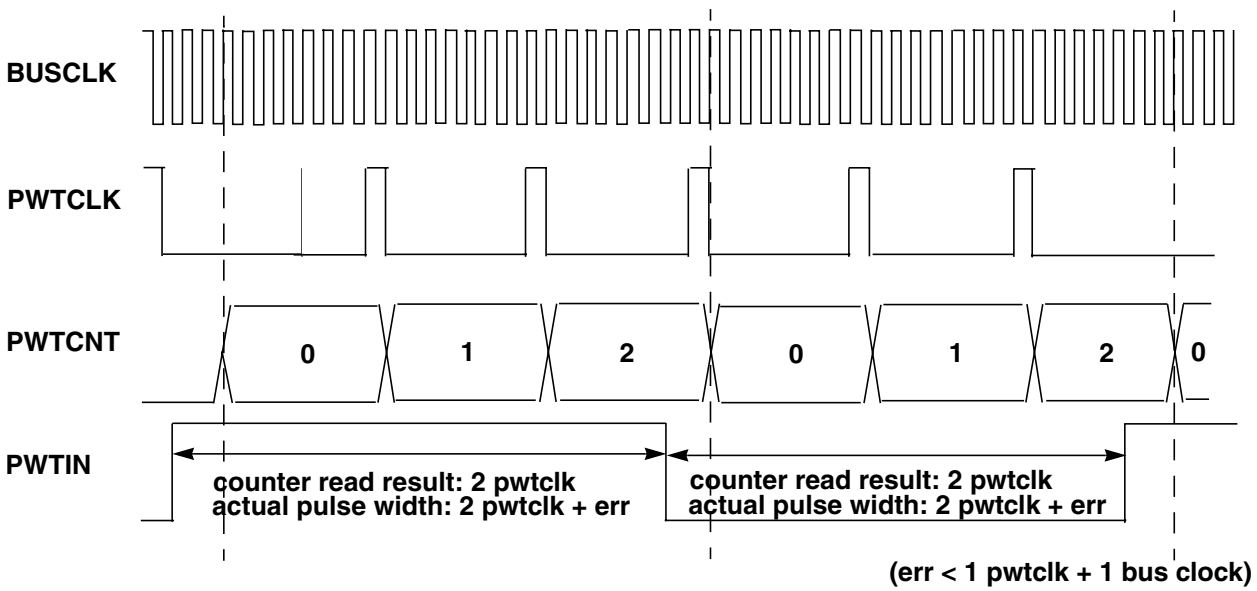


Figure 20-11. Example at PWTCLK is bus clock divided by 8

20.8 Initialization/Application information

Following are the recommended steps to initialize the PWT module:

1. Configure PWTxCR to select clock source, set pre-scaler rate, select PWT input pin and edge detection mode.
2. Set PWTIE, PRDYIE and POVIE bits in PWTxCS if corresponding interrupt is desired to be generated.
3. Set PWTEN bit in PWTxCS to enable the pulse width measurement.

The step 1 and 2 can be sequential or not, but they must be completed before step 3 to ensure all settings are ready before pulse width measurement is enabled.

Chapter 21
Inter-Integrated Circuit (I2C)

21.1 Chip specific inter-integrated circuit

This device contains an inter-integrated circuit (I2C) module for communication with other integrated circuits. It supports up to 400 kb/s communication speed. The digital glitch filter implemented in the PORT module, controlled by the PORT registers, is clocked from the bus clock and thus has filter granularity in bus clock cycle counts.

Customization:

- Primary clock: BUSCLK (20 MHz)
- Alternate clock: None
- Enable SMBus feature
- The following table summarizes the signal connection of I2C module.

Table 21-1. I2C module signals connection

Module	Signal	Connect to
I2C	SDA	PTA5/SDA
	SCL	PTA4/SCL
	Internal clock	BUSCLK

NOTE

This module supports wakeup in Wait and Stop modes.

21.2 Introduction

NOTE

For the chip-specific implementation details of this module's instances, see the chip configuration information.

The inter-integrated circuit (I^2C, I2C, or IIC) module provides a method of communication between a number of devices.

The interface is designed to operate up to at least 400 kbit/s with maximum bus loading and timing. The I2C device is capable of operating at higher baud rates, up to a maximum of clock/20, with reduced bus loading. The maximum communication length and the number of devices that can be connected are limited by a maximum bus capacitance of 400 pF. The I2C module also complies with the *System Management Bus (SMBus) Specification, version 2*.

21.2.1 Features

The I2C module has the following features:

- Compatible with *The I^2C-Bus Specification*
- Multimaster operation
- Software programmable for one of 64 different serial clock frequencies
- Software-selectable acknowledge bit
- Interrupt-driven byte-by-byte data transfer
- Arbitration-lost interrupt with automatic mode switching from master to slave
- Calling address identification interrupt
- START and STOP signal generation and detection
- Repeated START signal generation and detection
- Acknowledge bit generation and detection
- Bus busy detection
- General call recognition
- 10-bit address extension
- Support for *System Management Bus (SMBus) Specification, version 2*
- Low power mode wakeup on slave address match
- Range slave address support
- Double buffering support to achieve higher baud rate

21.2.2 Modes of operation

The I2C module's operation in various low power modes is as follows:

- Run mode: This is the basic mode of operation. To conserve power in this mode, disable the module.

- Wait mode: The module continues to operate when the core is in Wait mode and can provide a wakeup interrupt.
- Stop mode: The module is inactive in Stop3 mode for reduced power consumption, except that address matching is enabled in Stop3 mode. The STOP instruction does not affect the I2C module's register states.

21.2.3 Block diagram

The following figure is a functional block diagram of the I2C module.

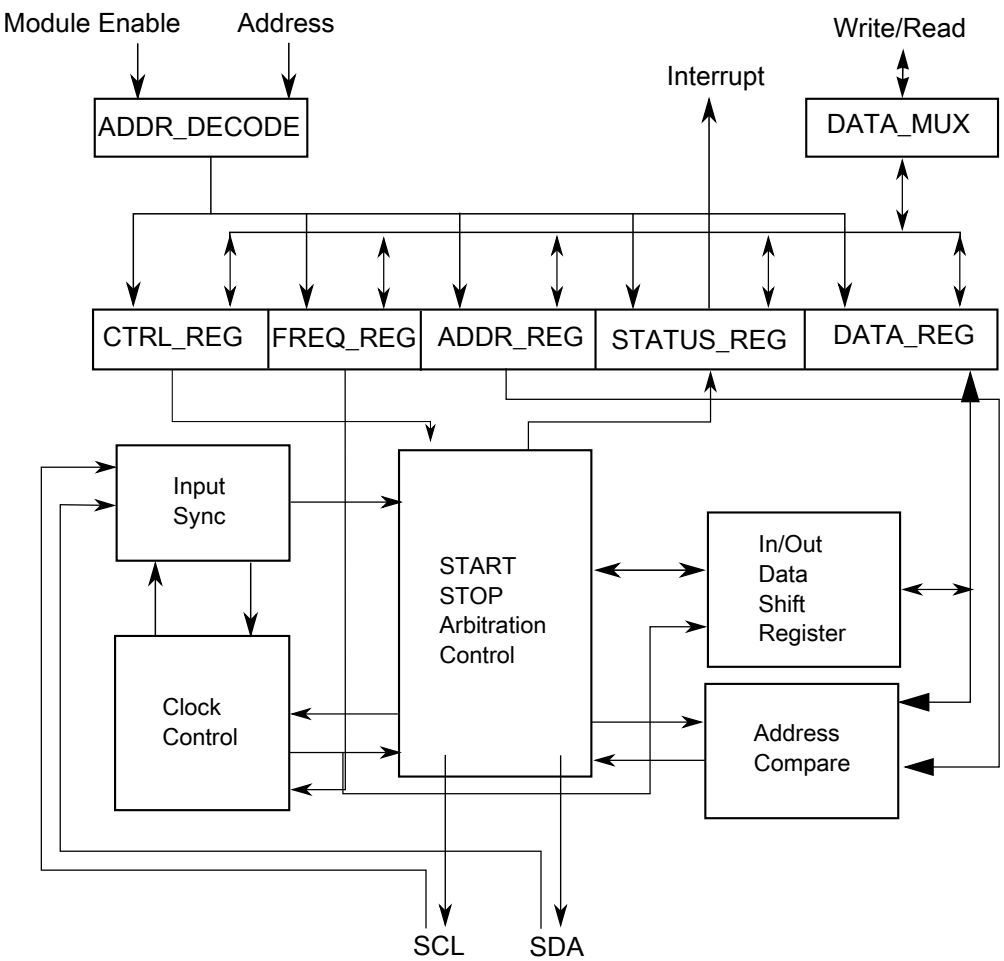


Figure 21-1. I2C Functional block diagram

21.3 I²C signal descriptions

The signal properties of I²C are shown in the table found here.

Table 21-2. I²C signal descriptions

Signal	Description	I/O
SCL	Bidirectional serial clock line of the I²C system.	I/O
SDA	Bidirectional serial data line of the I²C system.	I/O

21.4 Memory map/register definition

This section describes in detail all I2C registers accessible to the end user.

I2C memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
18B0	I2C Address Register 1 (I2C_A1)	8	R/W	00h	21.4.1/365
18B1	I2C Frequency Divider register (I2C_F)	8	R/W	00h	21.4.2/365
18B2	I2C Control Register 1 (I2C_C1)	8	R/W	00h	21.4.3/366
18B3	I2C Status register (I2C_S)	8	R/W	80h	21.4.4/368
18B4	I2C Data I/O register (I2C_D)	8	R/W	00h	21.4.5/369
18B5	I2C Control Register 2 (I2C_C2)	8	R/W	00h	21.4.6/370
18B6	I2C Stop Control and Status Register (I2C_SCS)	8	R/W	00h	21.4.7/371
18B7	I2C Range Address register (I2C_RA)	8	R/W	00h	21.4.8/372
18B8	I2C SMBus Control and Status register (I2C_SMB)	8	R/W	00h	21.4.9/373
18B9	I2C Address Register 2 (I2C_A2)	8	R/W	C2h	21.4.10/374
18BA	I2C SCL Low Timeout Register High (I2C_SLTH)	8	R/W	00h	21.4.11/375
18BB	I2C SCL Low Timeout Register Low (I2C_SLTL)	8	R/W	00h	21.4.12/375
18BC	I2C Status register 2 (I2C_S2)	8	R/W	01h	21.4.13/376

21.4.1 I2C Address Register 1 (I2C_A1)

This register contains the slave address to be used by the I2C module.

Address: 18B0h base + 0h offset = 18B0h

Bit	7	6	5	4	3	2	1	0
Read	AD[7:1]							0
Write								
Reset	0	0	0	0	0	0	0	0

I2C_A1 field descriptions

Field	Description
7–1 AD[7:1]	Address Contains the primary slave address used by the I2C module when it is addressed as a slave. This field is used in the 7-bit address scheme and the lower seven bits in the 10-bit address scheme.
0 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

21.4.2 I2C Frequency Divider register (I2C_F)

Address: 18B0h base + 1h offset = 18B1h

Bit	7	6	5	4	3	2	1	0
Read	MULT		ICR					
Write								
Reset	0	0	0	0	0	0	0	0

I2C_F field descriptions

Field	Description
7–6 MULT	Multiplier Factor Defines the multiplier factor (mul). This factor is used along with the SCL divider to generate the I2C baud rate. 00 mul = 1 01 mul = 2 10 mul = 4 11 Reserved
ICR	ClockRate Prescales the I2C module clock for bit rate selection. This field and the MULT field determine the I2C baud rate, the SDA hold time, the SCL start hold time, and the SCL stop hold time. For a list of values corresponding to each ICR setting, see I2C divider and hold values. The SCL divider multiplied by multiplier factor (mul) determines the I2C baud rate. `I2C baud rate = I2C module clock speed (Hz)/(mul × SCL divider)`

Table continues on the next page...

I2C_F field descriptions (continued)

Field	Description

	The SDA hold time is the delay from the falling edge of SCL (I2C clock) to the changing of SDA (I2C data).
	
	`SDA hold time = I2C module clock period (s) × mul × SDA hold value`
	
	The SCL start hold time is the delay from the falling edge of SDA (I2C data) while SCL is high (start condition) to the falling edge of SCL (I2C clock).
	
	`SCL start hold time = I2C module clock period (s) × mul × SCL start hold value`
	
	The SCL stop hold time is the delay from the rising edge of SCL (I2C clock) to the rising edge of SDA (I2C data) while SCL is high (stop condition).
	
	`SCL stop hold time = I2C module clock period (s) × mul × SCL stop hold value`
	
	For example, if the I2C module clock speed is 8 MHz, the following table shows the possible hold time values with different ICR and MULT selections to achieve an I^2C baud rate of 100 kbit/s.

MULT	ICR	Hold times (µs)		
		SDA	SCL Start	SCL Stop
2h	00h	3.500	3.000	5.500
1h	07h	2.500	4.000	5.250
1h	0Bh	2.250	4.000	5.250
0h	14h	2.125	4.250	5.125
0h	18h	1.125	4.750	5.125

21.4.3 I2C Control Register 1 (I2C_C1)

Address: 18B0h base + 2h offset = 18B2h

Bit	7	6	5	4	3	2	1	0
Read	IICEN	IICIE	MST	TX	TXAK	0	WUEN	0
Write						RSTA		
Reset	0	0	0	0	0	0	0	0

I2C_C1 field descriptions

Field	Description
7 IICEN	I2C Enable Enables I2C module operation. 0 Disabled 1 Enabled
6 IICIE	I2C Interrupt Enable Enables I2C interrupt requests.

Table continues on the next page...

I2C_C1 field descriptions (continued)

Field	Description
	0 Disabled 1 Enabled
5 MST	Master Mode Select When MST is changed from 0 to 1, a START signal is generated on the bus and master mode is selected. When this bit changes from 1 to 0, a STOP signal is generated and the mode of operation changes from master to slave. 0 Slave mode 1 Master mode
4 TX	Transmit Mode Select Selects the direction of master and slave transfers. In master mode this bit must be set according to the type of transfer required. Therefore, for address cycles, this bit is always set. When addressed as a slave this bit must be set by software according to the SRW bit in the status register. 0 Receive 1 Transmit
3 TXAK	Transmit Acknowledge Enable Specifies the value driven onto the SDA during data acknowledge cycles for both master and slave receivers. The value of SMB[FACK] affects NACK/ACK generation. **NOTE:** SCL is held low until TXAK is written. 0 An acknowledge signal is sent to the bus on the following receiving byte (if FACK is cleared) or the current receiving byte (if FACK is set). 1 No acknowledge signal is sent to the bus on the following receiving data byte (if FACK is cleared) or the current receiving data byte (if FACK is set).
2 RSTA	Repeat START Writing 1 to this bit generates a repeated START condition provided it is the current master. This bit will always be read as 0. Attempting a repeat at the wrong time results in loss of arbitration.
1 WUEN	Wakeup Enable The I2C module can wake the MCU from low power mode with no peripheral bus running when slave address matching occurs. 0 Normal operation. No interrupt generated when address matching in low power mode. 1 Enables the wakeup function in low power mode.
0 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

21.4.4 I2C Status register (I2C_S)

Address: 18B0h base + 3h offset = 18B3h

Bit	7	6	5	4	3	2	1	0
Read	TCF	IAAS	BUSY	ARBL	RAM	SRW	IICIF	RXAK
Write				w1c			w1c	
Reset	1	0	0	0	0	0	0	0

I2C_S field descriptions

Field	Description
7 TCF	Transfer Complete Flag Acknowledges a byte transfer; TCF is set on the completion of a byte transfer. This bit is valid only during or immediately following a transfer to or from the I2C module. TCF is cleared by reading the I2C data register in receive mode or by writing to the I2C data register in transmit mode. **NOTE:** In the buffer mode, TCF is cleared automatically by internal reading or writing the data register I2C_D, with no need waiting for manually reading/writing the I2C data register in Rx/Tx mode. 0 Transfer in progress 1 Transfer complete
6 IAAS	Addressed As A Slave This bit is set by one of the following conditions: • The calling address matches the programmed primary slave address in the A1 register, or matches the range address in the RA register (which must be set to a nonzero value and under the condition I2C_C2[RMEN] = 1). • C2[GCAEN] is set and a general call is received. • SMB[SIICAEN] is set and the calling address matches the second programmed slave address. • ALERTEN is set and an SMBus alert response address is received • RMEN is set and an address is received that is within the range between the values of the A1 and RA registers. IAAS sets before the ACK bit. The CPU must check the SRW bit and set TX/RX accordingly. Writing the C1 register with any value clears this bit. 0 Not addressed 1 Addressed as a slave
5 BUSY	Bus Busy Indicates the status of the bus regardless of slave or master mode. This bit is set when a START signal is detected and cleared when a STOP signal is detected. 0 Bus is idle 1 Bus is busy
4 ARBL	Arbitration Lost This bit is set by hardware when the arbitration procedure is lost. The ARBL bit must be cleared by software, by writing 1 to it.

Table continues on the next page...

I2C_S field descriptions (continued)

Field	Description
	0 Standard bus operation. 1 Loss of arbitration.
3 RAM	Range Address Match This bit is set to 1 by any of the following conditions, if I2C_C2[RMEN] = 1: • Any nonzero calling address is received that matches the address in the RA register. • The calling address is within the range of values of the A1 and RA registers. Writing the C1 register with any value clears this bit to 0. 0 Not addressed 1 Addressed as a slave
2 SRW	Slave Read/Write When addressed as a slave, SRW indicates the value of the R/$\overline{W}$ command bit of the calling address sent to the master. 0 Slave receive, master writing to slave 1 Slave transmit, master reading from slave
1 IICIF	Interrupt Flag This bit sets when an interrupt is pending. This bit must be cleared by software by writing 1 to it, such as in the interrupt routine. One of the following events can set this bit: • One byte transfer, including ACK/NACK bit, completes if FACK is 0. An ACK or NACK is sent on the bus by writing 0 or 1 to TXAK after this bit is set in receive mode. • One byte transfer, excluding ACK/NACK bit, completes if FACK is 1. • Match of slave address to calling address including primary slave address, range slave address, alert response address, second slave address, or general call address. • Arbitration lost • In SMBus mode, any timeouts except SCL and SDA high timeouts • I2C bus stop or start detection if the SSIE bit in the Input Glitch Filter register is 1 **NOTE:** To clear the I2C bus stop or start detection interrupt: In the interrupt service routine, first clear the STOPF or STARTF bit in the Input Glitch Filter register by writing 1 to it, and then clear the IICIF bit. If this sequence is reversed, the IICIF bit is asserted again. 0 No interrupt pending 1 Interrupt pending
0 RXAK	Receive Acknowledge 0 Acknowledge signal was received after the completion of one byte of data transmission on the bus 1 No acknowledge signal detected

21.4.5 I2C Data I/O register (I2C_D)

Address: 18B0h base + 4h offset = 18B4h

Bit	7	6	5	4	3	2	1	0
Read Write	DATA							
Reset	0	0	0	0	0	0	0	0

I2C_D field descriptions

Field	Description
DATA	Data In master transmit mode, when data is written to this register, a data transfer is initiated. The most significant bit is sent first. In master receive mode, reading this register initiates receiving of the next byte of data. **NOTE:** When making the transition out of master receive mode, switch the I2C mode before reading the Data register to prevent an inadvertent initiation of a master receive data transfer. In slave mode, the same functions are available after an address match occurs. The C1[TX] bit must correctly reflect the desired direction of transfer in master and slave modes for the transmission to begin. For example, if the I2C module is configured for master transmit but a master receive is desired, reading the Data register does not initiate the receive. Reading the Data register returns the last byte received while the I2C module is configured in master receive or slave receive mode. The Data register does not reflect every byte that is transmitted on the I2C bus, and neither can software verify that a byte has been written to the Data register correctly by reading it back. In master transmit mode, the first byte of data written to the Data register following assertion of MST (start bit) or assertion of RSTA (repeated start bit) is used for the address transfer and must consist of the calling address (in bits 7-1) concatenated with the required R/$\overline{W}$ bit (in position bit 0).

21.4.6 I2C Control Register 2 (I2C_C2)

Address: 18B0h base + 5h offset = 18B5h

Bit	7	6	5	4	3	2	1	0
Read Write	GCAEN	ADEXT	0	0	RMEN	AD[10:8]		
Reset	0	0	0	0	0	0	0	0

I2C_C2 field descriptions

Field	Description
7 GCAEN	General Call Address Enable Enables general call address. 0 Disabled 1 Enabled
6 ADEXT	Address Extension Controls the number of bits used for the slave address. 0 7-bit address scheme 1 10-bit address scheme
5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

Table continues on the next page...

I2C_C2 field descriptions (continued)

Field	Description
3 RMEN	Range Address Matching Enable This bit controls the slave address matching for addresses between the values of the A1 and RA registers. When this bit is set, a slave address matching occurs for any address greater than the value of the A1 register and less than or equal to the value of the RA register. 0 Range mode disabled. No address matching occurs for an address within the range of values of the A1 and RA registers. 1 Range mode enabled. Address matching occurs when a slave receives an address within the range of values of the A1 and RA registers.
AD[10:8]	Slave Address Contains the upper three bits of the slave address in the 10-bit address scheme. This field is valid only while the ADEXT bit is set.

21.4.7 I2C Stop Control and Status Register (I2C_SCS)

Address: 18B0h base + 6h offset = 18B6h

Bit	7	6	5	4	3	2	1	0
Read	SHEN	STOPF	SSIE	STARTF	0			
Write		w1c		w1c				
Reset	0	0	0	0	0	0	0	0

I2C_SCS field descriptions

Field	Description
7 SHEN	Stop Hold Enable Set this bit to hold off entry to stop mode when any data transmission or reception is occurring. The following scenario explains the holdoff functionality: 1. The I2C module is configured for a basic transfer, and the SHEN bit is set to 1. 2. A transfer begins. 3. The MCU signals the I2C module to enter stop mode. 4. The byte currently being transferred, including both address and data, completes its transfer. 5. The I2C slave or master acknowledges that the in-transfer byte completed its transfer and acknowledges the request to enter stop mode. 6. After receiving the I2C module's acknowledgment of the request to enter stop mode, the MCU determines whether to shut off the I2C module's clock. If the SHEN bit is set to 1 and the I2C module is in an idle or disabled state when the MCU signals to enter stop mode, the module immediately acknowledges the request to enter stop mode. If SHEN is cleared to 0 and the overall data transmission or reception that was suspended by stop mode entry was incomplete: To resume the overall transmission or reception after the MCU exits stop mode, software must reinitialize the transfer by resending the address of the slave. If the I2C Control Register 1's IICIE bit was set to 1 before the MCU entered stop mode, system software will receive the interrupt triggered by the I2C Status Register's TCF bit after the MCU wakes from the stop mode.

Table continues on the next page...

I2C_SCS field descriptions (continued)

Field	Description
	0 Stop holdoff is disabled. The MCU's entry to stop mode is not gated. 1 Stop holdoff is enabled.
6 STOPF	I2C Bus Stop Detect Flag Hardware sets this bit when the I2C bus's stop status is detected. The STOPF bit must be cleared by writing 1 to it. **NOTE:** The stop flag is only for the matched slave devices, therefore the master will not respond for it. 0 No stop happens on I2C bus 1 Stop detected on I2C bus
5 SSIE	I2C Bus Stop or Start Interrupt Enable This bit enables the interrupt for I2C bus stop or start detection. **NOTE:** To clear the I2C bus stop or start detection interrupt: In the interrupt service routine, first clear the STOPF or STARTF bit by writing 1 to it, and then clear the IICIF bit in the status register. If this sequence is reversed, the IICIF bit is asserted again. 0 Stop or start detection interrupt is disabled 1 Stop or start detection interrupt is enabled
4 STARTF	I2C Bus Start Detect Flag Hardware sets this bit when the I2C bus's start status is detected. The STARTF bit must be cleared by writing 1 to it. 0 No start happens on I2C bus 1 Start detected on I2C bus
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

21.4.8 I2C Range Address register (I2C_RA)

Address: 18B0h base + 7h offset = 18B7h

Bit	7	6	5	4	3	2	1	0
Read	RAD							0
Write								
Reset	0	0	0	0	0	0	0	0

I2C_RA field descriptions

Field	Description
7–1 RAD	Range Slave Address This field contains the slave address to be used by the I2C module. The field is used in the 7-bit address scheme. If I2C_C2[RMEN] is set to 1, any nonzero value write enables this register. This register value can be considered as a maximum boundary in the range matching mode.
0 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

21.4.9 I2C SMBus Control and Status register (I2C_SMB)

NOTE

When the SCL and SDA signals are held high for a length of time greater than the high timeout period, the SHTF1 flag sets. Before reaching this threshold, while the system is detecting how long these signals are being held high, a master assumes that the bus is free. However, the SHTF1 bit is set to 1 in the bus transmission process with the idle bus state.

NOTE

When the TCKSEL bit is set, there is no need to monitor the SHTF1 bit because the bus speed is too high to match the protocol of SMBus.

Address: 18B0h base + 8h offset = 18B8h

Bit	7	6	5	4	3	2	1	0
Read	FACK	ALERTEN	SIICAEN	TCKSEL	SLTF	SHTF1	SHTF2	SHTF2IE
Write					w1c		w1c	
Reset	0	0	0	0	0	0	0	0

I2C_SMB field descriptions

Field	Description
7 FACK	Fast NACK/ACK Enable For SMBus packet error checking, the CPU must be able to issue an ACK or NACK according to the result of receiving data byte. 0 An ACK or NACK is sent on the following receiving data byte 1 Writing 0 to TXAK after receiving a data byte generates an ACK. Writing 1 to TXAK after receiving a data byte generates a NACK.
6 ALERTEN	SMBus Alert Response Address Enable Enables or disables SMBus alert response address matching. **NOTE:** After the host responds to a device that used the alert response address, you must use software to put the device's address on the bus. The alert protocol is described in the SMBus specification. 0 SMBus alert response address matching is disabled 1 SMBus alert response address matching is enabled
5 SIICAEN	Second I2C Address Enable Enables or disables SMBus device default address. 0 I2C address register 2 matching is disabled 1 I2C address register 2 matching is enabled

Table continues on the next page...

I2C_SMB field descriptions (continued)

Field	Description
4 TCKSEL	Timeout Counter Clock Select Selects the clock source of the timeout counter. 0 Timeout counter counts at the frequency of the I2C module clock / 64 1 Timeout counter counts at the frequency of the I2C module clock
3 SLTF	SCL Low Timeout Flag This bit is set when the SLT register (consisting of the SLTH and SLTL registers) is loaded with a non-zero value (LoValue) and an SCL low timeout occurs. Software clears this bit by writing a logic 1 to it. **NOTE:** The low timeout function is disabled when the SLT register's value is 0. 0 No low timeout occurs 1 Low timeout occurs
2 SHTF1	SCL High Timeout Flag 1 This read-only bit sets when SCL and SDA are held high more than clock × LoValue / 512, which indicates the bus is free. This bit is cleared automatically. 0 No SCL high and SDA high timeout occurs 1 SCL high and SDA high timeout occurs
1 SHTF2	SCL High Timeout Flag 2 This bit sets when SCL is held high and SDA is held low more than clock × LoValue / 512. Software clears this bit by writing 1 to it. 0 No SCL high and SDA low timeout occurs 1 SCL high and SDA low timeout occurs
0 SHTF2IE	SHTF2 Interrupt Enable Enables SCL high and SDA low timeout interrupt. 0 SHTF2 interrupt is disabled 1 SHTF2 interrupt is enabled

21.4.10 I2C Address Register 2 (I2C_A2)

Address: 18B0h base + 9h offset = 18B9h

Bit	7	6	5	4	3	2	1	0
Read	SAD							0
Write								
Reset	1	1	0	0	0	0	1	0

I2C_A2 field descriptions

Field	Description
7–1 SAD	SMBus Address

Table continues on the next page...

I2C_A2 field descriptions (continued)

Field	Description
	Contains the slave address used by the SMBus. This field is used on the device default address or other related addresses.
0 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

21.4.11 I2C SCL Low Timeout Register High (I2C_SLTH)

Address: 18B0h base + Ah offset = 18BAh

Bit	7	6	5	4	3	2	1	0
Read Write	SSLT[15:8]							
Reset	0	0	0	0	0	0	0	0

I2C_SLTH field descriptions

Field	Description
SSLT[15:8]	SSLT[15:8] Most significant byte of SCL low timeout value that determines the timeout period of SCL low.

21.4.12 I2C SCL Low Timeout Register Low (I2C_SLTL)

Address: 18B0h base + Bh offset = 18BBh

Bit	7	6	5	4	3	2	1	0
Read Write	SSLT[7:0]							
Reset	0	0	0	0	0	0	0	0

I2C_SLTL field descriptions

Field	Description
SSLT[7:0]	SSLT[7:0] Least significant byte of SCL low timeout value that determines the timeout period of SCL low.

21.4.13 I2C Status register 2 (I2C_S2)

Address: 18B0h base + Ch offset = 18BCh

Bit	7	6	5	4	3	2	1	0
Read	0	0	0	0	0	0	ERROR	EMPTY
Write							w1c	
Reset	0	0	0	0	0	0	0	1

I2C_S2 field descriptions

Field	Description
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 ERROR	Error flag Indicates if there are read or write errors with the Tx and Rx buffers. 0 The buffer is not full and all write/read operations have no errors. 1 There are 3 or more write/read errors during the data transfer phase (when the Empty flag is not set and the buffer is busy).
0 EMPTY	Empty flag Indicates if the Tx or Rx buffer is empty. 0 Tx or Rx buffer is not empty and cannot be written to, that is new data cannot be loaded into the buffer. 1 Tx or Rx buffer is empty and can be written to, that is new data can be loaded into the buffer.

21.5 Functional description

This section provides a comprehensive functional description of the I2C module.

21.5.1 I2C protocol

The I2C bus system uses a serial data line (SDA) and a serial clock line (SCL) for data transfers.

All devices connected to it must have open drain or open collector outputs. A logic AND function is exercised on both lines with external pull-up resistors. The value of these resistors depends on the system.

Normally, a standard instance of communication is composed of four parts:

1. START signal
2. Slave address transmission
3. Data transfer
4. STOP signal

The STOP signal should not be confused with the CPU STOP instruction. The following figure illustrates I2C bus system communication.

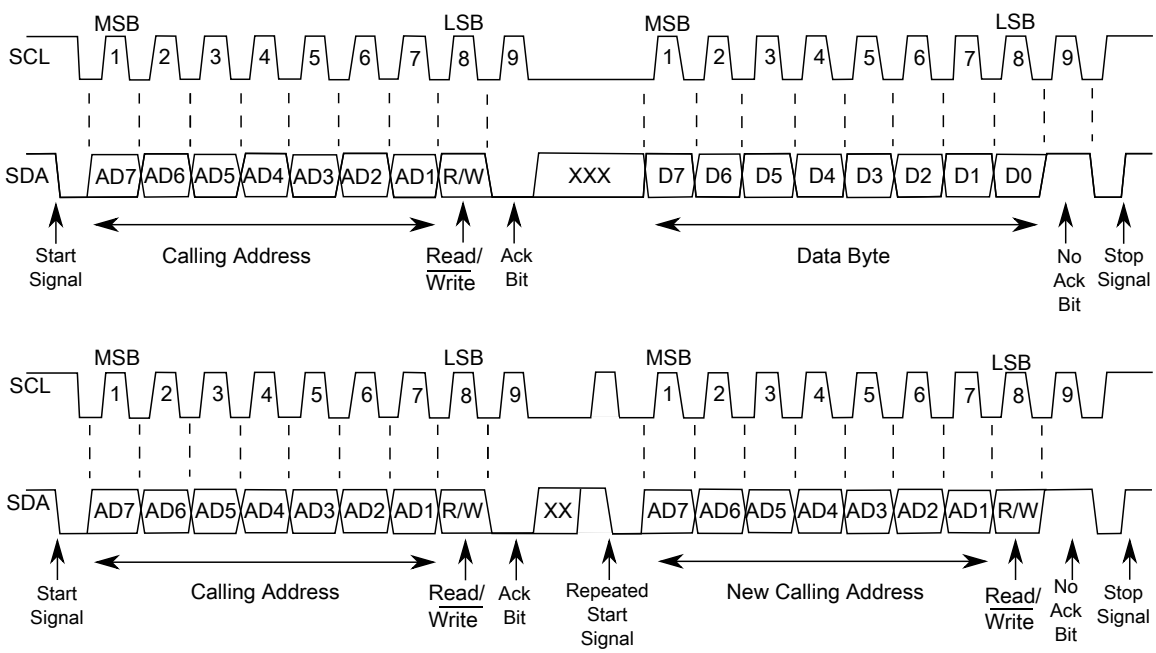


Figure 21-2. I2C bus transmission signals

21.5.1.1 START signal

The bus is free when no master device is engaging the bus (both SCL and SDA are high). When the bus is free, a master may initiate communication by sending a START signal. A START signal is defined as a high-to-low transition of SDA while SCL is high. This signal denotes the beginning of a new data transfer—each data transfer might contain several bytes of data—and brings all slaves out of their idle states.

21.5.1.2 Slave address transmission

Immediately after the START signal, the first byte of a data transfer is the slave address transmitted by the master. This address is a 7-bit calling address followed by an R/$\overline{W}$ bit. The R/$\overline{W}$ bit tells the slave the desired direction of data transfer.

- 1 = Read transfer: The slave transmits data to the master
- 0 = Write transfer: The master transmits data to the slave

Only the slave with a calling address that matches the one transmitted by the master responds by sending an acknowledge bit. The slave sends the acknowledge bit by pulling SDA low at the ninth clock.

No two slaves in the system can have the same address. If the I2C module is the master, it must not transmit an address that is equal to its own slave address. The I2C module cannot be master and slave at the same time. However, if arbitration is lost during an address cycle, the I2C module reverts to slave mode and operates correctly even if it is being addressed by another master.

21.5.1.3 Data transfers

When successful slave addressing is achieved, data transfer can proceed on a byte-by-byte basis in the direction specified by the R/$\overline{W}$ bit sent by the calling master.

All transfers that follow an address cycle are referred to as data transfers, even if they carry subaddress information for the slave device.

Each data byte is 8 bits long. Data may be changed only while SCL is low. Data must be held stable while SCL is high. There is one clock pulse on SCL for each data bit, and the MSB is transferred first. Each data byte is followed by a ninth (acknowledge) bit, which is signaled from the receiving device by pulling SDA low at the ninth clock. In summary, one complete data transfer needs nine clock pulses.

If the slave receiver does not acknowledge the master in the ninth bit, the slave must leave SDA high. The master interprets the failed acknowledgement as an unsuccessful data transfer.

If the master receiver does not acknowledge the slave transmitter after a data byte transmission, the slave interprets it as an end to data transfer and releases the SDA line.

In the case of a failed acknowledgement by either the slave or master, the data transfer is aborted and the master does one of two things:

- Relinquishes the bus by generating a STOP signal.
- Commences a new call by generating a repeated START signal.

21.5.1.4 STOP signal

The master can terminate the communication by generating a STOP signal to free the bus. A STOP signal is defined as a low-to-high transition of SDA while SCL is asserted.

21.5.1.5 Repeated START signal

The master may generate a START signal followed by a calling command without generating a STOP signal first. This action is called a repeated START. The master uses a repeated START to communicate with another slave or with the same slave in a different mode (transmit/receive mode) without releasing the bus.The master needs to send a NACK signal before sending repeated-START in the buffering mode.

21.5.1.6 Arbitration procedure

The I2C bus is a true multimaster bus that allows more than one master to be connected on it.

If two or more masters try to control the bus at the same time, a clock synchronization procedure determines the bus clock. The bus clock's low period is equal to the longest clock low period, and the high period is equal to the shortest one among the masters.

The relative priority of the contending masters is determined by a data arbitration procedure. A bus master loses arbitration if it transmits logic level 1 while another master transmits logic level 0. The losing masters immediately switch to slave receive mode and

stop driving SDA output. In this case, the transition from master to slave mode does not generate a STOP condition. Meanwhile, hardware sets a status bit to indicate the loss of arbitration.

21.5.1.7 Clock synchronization

Because wire AND logic is performed on SCL, a high-to-low transition on SCL affects all devices connected on the bus. The devices start counting their low period and, after a device's clock has gone low, that device holds SCL low until the clock reaches its high state. However, the change of low to high in this device clock might not change the state of SCL if another device clock is still within its low period. Therefore, the synchronized clock SCL is held low by the device with the longest low period. Devices with shorter low periods enter a high wait state during this time; see the following diagram. When all applicable devices have counted off their low period, the synchronized clock SCL is released and pulled high. Afterward there is no difference between the device clocks and the state of SCL, and all devices start counting their high periods. The first device to complete its high period pulls SCL low again.

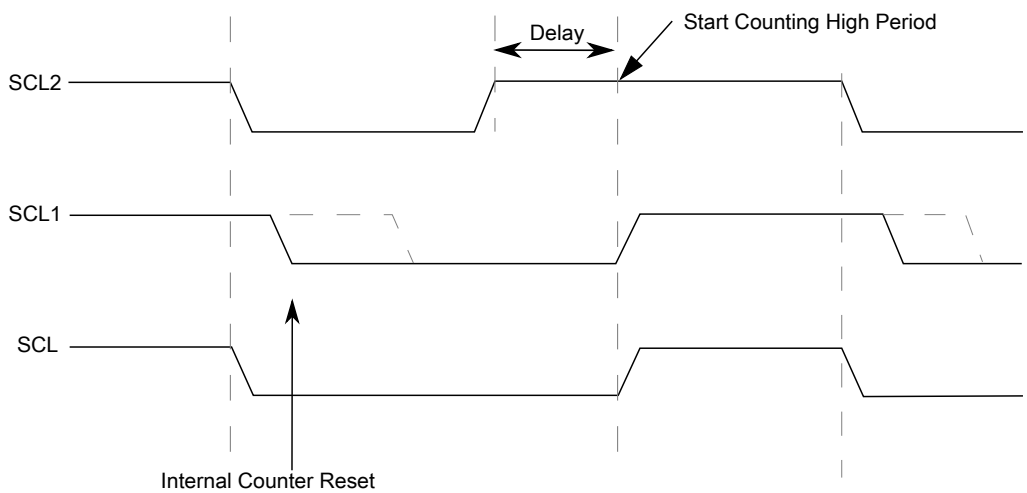


Figure 21-3. I2C clock synchronization

21.5.1.8 Handshaking

The clock synchronization mechanism can be used as a handshake in data transfers. A slave device may hold SCL low after completing a single byte transfer (9 bits). In this case, it halts the bus clock and forces the master clock into wait states until the slave releases SCL.

21.5.1.9 Clock stretching

The clock synchronization mechanism can be used by slaves to slow down the bit rate of a transfer. After the master drives SCL low, a slave can drive SCL low for the required period and then release it. If the slave's SCL low period is greater than the master's SCL low period, the resulting SCL bus signal's low period is stretched. In other words, the SCL bus signal's low period is increased to be the same length as the slave's SCL low period.

21.5.1.10 I2C divider and hold values

NOTE

For some cases on some devices, the SCL divider value may vary by ±2 or ±4 when ICR's value ranges from 00h to 0Fh. These potentially varying SCL divider values are highlighted in the following table. For the actual SCL divider values for your device, see the chip-specific details about the I2C module.

Table 21-3. I2C divider and hold values

ICR (hex)	SCL divider	SDA hold value	SCL hold (start) value	SCL hold (stop) value	ICR (hex)	SCL divider (clocks)	SDA hold (clocks)	SCL hold (start) value	SCL hold (stop) value
00	20	7	6	11	20	160	17	78	81
01	22	7	7	12	21	192	17	94	97
02	24	8	8	13	22	224	33	110	113
03	26	8	9	14	23	256	33	126	129
04	28	9	10	15	24	288	49	142	145
05	30	9	11	16	25	320	49	158	161
06	34	10	13	18	26	384	65	190	193
07	40	10	16	21	27	480	65	238	241
08	28	7	10	15	28	320	33	158	161
09	32	7	12	17	29	384	33	190	193
0A	36	9	14	19	2A	448	65	222	225
0B	40	9	16	21	2B	512	65	254	257
0C	44	11	18	23	2C	576	97	286	289
0D	48	11	20	25	2D	640	97	318	321
0E	56	13	24	29	2E	768	129	382	385
0F	68	13	30	35	2F	960	129	478	481
10	48	9	18	25	30	640	65	318	321
11	56	9	22	29	31	768	65	382	385
12	64	13	26	33	32	896	129	446	449
13	72	13	30	37	33	1024	129	510	513

Table continues on the next page...

Table 21-3. I2C divider and hold values (continued)

ICR (hex)	SCL divider	SDA hold value	SCL hold (start) value	SCL hold (stop) value	ICR (hex)	SCL divider (clocks)	SDA hold (clocks)	SCL hold (start) value	SCL hold (stop) value
14	80	17	34	41	34	1152	193	574	577
15	88	17	38	45	35	1280	193	638	641
16	104	21	46	53	36	1536	257	766	769
17	128	21	58	65	37	1920	257	958	961
18	80	9	38	41	38	1280	129	638	641
19	96	9	46	49	39	1536	129	766	769
1A	112	17	54	57	3A	1792	257	894	897
1B	128	17	62	65	3B	2048	257	1022	1025
1C	144	25	70	73	3C	2304	385	1150	1153
1D	160	25	78	81	3D	2560	385	1278	1281
1E	192	33	94	97	3E	3072	513	1534	1537
1F	240	33	118	121	3F	3840	513	1918	1921

21.5.2 10-bit address

For 10-bit addressing, 0x11110 is used for the first 5 bits of the first address byte. Various combinations of read/write formats are possible within a transfer that includes 10-bit addressing.

21.5.2.1 Master-transmitter addresses a slave-receiver

The transfer direction is not changed. When a 10-bit address follows a START condition, each slave compares the first 7 bits of the first byte of the slave address (11110XX) with its own address and tests whether the eighth bit (R/$\overline{W}$ direction bit) is 0. It is possible that more than one device finds a match and generates an acknowledge (A1). Each slave that finds a match compares the 8 bits of the second byte of the slave address with its own address, but only one slave finds a match and generates an acknowledge (A2). The matching slave remains addressed by the master until it receives a STOP condition (P) or a repeated START condition (Sr) followed by a different slave address.

Table 21-4. Master-transmitter addresses slave-receiver with a 10-bit address

S	Slave address first 7 bits 11110 + AD10 + AD9	R/$\overline{W}$ 0	A1	Slave address second byte AD[8:1]	A2	Data	A	...	Data	A/A	P

After the master-transmitter has sent the first byte of the 10-bit address, the slave-receiver sees an I2C interrupt. User software must ensure that for this interrupt, the contents of the Data register are ignored and not treated as valid data.

21.5.2.2 Master-receiver addresses a slave-transmitter

The transfer direction is changed after the second R/$\overline{W}$ bit. Up to and including acknowledge bit A2, the procedure is the same as that described for a master-transmitter addressing a slave-receiver. After the repeated START condition (Sr), a matching slave remembers that it was addressed before. This slave then checks whether the first seven bits of the first byte of the slave address following Sr are the same as they were after the START condition (S), and it tests whether the eighth (R/$\overline{W}$) bit is 1. If there is a match, the slave considers that it has been addressed as a transmitter and generates acknowledge A3. The slave-transmitter remains addressed until it receives a STOP condition (P) or a repeated START condition (Sr) followed by a different slave address.

After a repeated START condition (Sr), all other slave devices also compare the first seven bits of the first byte of the slave address with their own addresses and test the eighth (R/$\overline{W}$) bit. However, none of them are addressed because R/$\overline{W}$ = 1 (for 10-bit devices), or the 11110XX slave address (for 7-bit devices) does not match.

Table 21-5. Master-receiver addresses a slave-transmitter with a 10-bit address

S	Slave address first 7 bits 11110 + AD10 + AD9	R/$\overline{W}$ 0	A1	Slave address second byte AD[8:1]	A2	Sr	Slave address first 7 bits 11110 + AD10 + AD9	R/$\overline{W}$ 1	A3	Data	A	...	Data	A	P

After the master-receiver has sent the first byte of the 10-bit address, the slave-transmitter sees an I2C interrupt. User software must ensure that for this interrupt, the contents of the Data register are ignored and not treated as valid data.

21.5.3 Address matching

All received addresses can be requested in 7-bit or 10-bit address format.

- AD[7:1] in Address Register 1, which contains the I2C primary slave address, always participates in the address matching process. It provides a 7-bit address.
- If the ADEXT bit is set, AD[10:8] in Control Register 2 participates in the address matching process. It extends the I2C primary slave address to a 10-bit address.

Additional conditions that affect address matching include:

- If the GCAEN bit is set, general call participates the address matching process.
- If the ALERTEN bit is set, alert response participates the address matching process.
- If the SIICAEN bit is set, Address Register 2 participates in the address matching process.
- If the RMEN bit is set, when the Range Address register is programmed to a nonzero value, any address within the range of values of Address Register 1 (excluded) and the Range Address register (included) participates in the address matching process. The Range Address register must be programmed to a value greater than the value of Address Register 1.

When the I2C module responds to one of these addresses, it acts as a slave-receiver and the IAAS bit is set after the address cycle. Software must read the Data register after the first byte transfer to determine that the address is matched.

21.5.4 System management bus specification

SMBus provides a control bus for system and power management related tasks. A system can use SMBus to pass messages to and from devices instead of tripping individual control lines.

Removing the individual control lines reduces pin count. Accepting messages ensures future expandability. With the system management bus, a device can provide manufacturer information, tell the system what its model/part number is, save its state for a suspend event, report different types of errors, accept control parameters, and return its status.

21.5.4.1 Timeouts

The $T_{TIMEOUT,MIN}$ parameter allows a master or slave to conclude that a defective device is holding the clock low indefinitely or a master is intentionally trying to drive devices off the bus. The slave device must release the bus (stop driving the bus and let SCL and SDA float high) when it detects any single clock held low longer than $T_{TIMEOUT,MIN}$. Devices that have detected this condition must reset their communication and be able to receive a new START condition within the timeframe of $T_{TIMEOUT,MAX}$.

SMBus defines a clock low timeout, $T_{TIMEOUT}$, of 35 ms, specifies $T_{LOW:SEXT}$ as the cumulative clock low extend time for a slave device, and specifies $T_{LOW:MEXT}$ as the cumulative clock low extend time for a master device.

21.5.4.1.1 SCL low timeout

If the SCL line is held low by a slave device on the bus, no further communication is possible. Furthermore, the master cannot force the SCL line high to correct the error condition. To solve this problem, the SMBus protocol specifies that devices participating in a transfer must detect any clock cycle held low longer than a timeout value condition. Devices that have detected the timeout condition must reset the communication. When the I2C module is an active master, if it detects that SMBCLK low has exceeded the value of $T_{TIMEOUT,MIN}$, it must generate a stop condition within or after the current data byte in the transfer process. When the I2C module is a slave, if it detects the $T_{TIMEOUT,MIN}$ condition, it resets its communication and is then able to receive a new START condition.

21.5.4.1.2 SCL high timeout

When the I2C module has determined that the SMBCLK and SMBDAT signals have been high for at least $T_{HIGH:MAX}$, it assumes that the bus is idle.

A HIGH timeout occurs after a START condition appears on the bus but before a STOP condition appears on the bus. Any master detecting this scenario can assume the bus is free when either of the following occurs:

- SHTF1 rises.
- The BUSY bit is high and SHTF1 is high.

When the SMBDAT signal is low and the SMBCLK signal is high for a period of time, another kind of timeout occurs. The time period must be defined in software. SHTF2 is used as the flag when the time limit is reached. This flag is also an interrupt resource, so it triggers IICIF.

21.5.4.1.3 CSMBCLK TIMEOUT MEXT and CSMBCLK TIMEOUT SEXT

The following figure illustrates the definition of the timeout intervals $T_{LOW:SEXT}$ and $T_{LOW:MEXT}$. When in master mode, the I2C module must not cumulatively extend its clock cycles for a period greater than $T_{LOW:MEXT}$ within a byte, where each byte is defined as START-to-ACK, ACK-to-ACK, or ACK-to-STOP. When CSMBCLK TIMEOUT MEXT occurs, SMBus MEXT rises and also triggers the SLTF.

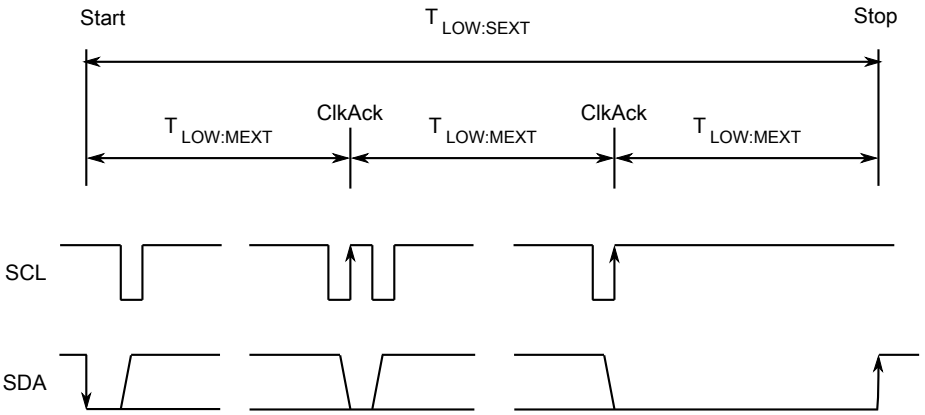


Figure 21-4. Timeout measurement intervals

A master is allowed to abort the transaction in progress to any slave that violates the $T_{LOW:SEXT}$ or $T_{TIMEOUT,MIN}$ specifications. To abort the transaction, the master issues a STOP condition at the conclusion of the byte transfer in progress. When a slave, the I2C module must not cumulatively extend its clock cycles for a period greater than $T_{LOW:SEXT}$ during any message from the initial START to the STOP. When CSMBCLK TIMEOUT SEXT occurs, SEXT rises and also triggers SLTF.

NOTE

CSMBCLK TIMEOUT SEXT and CSMBCLK TIMEOUT MEXT are optional functions that are implemented in the second step.

21.5.4.2 FAST ACK and NACK

To improve reliability and communication robustness, implementation of packet error checking (PEC) by SMBus devices is optional for SMBus devices but required for devices participating in and only during the address resolution protocol (ARP) process. The PEC is a CRC-8 error checking byte, calculated on all the message bytes. The PEC is appended to the message by the device that supplied the last data byte. If the PEC is present but not correct, a NACK is issued by the receiver. Otherwise an ACK is issued. To calculate the CRC-8 by software, this module can hold the SCL line low after receiving the eighth SCL (8th bit) if this byte is a data byte. So software can determine whether an ACK or NACK should be sent to the bus by setting or clearing the TXAK bit if the FACK (fast ACK/NACK enable) bit is enabled.

SMBus requires a device always to acknowledge its own address, as a mechanism to detect the presence of a removable device (such as a battery or docking station) on the bus. In addition to indicating a slave device busy condition, SMBus uses the NACK mechanism to indicate the reception of an invalid command or invalid data. Because such a condition may occur on the last byte of the transfer, SMBus devices are required to

have the ability to generate the not acknowledge after the transfer of each byte and before the completion of the transaction. This requirement is important because SMBus does not provide any other resend signaling. This difference in the use of the NACK signaling has implications on the specific implementation of the SMBus port, especially in devices that handle critical system data such as the SMBus host and the SBS components.

NOTE

In the last byte of master receive slave transmit mode, the master must send a NACK to the bus, so FACK must be switched off before the last byte transmits.

21.5.5 Resets

The I2C module is disabled after a reset. The I2C module cannot cause a core reset.

21.5.6 Interrupts

The I2C module generates an interrupt when any of the events in the table found here occur, provided that the IICIE bit is set.

The interrupt is driven by the IICIF bit (of the I2C Status Register) and masked with the IICIE bit (of the I2C Control Register 1). The IICIF bit must be cleared (by software) by writing 1 to it in the interrupt routine. The SMBus timeouts interrupt is driven by SLTF and masked with the IICIE bit. The SLTF bit must be cleared by software by writing 1 to it in the interrupt routine. You can determine the interrupt type by reading the Status Register.

NOTE

In master receive mode, the FACK bit must be set to zero before the last byte transfer.

Table 21-6. Interrupt summary

Interrupt source	Status	Flag	Local enable
Complete 1-byte transfer	TCF	IICIF	IICIE
Match of received calling address	IAAS	IICIF	IICIE
Arbitration lost	ARBL	IICIF	IICIE
I^2C bus stop detection	STOPF	IICIF	IICIE & SSIE
I^2C bus start detection	STARTF	IICIF	IICIE & SSIE
SMBus SCL low timeout	SLTF	IICIF	IICIE
SMBus SCL high SDA low timeout	SHTF2	IICIF	IICIE & SHTF2IE
Wakeup from stop3 or wait mode	IAAS	IICIF	IICIE & WUEN

21.5.6.1 Byte transfer interrupt

The Transfer Complete Flag (TCF) bit is set at the falling edge of the ninth clock to indicate the completion of a byte and acknowledgement transfer. When FACK is enabled, TCF is then set at the falling edge of eighth clock to indicate the completion of byte.

21.5.6.2 Address detect interrupt

When the calling address matches the programmed slave address (I2C Address Register) or when the GCAEN bit is set and a general call is received, the IAAS bit in the Status Register is set. The CPU is interrupted, provided the IICIE bit is set. The CPU must check the SRW bit and set its Tx mode accordingly.

21.5.6.3 Stop Detect Interrupt

When the stop status is detected on the I^2C bus, the STOPF bit is set to 1. The CPU is interrupted, provided the IICIE and SSIE bits are both set to 1.

21.5.6.4 Exit from low-power/stop modes

The slave receive input detect circuit and address matching feature are still active on low power modes (wait and stop). An asynchronous input matching slave address or general call address brings the CPU out of low power/stop mode if the interrupt is not masked. Therefore, TCF and IAAS both can trigger this interrupt.

21.5.6.5 Arbitration lost interrupt

The I2C is a true multimaster bus that allows more than one master to be connected on it. If two or more masters try to control the bus at the same time, the relative priority of the contending masters is determined by a data arbitration procedure. The I2C module asserts the arbitration-lost interrupt when it loses the data arbitration process and the ARBL bit in the Status Register is set.

Arbitration is lost in the following circumstances:

1. SDA is sampled as low when the master drives high during an address or data transmit cycle.

2. SDA is sampled as low when the master drives high during the acknowledge bit of a data receive cycle.
3. A START cycle is attempted when the bus is busy.
4. A repeated START cycle is requested in slave mode.
5. A STOP condition is detected when the master did not request it.

The ARBL bit must be cleared (by software) by writing 1 to it.

21.5.6.6 Timeout interrupt in SMBus

When the IICIE bit is set, the I2C module asserts a timeout interrupt (outputs SLTF and SHTF2) upon detection of any of the mentioned timeout conditions, with one exception. The SCL high and SDA high TIMEOUT mechanism must not be used to influence the timeout interrupt output, because this timeout indicates an idle condition on the bus. SHTF1 rises when it matches the SCL high and SDA high TIMEOUT and falls automatically just to indicate the bus status. The SHTF2's timeout period is the same as that of SHTF1, which is short compared to that of SLTF, so another control bit, SHTF2IE, is added to enable or disable it.

21.5.7 Address matching wake-up

When a primary, range, or general call address match occurs when the I2C module is in slave receive mode, the MCU wakes from a low power mode where no peripheral bus is running.

Data sent on the bus that is the same as a target device address might also wake the target MCU.

After the address matching IAAS bit is set, an interrupt is sent at the end of address matching to wake the core. The IAAS bit must be cleared after the clock recovery.

NOTE

After the system recovers and is in Run mode, restart the I2C module if it is needed to transfer packets. To avoid I2C transfer problems resulting from the situation, firmware should prevent the MCU execution of a STOP instruction when the I2C module is in the middle of a transfer unless the Stop mode holdoff feature is used during this period (set FLT[SHEN] to 1).

NOTE

After I2C address matching wake-up, the master must wait a time long enough for the slave ISR to finish running and resend start or repeat start signals.

For the SRW bit to function properly, it only supports Address +Write to wake up by I2C address matching. Before entering the next low power mode, Address+Write must be sent to change the SRW status.

21.5.8 Double buffering mode

In the double buffering mode, the data transfer is processed byte by byte. However, the data can be transferred without waiting for the interrupt or the polling to finish. This means the write/read I2C_D operation will not block the data transfer, as the hardware has already finished the internal write or read. The benefit is that the baud rate is able to achieve higher speed.

There are several items to consider as follows:

- When initiating a double buffering transfer at Tx side, the user can write 2 values to the I2C_D buffer before transfer. However, that is allowed only at one time per package frame (due to the buffer depth, and because two-times writes in each ISR are not allowed). The second write to the I2C_D buffer must wait for the Empty flag. On the other hand, at Rx side the user can read twice in a one-byte transfer (if needed).

NOTE

Check Empty flag before write to I2C_D.

Write twice to the I2C_D buffer ONLY after the address matching byte. Do not write twice (Address+Data) before START or at the beginning of I2C transfer, especially when the baud rate is very slow.

- To write twice in one frame, during the next-to-last ISR, do a dummy read from the I2C_D buffer at Tx side (or the TCF will stay high, because the TCF is cleared by write/read operation). In the next-to-last ISR, do not send data again (the buffer data will be under running).
- To keep new ISRs software-compatible with previous ISRs, the write/read I2C_D operation will not block the internal-hardware-released SCL/SDA signals. At the

ACK phase, the bus is released to accept the next byte if the master can send the clock immediately.

- On the slave side, two-times writes to the I2C_D buffer may be limited by the master's clock and START/repeated-START signal. This is not currently supported, and the master's START/repeated-START signal will break data transfers. To release the bus, do a dummy read or write to the I2C_D buffer again. It is suggested to send repeated-START/START during intervals as before.
- The master receive should send a NACK in the next-to-last ISR, if it wants to do the STOP or the repeated-START work. The transmitting slave which receives the NACK, will switch to receive mode, and do a dummy read to release SCL and SDA signals.

21.6 Initialization/application information

Module Initialization (Slave)

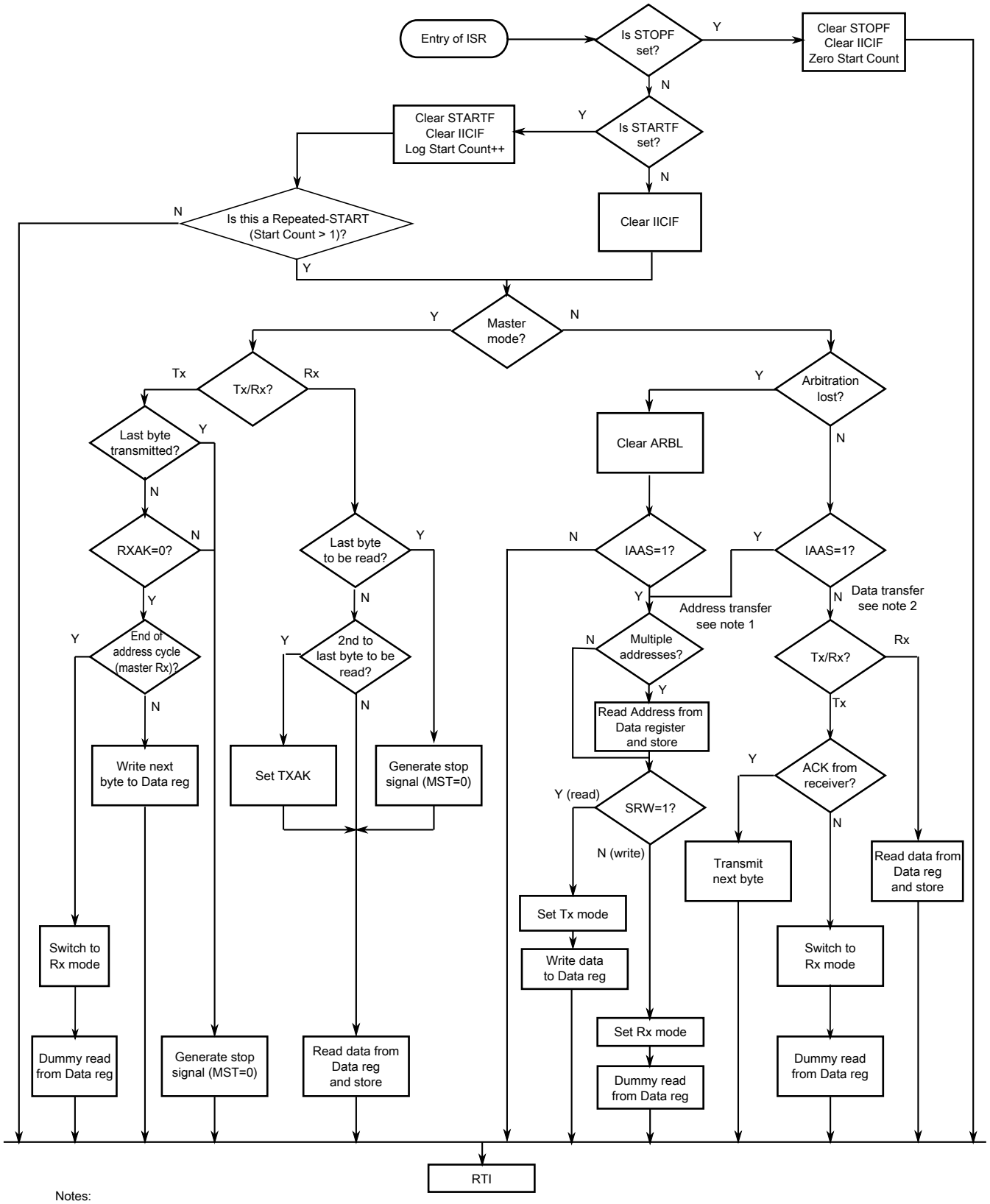
1. Write: Control Register 2
   - to enable or disable general call
   - to select 10-bit or 7-bit addressing mode
2. Write: Address Register 1 to set the slave address
3. Write: Control Register 1 to enable the I2C module and interrupts
4. Initialize RAM variables (IICEN = 1 and IICIE = 1) for transmit data
5. Initialize RAM variables used to achieve the routine shown in the following figure

Module Initialization (Master)

1. Write: Frequency Divider register to set the I2C baud rate (see example in description of ICR)
2. Write: Control Register 1 to enable the I2C module and interrupts
3. Initialize RAM variables (IICEN = 1 and IICIE = 1) for transmit data
4. Initialize RAM variables used to achieve the routine shown in the following figure
5. Write: Control Register 1 to enable TX
6. Write: Control Register 1 to enable MST (master mode)
7. Write: Data register with the address of the target slave (the LSB of this byte determines whether the communication is master receive or transmit)

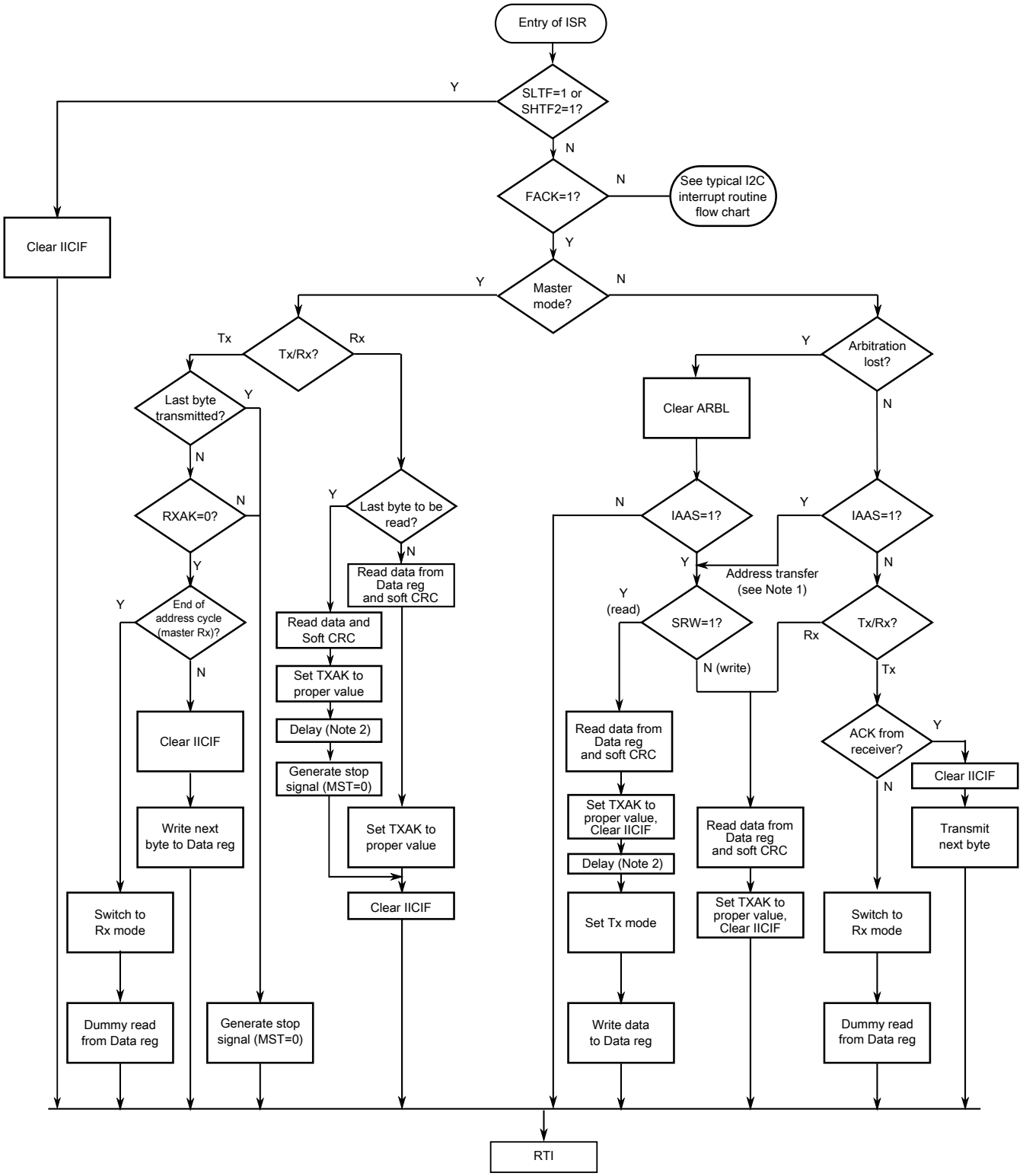
The routine shown in the following figure encompasses both master and slave I2C operations. For slave operation, an incoming I2C message that contains the proper address begins I2C communication. For master operation, communication must be

initiated by writing the Data register. An example of an I2C driver which implements many of the steps described here is available in AN4342: Using the Inter-Integrated Circuit on ColdFire+ and Kinetis .



Notes:
1. If general call is enabled, check to determine if the received address is a general call address (0x00).
   If the received address is a general call address, the general call must be handled by user software.
2. When 10-bit addressing addresses a slave, the slave sees an interrupt following the first byte of the extended address.
   Ensure that for this interrupt, the contents of the Data register are ignored and not treated as a valid data transfer.

Figure 21-5. Typical I2C interrupt routine



Notes:
1. If general call or SIICAEN is enabled, check to determine if the received address is a general call address (0x00) or an SMBus device default address. In either case, they must be handled by user software.
2. In receive mode, one bit time delay may be needed before the stop signal generation, to wait for the possible longest time period (in worst case) of the 9th SCL cycle.

Figure 21-6. Typical I2C SMBus interrupt routine

Chapter 22
Serial Communications Interface (SCI)

22.1 Chip specific serial communications interface

This device includes one independent serial communications interface (SCI) module. Typically, these systems are used to connect to the RS232 serial input/output port of a personal computer or workstation. They can also be used to communicate with other embedded controllers.

A flexible, 13-bit, modulo-based baud rate generator supports a broad range of standard baud rates beyond 115.2 kBD. Programmable 8-bit or 9-bit format and 1-bit or 2-bit Stop are supported.

This SCI system offers many advanced features not commonly found on other asynchronous serial I/O peripherals of the embedded controllers. The receiver employs an advanced data sampling technique that ensures reliable communication and noise detection. Hardware parity, receiver wakeup, and double buffering on transmit and receive are also included.

SCI has option to be either continuously clocked or frozen in Wait mode. It is controlled by SC1_C1[SCISWAI].

- SC1_C1[SCISWAI] = 0: SCI clocks continue to run in Wait mode so the SCI can be the source of an interrupt that wakes the CPU
- SC1_C1[SCISWAI] = 1: SCI clocks freeze when CPU is in Wait mode.

Customization:

- Primary clock: BUSCLK (20 MHz)
- Alternate clock: None

- LIN capability
- The following table summarizes the signal connection of SCI module.

Table 22-1. SCI module signals connection

Module	Signal	Connect to
SCI	TX	PA5/TX, PA7/TX
	RX	PA4/RX, PA6/RX
	Internal clock	BUSCLK

NOTE

This module supports wakeup in Wait and Stop modes.

22.2 Introduction

22.2.1 Features

Features of SCI module include:

- Full-duplex, standard non-return-to-zero (NRZ) format
- Double-buffered transmitter and receiver with separate enables
- Programmable baud rates (13-bit modulo divider)
- Interrupt-driven or polled operation:
  - Transmit data register empty and transmission complete
  - Receive data register full
  - Receive overrun, parity error, framing error, and noise error
  - Idle receiver detect
  - Active edge on receive pin
  - Break detect supporting LIN
- Hardware parity generation and checking
- Programmable 8-bit or 9-bit character length
- Programmable 1-bit or 2-bit stop bits
- Receiver wakeup by idle-line or address-mark
- Optional 13-bit break character generation / 11-bit break character detection
- Selectable transmitter output polarity

22.2.2 Modes of operation

See Section Functional description for details concerning SCI operation in these modes:

- 8- and 9-bit data modes
- Stop mode operation
- Loop mode
- Single-wire mode

22.2.3 Block diagram

The following figure shows the transmitter portion of the SCI.

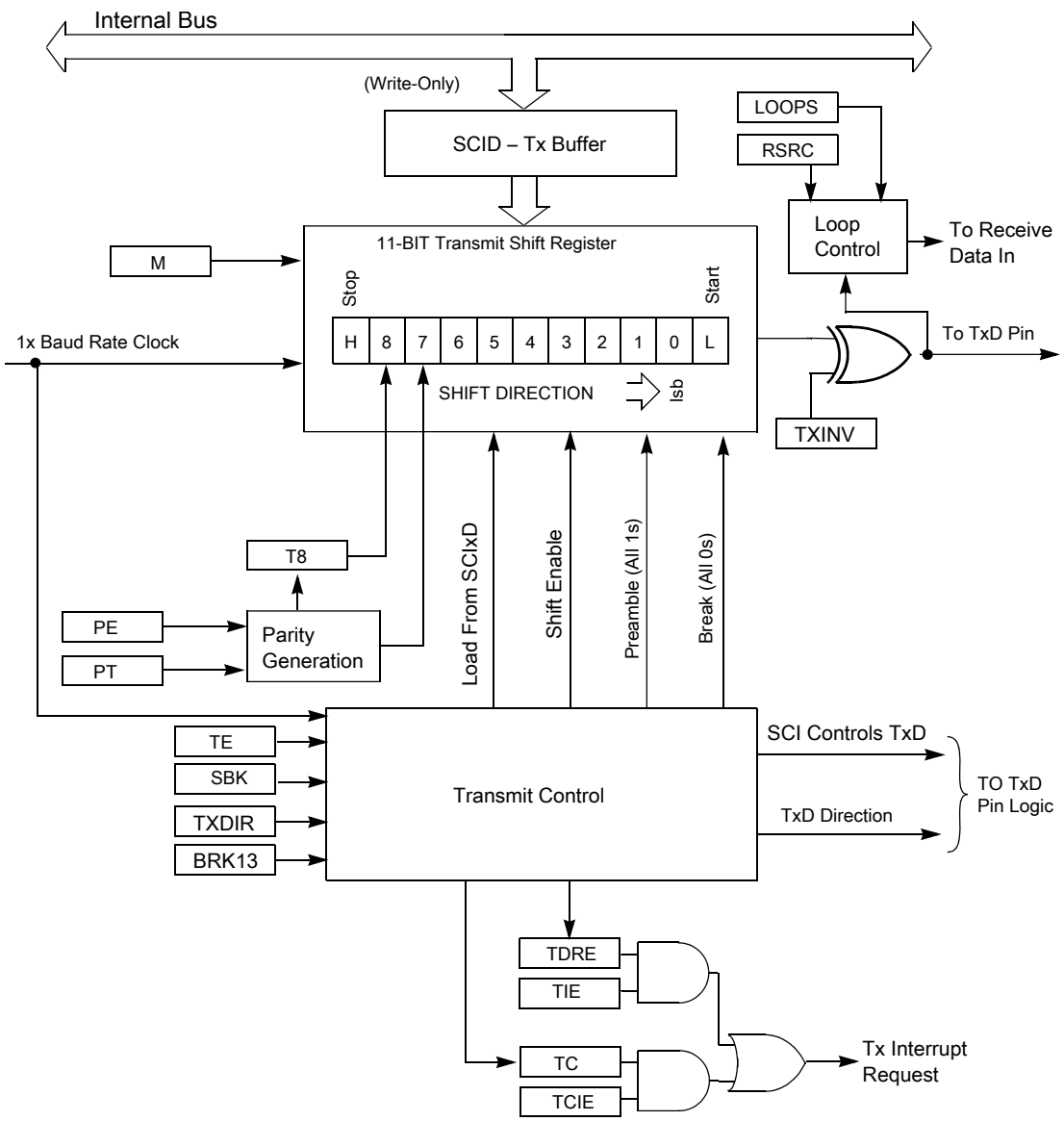


Figure 22-1. SCI transmitter block diagram

The following figure shows the receiver portion of the SCI.

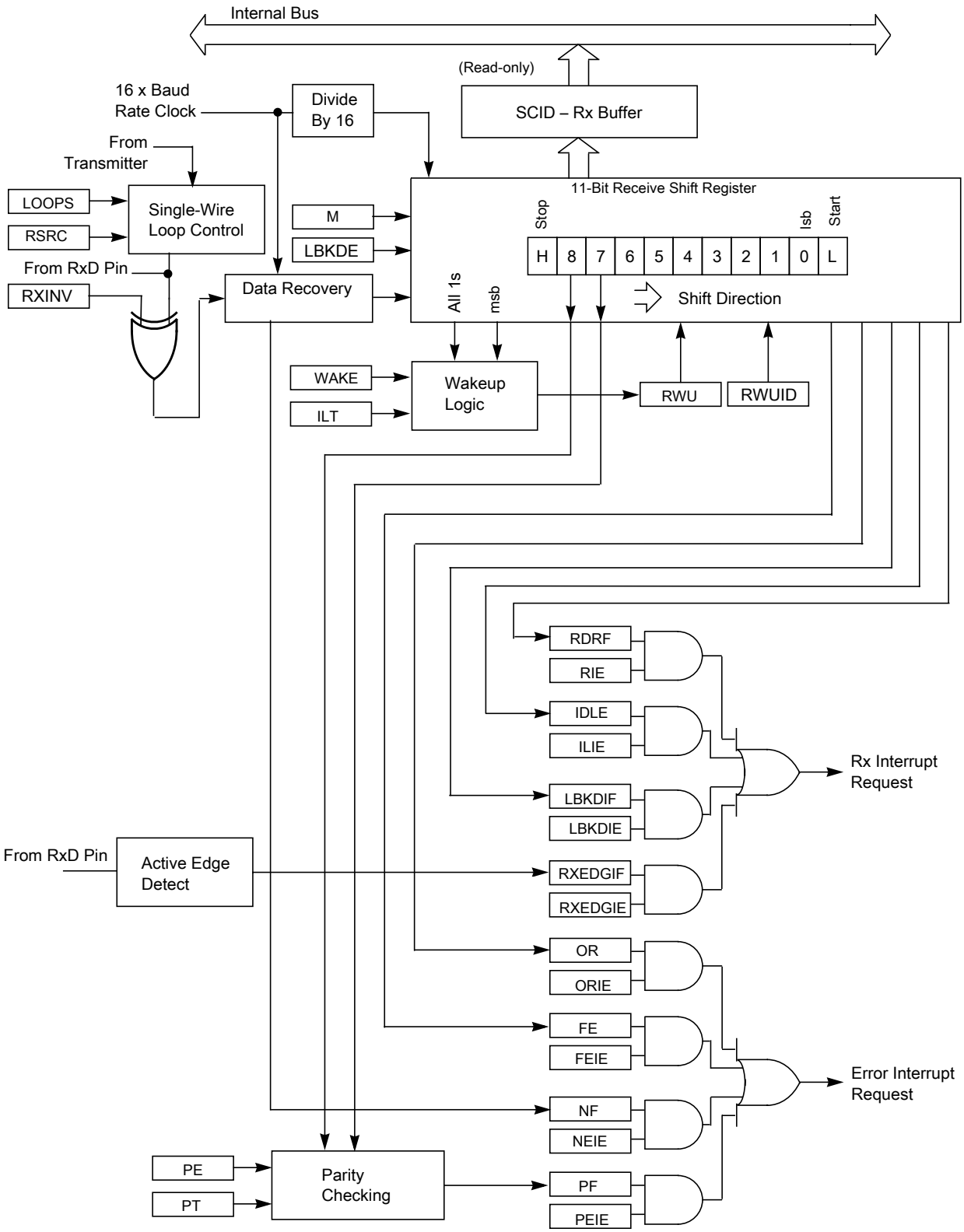


Figure 22-2. SCI receiver block diagram

22.3 SCI signal descriptions

The SCI signals are shown in the table found here.

Table 22-2. SCI signal descriptions

Signal	Description	I/O
RxD	Receive data	I
TxD	Transmit data	I/O

22.3.1 Detailed signal descriptions

The detailed signal descriptions of the SCI are shown in the following table.

Table 22-3. SCI—Detailed signal descriptions

Signal	I/O	Description	
RxD	I	Receive data. Serial data input to receiver.	
		State meaning	Whether RxD is interpreted as a 1 or 0 depends on the bit encoding method along with other configuration settings.
		Timing	Sampled at a frequency determined by the module clock divided by the baud rate.
TxD	I/O	Transmit data. Serial data output from transmitter.	
		State meaning	Whether TxD is interpreted as a 1 or 0 depends on the bit encoding method along with other configuration settings.
		Timing	Driven at the beginning or within a bit time according to the bit encoding method along with other configuration settings. Otherwise, transmissions are independent of reception timing.

22.4 Register definition

The SCI has 8-bit registers to control baud rate, select SCI options, report SCI status, and for transmit/receive data.

Refer to the direct-page register summary in the memory chapter of this document or the absolute address assignments for all SCI registers. This section refers to registers and control bits only by their names. An NXP-provided equate or header file is used to translate these names into the appropriate absolute addresses.

SCI memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
1868	SCI Baud Rate Register: High (SCI0_BDH)	8	R/W	00h	22.4.1/400
1869	SCI Baud Rate Register: Low (SCI0_BDL)	8	R/W	04h	22.4.2/401
186A	SCI Control Register 1 (SCI0_C1)	8	R/W	00h	22.4.3/401
186B	SCI Control Register 2 (SCI0_C2)	8	R/W	00h	22.4.4/403
186C	SCI Status Register 1 (SCI0_S1)	8	R	C0h	22.4.5/404
186D	SCI Status Register 2 (SCI0_S2)	8	R/W	00h	22.4.6/406
186E	SCI Control Register 3 (SCI0_C3)	8	R/W	00h	22.4.7/407
186F	SCI Data Register (SCI0_D)	8	R/W	00h	22.4.8/409

22.4.1 SCI Baud Rate Register: High (SCIx_BDH)

This register, along with SCI_BDL, controls the prescale divisor for SCI baud rate generation. To update the 13-bit baud rate setting [SBR12:SBR0], first write to SCI_BDH to buffer the high half of the new value and then write to SCI_BDL. The working value in SCI_BDH does not change until SCI_BDL is written.

Address: 1868h base + 0h offset = 1868h

Bit	7	6	5	4	3	2	1	0
Read Write	LBKDIE	RXEDGIE	SBNS	SBR				
Reset	0	0	0	0	0	0	0	0

SCIx_BDH field descriptions

Field	Description
7 LBKDIE	LIN Break Detect Interrupt Enable (for LBKDIF) 0 Hardware interrupts from SCI_S2[LBKDIF] disabled (use polling). 1 Hardware interrupt requested when SCI_S2[LBKDIF] flag is 1.
6 RXEDGIE	RxD Input Active Edge Interrupt Enable (for RXEDGIF) 0 Hardware interrupts from SCI_S2[RXEDGIF] disabled (use polling). 1 Hardware interrupt requested when SCI_S2[RXEDGIF] flag is 1.
5 SBNS	Stop Bit Number Select SBNS determines whether data characters are one or two stop bits. 0 One stop bit. 1 Two stop bit.
SBR	Baud Rate Modulo Divisor.

Table continues on the next page...

SCIx_BDH field descriptions (continued)

Field	Description
	The 13 bits in SBR[12:0] are referred to collectively as BR, and they set the modulo divide rate for the SCI baud rate generator. When BR is cleared, the SCI baud rate generator is disabled to reduce supply current. When BR is 1 - 8191, the SCI baud rate equals BUSCLK/(16×BR).

22.4.2 SCI Baud Rate Register: Low (SCIx_BDL)

This register, along with SCI_BDH, control the prescale divisor for SCI baud rate generation. To update the 13-bit baud rate setting [SBR12:SBR0], first write to SCI_BDH to buffer the high half of the new value and then write to SCI_BDL. The working value in SCI_BDH does not change until SCI_BDL is written.

SCI_BDL is reset to a non-zero value, so after reset the baud rate generator remains disabled until the first time the receiver or transmitter is enabled; that is, SCI_C2[RE] or SCI_C2[TE] bits are written to 1.

Address: 1868h base + 1h offset = 1869h

Bit	7	6	5	4	3	2	1	0
Read Write	SBR							
Reset	0	0	0	0	0	1	0	0

SCIx_BDL field descriptions

Field	Description
SBR	Baud Rate Modulo Divisor These 13 bits in SBR[12:0] are referred to collectively as BR. They set the modulo divide rate for the SCI baud rate generator. When BR is cleared, the SCI baud rate generator is disabled to reduce supply current. When BR is 1 - 8191, the SCI baud rate equals BUSCLK/(16×BR).

22.4.3 SCI Control Register 1 (SCIx_C1)

This read/write register controls various optional features of the SCI system.

Address: 1868h base + 2h offset = 186Ah

Bit	7	6	5	4	3	2	1	0
Read Write	LOOPS	SCISWAI	RSRC	M	WAKE	ILT	PE	PT
Reset	0	0	0	0	0	0	0	0

SCIx_C1 field descriptions

Field	Description
7 LOOPS	Loop Mode Select Selects between loop back modes and normal 2-pin full-duplex modes. When LOOPS is set, the transmitter output is internally connected to the receiver input. 0 Normal operation - RxD and TxD use separate pins. 1 Loop mode or single-wire mode where transmitter outputs are internally connected to receiver input. (See RSRC bit.) RxD pin is not used by SCI.
6 SCISWAI	SCI Stops in Wait Mode 0 SCI clocks continue to run in wait mode so the SCI can be the source of an interrupt that wakes up the CPU. 1 SCI clocks freeze while CPU is in wait mode.
5 RSRC	Receiver Source Select This bit has no meaning or effect unless the LOOPS bit is set to 1. When LOOPS is set, the receiver input is internally connected to the TxD pin and RSRC determines whether this connection is also connected to the transmitter output. 0 Provided LOOPS is set, RSRC is cleared, selects internal loop back mode and the SCI does not use the RxD pins. 1 Single-wire SCI mode where the TxD pin is connected to the transmitter output and receiver input.
4 M	9-Bit or 8-Bit Mode Select 0 Normal - start + 8 data bits (lsb first) + stop. 1 Receiver and transmitter use 9-bit data characters start + 8 data bits (lsb first) + 9th data bit + stop.
3 WAKE	Receiver Wakeup Method Select 0 Idle-line wakeup. 1 Address-mark wakeup.
2 ILT	Idle Line Type Select Setting this bit to 1 ensures that the stop bits and logic 1 bits at the end of a character do not count toward the 10 or 11 bit times of logic high level needed by the idle line detection logic. 0 Idle character bit count starts after start bit. 1 Idle character bit count starts after stop bit.
1 PE	Parity Enable Enables hardware parity generation and checking. When parity is enabled, the most significant bit (msb) of the data character, eighth or ninth data bit, is treated as the parity bit. 0 No hardware parity generation or checking. 1 Parity enabled.
0 PT	Parity Type Provided parity is enabled (PE = 1), this bit selects even or odd parity. Odd parity means the total number of 1s in the data character, including the parity bit, is odd. Even parity means the total number of 1s in the data character, including the parity bit, is even. 0 Even parity. 1 Odd parity.

22.4.4 SCI Control Register 2 (SCIx_C2)

This register can be read or written at any time.

Address: 1868h base + 3h offset = 186Bh

Bit	7	6	5	4	3	2	1	0
Read Write	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
Reset	0	0	0	0	0	0	0	0

SCIx_C2 field descriptions

Field	Description
7 TIE	Transmit Interrupt Enable for TDRE 0 Hardware interrupts from TDRE disabled; use polling. 1 Hardware interrupt requested when TDRE flag is 1.
6 TCIE	Transmission Complete Interrupt Enable for TC 0 Hardware interrupts from TC disabled; use polling. 1 Hardware interrupt requested when TC flag is 1.
5 RIE	Receiver Interrupt Enable for RDRF 0 Hardware interrupts from RDRF disabled; use polling. 1 Hardware interrupt requested when RDRF flag is 1.
4 ILIE	Idle Line Interrupt Enable for IDLE 0 Hardware interrupts from IDLE disabled; use polling. 1 Hardware interrupt requested when IDLE flag is 1.
3 TE	Transmitter Enable TE must be 1 to use the SCI transmitter. When TE is set, the SCI forces the TxD pin to act as an output for the SCI system. When the SCI is configured for single-wire operation (LOOPS = RSRC = 1), TXDIR controls the direction of traffic on the single SCI communication line (TxD pin). TE can also queue an idle character by clearing TE then setting TE while a transmission is in progress. When TE is written to 0, the transmitter keeps control of the port TxD pin until any data, queued idle, or queued break character finishes transmitting before allowing the pin to revert to a general-purpose I/O pin. 0 Transmitter off. 1 Transmitter on.
2 RE	Receiver Enable When the SCI receiver is off, the RxD pin reverts to being a general-purpose port I/O pin. If LOOPS is set the RxD pin reverts to being a general-purpose I/O pin even if RE is set. 0 Receiver off. 1 Receiver on.
1 RWU	Receiver Wakeup Control

Table continues on the next page...

SCIx_C2 field descriptions (continued)

Field	Description
	This bit can be written to 1 to place the SCI receiver in a standby state where it waits for automatic hardware detection of a selected wakeup condition. The wakeup condition is an idle line between messages, WAKE = 0, idle-line wakeup, or a logic 1 in the most significant data bit in a character, WAKE = 1, address-mark wakeup. Application software sets RWU and, normally, a selected hardware condition automatically clears RWU. 0 Normal SCI receiver operation. 1 SCI receiver in standby waiting for wakeup condition.
0 SBK	Send Break Writing a 1 and then a 0 to SBK queues a break character in the transmit data stream. Additional break characters of 10 or 11 or 12, 13 or 14 or 15 if BRK13 = 1, bit times of logic 0 are queued as long as SBK is set. Depending on the timing of the set and clear of SBK relative to the information currently being transmitted, a second break character may be queued before software clears SBK. 0 Normal transmitter operation. 1 Queue break character(s) to be sent.

22.4.5 SCI Status Register 1 (SCIx_S1)

This register has eight read-only status flags. Writes have no effect. Special software sequences, which do not involve writing to this register, clear these status flags.

Address: 1868h base + 4h offset = 186Ch

Bit	7	6	5	4	3	2	1	0
Read	TDRE	TC	RDRF	IDLE	OR	NF	FE	PF
Write								
Reset	1	1	0	0	0	0	0	0

SCIx_S1 field descriptions

Field	Description
7 TDRE	Transmit Data Register Empty Flag TDRE is set out of reset and when a transmit data value transfers from the transmit data buffer to the transmit shifter, leaving room for a new character in the buffer. To clear TDRE, read SCI_S1 with TDRE set and then write to the SCI data register (SCI_D). 0 Transmit data register (buffer) full. 1 Transmit data register (buffer) empty.
6 TC	Transmission Complete Flag TC is set out of reset and when TDRE is set and no data, preamble, or break character is being transmitted. TC is cleared automatically by reading SCI_S1 with TC set and then doing one of the following: • Write to the SCI data register (SCI_D) to transmit new data • Queue a preamble by changing TE from 0 to 1 • Queue a break character by writing 1 to SCI_C2[SBK]

Table continues on the next page...

SCIx_S1 field descriptions (continued)

Field	Description
	0 Transmitter active (sending data, a preamble, or a break). 1 Transmitter idle (transmission activity complete).
5 RDRF	Receive Data Register Full Flag RDRF becomes set when a character transfers from the receive shifter into the receive data register (SCI_D). To clear RDRF, read SCI_S1 with RDRF set and then read the SCI data register (SCI_D). 0 Receive data register empty. 1 Receive data register full.
4 IDLE	Idle Line Flag IDLE is set when the SCI receive line becomes idle for a full character time after a period of activity. When ILT is cleared, the receiver starts counting idle bit times after the start bit. If the receive character is all 1s, these bit times and the stop bits time count toward the full character time of logic high, 10 or 11 bit times depending on the M control bit, needed for the receiver to detect an idle line. When ILT is set, the receiver doesn't start counting idle bit times until after the stop bits. The stop bits and any logic high bit times at the end of the previous character do not count toward the full character time of logic high needed for the receiver to detect an idle line. To clear IDLE, read SCI_S1 with IDLE set and then read the SCI data register (SCI_D). After IDLE has been cleared, it cannot become set again until after a new character has been received and RDRF has been set. IDLE is set only once even if the receive line remains idle for an extended period. 0 No idle line detected. 1 Idle line was detected.
3 OR	Receiver Overrun Flag OR is set when a new serial character is ready to be transferred to the receive data register (buffer), but the previously received character has not been read from SCI_D yet. In this case, the new character, and all associated error information, is lost because there is no room to move it into SCI_D. To clear OR, read SCI_S1 with OR set and then read the SCI data register (SCI_D). 0 No overrun. 1 Receive overrun (new SCI data lost).
2 NF	Noise Flag The advanced sampling technique used in the receiver takes seven samples during the start bit and three samples in each data bit and the stop bits. If any of these samples disagrees with the rest of the samples within any bit time in the frame, the flag NF is set at the same time as RDRF is set for the character. To clear NF, read SCI_S1 and then read the SCI data register (SCI_D). 0 No noise detected. 1 Noise detected in the received character in SCI_D.
1 FE	Framing Error Flag FE is set at the same time as RDRF when the receiver detects a logic 0 where the stop bits was expected. This suggests the receiver was not properly aligned to a character frame. To clear FE, read SCI_S1 with FE set and then read the SCI data register (SCI_D). 0 No framing error detected. This does not guarantee the framing is correct. 1 Framing error.
0 PF	Parity Error Flag

Table continues on the next page...

SCIx_S1 field descriptions (continued)

Field	Description
	PF is set at the same time as RDRF when parity is enabled (PE = 1) and the parity bit in the received character does not agree with the expected parity value. To clear PF, read SCI_S1 and then read the SCI data register (SCI_D). 0 No parity error. 1 Parity error.

22.4.6 SCI Status Register 2 (SCIx_S2)

This register contains one read-only status flag.

When using an internal oscillator in a LIN system, it is necessary to raise the break detection threshold one bit time. Under the worst case timing conditions allowed in LIN, it is possible that a 0x00 data character can appear to be 10.26 bit times long at a slave running 14% faster than the master. This would trigger normal break detection circuitry designed to detect a 10-bit break symbol. When the LBKDE bit is set, framing errors are inhibited and the break detection threshold changes from 10 bits to 11 bits, preventing false detection of a 0x00 data character as a LIN break symbol.

Address: 1868h base + 5h offset = 186Dh

Bit	7	6	5	4	3	2	1	0
Read	LBKDIF	RXEDGIF	0	RXINV	RWUID	BRK13	LBKDE	RAF
Write								
Reset	0	0	0	0	0	0	0	0

SCIx_S2 field descriptions

Field	Description
7 LBKDIF	LIN Break Detect Interrupt Flag LBKDIF is set when the LIN break detect circuitry is enabled and a LIN break character is detected. LBKDIF is cleared by writing a 1 to it. 0 No LIN break character has been detected. 1 LIN break character has been detected.
6 RXEDGIF	RxD Pin Active Edge Interrupt Flag RXEDGIF is set when an active edge, falling if RXINV = 0, rising if RXINV=1, on the RxD pin occurs. RXEDGIF is cleared by writing a 1 to it. 0 No active edge on the receive pin has occurred. 1 An active edge on the receive pin has occurred.
5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

Table continues on the next page...

SCIx_S2 field descriptions (continued)

Field	Description
4 RXINV	Receive Data Inversion Setting this bit reverses the polarity of the received data input. **NOTE:** Setting RXINV inverts the RxD input for all cases: data bits, start and stop bits, break, and idle. 0 Receive data not inverted. 1 Receive data inverted.
3 RWUID	Receive Wake Up Idle Detect RWUID controls whether the idle character that wakes up the receiver sets the IDLE bit. 0 During receive standby state (RWU = 1), the IDLE bit does not get set upon detection of an idle character. 1 During receive standby state (RWU = 1), the IDLE bit gets set upon detection of an idle character.
2 BRK13	Break Character Generation Length BRK13 selects a longer transmitted break character length. Detection of a framing error is not affected by the state of this bit. 0 Break character is transmitted with length of 10 bit times (if M = 0, SBNS = 0) or 11 (if M = 1, SBNS = 0 or M = 0, SBNS = 1) or 12 (if M = 1, SBNS = 1). 1 Break character is transmitted with length of 13 bit times (if M = 0, SBNS = 0) or 14 (if M = 1, SBNS = 0 or M = 0, SBNS = 1) or 15 (if M = 1, SBNS = 1).
1 LBKDE	LIN Break Detection Enable LBKDE selects a longer break character detection length. While LBKDE is set, framing error (FE) and receive data register full (RDRF) flags are prevented from setting. 0 Break character is detected at length 10 bit times (if M = 0, SBNS = 0) or 11 (if M = 1, SBNS = 0 or M = 0, SBNS = 1) or 12 (if M = 1, SBNS = 1). 1 Break character is detected at length of 11 bit times (if M = 0, SBNS = 0) or 12 (if M = 1, SBNS = 0 or M = 0, SBNS = 1) or 13 (if M = 1, SBNS = 1).
0 RAF	Receiver Active Flag RAF is set when the SCI receiver detects the beginning of a valid start bit, and RAF is cleared automatically when the receiver detects an idle line. This status flag can be used to check whether an SCI character is being received before instructing the MCU to go to stop mode. 0 SCI receiver idle waiting for a start bit. 1 SCI receiver active (RxD input not idle).

22.4.7 SCI Control Register 3 (SCIx_C3)

Address: 1868h base + 6h offset = 186Eh

Bit	7	6	5	4	3	2	1	0
Read	R8	T8	TXDIR	TXINV	ORIE	NEIE	FEIE	PEIE
Write								
Reset	0	0	0	0	0	0	0	0

SCIx_C3 field descriptions

Field	Description
7 R8	Ninth Data Bit for Receiver When the SCI is configured for 9-bit data (M = 1), R8 can be thought of as a ninth receive data bit to the left of the msb of the buffered data in the SCI_D register. When reading 9-bit data, read R8 before reading SCI_D because reading SCI_D completes automatic flag clearing sequences that could allow R8 and SCI_D to be overwritten with new data.
6 T8	Ninth Data Bit for Transmitter When the SCI is configured for 9-bit data (M = 1), T8 may be thought of as a ninth transmit data bit to the left of the msb of the data in the SCI_D register. When writing 9-bit data, the entire 9-bit value is transferred to the SCI shift register after SCI_D is written so T8 should be written, if it needs to change from its previous value, before SCI_D is written. If T8 does not need to change in the new value, such as when it is used to generate mark or space parity, it need not be written each time SCI_D is written.
5 TXDIR	TxD Pin Direction in Single-Wire Mode When the SCI is configured for single-wire half-duplex operation (LOOPS = RSRC = 1), this bit determines the direction of data at the TxD pin. 0 TxD pin is an input in single-wire mode. 1 TxD pin is an output in single-wire mode.
4 TXINV	Transmit Data Inversion Setting this bit reverses the polarity of the transmitted data output. **NOTE:** Setting TXINV inverts the TxD output for all cases: data bits, start and stop bits, break, and idle. 0 Transmit data not inverted. 1 Transmit data inverted.
3 ORIE	Overrun Interrupt Enable This bit enables the overrun flag (OR) to generate hardware interrupt requests. 0 OR interrupts disabled; use polling. 1 Hardware interrupt requested when OR is set.
2 NEIE	Noise Error Interrupt Enable This bit enables the noise flag (NF) to generate hardware interrupt requests. 0 NF interrupts disabled; use polling). 1 Hardware interrupt requested when NF is set.
1 FEIE	Framing Error Interrupt Enable This bit enables the framing error flag (FE) to generate hardware interrupt requests. 0 FE interrupts disabled; use polling). 1 Hardware interrupt requested when FE is set.
0 PEIE	Parity Error Interrupt Enable This bit enables the parity error flag (PF) to generate hardware interrupt requests. 0 PF interrupts disabled; use polling). 1 Hardware interrupt requested when PF is set.

22.4.8 SCI Data Register (SCIx_D)

This register is actually two separate registers. Reads return the contents of the read-only receive data buffer and writes go to the write-only transmit data buffer. Reads and writes of this register are also involved in the automatic flag clearing mechanisms for the SCI status flags.

Address: 1868h base + 7h offset = 186Fh

Bit	7	6	5	4	3	2	1	0
Read Write	R7T7	R6T6	R5T5	R4T4	R3T3	R2T2	R1T1	R0T0
Reset	0	0	0	0	0	0	0	0

SCIx_D field descriptions

Field	Description
7 R7T7	Read receive data buffer 7 or write transmit data buffer 7.
6 R6T6	Read receive data buffer 6 or write transmit data buffer 6.
5 R5T5	Read receive data buffer 5 or write transmit data buffer 5.
4 R4T4	Read receive data buffer 4 or write transmit data buffer 4.
3 R3T3	Read receive data buffer 3 or write transmit data buffer 3.
2 R2T2	Read receive data buffer 2 or write transmit data buffer 2.
1 R1T1	Read receive data buffer 1 or write transmit data buffer 1.
0 R0T0	Read receive data buffer 0 or write transmit data buffer 0.

22.5 Functional description

The SCI allows full-duplex, asynchronous, NRZ serial communication among the MCU and remote devices, including other MCUs.

The SCI comprises a baud rate generator, transmitter, and receiver block. The transmitter and receiver operate independently, although they use the same baud rate generator. During normal operation, the MCU monitors the status of the SCI, writes the data to be transmitted, and processes received data. The following describes each of the blocks of the SCI.

22.5.1 Baud rate generation

As shown in the figure found here, the clock source for the SCI baud rate generator is the bus-rate clock.

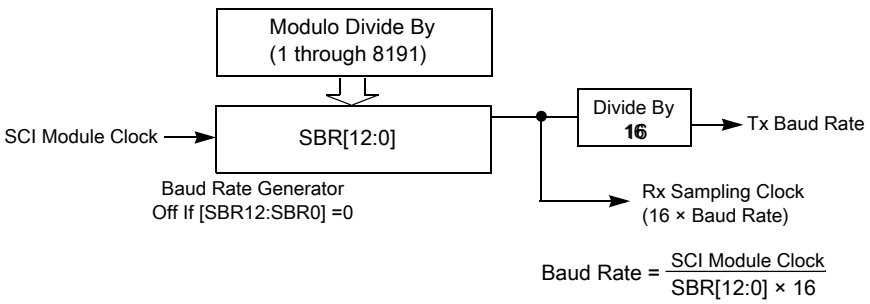


Figure 22-3. SCI baud rate generation

SCI communications require the transmitter and receiver, which typically derive baud rates from independent clock sources, to use the same baud rate. Allowed tolerance on this baud frequency depends on the details of how the receiver synchronizes to the leading edge of the start bit and how bit sampling is performed.

The MCU resynchronizes to bit boundaries on every high-to-low transition. In the worst case, there are no such transitions in the full 10- or 11-bit or 12-bittime character frame so any mismatch in baud rate is accumulated for the whole character time. For an NXP SCI system whose bus frequency is driven by a crystal, the allowed baud rate mismatch is about ±4.5 percent for 8-bit data format and about ±4 percent for 9-bit data format. Although baud rate modulo divider settings do not always produce baud rates that exactly match standard rates, it is normally possible to get within a few percent, which is acceptable for reliable communications.

22.5.2 Transmitter functional description

This section describes the overall block diagram for the SCI transmitter, as well as specialized functions for sending break and idle characters.

The transmitter output (TxD) idle state defaults to logic high, SCI_C3[TXINV] is cleared following reset. The transmitter output is inverted by setting SCI_C3[TXINV]. The transmitter is enabled by setting the TE bit in SCI_C2. This queues a preamble character that is one full character frame of the idle state. The transmitter then remains idle until data is available in the transmit data buffer. Programs store data into the transmit data buffer by writing to the SCI data register (SCI_D).

The central element of the SCI transmitter is the transmit shift register that is 10 or 11 or 12 bits long depending on the setting in the SCI_C1[M] control bit and SCI_BDH[SBNS] bit. For the remainder of this section, assume SCI_C1[M] is cleared, SCI_BDH[SBNS] is also cleared, selecting the normal 8-bit data mode. In 8-bit data mode, the shift register holds a start bit, eight data bits, and a stop bit. When the transmit shift register is available for a new SCI character, the value waiting in the transmit data register is transferred to the shift register, synchronized with the baud rate clock, and the transmit data register empty (SCI_S1[TDRE]) status flag is set to indicate another character may be written to the transmit data buffer at SCI_D.

NOTE

Always read SCI_S1 before writing to SCI_D to allow data to be transmitted.

If no new character is waiting in the transmit data buffer after a stop bit is shifted out the TxD pin, the transmitter sets the transmit complete flag and enters an idle mode, with TxD high, waiting for more characters to transmit.

Writing 0 to SCI_C2[TE] does not immediately release the pin to be a general-purpose I/O pin. Any transmit activity in progress must first be completed. This includes data characters in progress, queued idle characters, and queued break characters.

22.5.2.1 Send break and queued idle

SCI_C2[SBK] sends break characters originally used to gain the attention of old teletype receivers. Break characters are a full character time of logic 0, 10 bit times including the start and stop bits. A longer break of 13 bit times can be enabled by setting SCI_S2[BRK13]. Normally, a program would wait for SCI_S1[TDRE] to become set to indicate the last character of a message has moved to the transmit shifter, write 1, and then write 0 to SCI_C2[SBK]. This action queues a break character to be sent as soon as the shifter is available. If SCI_C2[SBK] remains 1 when the queued break moves into the shifter, synchronized to the baud rate clock, an additional break character is queued. If the receiving device is another NXP SCI, the break characters are received as 0s in all eight data bits and a framing error (SCI_S1[FE] = 1) occurs.

When idle-line wake-up is used, a full character time of idle (logic 1) is needed between messages to wake up any sleeping receivers. Normally, a program would wait for SCI_S1[TDRE] to become set to indicate the last character of a message has moved to the transmit shifter, then write 0 and then write 1 to the SCI_C2[TE] bit. This action queues an idle character to be sent as soon as the shifter is available. As long as the character in the shifter does not finish while SCI_C2[TE] is cleared, the SCI transmitter never actually releases control of the TxD pin. If there is a possibility of the shifter finishing while SCI_C2[TE] is cleared, set the general-purpose I/O controls so the pin shared with TxD is an output driving a logic 1. This ensures that the TxD line looks like a normal idle line even if the SCI loses control of the port pin between writing 0 and then 1 to SCI_C2[TE].

The length of the break character is affected by the SCI_S2[BRK13] and SCI_C1[M] as shown below.

Table 22-4. Break character length

BRK13	M	SBNS	Break character length
0	0	0	10 bit times
0	0	1	11 bit times
0	1	0	11 bit times
0	1	1	12 bit times
1	0	0	13 bit times
1	0	1	14 bit times
1	1	0	14 bit times
1	1	1	15 bit times

22.5.3 Receiver functional description

In this section, the receiver block diagram is a guide for the overall receiver functional description.

Next, the data sampling technique used to reconstruct receiver data is described in more detail. Finally, two variations of the receiver wakeup function are explained.

The receiver input is inverted by setting SCI_S2[RXINV]. The receiver is enabled by setting the SCI_C2[RE] bit. Character frames consist of a start bit of logic 0, eight (or nine) data bits (lsb first), and one (or two) stop bits of logic 1. For information about 9-bit data mode, refer to 8- and 9-bit data modes. For the remainder of this discussion, assume the SCI is configured for normal 8-bit data mode.

After receiving the stop bit into the receive shifter, and provided the receive data register is not already full, the data character is transferred to the receive data register and the receive data register full (SCI_S1[RDRF]) status flag is set. If SCI_S1[RDRF] was already set indicating the receive data register (buffer) was already full, the overrun (OR) status flag is set and the new data is lost. Because the SCI receiver is double-buffered, the program has one full character time after SCI_S1[RDRF] is set before the data in the receive data buffer must be read to avoid a receiver overrun.

When a program detects that the receive data register is full (SCI_S1[RDRF] = 1), it gets the data from the receive data register by reading SCI_D. The SCI_S1[RDRF] flag is cleared automatically by a two-step sequence normally satisfied in the course of the user's program that manages receive data. Refer to Interrupts and status flags for more details about flag clearing.

22.5.3.1 Data sampling technique

The SCI receiver uses a 16× baud rate clock for sampling. The oversampling ratio is fixed at 16. The receiver starts by taking logic level samples at 16 times the baud rate to search for a falling edge on the RxD serial data input pin. A falling edge is defined as a logic 0 sample after three consecutive logic 1 samples. The 16× baud rate clock divides the bit time into 16 segments labeled SCI_D[RT1] through SCI_D[RT16]. When a falling edge is located, three more samples are taken at SCI_D[RT3], SCI_D[RT5], and SCI_D[RT7] to make sure this was a real start bit and not merely noise. If at least two of these three samples are 0, the receiver assumes it is synchronized to a receive character.

The receiver then samples each bit time, including the start and stop bits, at SCI_D[RT8], SCI_D[RT9], and SCI_D[RT10] to determine the logic level for that bit. The logic level is interpreted to be that of the majority of the samples taken during the bit time. In the case of the start bit, the bit is assumed to be 0 if at least two of the samples at SCI_D[RT3], SCI_D[RT5], and SCI_D[RT7] are 0 even if one or all of the samples taken at SCI_D[RT8], SCI_D[RT9], and SCI_D[RT10] are 1s. If any sample in any bit time, including the start and stop bits, in a character frame fails to agree with the logic level for that bit, the noise flag (SCI_S1[NF]) is set when the received character is transferred to the receive data buffer.

The falling edge detection logic continuously looks for falling edges. If an edge is detected, the sample clock is resynchronized to bit times. This improves the reliability of the receiver in the presence of noise or mismatched baud rates. It does not improve worst case analysis because some characters do not have any extra falling edges anywhere in the character frame.

In the case of a framing error, provided the received character was not a break character, the sampling logic that searches for a falling edge is filled with three logic 1 samples so that a new start bit can be detected almost immediately.

In the case of a framing error, the receiver is inhibited from receiving any new characters until the framing error flag is cleared. The receive shift register continues to function, but a complete character cannot transfer to the receive data buffer if SCI_S1[FE] remains set.

22.5.3.2 Receiver wake-up operation

Receiver wake-up is a hardware mechanism that allows an SCI receiver to ignore the characters in a message intended for a different SCI receiver. In such a system, all receivers evaluate the first character(s) of each message, and as soon as they determine the message is intended for a different receiver, they write logic 1 to the receiver wake up control field (SCI_C2[RWU]). When SCI_C2[RWU] is set, the status flags associated with the receiver, (with the exception of the idle bit, IDLE, when SCI_S2[RWUID] is set), are inhibited from setting, thus eliminating the software overhead for handling the unimportant message characters. At the end of a message, or at the beginning of the next message, all receivers automatically force SCI_C2[RWU] to 0, so all receivers wake up in time to look at the first character(s) of the next message.

22.5.3.2.1 Idle-line wakeup

When wake is cleared, the receiver is configured for idle-line wakeup. In this mode, SCI_C2[RWU] is cleared automatically when the receiver detects a full character time of the idle-line level. The SCI_C1[M] control field selects 8-bit or 9-bit data mode and SCI_BDH[SBNS] selects 1-bit or 2-bit stop bit number that determines how many bit times of idle are needed to constitute a full character time, 10 or 11 or 12 bit times because of the start and stop bits.

When SCII_C2[RWU] is 1 and SCI_S2[RWUID] is 0, the idle condition that wakes up the receiver does not set SCI_S1[IDLE]. The receiver wakes up and waits for the first data character of the next message that sets SCI_S1[RDRF] and generates an interrupt, if enabled. When SCI_S2[RWUID] is 1, any idle condition sets SCI_S1[IDLE] flag and generates an interrupt if enabled, regardless of whether SCI_C2[RWU] is 0 or 1.

The idle-line type (SCI_C1[ILT]) control bit selects one of two ways to detect an idle line. When SCI_C1[ILT] is cleared, the idle bit counter starts after the start bit so the stop bit and any logic 1s at the end of a character count toward the full character time of idle. When SCI_C1[ILT] is set, the idle bit counter does not start until after a stop bit time, so the idle detection is not affected by the data in the last character of the previous message.

22.5.3.2.2 Address-mark wakeup

When wake is set, the receiver is configured for address-mark wakeup. In this mode, SCI_C2[RWU] is cleared automatically when the receiver detects a, or two, if SCI_BDH[SBNS] = 1, logic 1 in the most significant bits of a received character, eighth bit when SCI_C1[M] is cleared and ninth bit when SCI_C1[M] is set.

Address-mark wakeup allows messages to contain idle characters, but requires the msb be reserved for use in address frames. The one, or two, if SCI_BDH[SBNS] = 1, logic 1s msb of an address frame clears the SCI_C2[RWU] bit before the stop bits are received and sets the SCI_S1[RDRF] flag. In this case, the character with the msb set is received even though the receiver was sleeping during most of this character time.

22.5.4 Interrupts and status flags

The SCI system has three separate interrupt vectors to reduce the amount of software needed to isolate the cause of the interrupt.

One interrupt vector is associated with the transmitter for SCI_S1[TDRE] and SCI_S1[TC] events. Another interrupt vector is associated with the receiver for RDRF, IDLE, RXEDGIF, and LBKDIF events. A third vector is used for OR, NF, FE, and PF error conditions. Each of these ten interrupt sources can be separately masked by local interrupt enable masks. The flags can be polled by software when the local masks are cleared to disable generation of hardware interrupt requests.

The SCI transmitter has two status flags that can optionally generate hardware interrupt requests. Transmit data register empty (SCI_S1[TDRE]) indicates when there is room in the transmit data buffer to write another transmit character to SCI_D. If the transmit interrupt enable (SCI_C2[TIE]) bit is set, a hardware interrupt is requested when SCI_S1[TDRE] is set. Transmit complete (SCI_S1[TC]) indicates that the transmitter is finished transmitting all data, preamble, and break characters and is idle with TxD at the inactive level. This flag is often used in systems with modems to determine when it is safe to turn off the modem. If the transmit complete interrupt enable (SCI_C2[TCIE]) bit is set, a hardware interrupt is requested when SCI_S1[TC] is set. Instead of hardware interrupts, software polling may be used to monitor the SCI_S1[TDRE] and SCI_S1[TC] status flags if the corresponding SCI_C2[TIE] or SCI_C2[TCIE] local interrupt masks are cleared.

When a program detects that the receive data register is full (SCI_S1[RDRF] = 1), it gets the data from the receive data register by reading SCI_D. The SCI_S1[RDRF] flag is cleared by reading SCI_S1 while SCI_S1[RDRF] is set and then reading SCI_D.

When polling is used, this sequence is naturally satisfied in the normal course of the user program. If hardware interrupts are used, SCI_S1 must be read in the interrupt service routine (ISR). Normally, this is done in the ISR anyway to check for receive errors, so the sequence is automatically satisfied.

The IDLE status flag includes logic that prevents it from getting set repeatedly when the RxD line remains idle for an extended period of time. IDLE is cleared by reading SCI_S1 while SCI_S1[IDLE] is set and then reading SCI_D. After SCI_S1[IDLE] has been cleared, it cannot become set again until the receiver has received at least one new character and has set SCI_S1[RDRF].

If the associated error was detected in the received character that caused SCI_S1[RDRF] to be set, the error flags - noise flag (SCI_S1[NF]), framing error (SCI_S1[FE]), and parity error flag (SCI_S1[PF]) - are set at the same time as SCI_S1[RDRF]. These flags are not set in overrun cases.

If SCI_S1[RDRF] was already set when a new character is ready to be transferred from the receive shifter to the receive data buffer, the overrun (SCI_S1[OR]) flag is set instead of the data along with any associated NF, FE, or PF condition is lost.

At any time, an active edge on the RxD serial data input pin causes the SCI_S2[RXEDGIF] flag to set. The SCI_S2[RXEDGIF] flag is cleared by writing a 1 to it. This function depends on the receiver being enabled (SCI_C2[RE] = 1).

22.5.5 Baud rate tolerance

A transmitting device may operate at a baud rate below or above that of the receiver.

Accumulated bit time misalignment can cause one of the three stop bit data samples (RT8, RT9, and RT10) to fall outside the actual stop bit. A noise error will occur if the RT8, RT9, and RT10 samples are not all the same logical values. A framing error will occur if the receiver clock is misaligned in such a way that the majority of the RT8, RT9, and RT10 stop bit samples are a logic zero.

As the receiver samples an incoming frame, it re-synchronizes the RT clock on any valid falling edge within the frame. Resynchronization within frames will correct a misalignment between transmitter bit times and receiver bit times.

22.5.5.1 Slow data tolerance

Figure 22-4 shows how much a slow received frame can be misaligned without causing a noise error or a framing error. The slow stop bit begins at RT8 instead of RT1 but arrives in time for the stop bit data samples at RT8, RT9, and RT10.

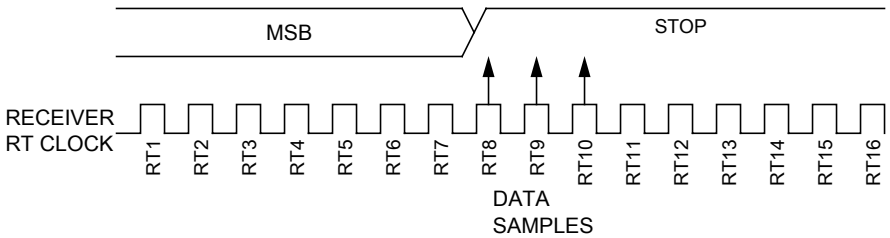


Figure 22-4. Slow data

For an 8-bit data and 1 stop bit character, data sampling of the stop bit takes the receiver 9 bit times x 16 RT cycles +10 RT cycles =154 RT cycles.

With the misaligned character shown in Figure 22-4, the receiver counts 154 RT cycles at the point when the count of the transmitting device is 9 bit times x 16 RT cycles + 3 RT cycles = 147 RT cycles.

The maximum percent difference between the receiver count and the transmitter count of a slow 8-bit data and 1 stop bit character with no errors is:

((154 - 147) / 154) x 100 = 4.54%

For a 9-bit data or 2 stop bits character, data sampling of the stop bit takes the receiver 10 bit times x 16 RT cycles + 10 RT cycles = 170 RT cycles.

With the misaligned character shown in Figure 22-4, the receiver counts 170 RT cycles at the point when the count of the transmitting device is 10 bit times x 16 RT cycles + 3 RT cycles = 163 RT cycles.

The maximum percent difference between the receiver count and the transmitter count of a slow 9-bit or 2 stop bits character with no errors is:

((170 - 163) / 170) X 100 = 4.12%

For a 9-bit data and 2 stop bit character, data sampling of the stop bit takes the receiver 11 bit times x 16 RT cycles + 10 RT cycles = 186 RT cycles.

With the misaligned character shown in Figure 22-4, the receiver counts 186 RT cycles at the point when the count of the transmitting device is 11 bit times x 16 RT cycles + 3 RT cycles = 179 RT cycles.

The maximum percent difference between the receiver count and the transmitter count of a slow 9-bit and 2 stop bits character with no errors is: ((186 - 179) / 186) X 100 = 3.76%

22.5.5.2 Fast data tolerance

Figure 22-5 shows how much a fast received frame can be misaligned. The fast stop bit ends at RT10 instead of RT16 but is still sampled at RT8, RT9, and RT10.

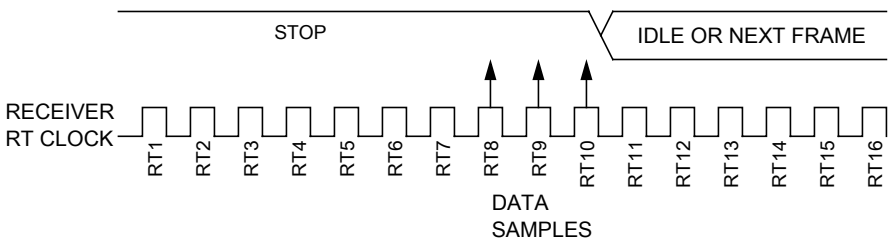


Figure 22-5. Fast data

For an 8-bit data and 1 stop bit character, data sampling of the stop bit takes the receiver 9 bit times x 16 RT cycles + 10 RT cycles = 154 RT cycles.

With the misaligned character shown in Figure 22-5, the receiver counts 154 RT cycles at the point when the count of the transmitting device is 10 bit times x 16 RT cycles = 160 RT cycles.

The maximum percent difference between the receiver count and the transmitter count of a fast 8-bit and 1 stop bit character with no errors is:

((154 - 160) / 154) x 100 = 3.90%

For a 9-bit data or 2 stop bits character, data sampling of the stop bit takes the receiver 10 bit times x 16 RT cycles + 10 RT cycles = 170 RT cycles.

With the misaligned character shown in, the receiver counts 170 RT cycles at the point when the count of the transmitting device is 11 bit times x 16 RT cycles = 176 RT cycles.

The maximum percent difference between the receiver count and the transmitter count of a fast 9-bit or 2 stop bits character with no errors is:

((170 - 176) / 170) x 100 = 3.53%

For a 9-bit data and 2 stop bits character, data sampling of the stop bit takes the receiver 11 bit times x 16 RT cycles + 10 RT cycles = 186 RT cycles.

With the misaligned character shown in, the receiver counts 186 RT cycles at the point when the count of the transmitting device is 12 bit times x 16 RT cycles = 192 RT cycles.

The maximum percent difference between the receiver count and the transmitter count of a fast 9-bit and 2 stop bits character with no errors is:

((186 - 192) / 186) x 100 = 3.23%

22.5.6 Additional SCI functions

The following sections describe additional SCI functions.

22.5.6.1 8- and 9-bit data modes

The SCI system, transmitter and receiver, can be configured to operate in 9-bit data mode by setting SCI_C1[M]. In 9-bit mode, there is a ninth data bit to the left of the most significant bit of the SCI data register. For the transmit data buffer, this bit is stored in T8 in SCI_C3. For the receiver, the ninth bit is held in SCI_C3[R8].

For coherent writes to the transmit data buffer, write to SCI_C3[T8] before writing to SCI_D.

If the bit value to be transmitted as the ninth bit of a new character is the same as for the previous character, it is not necessary to write to SCI_C3[T8] again. When data is transferred from the transmit data buffer to the transmit shifter, the value in SCI_C3[T8] is copied at the same time data is transferred from SCI_D to the shifter.

The 9-bit data mode is typically used with parity to allow eight bits of data plus the parity in the ninth bit, or it is used with address-mark wake-up so the ninth data bit can serve as the wakeup bit. In custom protocols, the ninth bit can also serve as a software-controlled marker.

22.5.6.2 Stop mode operation

During all stop modes, clocks to the SCI module are halted.

No SCI module registers are affected in Stop mode.

The receive input active edge detect circuit remains active in Stop mode. An active edge on the receive input brings the CPU out of Stop mode if the interrupt is not masked (SCI_BDH[RXEDGIE] = 1).

Because the clocks are halted, the SCI module resumes operation upon exit from stop, only in Stop mode. Software must ensure stop mode is not entered while there is a character (including preamble, break and normal data) being transmitted out of or received into the SCI module, that means SCI_S1[TC] =1, SCI_S1[TDRE] = 1, and SCI_S2[RAF] = 0 must all meet before entering stop mode.

22.5.6.3 Loop mode

When SCI_C1[LOOPS] is set, the SCI_C1[RSRC] bit in the same register chooses between loop mode (SCI_C1[RSRC] = 0) or single-wire mode (SCI_C1[RSRC] = 1). Loop mode is sometimes used to check software, independent of connections in the external system, to help isolate system problems. In this mode, the internal loop back connection from the transmitter to the receiver causes the receiver to receive characters that are sent out by the transmitter.

22.5.6.4 Single-wire operation

When SCI_C1[LOOPS] is set, SCI_C1[RSRC] chooses between loop mode (SCI_C1[RSRC] = 0) or single-wire mode (SCI_C1[RSRC] = 1). Single-wire mode implements a half-duplex serial connection. The receiver is internally connected to the transmitter output and to the TxD pin. The RxD pin is not used and reverts to a general-purpose port I/O pin.

In single-wire mode, the SCI_C3[TXDIR] bit controls the direction of serial data on the TxD pin. When SCI_C3[TXDIR] is cleared, the TxD pin is an input to the SCI receiver and the transmitter is temporarily disconnected from the TxD pin so an external device can send serial data to the receiver. When SCI_C3[TXDIR] is set, the TxD pin is an output driven by the transmitter. In single-wire mode, the transmitter output is internally connected to the receiver input and the RxD pin is not used by the SCI, so it reverts to a general-purpose port I/O pin.

Chapter 23
Programmable Delay Block (PDB)

23.1 Chip specific programmable delay block

This device has one programmable delay block (PDB)

The PDB is used to synchronize between PWM sync pulse and ADC samples or comparator sampling windows. The primary function of the programmable delay block is simply to provide a controllable delay from a trigger signal . The module includes two timebase. Each has its own counter and one comparator against the counter. Its operation modes include single shot or continuous mode. The input and output of PDBs are connected through intermodule crossbar to other modules, such as PWM, ADCs, Comparators.

Customization:

- Primary clock: HSCLK (up to 40 MHz)
- Alternate clock: No
- The following table summarizes the signal connection of PDB modules.

Table 23-1. PDB module signals Connection

Module	Signal	Connect to
PDB0	PDB0 HW Trigger	XBAR_OUT8
	PDB0 Output	XBAR_IN8
	Internal Clock	HSCLK
PDB1	PDB1 HW trigger	XBAR_OUT9
	PDB1 OUTPUT	XBAR_IN9
	Internal Clock	HSCLK

23.2 Introduction

The Programmable Delay Block (PDB) provides controllable delays from either an internal or an external trigger, to the hardware trigger inputs of ADCs and/or generates the interval triggers to DACs, so that the precise timing between ADC conversions and/or DAC updates can be achieved.

23.3 Features

The PDB has the following features:

- 16-bit resolution with a shared prescaler
- Support software and hardware trigger
- Support continuous count mode or single shot delay mode
- Supply on-fly delay value update
- Selective output mode: logic level or one-shot pulse

23.4 Block diagram

The following figure show the block diagram of the PDB.

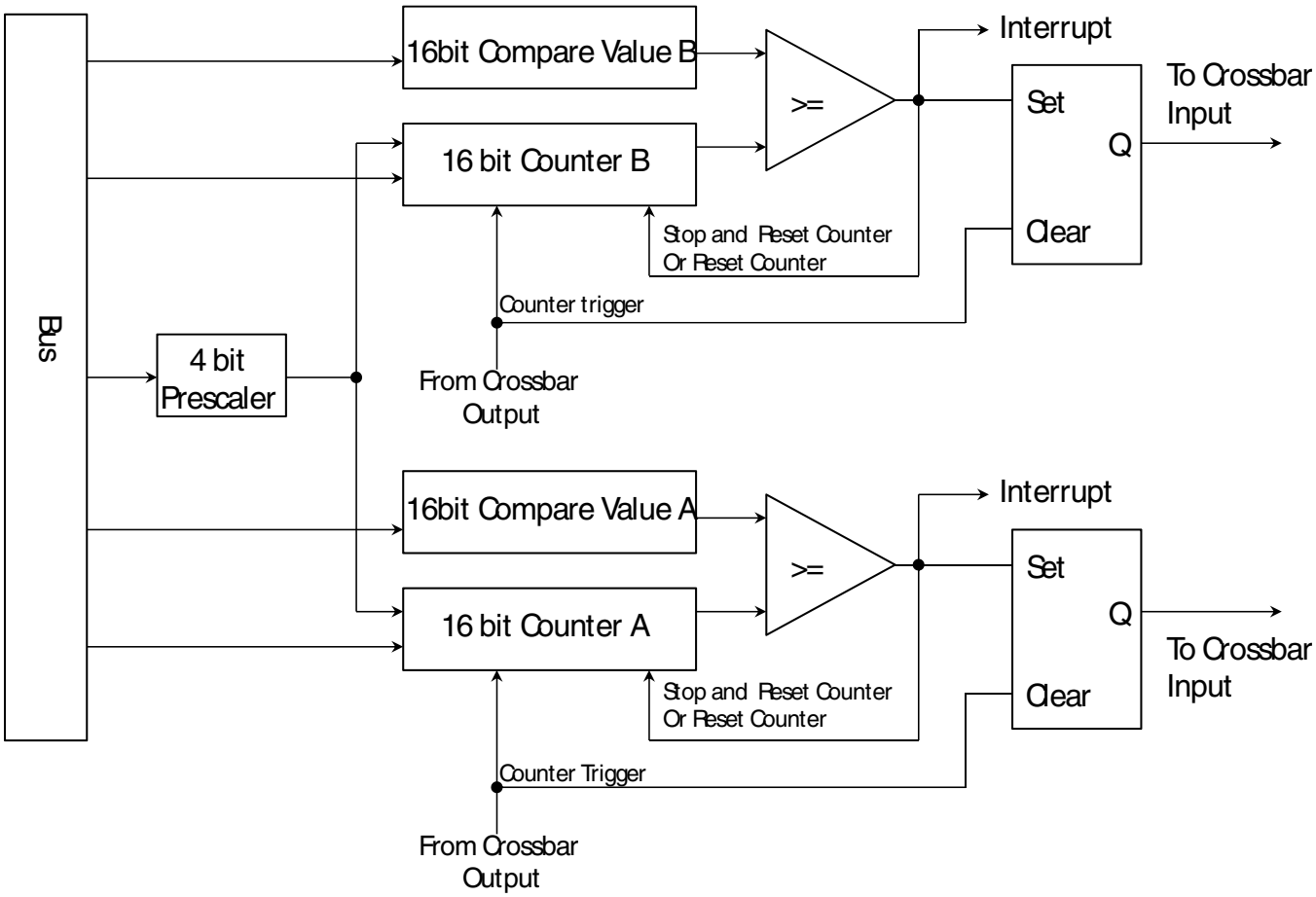


Figure 23-1. PDB block diagram

23.5 Mode of operation

23.5.1 Single shot delay mode

In the single shot delay mode:

1. Trigger pulse (include both HW and SW trigger) clear output and start counter from zero
2. Before counter reaches compare value, new trigger will restart counter from zero
3. When counter value is equal and greater than compare value, counter stops and output set to high
4. If Pulse output mode is set, pulse will be generated when counter value is equal and greater than compare value
5. A stop bit is needed to stop counter and reset counter to zero and clear output.

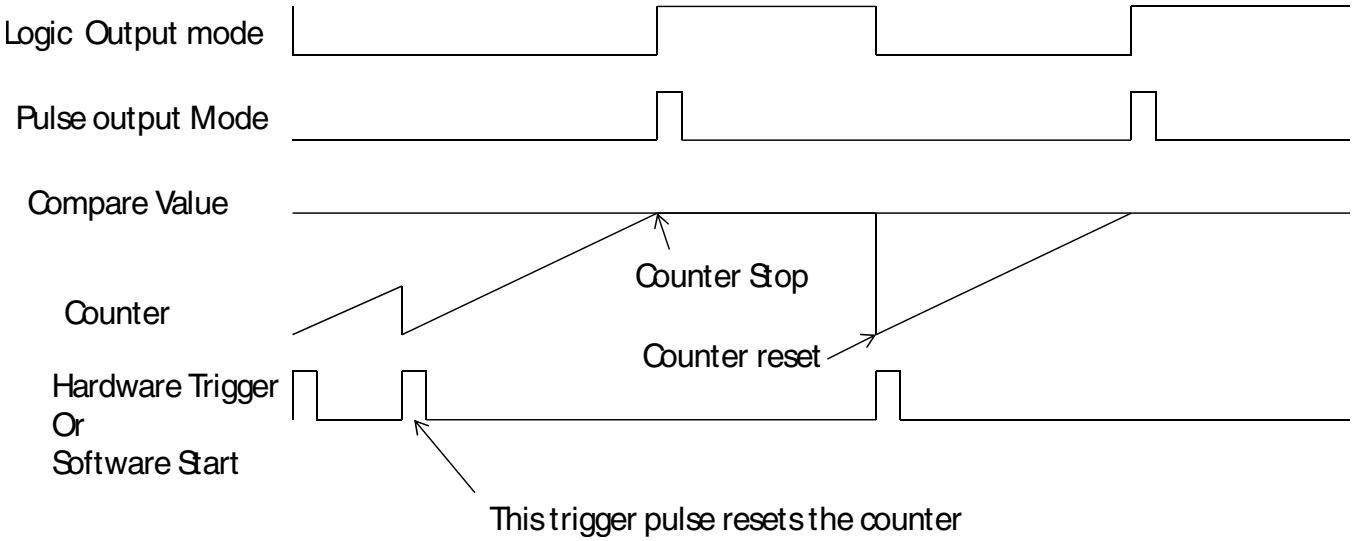


Figure 23-2. Single shot delay mode

23.5.2 Continuous count mode

In the continuous count mode:

1. After counter initial, trigger pulse (include both HW and SW trigger) clear output and start counter from zero
2. Before counter reaches compare value, a trigger will stop counter. Then a trigger will resume counter
3. When counter value is equal and greater than compare value, counter output will toggle. Counter will restart from zero
4. When pulse output mode is set, only rising edge toggle will generate a pulse.
5. A stop bit is needed to stop counter and reset counter to zero and clear output.

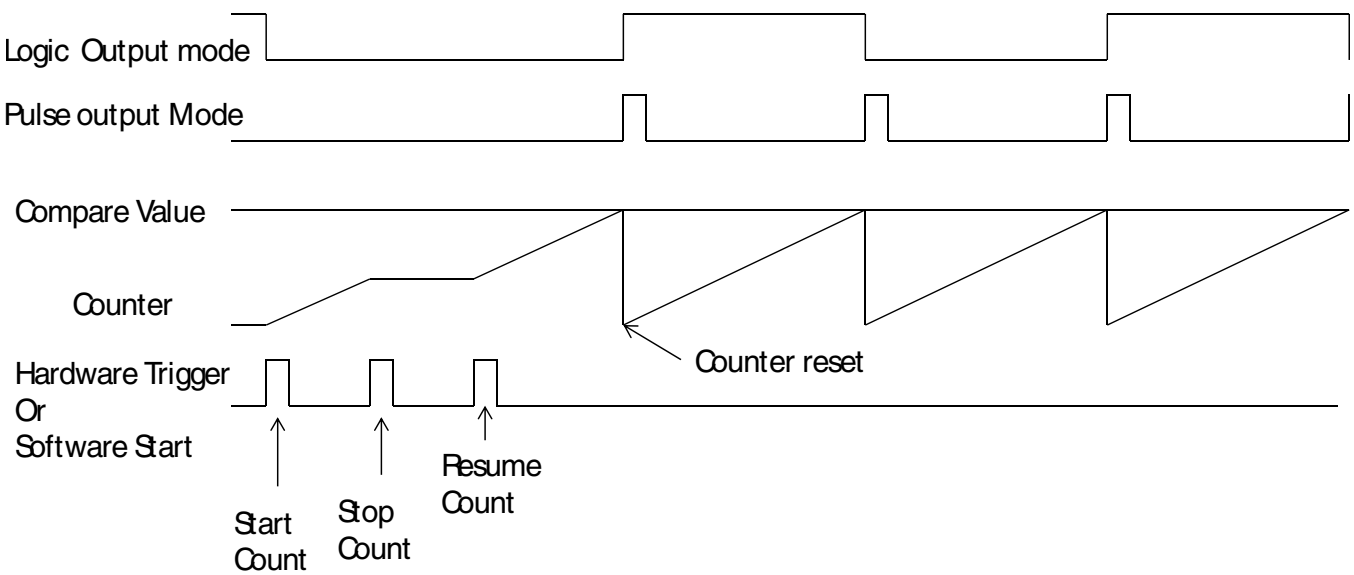


Figure 23-3. Continuous count mode

NOTE

The width of pulse output is between one and two clocks.

To use the I/O trigger to trig the PDB, the user needs to open the I/O filter in advance.

23.6 Memory Map and Register Descriptions

PDB memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
60	PDB Control Register 0 (PDB_CTRL0)	8	R/W	00h	23.6.1/425
61	PDB Control Register 1 (PDB_CTRL1)	8	R/W	00h	23.6.2/426
62	PDB0 Comparison Low Register (PDB_CMPL0)	8	R/W	FFh	23.6.3/427
63	PDB0 Comparison High Register (PDB_CMPH0)	8	R/W	FFh	23.6.4/428
64	PDB0 Counter High/Low (PDB_CNT0)	8	R/W	00h	23.6.5/428
65	PDB1 Comparison Low Register (PDB_CMPL1)	8	R/W	FFh	23.6.6/429
66	PDB1 Comparison High Register (PDB_CMPH1)	8	R/W	FFh	23.6.7/429
67	PDB1 Counter High/Low (PDB_CNT1)	8	R/W	00h	23.6.8/430

23.6.1 PDB Control Register 0 (PDB_CTRL0)

Address: 60h base + 0h offset = 60h

Bit	7	6	5	4	3	2	1	0
Read	TCF1	TCIE1	TRGOUT1	MOD1	TCF0	TCIE0	TRGOUT0	MOD0
Write	w1c				w1c			
Reset	0	0	0	0	0	0	0	0

PDB_CTRL0 field descriptions

Field	Description
7 TCF1	PDB1 Timer Compare Flag This bit is set when a successful compare occurs. Clear this bit by writing one to it.
6 TCIE1	PDB1 Timer Compare Interrupt Enable 0 Timer compare interrupt requests disabled. 1 Timer compare interrupt requests enabled.
5 TRGOUT1	PDB1 Trigger Output Configure PDB1 trigger output as an pulse or level when a successful compare occurs.

Table continues on the next page...

PDB_CTRL0 field descriptions (continued)

Field	Description
	0 Output is asserted and de-asserted by input trigger or counter rollover. 1 Output a true high pulse.
4 MOD1	PDB1 Counter Mode Enable 0 Module is in single shot delay mode. 1 Module is in continuous count mode.
3 TCF0	PDB0 Timer Compare Flag This bit is set when a successful compare occurs. Clear this bit by writing one to it.
2 TCIE0	PDB0 Timer Compare Interrupt Enable 0 Timer compare interrupt requests disabled. 1 Timer compare interrupt requests enabled.
1 TRGOUT0	PDB0 Trigger Output Configure PDB0 trigger output as an pulse or level when a successful compare occurs. 0 Logic output mode, output is asserted and de-asserted by input trigger or counter rollover. 1 Pulse output mode, output a true high pulse.
0 MOD0	PDB0 Counter Mode Enable 0 Module is in single shot delay mode. 1 Module is in continuous count mode.

23.6.2 PDB Control Register 1 (PDB_CTRL1)

NOTE

SWCLR0 and SWCLR1 are also used as PDB status bits to indicate whether PDB0 and PDB1 are enabled or not.

Do NOT use BSET and BCLR instruction to this register, which may cause unexpected software clear to SWCLR0 and SWCLR1.

Address: 60h base + 1h offset = 61h

Bit	7	6	5	4	3	2	1	0
Read Write	CNTSEL1	CNTSEL0	PRESCALER		SWCLR1	SWTRG1	SWCLR0	SWTRG0
Reset	0	0	0	0	0	0	0	0

PDB_CTRL1 field descriptions

Field	Description
7 CNTSEL1	PDB1 Counter Selection 0 Read the low byte of 16-bit counter of PDB1 via register CNT1 1 Read the high byte of 16-bit counter of PDB1 via register CNT1
6 CNTSEL0	PDB0 Counter Selection 0 Read the low byte of 16-bit counter of PDB0 via register CNT0 1 Read the high byte of 16-bit counter of PDB0 via register CNT0
5–4 PRESCALER	Clock Prescaler Selection 00 Counter uses peripheral clock. 01 Counter uses peripheral clock / 2. 10 Counter uses peripheral clock / 4. 11 Counter uses peripheral clock / 8.
3 SWCLR1	PDB1 Software Clear Writing a one to this field triggers a reset and stops the counter, and clear output to low. This bit reads as zero when pdb1 is disabled and reads one when pdb1 is enabled.
2 SWTRG1	PDB1 Software Trigger When the module is enabled, writing a one to this field generates a soft trigger to PDB1. This bit read is PDB1 output state in logic level output.
1 SWCLR0	PDB0 Software Clear Writing a one to this field triggers a reset and stops the counter, and clear output to low. This bit reads as zero when pdb0 is disabled and reads one when pdb0 is enabled.
0 SWTRG0	PDB0 Software Trigger When the module is enabled, writing a one to this field generates a soft trigger to PDB0. This bit read is PDB0 output state in logic level output.

23.6.3 PDB0 Comparison Low Register (PDB_CMPL0)

The Comparison registers contain the high and low bytes of the comparison value for the counter. After the counter reaches the comparison value, the timer comparison flag (TCF0) becomes set at the next clock.

NOTE

Writing to the CMPx0 registers would take effect immediately.

Address: 60h base + 2h offset = 62h

Bit	7	6	5	4	3	2	1	0
Read Write	CMPL0							
Reset	1	1	1	1	1	1	1	1

PDB_CMPL0 field descriptions

Field	Description
CMPL0	PDB0 low byte of the comparison value

23.6.4 PDB0 Comparison High Register (PDB_CMPH0)

The Comparison registers contain the high and low bytes of the comparison value for the counter. After the counter reaches the comparison value, the timer comparison flag (TCF0) becomes set at the next clock.

NOTE

Writing to the CMPx0 registers would take effect immediately.

Address: 60h base + 3h offset = 63h

Bit	7	6	5	4	3	2	1	0
Read Write	CMPH0							
Reset	1	1	1	1	1	1	1	1

PDB_CMPH0 field descriptions

Field	Description
CMPH0	PDB0 high byte of the comparison value

23.6.5 PDB0 Counter High/Low (PDB_CNT0)

The Counter registers may read the high or low bytes of the counter value which controlled by [CNTSEL]. When BDM is active, the counter is frozen.

NOTE

Reading either byte would not latch the contents of both bytes into a buffer.

This register is also used to enable PDB0 by writing 1 to bit 0, the writing operation doesn't take effect the counter's value.

Address: 60h base + 4h offset = 64h

Bit	7	6	5	4	3	2	1	0
Read	CNT0_7_1							CNT0_0_ PDBEN0
Write								
Reset	0	0	0	0	0	0	0	0

PDB_CNT0 field descriptions

Field	Description
7–1 CNT0_7_1	PDB0 counter value [7:1] Control bit [CNTSEL] decides read counter high or low.
0 CNT0_0_ PDBEN0	PDB0 Counter Value 0 and Module Enable This bit read as counter value 0 and is also used to enable the PDB module 0, the writing operation doesn't take effect the counter's value. Writing 0 to bit0 of this register: Counter is off and Trigger output is low. Writing 1 to bit0 of this register: Counter is enabled.

23.6.6 PDB1 Comparison Low Register (PDB_CMPL1)

The Comparison registers contain the high and low bytes of the comparison value for the counter. After the counter reaches the comparison value, the timer comparison flag (TCF0) becomes set at the next clock.

NOTE

Writing to the CMPx1 registers would take effect immediately.

Address: 60h base + 5h offset = 65h

Bit	7	6	5	4	3	2	1	0
Read Write	CMPL1							
Reset	1	1	1	1	1	1	1	1

PDB_CMPL1 field descriptions

Field	Description
CMPL1	PDB1 low byte of the comparison value

23.6.7 PDB1 Comparison High Register (PDB_CMPH1)

The Comparison registers contain the high and low bytes of the comparison value for the counter. After the counter reaches the comparison value, the timer comparison flag (TCF0) becomes set at the next clock.

NOTE

Writing to the CMPx1 registers would take effect immediately.

Address: 60h base + 6h offset = 66h

Bit	7	6	5	4	3	2	1	0
Read Write	CMPH1							
Reset	1	1	1	1	1	1	1	1

PDB_CMPH1 field descriptions

Field	Description
CMPH1	PDB1 high byte of the comparison value

23.6.8 PDB1 Counter High/Low (PDB_CNT1)

The Counter registers may read the high or low bytes of the counter value which controlled by [CNTSEL]. When BDM is active, the counter isn't frozen.

NOTE

Reading either byte would not latch the contents of both bytes into a buffer.

This register is also used to enable PDB1 by writing 1 to bit 0, the writing operation doesn’t take effect the counter’s value.

Address: 60h base + 7h offset = 67h

Bit	7	6	5	4	3	2	1	0
Read	CNT1_7_1							CNT1_0_ PDBEN1
Write								
Reset	0	0	0	0	0	0	0	0

PDB_CNT1 field descriptions

Field	Description
7–1 CNT1_7_1	PDB1 counter value [7:1] Control bit [CNTSEL] decides read counter high or low.
0 CNT1_0_ PDBEN1	PDB1 Counter Value 0 and Module Enable This bit read as counter value 0 and is also used to enable the PDB module 1, the writing operation doesn’t take effect the counter’s value. Writing 0 to bit0 of this register: Counter is off and Trigger output is low. Writing 1 to bit0 of this register: Counter is enabled.

Chapter 24
Inter-peripheral Crossbar Switch (XBAR)

24.1 Introduction

This module implements an array of M(16) N(16)-input combinational muxes. All muxes share the same N(16) inputs in the same order, however, each mux has its own independent select field.

24.2 Features

The XBAR has the following features:

- M(16) identical N(16)-input muxes with individual select fields.

24.3 Block diagram

The following figure show the block diagram of the XBAR.

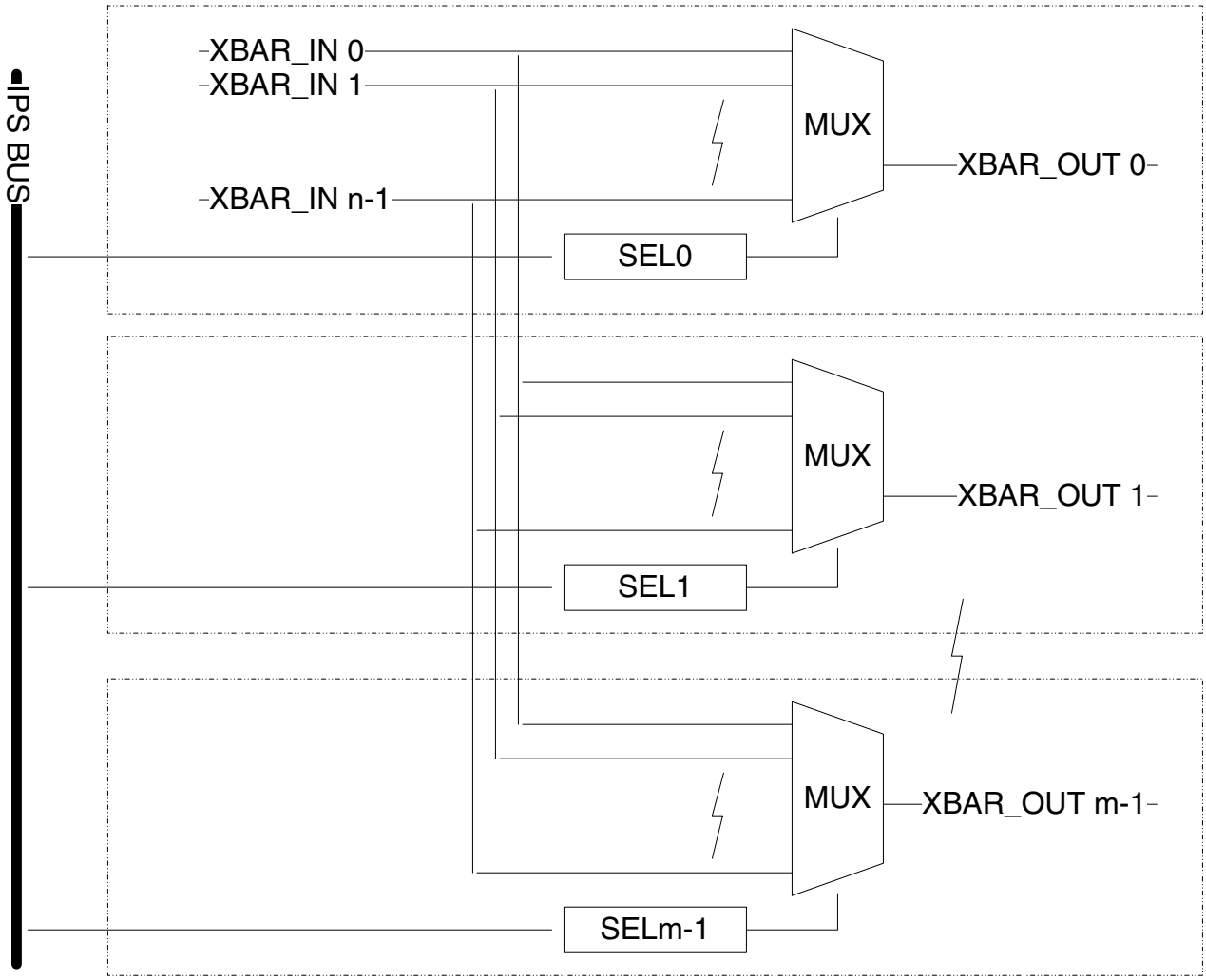


Figure 24-1. XBAR block diagram

24.4 Memory Map and Register Descriptions

XBAR memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
6	External Mux Selection Register (XBAR_EXTMUX)	8	R/W	00h	24.4.1/433
18D0	XBAR Selection Register (XBAR_SEL0)	8	R/W	02h	24.4.2/434
18D1	XBAR Selection Register (XBAR_SEL1)	8	R/W	02h	24.4.2/434
18D2	XBAR Selection Register (XBAR_SEL2)	8	R/W	02h	24.4.2/434
18D3	XBAR Selection Register (XBAR_SEL3)	8	R/W	02h	24.4.2/434
18D4	XBAR Selection Register (XBAR_SEL4)	8	R/W	02h	24.4.2/434
18D5	XBAR Selection Register (XBAR_SEL5)	8	R/W	02h	24.4.2/434

Table continues on the next page...

XBAR memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
18D6	XBAR Selection Register (XBAR_SEL6)	8	R/W	02h	24.4.2/434
18D7	XBAR Selection Register (XBAR_SEL7)	8	R/W	02h	24.4.2/434
18D8	XBAR Selection Register (XBAR_SEL8)	8	R/W	02h	24.4.2/434
18D9	XBAR Selection Register (XBAR_SEL9)	8	R/W	02h	24.4.2/434
18DA	XBAR Selection Register (XBAR_SEL10)	8	R/W	02h	24.4.2/434
18DB	XBAR Selection Register (XBAR_SEL11)	8	R/W	02h	24.4.2/434
18DC	XBAR Selection Register (XBAR_SEL12)	8	R/W	02h	24.4.2/434
18DD	XBAR Selection Register (XBAR_SEL13)	8	R/W	02h	24.4.2/434
18DE	XBAR Selection Register (XBAR_SEL14)	8	R/W	02h	24.4.2/434
18DF	XBAR Selection Register (XBAR_SEL15)	8	R/W	02h	24.4.2/434

24.4.1 External Mux Selection Register (XBAR_EXTMUX)

Address: 0h base + 6h offset = 6h

Bit	7	6	5	4	3	2	1	0
Read	0					SEL2	SEL1	SEL0
Write								
Reset	0	0	0	0	0	0	0	0

XBAR_EXTMUX field descriptions

Field	Description
7–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 SEL2	External mux selection 2 for PWM channels 0 PWM output channel 4. 1 PWM output channel 5.
1 SEL1	External mux selection 1 for PWM channels 0 PWM output channel 2. 1 PWM output channel 3.
0 SEL0	External mux selection 0 for PWM channels 0 PWM output channel 0. 1 PWM output channel 1.

24.4.2 XBAR Selection Register (XBAR_SEL*n*)

Address: 0h base + 18D0h offset + (1d × i), where i=0d to 15d

Bit	7	6	5	4	3	2	1	0
Read	0				SEL			
Write								
Reset	0	0	0	0	0	0	1	0

XBAR_SEL*n* field descriptions

Field	Description
7–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
SEL	Mux Selection Select inputs of XBAR_IN to be muxed to XBAR_OUT.

Chapter 25
Gate Drive Unit (GDU)

25.1 Chip specific GDU information

This device has one GDU

The following table summarizes the signal connection of GDU module.

Table 25-1. GDU module signals connection

Signal	Connect to
Predriver Phase U Top input	PWM0 (PWMA0)
Predriver Phase U Bottom input	PWM1 (PWMA1)
Predriver Phase V Top input	PWM2 (PWMB0)
Predriver Phase V Bottom input	PWM3 (PWMB1)
Predriver Phase W Top input	PWM4 (PWMC0)
Predriver Phase W Bottom input	PWM5 (PWMC1)
Limit ACMP0 output	PWM_Fault2
Limit ACMP1 output	PWM_Fault3
AMP0 analog output	ADCA_AD5 and ADCB_AD5
AMP1 analog output	ADCA_AD6 and ADCB_AD6
Phase detection ACMP0 output	XB_IN12
Phase detection ACMP1 output	XB_IN13
Phase detection ACMP2 output	XB_IN14
Phase detection ACMP0 window input	XB_OUT10
Phase detection ACMP1 window input	XB_OUT11
Phase detection ACMP2 window input	XB_OUT12

25.2 Introduction

The Gate drive Unit (GDU) module is designed for power conversion and three phase motor control applications. It includes high side and low side Field Effect Transistor (FET) pre-drivers, motor BEMF zero crossing detection circuit, two current sensing amplifiers with per-configured 20x gain and limitation detection. GDU is also internally connected to Crossbar module and PWM module and ADC module so the signal propagation delays can be effectively reduced or minimized.

25.3 Features

The module includes the following distinctive features:

- 4.5 V~18 V supply voltage
- 3-phase bridge MOSTFET pre-drivers
  - 3 high side P-MOSFET logic gate drivers
  - 3 low side N-MOSFET logic gate drivers
  - Controllable drive strength
- Voltage clamp circuit for high-side P-MOSFET pre-drivers with external clamp capacitor
- Two low-side current measurement amplifiers for current measurement
  - Differential input
  - Pre-configured 20x gain
  - Selectable offset sources
  - Two over-limit comparators with programmable voltage threshold and digital filter and selectable output level and programmable hysteresis control
- BEMF zero crossing detection in sensor-less BLDC applications
  - Three Phase comparator with Digital filter and selectable output level and programmable hysteresis control and windowed output
  - Virtual neutral network circuit reconstructing 3 phase motor neutral
  - Programmable Phase selection allows the maximize the gain of BEMF zero crossing detection
  - Selectable offset from internal 6bit DAC or external pin
- Over-voltage detection on supply VDD pin

25.4 Block diagram

The following figure shows the block diagram of GDU.

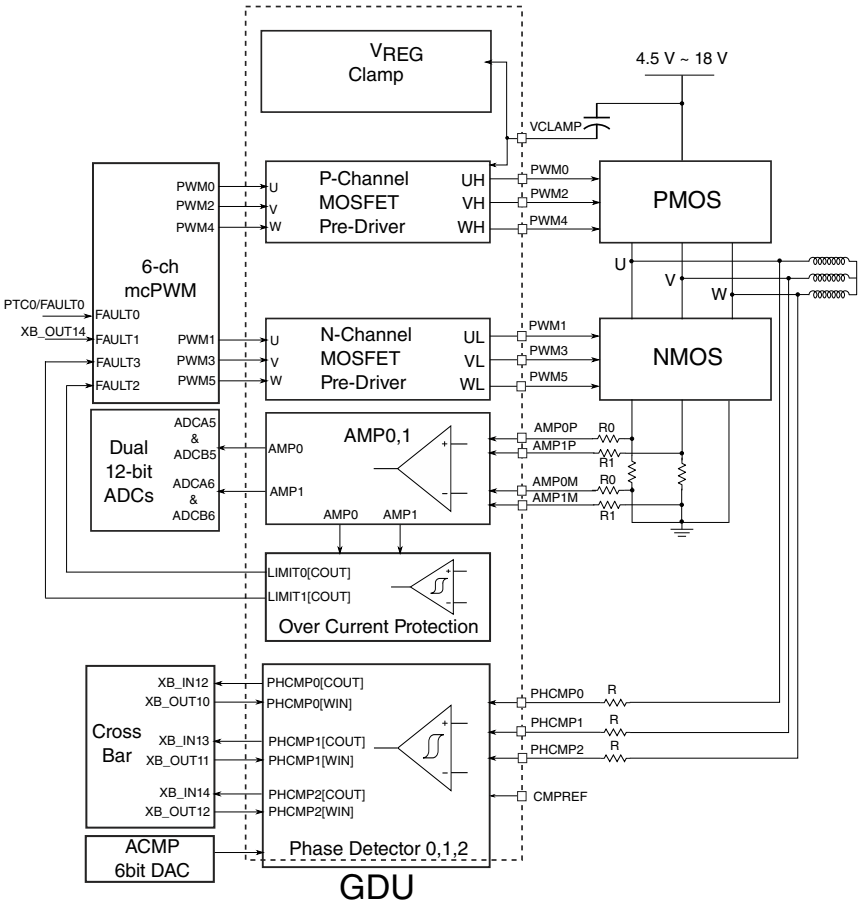


Figure 25-1. GDU block diagram

25.5 Modes of operation

Under different system power modes, the module behaves as follows:

1. Run mode

   All features of the module are available.

   NOTE

   In Run mode, there are two sub-operational modes. One is the regulation mode when V_{DD} is greater than 5.5 V, and the other is the open-loop mode when V_{DD} is less than 5.5 V. In the open-loop mode, the clamped voltage follows with V_{DD}.

2. Stop mode

   The module is disabled in Stop mode. All internal analog circuits are switched off.

25.6 Memory map and register definition

This section includes the module memory map and detailed descriptions of all registers.

GDU memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
20	PHCMP0 Control Register 0 (GDU_PHCMP0CR0)	8	R/W	00h	25.6.1/439
21	PHCMP0 Control Register 1 (GDU_PHCMP0CR1)	8	R/W	00h	25.6.2/439
22	PHCMP0 Filter Period Register (GDU_PHCMP0FPR)	8	R/W	00h	25.6.3/441
23	PHCMP0 Status and Control Register (GDU_PHCMP0SCR)	8	R/W	00h	25.6.4/441
24	PHCMP1 Control Register 0 (GDU_PHCMP1CR0)	8	R/W	00h	25.6.5/442
25	PHCMP1 Control Register 1 (GDU_PHCMP1CR1)	8	R/W	00h	25.6.6/443
26	PHCMP1 Filter Period Register (GDU_PHCMP1FPR)	8	R/W	00h	25.6.7/444
27	PHCMP1 Status and Control Register (GDU_PHCMP1SCR)	8	R/W	00h	25.6.8/445
28	PHCMP2 Control Register 0 (GDU_PHCMP2CR0)	8	R/W	00h	25.6.9/446
29	PHCMP2 Control Register 1 (GDU_PHCMP2CR1)	8	R/W	00h	25.6.10/446
2A	PHCMP2 Filter Period Register (GDU_PHCMP2FPR)	8	R/W	00h	25.6.11/448
2B	PHCMP2 Status and Control Register (GDU_PHCMP2SCR)	8	R/W	00h	25.6.12/448
1870	Clamp Control Register (GDU_CLMPCTRL)	8	R/W	86h	25.6.13/449
1871	I/O Control Register (GDU_IOCTRL)	8	R/W	01h	25.6.14/450
1872	Virtual Network Phase Detection Control (GDU_PHASECTRL)	8	R/W	00h	25.6.15/451
1873	Current Sensor and Overcurrent Protection Control Register (GDU_CURCTRL)	8	R/W	00h	25.6.16/452
1874	LIMIT0 CMP Control Register 0 (GDU_LIMIT0CR0)	8	R/W	00h	25.6.17/452
1875	LIMIT0 CMP Control Register 1 (GDU_LIMIT0CR1)	8	R/W	00h	25.6.18/453
1876	LIMIT0 CMP Filter Period Register (GDU_LIMIT0FPR)	8	R/W	00h	25.6.19/454
1877	LIMIT0 CMP Status and Control Register (GDU_LIMIT0SCR)	8	R/W	00h	25.6.20/455
1878	LIMIT0 DAC Control Register (GDU_LIMIT0DACCR)	8	R/W	00h	25.6.21/456
1879	LIMIT1 CMP Control Register 0 (GDU_LIMIT1CR0)	8	R/W	00h	25.6.22/456
187A	LIMIT1 CMP Control Register 1 (GDU_LIMIT1CR1)	8	R/W	00h	25.6.23/457
187B	LIMIT1 CMP Filter Period Register (GDU_LIMIT1FPR)	8	R/W	00h	25.6.24/458
187C	LIMIT1 CMP Status and Control Register (GDU_LIMIT1SCR)	8	R/W	00h	25.6.25/459
187D	LIMIT1 DAC Control Register (GDU_LIMIT1DACCR)	8	R/W	00h	25.6.26/460
187E	PDCS and Clamp Status Register (GDU_STATREG)	8	R	00h	25.6.27/460
187F	LIMIT CMP BIAS Register (GDU_SIGBIAS)	8	R/W	00h	25.6.28/461

25.6.1 PHCMP0 Control Register 0 (GDU_PHCMP0CR0)

Address: 20h base + 0h offset = 20h

Bit	7	6	5	4	3	2	1	0
Read	0	FILTER_CNT			0			HYSTCTR
Write								
Reset	0	0	0	0	0	0	0	0

GDU_PHCMP0CR0 field descriptions

Field	Description
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6–4 FILTER_CNT	Filter Sample Count These bits represent the number of consecutive samples that must agree prior to the comparator ouput filter accepting a new output state. For information regarding filter programming and latency reference the Functional Description. 000 Filter is disabled. If SE = 1, then COUT is a logic zero (this is not a legal state, and is not recommended). If SE = 0, COUT = COUTA. 001 1 consecutive sample must agree (comparator output is simply sampled). 010 2 consecutive samples must agree. 011 3 consecutive samples must agree. 100 4 consecutive samples must agree. 101 5 consecutive samples must agree. 110 6 consecutive samples must agree. 111 7 consecutive samples must agree.
3–1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 HYSTCTR	Comparator hard block hysteresis control Defines the programmable hysteresis level. The hysteresis values associated with each level is device-specific. See the device's data sheet for the exact values. 0 Level 0 1 Level 1

25.6.2 PHCMP0 Control Register 1 (GDU_PHCMP0CR1)

Address: 20h base + 1h offset = 21h

Bit	7	6	5	4	3	2	1	0
Read	SE	WE	0	PMODE	INV	COS	OPE	EN
Write								
Reset	0	0	0	0	0	0	0	0

GDU_PHCMP0CR1 field descriptions

Field	Description
7 SE	Sample Enable At any given time, either SE or WE can be set. If a write to this register attempts to set both, then SE is set and WE is cleared. However, avoid writing ones to both bit locations because this "11" case is reserved and may change in future implementations. 0 Sampling mode not selected. 1 Sampling mode selected.
6 WE	Windowing Enable At any given time, either SE or WE can be set. If a write to this register attempts to set both, then SE is set and WE is cleared. However, avoid writing ones to both bit locations because this "11" case is reserved and may change in future implementations. 0 Windowing mode not selected. 1 Windowing mode selected.
5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
4 PMODE	Power Mode Select 0 Low Speed (LS) comparison mode selected. 1 High Speed (HS) comparison mode selected.
3 INV	GDU Comparator INVERT This bit allows you to select the polarity of the GDU analog comparator function. It is also driven to the COUT output (on both the device pin and as SCR[COUT]) when CR1[OPE]=0. 0 Does not invert the GDU comparator output. 1 Inverts the GDU comparator output.
2 COS	Comparator Output Select **NOTE:** This field is always disabled in this device, writing 1 to this field does not take effect. 0 Set CMPO to equal COUT (filtered comparator output). 1 Set CMPO to equal COUTA (unfiltered comparator output).
1 OPE	Comparator Output Pin Enable **NOTE:** This field is always disabled in this device, writing 1 to this field does not take effect. 0 The comparator output (CMPO) is not available on the associated CMPO output pin. 1 The comparator output (CMPO) is available on the associated CMPO output pin.
0 EN	Comparator Module Enable The EN bit enables the Analog Comparator Module. When the module is not enabled, it remains in the off state, and consumes no power. When you select the same input from analog mux to the positive and negative port, the comparator is disabled automatically. 0 Analog Comparator disabled. 1 Analog Comparator enabled.

25.6.3 PHCMP0 Filter Period Register (GDU_PHCMP0FPR)

Address: 20h base + 2h offset = 22h

Bit	7	6	5	4	3	2	1	0
Read Write	FILT_PER							
Reset	0	0	0	0	0	0	0	0

GDU_PHCMP0FPR field descriptions

Field	Description
FILT_PER	Filter Sample Period When CR1[SE] is equal to zero, this field specifies the sampling period, in bus clock cycles, of the comparator output filter. Setting FILT_PER to 0x0 disables the filter. Filter programming and latency details appear in the Functional Description. This field has no effect when CR1[SE] is equal to one. In that case, the external SAMPLE signal is used to determine the sampling period.

25.6.4 PHCMP0 Status and Control Register (GDU_PHCMP0SCR)

Address: 20h base + 3h offset = 23h

Bit	7	6	5	4	3	2	1	0
Read	0		0	IER	IEF	CFR	CFF	COUT
Write						w1c	w1c	
Reset	0	0	0	0	0	0	0	0

GDU_PHCMP0SCR field descriptions

Field	Description
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
4 IER	Comparator Interrupt Enable Rising The IER bit enables the CFR interrupt from the CMP. When this bit is set, an interrupt will be asserted when the CFR bit is set. 0 Interrupt disabled. 1 Interrupt enabled.
3 IEF	Comparator Interrupt Enable Falling The IEF bit enables the CFF interrupt from the CMP. When this bit is set, an interrupt will be asserted when the CFF bit is set.

Table continues on the next page...

GDU_PHCMP0SCR field descriptions (continued)

Field	Description
	0 Interrupt disabled. 1 Interrupt enabled.
2 CFR	Analog Comparator Flag Rising During normal operation, the CFR bit is set when a rising edge on COUT has been detected. The CFR bit is cleared by writing a logic one to the bit. 0 Rising edge on COUT has not been detected. 1 Rising edge on COUT has occurred.
1 CFF	Analog Comparator Flag Falling During normal operation, the CFF bit is set when a falling edge on COUT has been detected. The CFF bit is cleared by writing a logic one to the bit. 0 Falling edge on COUT has not been detected. 1 Falling edge on COUT has occurred.
0 COUT	Analog Comparator Output Reading the COUT bit will return the current value of the analog comparator output. The register bit is reset to zero and will read as CR1[INV] when the Analog Comparator module is disabled (CR1[EN] = 0). Writes to this bit are ignored.

25.6.5 PHCMP1 Control Register 0 (GDU_PHCMP1CR0)

Address: 20h base + 4h offset = 24h

Bit	7	6	5	4	3	2	1	0
Read	0	FILTER_CNT			0			HYSTCTR
Write								
Reset	0	0	0	0	0	0	0	0

GDU_PHCMP1CR0 field descriptions

Field	Description
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6–4 FILTER_CNT	Filter Sample Count These bits represent the number of consecutive samples that must agree prior to the comparator ouput filter accepting a new output state. For information regarding filter programming and latency reference the Functional Description. 000 Filter is disabled. If SE = 1, then COUT is a logic zero (this is not a legal state, and is not recommended). If SE = 0, COUT = COUTA. 001 1 consecutive sample must agree (comparator output is simply sampled). 010 2 consecutive samples must agree. 011 3 consecutive samples must agree. 100 4 consecutive samples must agree. 101 5 consecutive samples must agree.

Table continues on the next page...

GDU_PHCMP1CR0 field descriptions (continued)

Field	Description
	110 6 consecutive samples must agree. 111 7 consecutive samples must agree.
3–1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 HYSTCTR	Comparator hard block hysteresis control Defines the programmable hysteresis level. The hysteresis values associated with each level is device-specific. See the device's data sheet for the exact values. 0 Level 0 1 Level 1

25.6.6 PHCMP1 Control Register 1 (GDU_PHCMP1CR1)

Address: 20h base + 5h offset = 25h

Bit	7	6	5	4	3	2	1	0
Read	SE	WE	0	PMODE	INV	COS	OPE	EN
Write								
Reset	0	0	0	0	0	0	0	0

GDU_PHCMP1CR1 field descriptions

Field	Description
7 SE	Sample Enable At any given time, either SE or WE can be set. If a write to this register attempts to set both, then SE is set and WE is cleared. However, avoid writing ones to both bit locations because this "11" case is reserved and may change in future implementations. 0 Sampling mode not selected. 1 Sampling mode selected.
6 WE	Windowing Enable At any given time, either SE or WE can be set. If a write to this register attempts to set both, then SE is set and WE is cleared. However, avoid writing ones to both bit locations because this "11" case is reserved and may change in future implementations. 0 Windowing mode not selected. 1 Windowing mode selected.
5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
4 PMODE	Power Mode Select 0 Low Speed (LS) comparison mode selected. 1 High Speed (HS) comparison mode selected.
3 INV	GDU Comparator INVERT

Table continues on the next page...

GDU_PHCMP1CR1 field descriptions (continued)

Field	Description
	This bit allows you to select the polarity of the GDU analog comparator function. It is also driven to the COUT output (on both the device pin and as SCR[COUT]) when CR1[OPE]=0. 0 Does not invert the GDU comparator output. 1 Inverts the GDU comparator output.
2 COS	Comparator Output Select **NOTE:** This field is always disabled in this device, writing 1 to this field does not take effect. 0 Set CMPO to equal COUT (filtered comparator output). 1 Set CMPO to equal COUTA (unfiltered comparator output).
1 OPE	Comparator Output Pin Enable **NOTE:** This field is always disabled in this device, writing 1 to this field does not take effect. 0 The comparator output (CMPO) is not available on the associated CMPO output pin. 1 The comparator output (CMPO) is available on the associated CMPO output pin.
0 EN	Comparator Module Enable The EN bit enables the Analog Comparator Module. When the module is not enabled, it remains in the off state, and consumes no power. When you select the same input from analog mux to the positive and negative port, the comparator is disabled automatically. 0 Analog Comparator disabled. 1 Analog Comparator enabled.

25.6.7 PHCMP1 Filter Period Register (GDU_PHCMP1FPR)

Address: 20h base + 6h offset = 26h

Bit	7	6	5	4	3	2	1	0
Read Write	FILT_PER							
Reset	0	0	0	0	0	0	0	0

GDU_PHCMP1FPR field descriptions

Field	Description
FILT_PER	Filter Sample Period When CR1[SE] is equal to zero, this field specifies the sampling period, in bus clock cycles, of the comparator output filter. Setting FILT_PER to 0x0 disables the filter. Filter programming and latency details appear in the Functional Description. This field has no effect when CR1[SE] is equal to one. In that case, the external SAMPLE signal is used to determine the sampling period.

25.6.8 PHCMP1 Status and Control Register (GDU_PHCMP1SCR)

Address: 20h base + 7h offset = 27h

Bit	7	6	5	4	3	2	1	0
Read	0		0	IER	IEF	CFR	CFF	COUT
Write						w1c	w1c	
Reset	0	0	0	0	0	0	0	0

GDU_PHCMP1SCR field descriptions

Field	Description
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
4 IER	Comparator Interrupt Enable Rising The IER bit enables the CFR interrupt from the CMP. When this bit is set, an interrupt will be asserted when the CFR bit is set. 0 Interrupt disabled. 1 Interrupt enabled.
3 IEF	Comparator Interrupt Enable Falling The IEF bit enables the CFF interrupt from the CMP. When this bit is set, an interrupt will be asserted when the CFF bit is set. 0 Interrupt disabled. 1 Interrupt enabled.
2 CFR	Analog Comparator Flag Rising During normal operation, the CFR bit is set when a rising edge on COUT has been detected. The CFR bit is cleared by writing a logic one to the bit. 0 Rising edge on COUT has not been detected. 1 Rising edge on COUT has occurred.
1 CFF	Analog Comparator Flag Falling During normal operation, the CFF bit is set when a falling edge on COUT has been detected. The CFF bit is cleared by writing a logic one to the bit. 0 Falling edge on COUT has not been detected. 1 Falling edge on COUT has occurred.
0 COUT	Analog Comparator Output Reading the COUT bit will return the current value of the analog comparator output. The register bit is reset to zero and will read as CR1[INV] when the Analog Comparator module is disabled (CR1[EN] = 0). Writes to this bit are ignored.

25.6.9 PHCMP2 Control Register 0 (GDU_PHCMP2CR0)

Address: 20h base + 8h offset = 28h

Bit	7	6	5	4	3	2	1	0
Read	0	FILTER_CNT			0			HYSTCTR
Write								
Reset	0	0	0	0	0	0	0	0

GDU_PHCMP2CR0 field descriptions

Field	Description
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6–4 FILTER_CNT	Filter Sample Count These bits represent the number of consecutive samples that must agree prior to the comparator ouput filter accepting a new output state. For information regarding filter programming and latency reference the Functional Description. 000 Filter is disabled. If SE = 1, then COUT is a logic zero (this is not a legal state, and is not recommended). If SE = 0, COUT = COUTA. 001 1 consecutive sample must agree (comparator output is simply sampled). 010 2 consecutive samples must agree. 011 3 consecutive samples must agree. 100 4 consecutive samples must agree. 101 5 consecutive samples must agree. 110 6 consecutive samples must agree. 111 7 consecutive samples must agree.
3–1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 HYSTCTR	Comparator hard block hysteresis control Defines the programmable hysteresis level. The hysteresis values associated with each level is device-specific. See the device's data sheet for the exact values. 0 Level 0 1 Level 1

25.6.10 PHCMP2 Control Register 1 (GDU_PHCMP2CR1)

Address: 20h base + 9h offset = 29h

Bit	7	6	5	4	3	2	1	0
Read	SE	WE	0	PMODE	INV	COS	OPE	EN
Write								
Reset	0	0	0	0	0	0	0	0

GDU_PHCMP2CR1 field descriptions

Field	Description
7 SE	Sample Enable At any given time, either SE or WE can be set. If a write to this register attempts to set both, then SE is set and WE is cleared. However, avoid writing ones to both bit locations because this "11" case is reserved and may change in future implementations. 0 Sampling mode not selected. 1 Sampling mode selected.
6 WE	Windowing Enable At any given time, either SE or WE can be set. If a write to this register attempts to set both, then SE is set and WE is cleared. However, avoid writing ones to both bit locations because this "11" case is reserved and may change in future implementations. 0 Windowing mode not selected. 1 Windowing mode selected.
5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
4 PMODE	Power Mode Select 0 Low Speed (LS) comparison mode selected. 1 High Speed (HS) comparison mode selected.
3 INV	GDU Comparator INVERT This bit allows you to select the polarity of the GDU analog comparator function. It is also driven to the COUT output (on both the device pin and as SCR[COUT]) when CR1[OPE]=0. 0 Does not invert the GDU comparator output. 1 Inverts the GDU comparator output.
2 COS	Comparator Output Select **NOTE:** This field is always disabled in this device, writing 1 to this field does not take effect. 0 Set CMPO to equal COUT (filtered comparator output). 1 Set CMPO to equal COUTA (unfiltered comparator output).
1 OPE	Comparator Output Pin Enable **NOTE:** This field is always disabled in this device, writing 1 to this field does not take effect. 0 The comparator output (CMPO) is not available on the associated CMPO output pin. 1 The comparator output (CMPO) is available on the associated CMPO output pin.
0 EN	Comparator Module Enable The EN bit enables the Analog Comparator Module. When the module is not enabled, it remains in the off state, and consumes no power. When you select the same input from analog mux to the positive and negative port, the comparator is disabled automatically. 0 Analog Comparator disabled. 1 Analog Comparator enabled.

25.6.11 PHCMP2 Filter Period Register (GDU_PHCMP2FPR)

Address: 20h base + Ah offset = 2Ah

Bit	7	6	5	4	3	2	1	0
Read Write	FILT_PER							
Reset	0	0	0	0	0	0	0	0

GDU_PHCMP2FPR field descriptions

Field	Description
FILT_PER	Filter Sample Period When CR1[SE] is equal to zero, this field specifies the sampling period, in bus clock cycles, of the comparator output filter. Setting FILT_PER to 0x0 disables the filter. Filter programming and latency details appear in the Functional Description. This field has no effect when CR1[SE] is equal to one. In that case, the external SAMPLE signal is used to determine the sampling period.

25.6.12 PHCMP2 Status and Control Register (GDU_PHCMP2SCR)

Address: 20h base + Bh offset = 2Bh

Bit	7	6	5	4	3	2	1	0
Read	0		0	IER	IEF	CFR	CFF	COUT
Write						w1c	w1c	
Reset	0	0	0	0	0	0	0	0

GDU_PHCMP2SCR field descriptions

Field	Description
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
4 IER	Comparator Interrupt Enable Rising The IER bit enables the CFR interrupt from the CMP. When this bit is set, an interrupt will be asserted when the CFR bit is set. 0 Interrupt disabled. 1 Interrupt enabled.
3 IEF	Comparator Interrupt Enable Falling The IEF bit enables the CFF interrupt from the CMP. When this bit is set, an interrupt will be asserted when the CFF bit is set.

Table continues on the next page...

GDU_PHCMP2SCR field descriptions (continued)

Field	Description
	0 Interrupt disabled. 1 Interrupt enabled.
2 CFR	Analog Comparator Flag Rising During normal operation, the CFR bit is set when a rising edge on COUT has been detected. The CFR bit is cleared by writing a logic one to the bit. 0 Rising edge on COUT has not been detected. 1 Rising edge on COUT has occurred.
1 CFF	Analog Comparator Flag Falling During normal operation, the CFF bit is set when a falling edge on COUT has been detected. The CFF bit is cleared by writing a logic one to the bit. 0 Falling edge on COUT has not been detected. 1 Falling edge on COUT has occurred.
0 COUT	Analog Comparator Output Reading the COUT bit will return the current value of the analog comparator output. The register bit is reset to zero and will read as CR1[INV] when the Analog Comparator module is disabled (CR1[EN] = 0). Writes to this bit are ignored.

25.6.13 Clamp Control Register (GDU_CLMPCTRL)

Address: 20h base + 1850h offset = 1870h

Bit	7	6	5	4	3	2	1	0
Read	OVPEN	OVPIE	0		TUNE			CLAMPEN
Write								
Reset	1	0	0	0	0	1	1	0

GDU_CLMPCTRL field descriptions

Field	Description
7 OVPEN	Overvoltage Protection Enable 0 Disables the overvoltage protection (OVP). 1 Enables the OVP.
6 OVPIE	Overvoltage Protection Interrupt Enable 0 Disables the OVP interrupt. 1 Enables the OVP interrupt.
5–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3–1 TUNE	Tunes the clamped output voltage when the clamp works at regulation mode. The tuning range is 5 V ±20% with step of 5%.
0 CLAMPEN	Clamp Enable

Table continues on the next page...

GDU_CLMPCTRL field descriptions (continued)

Field	Description
	0 Disables the clamp control. 1 Enables the clamp control.

25.6.14 I/O Control Register (GDU_IOCTRL)

NOTE

Bit 7 to bit 4 are control bits for high-side (HS) pre-driver output, while bit 3 to bit 0 are control bits for low-side (LS) pre-driver output.

Address: 20h base + 1851h offset = 1871h

Bit	7	6	5	4	3	2	1	0
Read	HSDE	0	HSDS		0	LSDS		PDE
Write								
Reset	0	0	0	0	0	0	0	1

GDU_IOCTRL field descriptions

Field	Description
7 HSDE	High-side (HS) pre-driver Output Buffer Enable 0 Disables the output buffer. 1 Enables the output buffer.
6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5–4 HSDS	High-Side pre-driver Drive Strength The value setting of HSDS1 (higher bit) and HSDS0 (lower bit) is as follows: 00 Lowest drive strength (default), I_{peak}=25 mA 01 Middle low drive strength, I_{peak}=50 mA 10 Middle high drive strength, I_{peak}=75 mA 11 Highest drive strength, I_{peak}=100 mA
3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2–1 LSDS	Low-Side (LS) pre-driver Output Buffer Enable and Drive Strength The value setting of LSDS1 (higher bit) and LSDS0 (lower bit) is as follows: 00 Disable the output buffer 01 Lowest drive strength, I_{peak}=25 mA 10 Middle drive strength, I_{peak}=75 mA 11 Highest drive strength, I_{peak}=100 mA

Table continues on the next page...

GDU_IOCTRL field descriptions (continued)

Field	Description
0 PDE	High-Side pre-driver Drive Output Pulldown Enable 0 Disables the pulldown. 1 Enables the pulldown.

25.6.15 Virtual Network Phase Detection Control (GDU_PHASECTRL)

NOTE

V_{INM} is the comparator of minus input from DAC, ADC or virtual Neutral.

Address: 20h base + 1852h offset = 1872h

Bit	7	6	5	4	3	2	1	0
Read	0	VNEN	INTSEL	EXTSEL	0	PHSEL2	PHSEL1	PHSEL0
Write								
Reset	0	0	0	0	0	0	0	0

GDU_PHASECTRL field descriptions

Field	Description
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6 VNEN	Virtural Network Enable 0 Virtual resistor network disabled. 1 Virtual resistor network enabled.
5 INTSEL	Virtual Neutral Internal Connection 0 INTSEL MUX disabled . 1 INTSET MUX enabled, virtual neutral is connected to the internal 6-bit DAC in the CMP module.
4 EXTSEL	Virtual Neutral External Connection 0 EXTSEL MUX disabled. 1 EXTSET MUX enabled, virtual neutral is connected to the external pin.
3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 PHSEL2	Virtual Network Phase 2 Selection 0 Phase 2 is off. 1 Phase 2 is on.
1 PHSEL1	Virtual Network Phase 1 Selection

Table continues on the next page...

GDU_PHASECTRL field descriptions (continued)

Field	Description
	0 Phase 1 is off. 1 Phase 1 is on.
0 PHSEL0	Virtual Network Phase 0 Selection 0 Phase 0 is off. 1 Phase 1 is on.

25.6.16 Current Sensor and Overcurrent Protection Control Register (GDU_CURCTRL)

Address: 20h base + 1853h offset = 1873h

Bit	7	6	5	4	3	2	1	0
Read	0					AMP0EN	0	AMP1EN
Write								
Reset	0	0	0	0	0	0	0	0

GDU_CURCTRL field descriptions

Field	Description
7–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 AMP0EN	OPAMP0 Enable 0 Disables GDU AMP0. 1 Enables GDU AMP0.
1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 AMP1EN	OPAMP1 Enable 0 Disables GDU AMP1. 1 Enables GDU AMP1.

25.6.17 LIMIT0 CMP Control Register 0 (GDU_LIMIT0CR0)

Address: 20h base + 1854h offset = 1874h

Bit	7	6	5	4	3	2	1	0
Read	0	FLTCNT			0			HYST
Write								
Reset	0	0	0	0	0	0	0	0

GDU_LIMIT0CR0 field descriptions

Field	Description
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6–4 FLTCNT	Filter Sample Count This field represents the number of consecutive samples that must agree, prior to the comparator output filter accepting a new output state. For information regarding filter programming and latency, see the Functional Description. 000 Filter is disabled. If SE = 1, then COUT is a logic 0 (this is not a legal state, and is not recommended). If SE = 0, COUT = COUTA. 001 1 consecutive sample must agree (comparator output is simply sampled). 010 2 consecutive samples must agree. 011 3 consecutive samples must agree. 100 4 consecutive samples must agree. 101 5 consecutive samples must agree. 110 6 consecutive samples must agree. 111 7 consecutive samples must agree.
3–1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 HYST	Comparator hard block Hysteresis control Defines the programmable hysteresis level. The hysteresis value associated with each level is device-specific. See the device's data sheet for the exact values. 0 Level 0 1 Level 1

25.6.18 LIMIT0 CMP Control Register 1 (GDU_LIMIT0CR1)

Address: 20h base + 1855h offset = 1875h

Bit	7	6	5	4	3	2	1	0
Read	SE	WE	0	PMODE	INV	COS	OPE	EN
Write								
Reset	0	0	0	0	0	0	0	0

GDU_LIMIT0CR1 field descriptions

Field	Description
7 SE	Sampling Enable At any given time, either SE or WE can be set. If a write to this register attempts to set both, then SE is set and WE is cleared. However, avoid writing 1 to both bits because this "11" case is reserved and may change in future implementations. **NOTE:** Set this bit to zero, because the sample input is connected to the logic 1. 0 Sampling mode is not selected. 1 Sampling mode is selected.

Table continues on the next page...

GDU_LIMIT0CR1 field descriptions (continued)

Field	Description
6 WE	Windowing Enable At any given time, either SE or WE can be set. If a write to this register attempts to set both, then SE is set and WE is cleared. However, avoid writing 1 to both bits because this "11" case is reserved and may change in future implementations. **NOTE:** Set this bit to zero, because the window input is connected to the logic 1.
5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
4 PMODE	Power Mode select 0 Low-speed (LS) comparison mode is selected. 1 High-speed (HS) comparison mode is selected.
3 INV	Comparator Invert This bit allows to select the polarity of the analog comparator function. It is also driven to the COUT output (on both the device pin and as GDU_USCR[COUT]) when GDU_UCR1[OPE]=0. 0 Does not invert the comparator output. 1 Inverts the comparator output.
2 COS	Comparator Output Select **NOTE:** This field is always disabled in this device, writing 1 to this field does not take effect. 0 Sets CMPO to equal to COUT (filtered comparator output). 1 Sets CMPO to equal to COUTA (unfiltered comparator output).
1 OPE	Comparator Output Pin Enable **NOTE:** This field is always disabled in this device, writing 1 to this field does not take effect. 0 The comparator output (CMPO) is not available on the associated CMPO output pin. The pin is available for use by other on-chip functions. 1 The comparator output (CMPO) is available on the associated CMPO output pin. The comparator output (CMPO) is driven out on the associated CMPO output pin.
0 EN	Comparator Enable 0 Analog Comparator is disabled. 1 Analog Comparator is enabled.

25.6.19 LIMIT0 CMP Filter Period Register (GDU_LIMIT0FPR)

Address: 20h base + 1856h offset = 1876h

Bit	7	6	5	4	3	2	1	0
Read Write	FLTPER							
Reset	0	0	0	0	0	0	0	0

GDU_LIMIT0FPR field descriptions

Field	Description
FLTPER	Filter sampling Period When UCR1[SE] is equal to 0, this field specifies the sampling period, in bus clock cycles, of the comparator output filter. Setting FLTPER to 0 disables the filter. For the filter programming and latency details, see the Functional Description. This field has no effect when UCR1[SE] is equal to 1. In that case, the external SAMPLE signal is used to determine the sampling period.

25.6.20 LIMIT0 CMP Status and Control Register (GDU_LIMIT0SCR)

Address: 20h base + 1857h offset = 1877h

Bit	7	6	5	4	3	2	1	0
Read	0			IER	IEF	CFR	CFF	COUT
Write						w1c	w1c	
Reset	0	0	0	0	0	0	0	0

GDU_LIMIT0SCR field descriptions

Field	Description
7–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
4 IER	Comparator Interrupt Enable Rising The IER bit enables the CFR interrupt from the CMP. If this bit is set, an interrupt is asserted when the CFR bit is set. 0 Interrupt is disabled. 1 Interrupt is enabled.
3 IEF	Comparator Interrupt Enable Falling The IEF bit enables the CFF interrupt from the CMP. If this bit is set, an interrupt is asserted when the CFF bit is set. 0 Interrupt is disabled. 1 Interrupt is enabled.
2 CFR	Analog Comparator Flag Rising During normal operation, the CFR bit is set when a rising edge on COUT is detected. This bit is cleared by writing a logic 1 to it. 0 Rising edge on COUT is not detected. 1 Rising edge on COUT occurs.
1 CFF	Analog Comparator Flag Falling During normal operation, the CFF bit is set when a falling edge on COUT is detected. This bit is cleared by writing a logic 1 to it.

Table continues on the next page...

GDU_LIMIT0SCR field descriptions (continued)

Field	Description
	0 Falling edge on COUT is not detected. 1 Falling edge on COUT occurs.
0 COUT	Analog Comparator Output Reading the COUT bit returns the current value of the analog comparator output. This bit is reset to 0 and reads as UCR1[INV] when the Analog Comparator module is disabled (UCR1[EN] = 0). Writing to this bit has no effect.

25.6.21 LIMIT0 DAC Control Register (GDU_LIMIT0DACCR)

Address: 20h base + 1858h offset = 1878h

Bit	7	6	5	4	3	2	1	0
Read	0				VOSEL			
Write								
Reset	0	0	0	0	0	0	0	0

GDU_LIMIT0DACCR field descriptions

Field	Description
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
VOSEL	DAC Output Voltage Select This field selects an output voltage from one of the 64 distinct levels. DACO = (V_{DDX}/64) × (VOSEL[5:0] + 1), so the DACO range is from V_{DDX}/64 to V_{DDX}.

25.6.22 LIMIT1 CMP Control Register 0 (GDU_LIMIT1CR0)

Address: 20h base + 1859h offset = 1879h

Bit	7	6	5	4	3	2	1	0
Read	0	FLTCNT			0			HYST
Write								
Reset	0	0	0	0	0	0	0	0

GDU_LIMIT1CR0 field descriptions

Field	Description
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6–4 FLTCNT	Filter Sample Count

Table continues on the next page...

GDU_LIMIT1CR0 field descriptions (continued)

Field	Description
	This field represents the number of consecutive samples that must agree, prior to the comparator output filter accepting a new output state. For information regarding filter programming and latency, see the Functional Description. 000 Filter is disabled. If SE = 1, then COUT is a logic 0 (this is not a legal state, and is not recommended). If SE = 0, COUT = COUTA. 001 1 consecutive sample must agree (comparator output is simply sampled). 010 2 consecutive samples must agree. 011 3 consecutive samples must agree. 100 4 consecutive samples must agree. 101 5 consecutive samples must agree. 110 6 consecutive samples must agree. 111 7 consecutive samples must agree.
3–1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 HYST	Comparator hard block Hysteresis control Defines the programmable hysteresis level. The hysteresis value associated with each level is device-specific. See the device's data sheet for the exact values. 0 Level 0 1 Level 1

25.6.23 LIMIT1 CMP Control Register 1 (GDU_LIMIT1CR1)

Address: 20h base + 185Ah offset = 187Ah

Bit	7	6	5	4	3	2	1	0
Read	SE	WE	0	PMODE	INV	COS	OPE	EN
Write								
Reset	0	0	0	0	0	0	0	0

GDU_LIMIT1CR1 field descriptions

Field	Description
7 SE	Sampling Enable At any given time, either SE or WE can be set. If a write to this register attempts to set both, then SE is set and WE is cleared. However, avoid writing 1 to both bits because this "11" case is reserved and may change in future implementations. **NOTE:** Set this bit to zero, because the sample input is connected to the logic 1. 0 Sampling mode is not selected. 1 Sampling mode is selected.
6 WE	Windowing Enable

Table continues on the next page...

GDU_LIMIT1CR1 field descriptions (continued)

Field	Description
	At any given time, either SE or WE can be set. If a write to this register attempts to set both, then SE is set and WE is cleared. However, avoid writing 1 to both bits because this "11" case is reserved and may change in future implementations. **NOTE:** Set this bit to zero, because the window input is connected to the logic 1.
5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
4 PMODE	Power Mode select 0 Low-speed (LS) comparison mode is selected. 1 High-speed (HS) comparison mode is selected.
3 INV	Comparator Invert This bit allows to select the polarity of the analog comparator function. It is also driven to the COUT output (on both the device pin and as GDU_VSCR[COUT]) when GDU_VCR1[OPE]=0. 0 Does not invert the comparator output. 1 Inverts the comparator output.
2 COS	Comparator Output Select **NOTE:** This field is always disabled in this device, writing 1 to this field does not take effect. 0 Sets CMPO to equal to COUT (filtered comparator output). 1 Sets CMPO to equal to COUTA (unfiltered comparator output).
1 OPE	Comparator Output Pin Enable **NOTE:** This field is always disabled in this device, writing 1 to this field does not take effect. 0 The comparator output (CMPO) is not available on the associated CMPO output pin. The pin is available for use by other on-chip functions. 1 The comparator output (CMPO) is available on the associated CMPO output pin. The comparator output (CMPO) is driven out on the associated CMPO output pin.
0 EN	Comparator Enable 0 Analog Comparator is disabled. 1 Analog Comparator is enabled.

25.6.24 LIMIT1 CMP Filter Period Register (GDU_LIMIT1FPR)

Address: 20h base + 185Bh offset = 187Bh

Bit	7	6	5	4	3	2	1	0
Read Write	FLTPER							
Reset	0	0	0	0	0	0	0	0

GDU_LIMIT1FPR field descriptions

Field	Description
FLTPER	Filter sampling Period When VCR1[SE] is equal to 0, this field specifies the sampling period, in bus clock cycles, of the comparator output filter. Setting FLTPER to 0 disables the filter. For the filter programming and latency details, see the Functional Description. This field has no effect when VCR1[SE] is equal to 1. In that case, the external SAMPLE signal is used to determine the sampling period.

25.6.25 LIMIT1 CMP Status and Control Register (GDU_LIMIT1SCR)

Address: 20h base + 185Ch offset = 187Ch

Bit	7	6	5	4	3	2	1	0
Read	0			IER	IEF	CFR	CFF	COUT
Write						w1c	w1c	
Reset	0	0	0	0	0	0	0	0

GDU_LIMIT1SCR field descriptions

Field	Description
7–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
4 IER	Comparator Interrupt Enable Rising The IER bit enables the CFR interrupt from the CMP. If this bit is set, an interrupt is asserted when the CFR bit is set. 0 Interrupt is disabled. 1 Interrupt is enabled.
3 IEF	Comparator Interrupt Enable Falling The IEF bit enables the CFF interrupt from the CMP. If this bit is set, an interrupt is asserted when the CFF bit is set. 0 Interrupt is disabled. 1 Interrupt is enabled.
2 CFR	Analog Comparator Flag Rising During normal operation, the CFR bit is set when a rising edge on COUT is detected. This bit is cleared by writing a logic 1 to it. 0 Rising edge on COUT is not detected. 1 Rising edge on COUT occurs.
1 CFF	Analog Comparator Flag Falling During normal operation, the CFF bit is set when a falling edge on COUT is detected. This bit is cleared by writing a logic 1 to it.

Table continues on the next page...

GDU_LIMIT1SCR field descriptions (continued)

Field	Description
	0 Falling edge on COUT is not detected. 1 Falling edge on COUT occurs.
0 COUT	Analog Comparator Output Reading the COUT bit returns the current value of the analog comparator output. This bit is reset to 0 and reads as VCR1[INV] when the Analog Comparator module is disabled (VCR1[EN] = 0). Writing to this bit has no effect.

25.6.26 LIMIT1 DAC Control Register (GDU_LIMIT1DACCR)

Address: 20h base + 185Dh offset = 187Dh

Bit	7	6	5	4	3	2	1	0
Read	0		VOSEL					
Write								
Reset	0	0	0	0	0	0	0	0

GDU_LIMIT1DACCR field descriptions

Field	Description
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
VOSEL	DAC Output Voltage Select This field selects an output voltage from one of the 64 distinct levels. DACO = (V_{DDX}/64) × (VOSEL[5:0] + 1), so the DACO range is from V_{DDX}/64 to V_{DDX}.

25.6.27 PDCS and Clamp Status Register (GDU_STATREG)

Address: 20h base + 185Eh offset = 187Eh

Bit	7	6	5	4	3	2	1	0
Read	OVP22V	OVP24V	0			PHASE		
Write	w1c	w1c						
Reset	0	0	0	0	0	0	0	0

GDU_STATREG field descriptions

Field	Description
7 OVP22V	Overvoltage Protection 22 V This bit is set when a voltage over 22 V is detected. The bit is cleared by writing a logic 1 to it.

Table continues on the next page...

GDU_STATREG field descriptions (continued)

Field	Description
	0 A voltage over 22 V is not detected. 1 A voltage over 22 V is detected.
6 OVP24V	Overvoltage Protection 24 V This bit is set when a voltage over 24 V is detected. The bit is cleared by writing a logic 1 to it. 0 A voltage over 24 V is not detected. 1 A voltage over 24 V is detected.
5–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
PHASE	GDU Phase status 0 Phase detector comparator is in disable mode or in the enable mode and INP < INM - (absolute value of input offset) 1 Phase detector comparator is in enable mode and INP > INM + (absolute value of input offset)

25.6.28 LIMIT CMP BIAS Register (GDU_SIGBIAS)

NOTE

In Scan mode, safe protection should be applied.

Address: 20h base + 185Fh offset = 187Fh

Bit	7	6	5	4	3	2	1	0
Read	BIASSEL	0						
Write								
Reset	0	0	0	0	0	0	0	0

GDU_SIGBIAS field descriptions

Field	Description
7 BIASSEL	Current sensor Bias Voltage Selection bit **NOTE:** PMC V_{REFH} is disabled by default. Customer must connect external V_{REFH} or enable internal V_{REFH} regulator. 0 Bias voltage selected from V_{REFH}. 1 Bias voltage selected from V_{DDX}.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

25.7 Functional description

The following content describes functional details of the module.

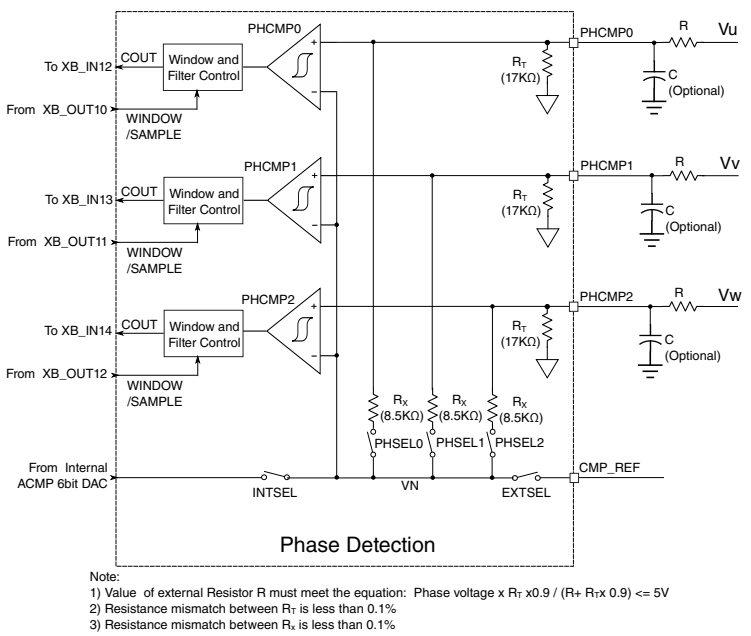
1. The generated 5 V CLAMP is between V_{DD} and V_{O_CLAMP}.
2. Drive 1/8 V_{DD} to ADC channel AD7.
3. CMP1 and CMP2 are used to generate 22 V over-voltage and 24 V over-voltage respectively.

25.7.1 Phase detection function descriptions

This section describes the phase detection function.

25.7.1.1 Phase detection diagram

The following figure shows the block diagram for the phase detection.



Note:
1) Value of external Resistor R must meet the equation: Phase voltage x R_T x0.9 / (R+ R_Tx 0.9) <= 5V
2) Resistance mismatch between R_T is less than 0.1%
3) Resistance mismatch between R_x is less than 0.1%

Figure 25-2. Phase detection diagram

25.7.1.2 Phase detection descriptions

Phase detection circuit is used to identify a zero-crossing event at a back-EMF (electro-motive force) signal generated at a winding of permanent magnet motor when the winding is not energized. A zero-crossing event at a back-EMF signal occurs when the signal level of the back-EMF signal equals to the voltage at the motor's common neutral connection. The comparators are provided with a virtual neutral reference signal that is generated at the virtual resistor network circuit and tracks the signal at the motor

winding's common neutral connection. The virtual neutral reference signal is generated by selectively communicating signals received from each motor winding to a virtual neutral circuit node.

Phase detection comparator is used to compare the virtual neutral and phase voltage input. Its output is connected to a window/filter circuit which provides window comparing and digital filter function.

25.7.2 OpAMP function descriptions

This section describes on-chip analog amplifies and over-limit protection comparators.

25.7.2.1 OpAMP diagram

The following figure shows the block diagram for the OpAMP.

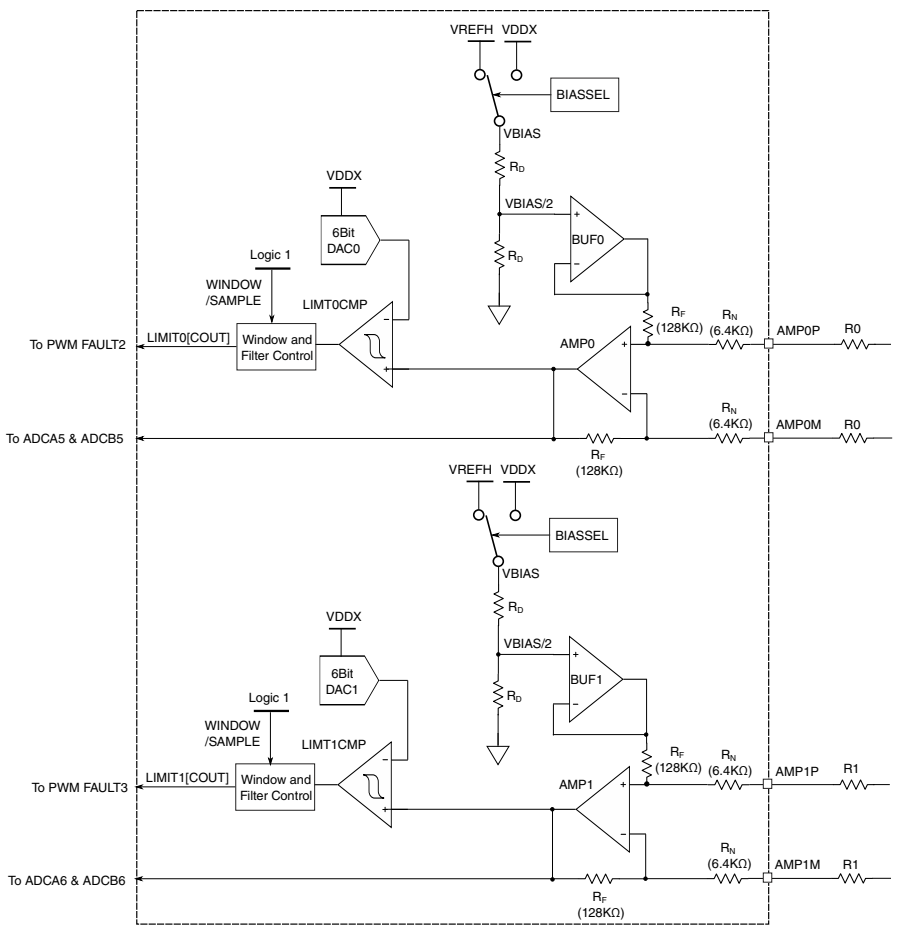


Figure 25-3. OpAMP diagram

25.7.2.2 OpAMP descriptions

The on-chip amplifier is usually connected as a differential amplifier. Its gain is internally set to 20. But the gain can be reduced by adding external resistors on its plus and minus input, as shown in Figure 25-3. it normally is used as current sense amplifier to sense the current flowing through the external resistor shunt as a voltage across the resistor. In order to measure both positive and negative currents, an internal bias reference must be used. This reference is divided either VREFH or VDDX to a half and added it to positive input of amplifier. The output of amplifier is connected to the ADC inputs and to the plus input of limitation comparator which can be used as over current protection. The minus input of limitation comparator is driven by a programmable 6-bit DAC. The output of the comparator is connected to a digital filter circuit.

25.7.3 Predrive function descriptions

This section describes the MOSFET predriver.

25.7.3.1 Predrive diagram

The following figure shows the predrive diagram

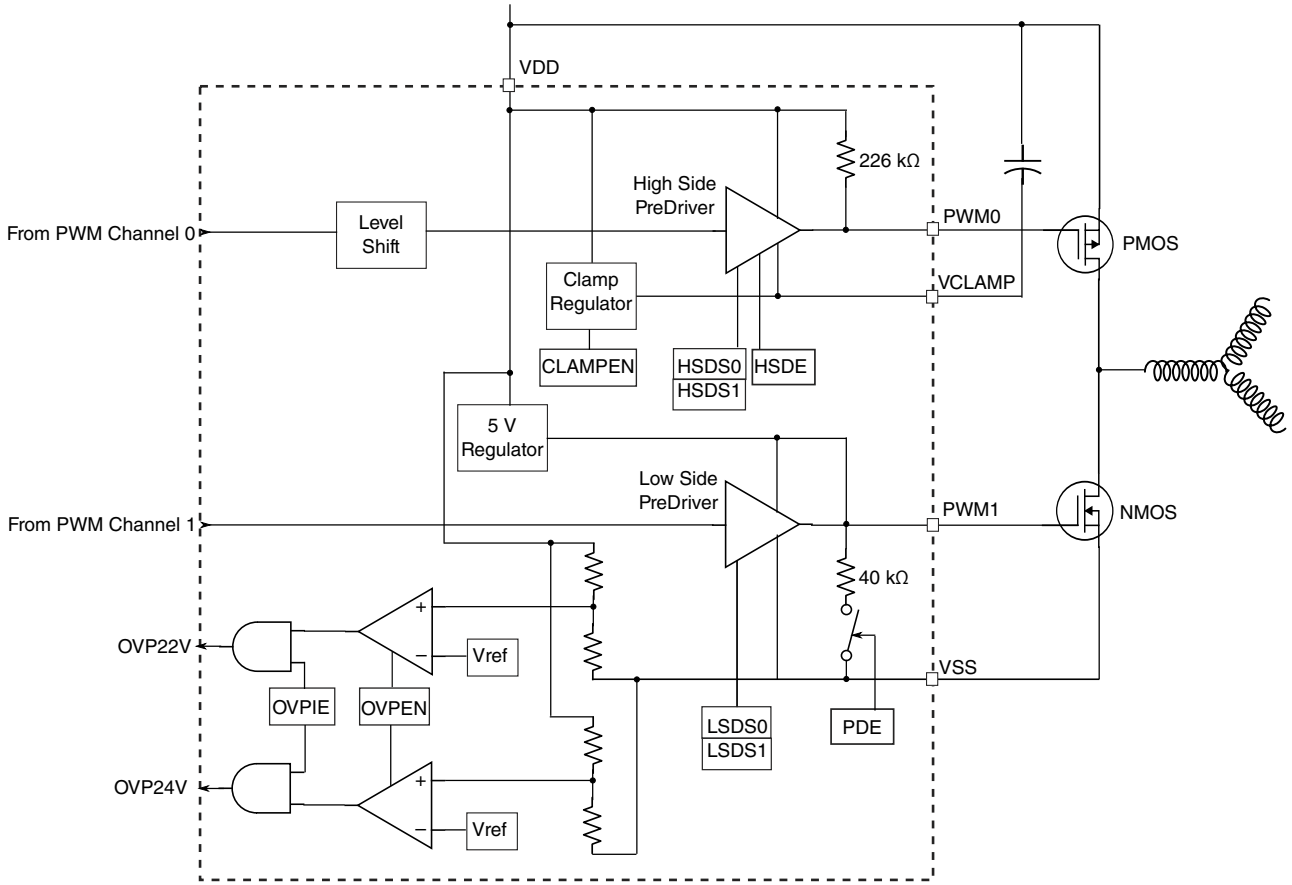


Figure 25-4. Predrive diagram

25.7.3.2 Predrive descriptions

25.7.4 GCMP functional description

The GDU Comparator can be used to compare two analog input voltages applied to INP and INM. The analog comparator output (CMPO) is high when the non-inverting input is greater than the inverting input, and is low when the non-inverting input is less than the inverting input. This signal can be selectively inverted by setting CR1[INV] = 1.

The SCR[IER], SCR[IEF] bits are used to select the condition which will cause the comparator module to assert an interrupt to the processor. SCR[CFF] is set on a falling edge and SCR[CFR] is set on rising edge of the comparator output. The (optionally filtered) comparator output can be read directly through the SCR[COUT] bit.

25.7.4.1 GCMP diagram

The following figure shows the block diagram for the GDU comparator module.

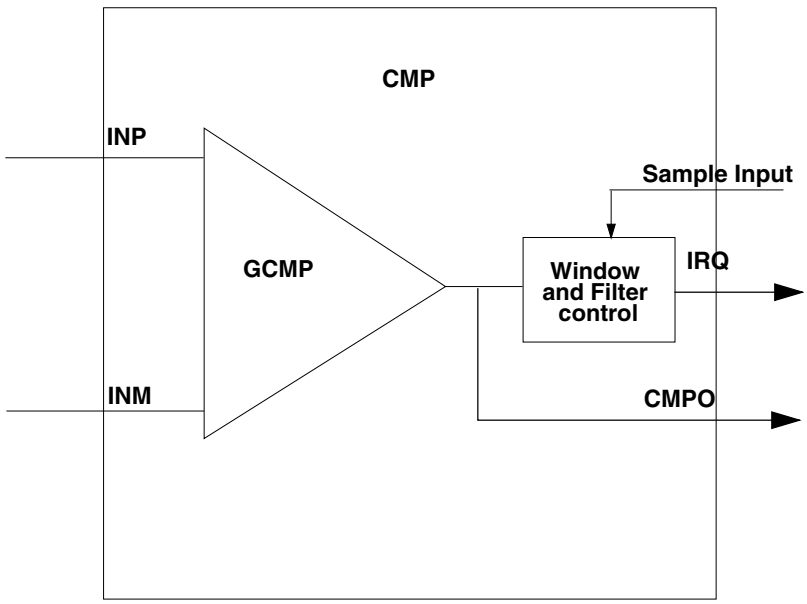


Figure 25-5. GCMP block diagram

25.7.4.2 GCMP block diagram

The following figure shows the block diagram for the GDU comparator module.

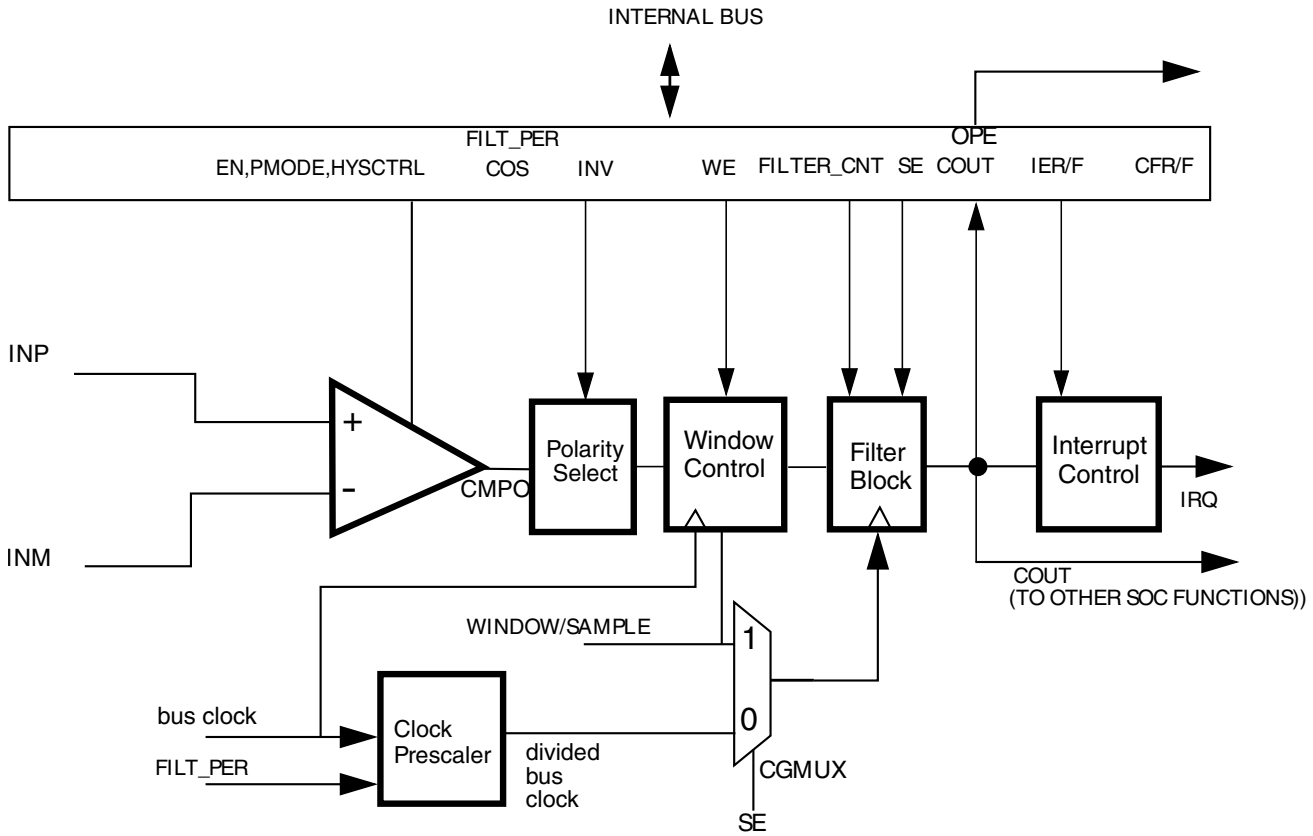


Figure 25-6. GDU comparator module block diagram

In the GCMP block diagram:

- The Window Control block is bypassed when CR1[WE] = 0
- If CR1[WE] = 1, the comparator output will be sampled on every bus clock when WINDOW=1 to generate COUTA. Sampling does NOT occur when WINDOW = 0.
- The Filter Block is bypassed when not in use.
- The Filter Block acts as a simple sampler if the filter is bypassed and CR0[FILTER_CNT] is set to 0x01.
- The Filter Block filters based on multiple samples when the filter is bypassed and CR0[FILTER_CNT] is set greater than 0x01.
  - If CR1[SE] = 1, the external SAMPLE input is used as sampling clock
  - IF CR1[SE] = 0, the divided bus clock is used as sampling clock

- If enabled, the Filter Block will incur up to 1 bus clock additional latency penalty on COUT due to the fact that COUT (which is crossing clock domain boundaries) must be resynchronized to the bus clock.
- CR1[WE] and CR1[SE] are mutually exclusive.

25.7.4.3 GCMP functional modes

There are three main sub-blocks to the comparator module: the comparator itself, the window function and the filter function. The filter, CR0[FILTER_CNT] can be clocked from an internally or external clock source. Additionally, the filter is programmable with respect to how many samples must agree before a change on the output is registered. In the simplest case, only 1 sample must agree. In this case, the filter acts as a simple sampler.

The external sample input is enabled using CR1[SE]. When set, the output of the comparator is sampled only on rising edges of the sample input.

The "windowing mode" is enabled by setting CR1[WE]. When set, the comparator output is sampled only when the WINDOW input signal is equal to one. This feature can be used to ignore the comparator output during time periods in which the input voltages are not valid. This is especially useful when implementing zero-crossing-detection for certain PWM applications.

The comparator filter and sampling features can be combined as shown in the following table. Individual modes are discussed below.

Table 25-2. GDU comparator sample/filter controls

Mode #	CR1[EN]	CR1[WE]	CR1[SE]	CR0[FILTER_CNT]	FPR[FILT_PER]	Operation
1	0	X	X	X	X	**Disabled** Refer to the Disabled Mode (# 1).
2A	1	0	0	0x00	X	**Continuous Mode**
2B	1	0	0	X	0x00	Refer to the Continuous mode (#s 2A & 2B).
3A	1	0	1	0x01	X	**Sampled, Non-Filtered mode**
3B	1	0	0	0x01	> 0x00	Refer to the Sampled, non-filtered mode (#s 3A & 3B).
4A	1	0	1	> 0x01	X	**Sampled, Filtered mode**
4B	1	0	0	> 0x01	> 0x00	Refer to the Sampled, filtered mode (#s 4A & 4B).
5A	1	1	0	0x00	X	**Windowed mode**

Table continues on the next page...

Table 25-2. GDU comparator sample/filter controls (continued)

Mode #	CR1[EN]	CR1[WE]	CR1[SE]	CR0[FILTER_CNT]	FPR[FILT_PER]	Operation
5B	1	1	0	X	0x00	Comparator output is sampled on every rising bus clock edge when SAMPLE=1 to generate COUTA Refer to the Windowed mode (#s 5A & 5B).
6	1	1	0	0x01	0x01 - 0xFF	**Windowed/Resampled mode** Comparator output is sampled on every rising bus clock edge when SAMPLE=1 to generate COUTA, which is then resampled on an interval determined by FILT_PER to generate COUT. Refer to the Windowed/Resampled mode (# 6).
7	1	1	0	> 0x01	0x01 - 0xFF	**Windowed/Filtered mode** Comparator output is sampled on every rising bus clock edge when SAMPLE=1 to generate COUTA, which is then resampled and filtered to generate COUT. Refer to the Windowed/filtered mode (#7).
All other combinations of CR1[EN], CR1[WE], CR1[SE], CR0[FILTER_CNT], and FPR[FILT_PER] are illegal.						

For cases where a GDU comparator is used to drive a fault input (for example, for a motor-control module such as FTM), it should generally be configured to operate in continuous mode, so that an external fault can immediately pass through the comparator to the target fault circuitry.

Note

Filtering and sampling settings should be changed only after setting CR1[SE]=0 and CR0[FILTER_CNT]=0x00. This has the effect of resetting the filter to a known state.

25.7.4.3.1 Disabled Mode (# 1)

In disabled mode, the GDU analog comparator is non-functional and consumes no power. The output of the analog comparator block (CMPO) is zero in this mode.

25.7.4.3.2 Continuous mode (#s 2A & 2B)

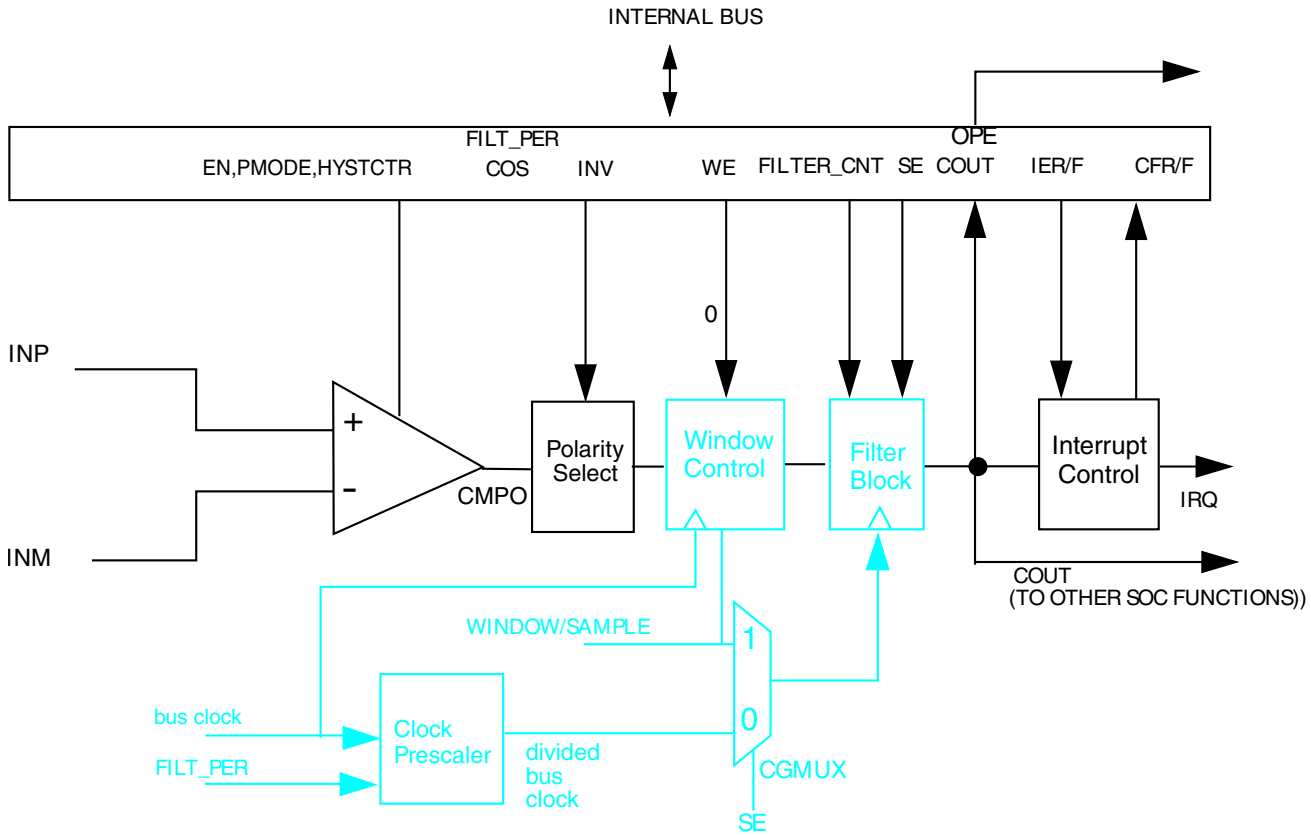


Figure 25-7. GDU comparator operation in continuous mode

NOTE

Refer to the chip configuration section for the source of sample/ window input.

The analog comparator block is powered and active. CMPO may be optionally inverted, but is not subject to external sampling or filtering. Both Window Control and Filter Blocks are completely bypassed. SCR[COUT] is updated continuously. The path from comparator inputs pins to output pin is operating in combinational (unclocked) mode. COUT and COUTA are identical.

For control configurations which result in disabling the Filter Block, refer to Filter Block Bypass Logic diagram.

25.7.4.3.3 Sampled, non-filtered mode (#s 3A & 3B)

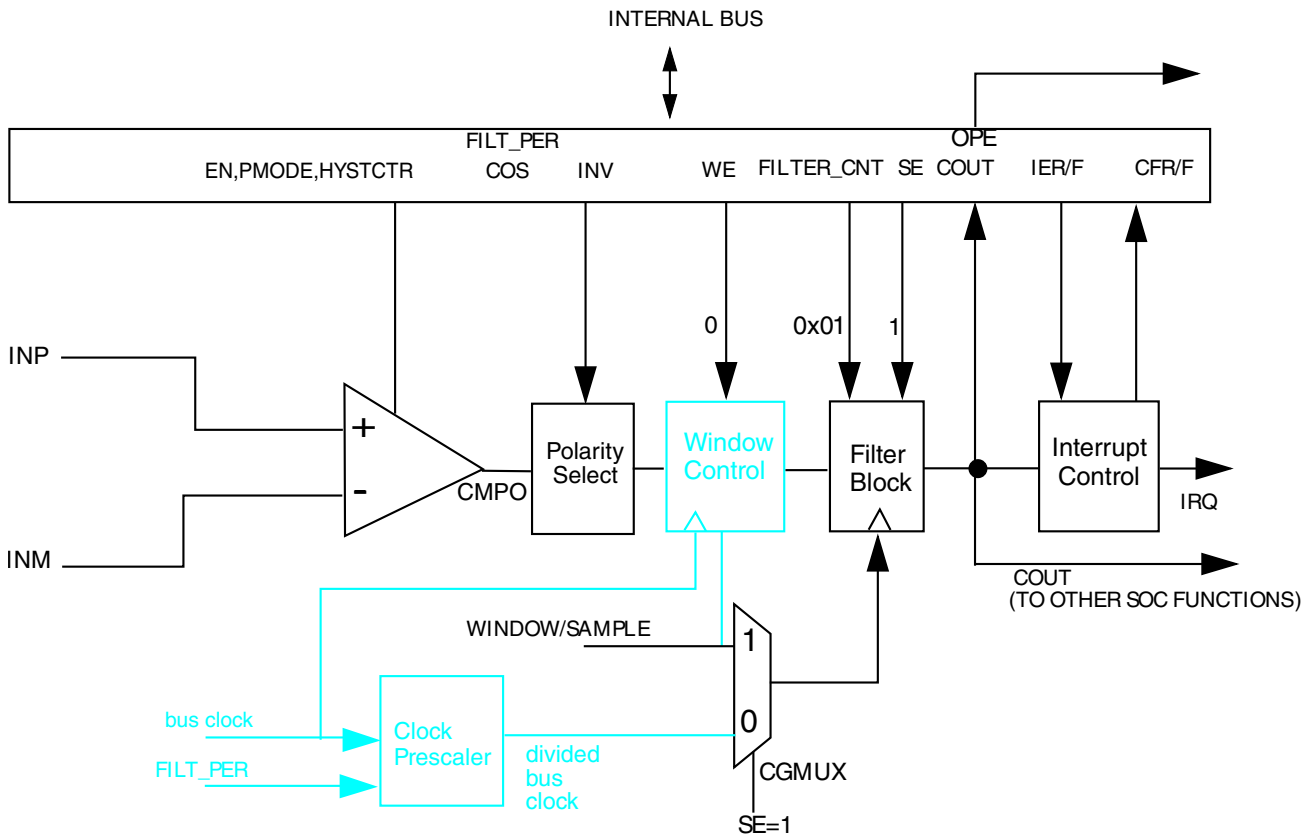


Figure 25-8. Sampled, non-filtered (# 3A): sampling point externally driven

In Sampled, Non-Filtered mode, the analog comparator block is powered and active. The path from analog inputs to COUTA is combinational (unclocked). Windowing Control is completely bypassed. COUTA is sampled whenever a rising edge is detected on the Filter Block clock input.

The only difference in operation between Sampled, Non-Filtered (# 3A) and Sampled, Non-Filtered (# 3B) is in how the clock to the Filter Block is derived. In #3A, the clock to filter block is externally derived while in #3B, the clock to filter block is internally derived.

The GDU comparator filter has no other function than sample/hold of the comparator output in this mode (# 3B).

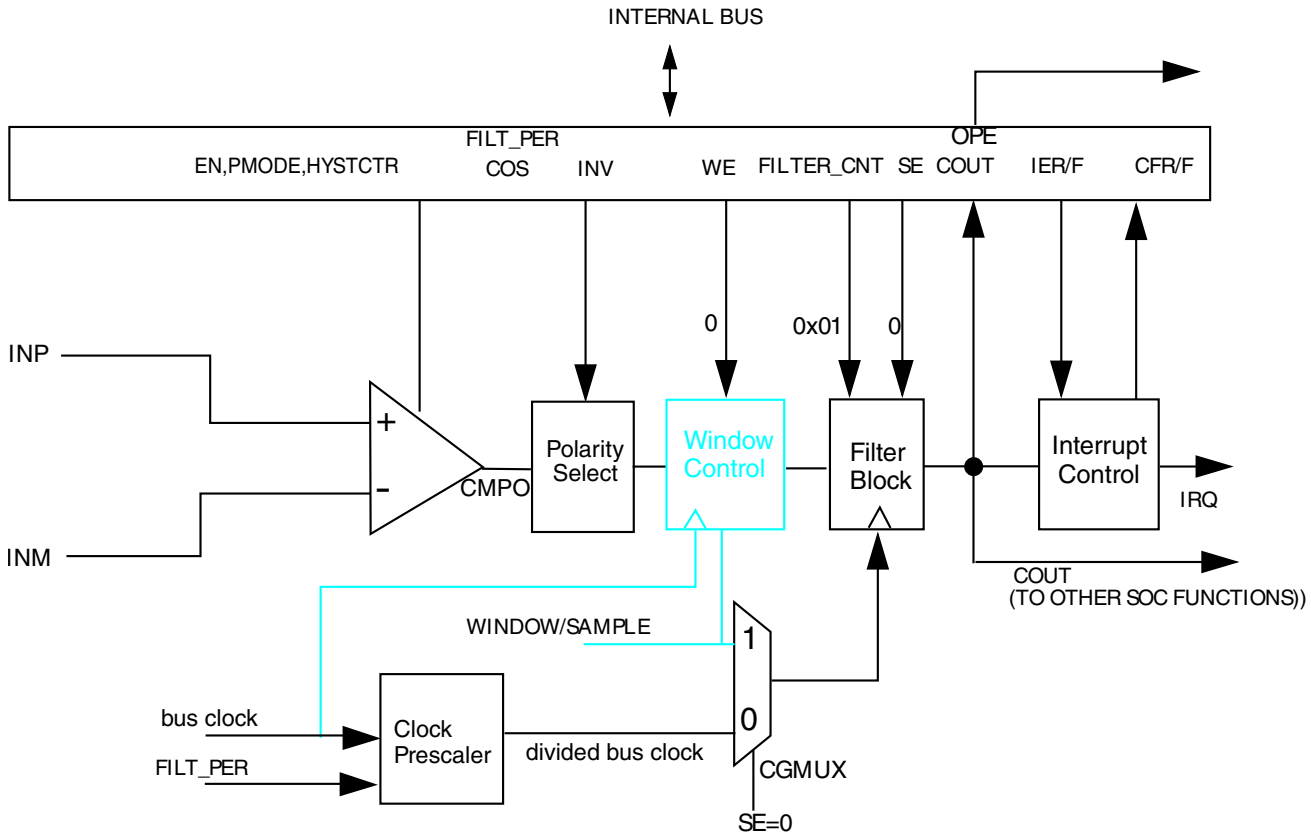


Figure 25-9. Sampled, non-filtered (# 3B): sampling interval internally derived

25.7.4.3.4 Sampled, filtered mode (#s 4A & 4B)

In Sampled, Filtered mode, the analog comparator block is powered and active. The path from analog inputs to COUTA is combinational (unclocked). Windowing Control is completely bypassed. COUTA is sampled whenever a rising edge is detected on the Filter Block clock input.

The only difference in operation between Sampled, Non-Filtered (# 3A) and Sampled, Filtered (# 4A) is that CR0[FILTER_CNT] is now greater than 1, which activates filter operation.

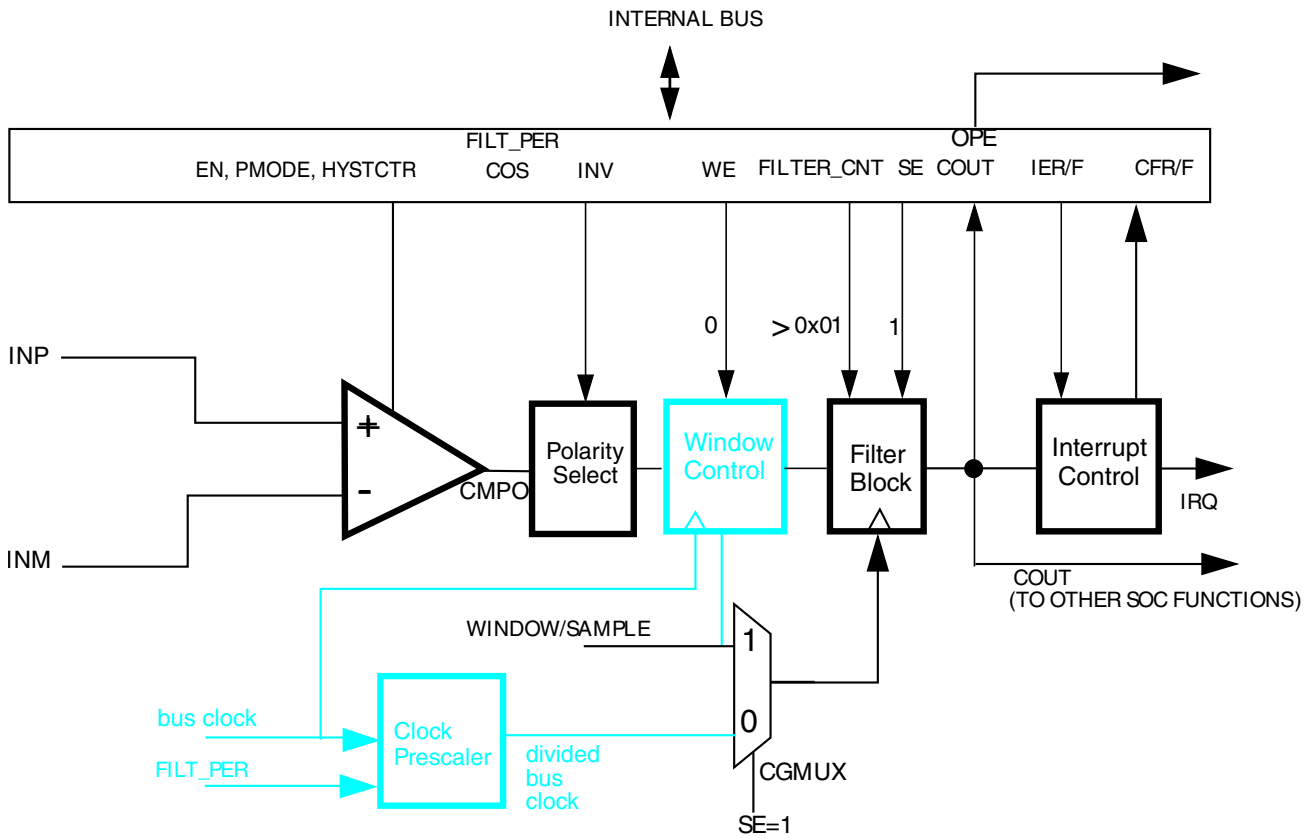


Figure 25-10. Sampled, filtered (# 4A): sampling point externally driven

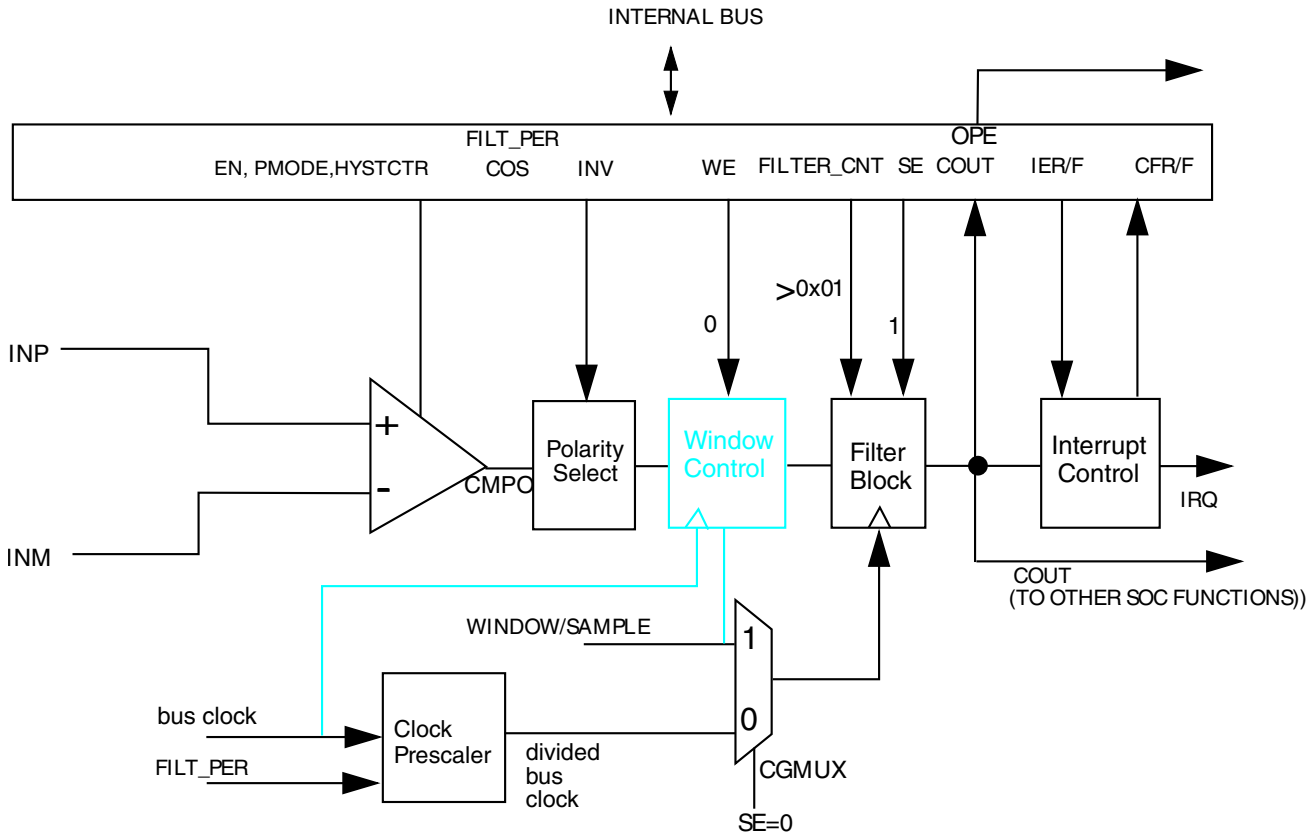


Figure 25-11. Sampled, filtered (# 4B): sampling point internally derived

The only difference in operation between Sampled, Non-Filtered (# 3B) and Sampled, Filtered (# 4B) is that CR0[FILTER_CNT] is now greater than 1, which activates filter operation.

25.7.4.3.5 Windowed mode (#s 5A & 5B)

The following figure illustrates comparator operation in the windowed mode, ignoring latency of the analog comparator, polarity select and Window Control block. It also assumes that the Polarity Select is set to "non-inverting". Note that the analog comparator output is passed to COUTA only when the WINDOW signal is high.

In actual operation, COUTA may lag the analog inputs by up to one bus clock cycle plus the combinational path delay through the comparator and polarity select logic.

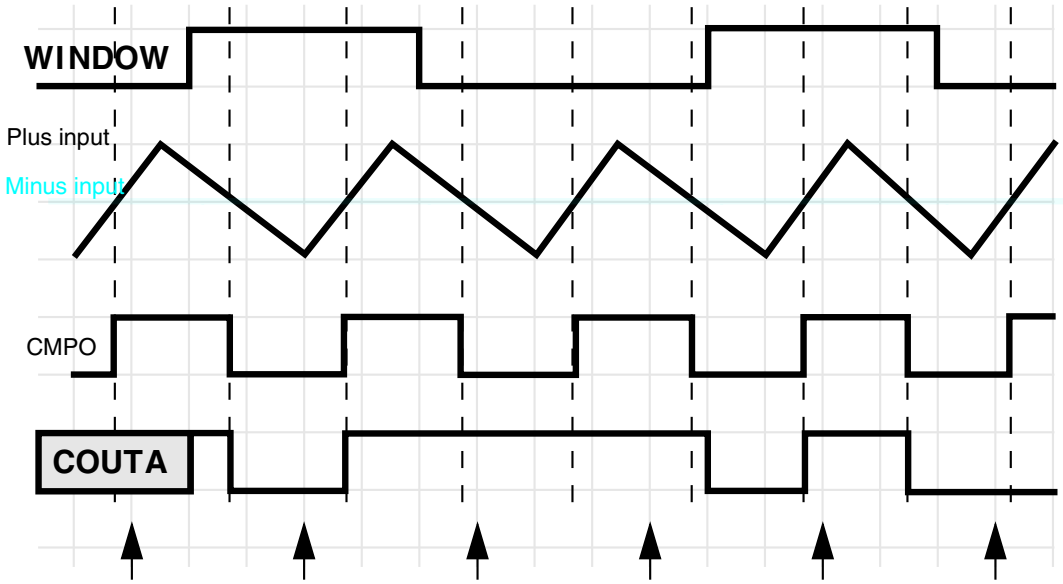


Figure 25-12. Windowed mode operation

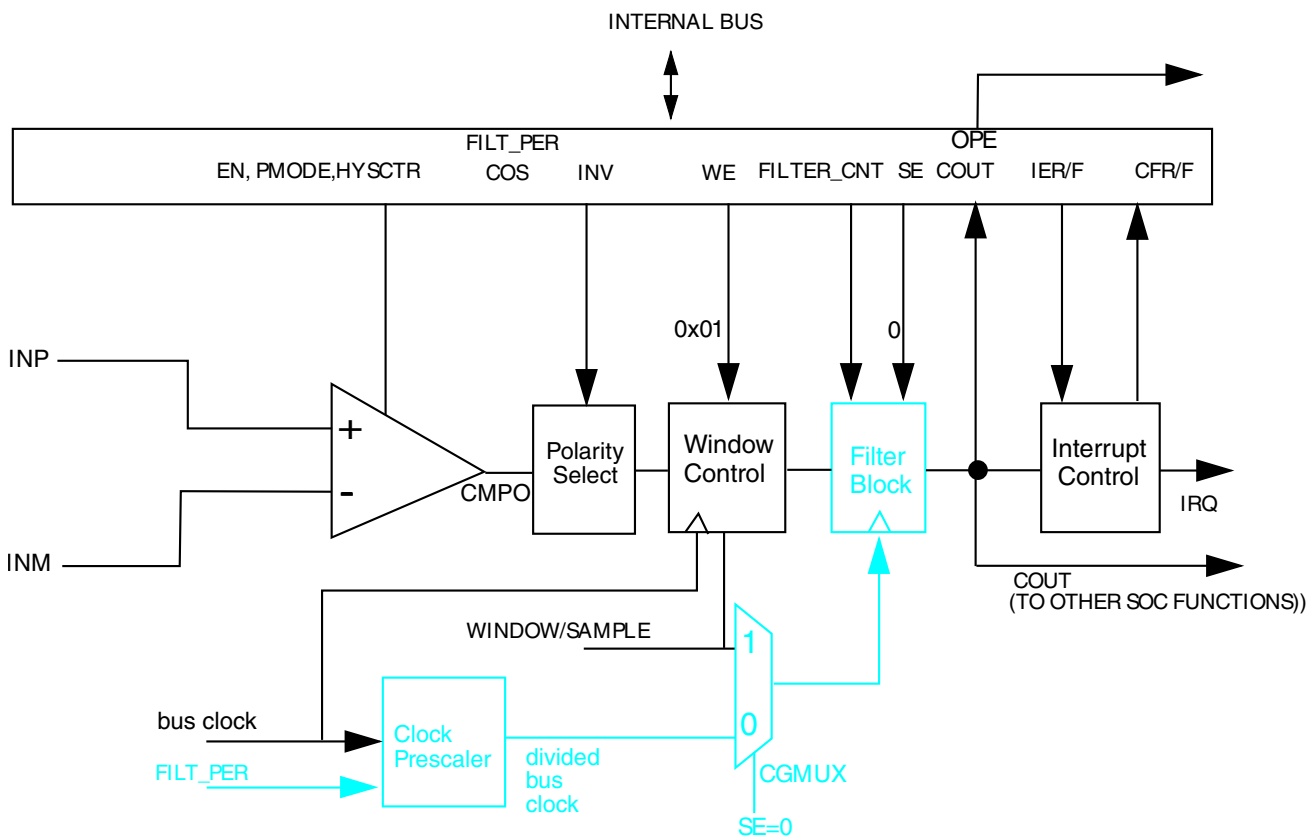


Figure 25-13. Windowed mode

For control configurations which result in disabling the Filter Block, refer to Filter Block Bypass Logic diagram.

When any windowed mode is active, COUTA is clocked by the bus clock whenever WINDOW = 1. The last latched value is held when WINDOW = 0.

25.7.4.3.6 Windowed/Resampled mode (# 6)

The following figure uses the same input stimulus shown in Figure 25-12, and adds resampling of COUTA to generate COUT. Samples are taken at the time points indicated by the arrows. Again, prop delays and latency is ignored for clarity's sake. This example was generated solely to demonstrate operation of the comparator in windowing / resampled mode, and does not reflect any specific application. Depending upon the sampling rate and window placement, COUT may not see zero-crossing events detected by the analog comparator. Sampling period and/or window placement must be carefully considered for a given application.

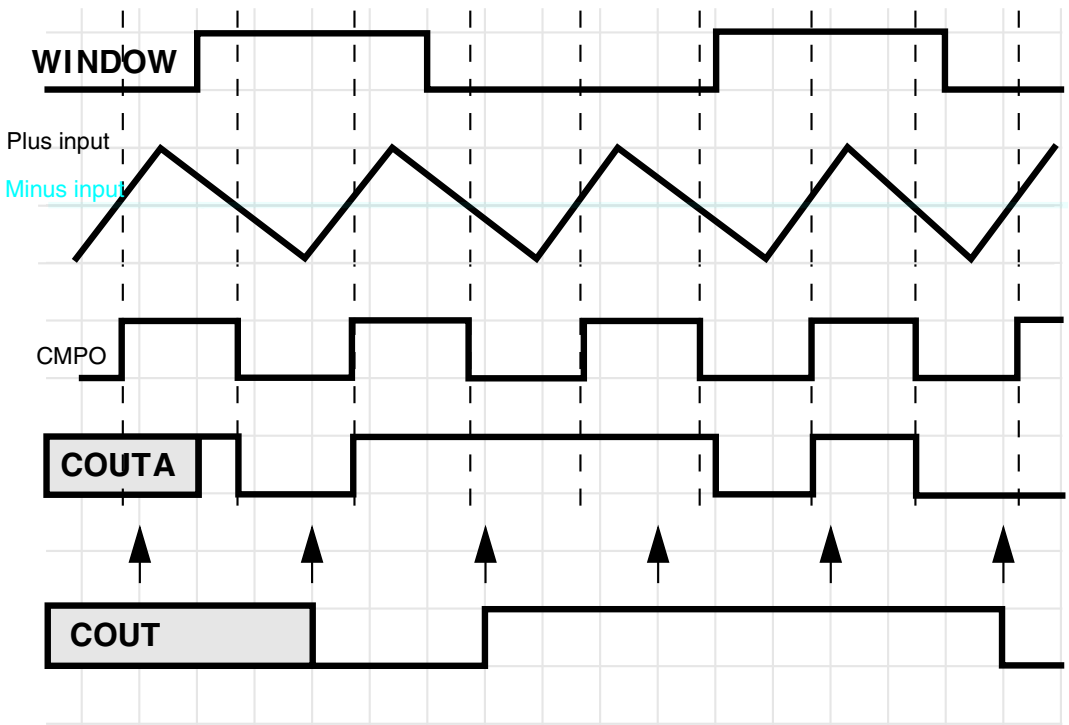


Figure 25-14. Windowed / Resampled Mode Operation

This mode of operation results in an unfiltered string of comparator samples where the interval between the samples is determined by FPR[FILT_PER] and the bus clock rate. Configuration for this mode is virtually identical to that for the Windowed/Filtered Mode shown in the next section. The only difference is that the value of CR0[FILTER_CNT] must be exactly one.

25.7.4.3.7 Windowed/filtered mode (#7)

This is the most complex mode of operation for the comparator block, as it utilizes both windowing and filtering features. It also has the highest latency of any of the modes. This can be approximated: up to 1 bus clock synchronization in the window function + ((CR0[FILTER_CNT] X FPR[FILT_PER]) + 1) X bus clock for the filter function.

When any windowed mode is active, COUTA is clocked by the bus clock whenever WINDOW = 1. The last latched value is held when WINDOW = 0.

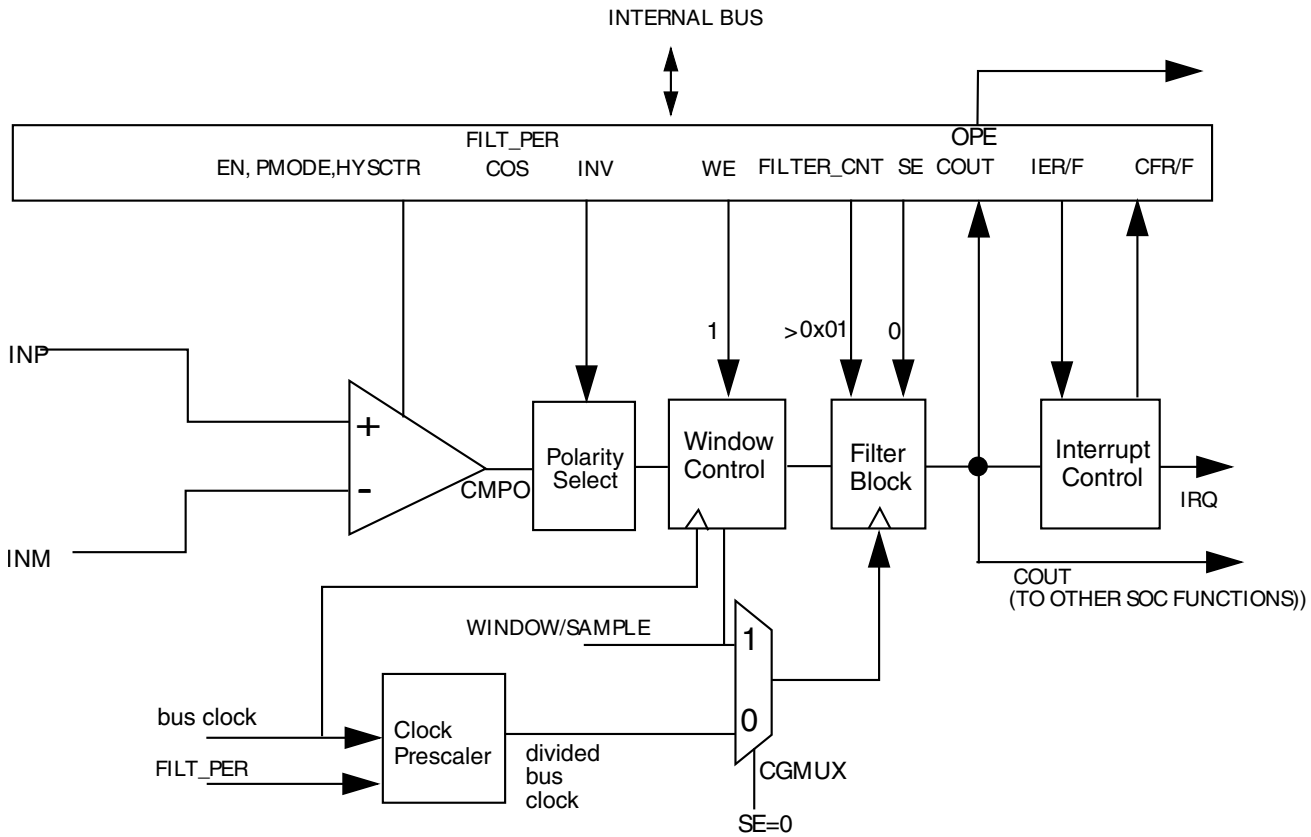


Figure 25-15. Windowed/filtered mode

25.7.4.4 Power modes

25.7.4.4.1 Wait mode operation

During Wait mode and if enabled, the GDU CMP continues to operate normally. Also, if enabled, a GDU CMP interrupt can wake the MCU.

25.7.4.4.2 Stop mode operation

This module is disabled in Stop mode. All internal analog circuits are switched off.

25.7.4.4.3 Background debug mode operation

When the microcontroller is in active background debug mode, the GCMP continues to operate normally.

25.7.4.5 Startup and operation

A typical startup sequence is as follows.

The time required to stabilize COUT will be the power-on delay of the comparators plus the largest propagation delay from a selected analog source through the analog comparator, windowing function and filter. Power on delay of the comparators are available from data sheets. The windowing function has a maximum of 1 bus clock period delay. Filter delay is specified in Low pass filter.

During operation, the propagation delay of the selected data paths must always be considered. It can take many bus clock cycles for COUT and the CFR/CFF status bits to reflect an input change or a configuration change to one of the components involved in the data path.

When programmed for filtering modes, COUT will initially be equal to zero until sufficient clock cycles have elapsed to fill all stages of the filter. This occurs even if COUTA is at a logic one.

25.7.4.6 Low pass filter

The low-pass filter operates on the unfiltered and unsynchronized and optionally inverted comparator output COUTA and generates the filtered and synchronized output COUT. Both COUTA and COUT can be configured as module outputs and are used for different purposes within the system.

Synchronization and edge detection are always used to determine status register bit values. They also apply to COUT for all sampling and windowed modes. Filtering can be performed using an internal timebase defined by FPR[FILT_PER], or using an external SAMPLE input to determine sample time.

The need for digital filtering and the amount of filtering is dependent on user requirements. Filtering can become more useful in the absence of an external hysteresis circuit. Without external hysteresis, high frequency oscillations can be generated at COUTA when the selected INM and INP input voltages differ by less than the offset voltage of the differential comparator.

25.7.4.6.1 Enabling filter modes

Filter Modes are enabled by setting CR0[FILTER_CNT] greater than 0x01 and (setting FPR[FILT_PER] to a non-zero value OR setting CR1[SE]=1). If using the divided bus clock to drive the filter, it will take samples of COUTA every FPR[FILT_PER] bus clock cycles.

The filter output will be at logic zero when first initalized, and will subsequently change when CR0[FILTER_CNT] consecutive samples all agree that the output value has changed. Said another way, SCR[COUT] will be zero for some initial period, even when COUTA is at logic one.

Setting both CR1[SE] and FPR[FILT_PER] to 0 disables the filter and eliminates switching current associated with the filtering process.

Note

Always switch to this setting prior to making any changes in filter parameters. This resets the filter to a known state. Switching CR0[FILTER_CNT] on the fly without this intermediate step can result in unexpected behavior.

If CR1[SE]=1, the filter takes samples of COUTA on each positive transition of the sample input. The output state of the filter changes when CR0[FILTER_CNT] consecutive samples all agree that the output value has changed.

25.7.4.6.2 Latency issues

The FPR[FILT_PER] value (or SAMPLE period) must be set such that the sampling period is just larger than the period of the expected noise. This way a noise spike will only corrupt one sample. The CR0[FILTER_CNT] value should be chosen to reduce the probability of noisy samples causing an incorrect transition to be recognized. The probability of an incorrect transition is defined as the probability of an incorrect sample raised to the CR0[FILTER_CNT] power.

The following table summarizes maximum latency values for the various modes of operation *in the absence of noise*. Filtering latency is restarted each time an actual output transition is masked by noise.

The values of FPR[FILT_PER] (or SAMPLE period) and CR0[FILTER_CNT] must also be traded off against the desire for minimal latency in recognizing actual comparator output transitions. The probability of detecting an actual output change within the nominal latency is the probability of a correct sample raised to the CR0[FILTER_CNT] power.

Table 25-3. Comparator Sample/Filter Maximum Latencies

Mode #	CR1[EN]	CR1[WE]	CR1[SE]	CR0[FILTER_CNT]	FPR[FILT_PER]	Operation	Maximum Latency[1]
1	0	X	X	X	X	Disabled	N/A
2A	1	0	0	0x00	X	Continuous Mode	T_{PD}
2B	1	0	0	X	0x00		
3A	1	0	1	0x01	X	Sampled, Non-Filtered mode	$T_{PD} + T_{SAMPLE} + T_{per}$
3B	1	0	0	0x01	> 0x00		T_{PD} + (FPR[FILT_PER] x T_{per}) + T_{per}
4A	1	0	1	> 0x01	X	Sampled, Filtered mode	T_{PD} + (CR0[FILTER_CNT] x T_{SAMPLE}) + T_{per}
4B	1	0	0	> 0x01	> 0x00		T_{PD} + (CR0[FILTER_CNT] x FPR[FILT_PER] x T_{per}) + T_{per}
5A	1	1	0	0x00	X	Windowed mode	$T_{PD} + T_{per}$
5B	1	1	0	X	0x00		$T_{PD} + T_{per}$
6	1	1	0	0x01	0x01 - 0xFF	Windowed / Resampled mode	T_{PD} + (FPR[FILT_PER] x T_{per}) + $2T_{per}$
7	1	1	0	> 0x01	0x01 - 0xFF	Windowed / Filtered mode	T_{PD} + (CR0[FILTER_CNT] x FPR[FILT_PER] x T_{per}) + $2T_{per}$

1. T_{PD} represents the intrinsic delay of the analog component plus the polarity select logic. T_{SAMPLE} is the clock period of the external sample clock. T_{per} is the period of the bus clock.

25.8 GCMP interrupts

The GCMP module is capable of generating an interrupt on either the rising or falling edge of the comparator output (or both). The interrupt request is asserted when both SCR[IER] bit and SCR[CFR] are set. It is also asserted when both SCR[IEF] bit and SCR[CFF] are set. The interrupt is de-asserted by clearing either SCR[IER] or SCR[CFR] for a rising edge interrupt, or SCR[IEF] and SCR[CFF] for a falling edge interrupt.

Chapter 26
Pulse Width Modulator (PWM)

26.1 Chip specific pulse width modulator

This device has one pulse width modulator (PWM).

The PWM can be configured as three complementary pairs, six independent PWM signals or their combinations, such as one complementary and four independent. Both edge- and center-aligned synchronous pulse width control, from 0 to 100% modulation, are supported.

A 15-bit common PWM counter is applied to all six channels. PWM resolution is one clock period for edge-aligned operation and two clock periods for center-aligned operation. The clock period is dependent on clock source frequency at HSCLK which is up to 40 MHz and a programmable prescaler.

When generating complementary PWM signals, the module features automatic deadtime insertion to PWM output pairs. Each PWM output can be controlled by PWM generator, timer or software manually and separate top and bottom output polarity control. Asymmetric PWM output is able to change PWM duty cycle alternatively at every half cycle without software involvement.

Table 26-1. PWM module signals connection

Module	Signal	Connect to
PWM	Fault0	PTC0/PWM_Fault0
	Fault1	XBAR_OUT14
	Fault2	GDU OPAMP0 ACMP OUT
	Fault3	GDU OPAMP1 ACMP OUT
	PWM_Synch	XBAR_IN4
	PWM0, PWM1	GDU inputs, XB_IN5 2-to-1 Mux
	PWM2, PWM3	GDU inputs, XB_IN6 2-to-1 Mux
	PWM4, PWM5	GDU inputs, XB_IN7 2-to-1 Mux
	Internal Clock	HSCLK

26.2 Introduction

26.2.1 Overview

This chapter describes the pulse width modulator (PWM) module. The PWM can be configured as three complementary pairs, six independent PWM signals, or their combinations (such as one complementary pair and four independent signals). Both edge- and center-aligned synchronous pulse-width control, from zero to 100 percent modulation, are supported.

A 15-bit common PWM counter is applied to all six channels. PWM resolution is one clock period for edge-aligned operation and two clock periods for center-aligned operation. The clock period is dependent on system clock frequency and a programmable prescaler.

When generating complementary PWM signals, the module features automatic deadtime insertion to PWM output pairs. Each PWM output can be controlled manually by a PWM generator or software, and separate top and bottom output-polarity control. Asymmetric PWM output is able to change the PWM duty cycle alternatively at every half cycle without software involvement.

26.2.2 Features

The PWM has the following features:

- PWM operation clock runs at system clock
- Six PWM signals
  - all independent
  - complementary pairs
  - mix independent and complementary
- Features of complementary channel operation
  - separate deadtime insertions for rising and falling edges
  - separate top and bottom pulse-width correction via software
  - asymmetric PWM output within center align operation
  - separate top and bottom polarity control
- Edge- or center-aligned PWM signals
- 15 bits of resolution
- Half-cycle reload capability
- Integral reload rates from 1 to 16
- Individual software controlled PWM output

- Programmable fault protection
- PWM compare output polarity control
- PWM output polarity control
- Write-protected registers

26.2.3 Modes of operation

Exercise care when using this module in certain chip operating modes. Some applications require regular software updates for proper operation. Failure to use caution could result in damaging the circuit. Because of this, PWM outputs are placed in their inactive states in stop mode, and optionally under wait and EOnCE modes. PWM outputs will be reactivated (assuming they were active to begin with) when these modes are exited.

Table 26-2. Modes when PWM operation is restricted

	Mode
Stop	PWM outputs are disabled
Wait	PWM outputs disabled as a function of CNFG WAIT_EN bit.
EOnCE	PWM outputs are disabled as a function of the CNFG DBG_EN bit.

26.2.4 Block diagram

The following figure show the block diagram of the PWM.

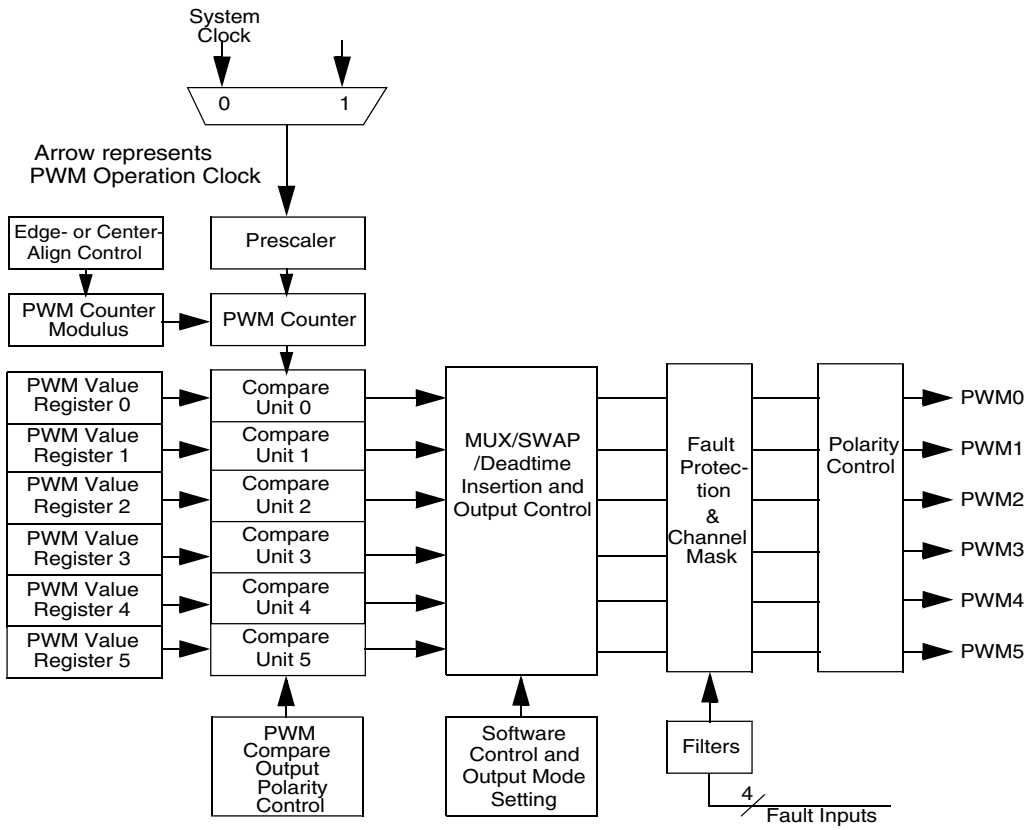


Figure 26-1. PWM block diagram

The following figure shows PWM SWAP and MASK functionality.

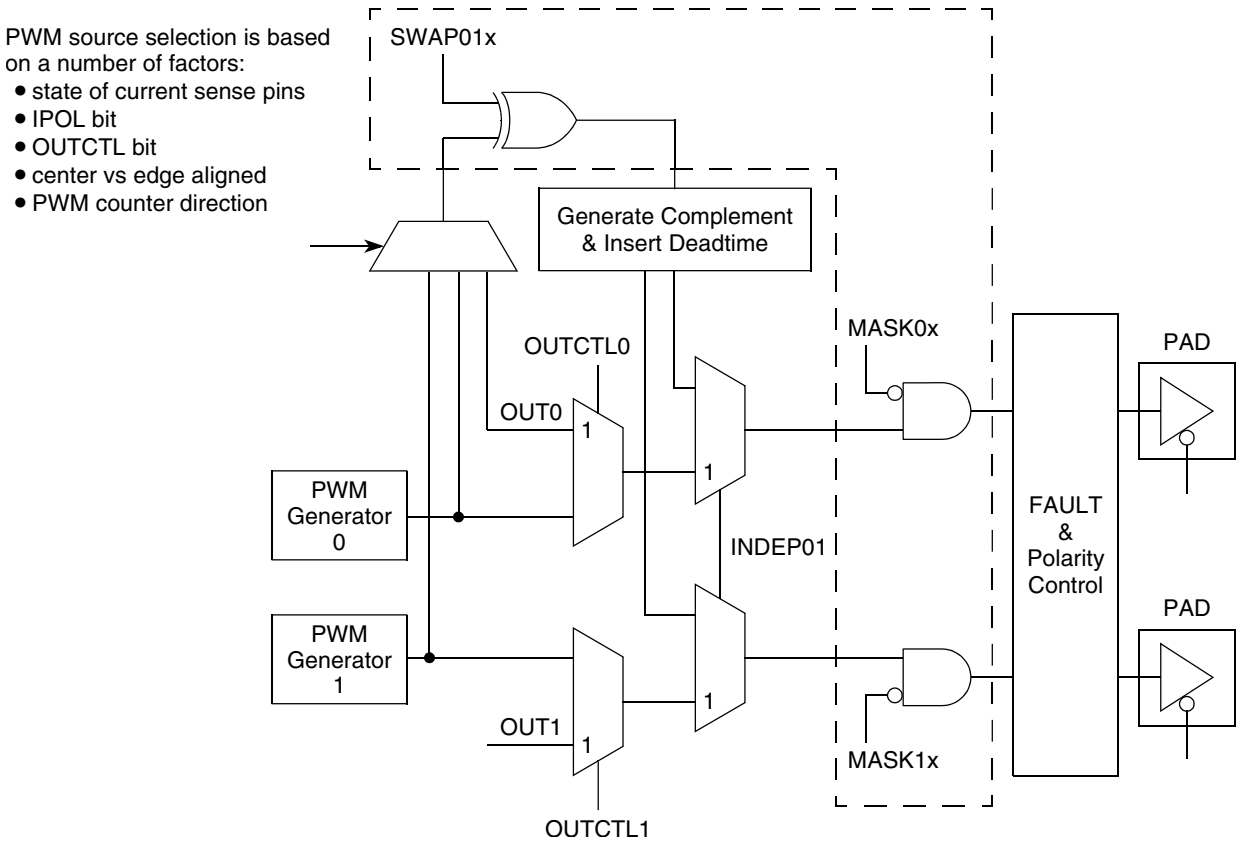


Figure 26-2. PWM SWAP

26.3 Functional description

26.3.1 Prescaler

To permit lower PWM frequencies, the prescaler produces the PWM clock frequency by dividing the PWM operation clock frequency by one, two, four, or eight. The prescaler bits, PRSC0 and PRSC1 in the control (CTRL) register, select the prescaler divisor. This prescaler is buffered and will not be used by the PWM generator until the LDOK bit is set and a new PWM reload cycle begins.

26.3.2 Generator

The PWM generator contains a 15-bit up/down PWM counter producing output signals with software selectable alignment, period, duty cycle, and the inversion of PWM signal generation.

26.3.2.1 Alignment and compare output polarity

The edge-align (EDG) bit in the configure (CNFG) register selects either center-aligned or edge-aligned PWM generator outputs.

PWM compare output polarity is selected by the CINVn bit field in the compare invert (CINV) register. Please see the output operations in the following two figures.

The PWM compare output is driven to high state when the value of PWM value (VAL0-5) register is greater than the value of PWM counter, and PWM compare is counting downwards if the corresponding channel CINVx=0. Or, the PWM compare output is driven to low state if the corresponding channel CINVx=1.

The PWM compare output is driven to low state when the value of PWM value (VAL0-5) register matches the value of PWM counter, and PWM counter is counting upwards if the corresponding channel CINVx=0. Or, the PWM compare output is driven to high state if the corresponding channel CINVx=1.

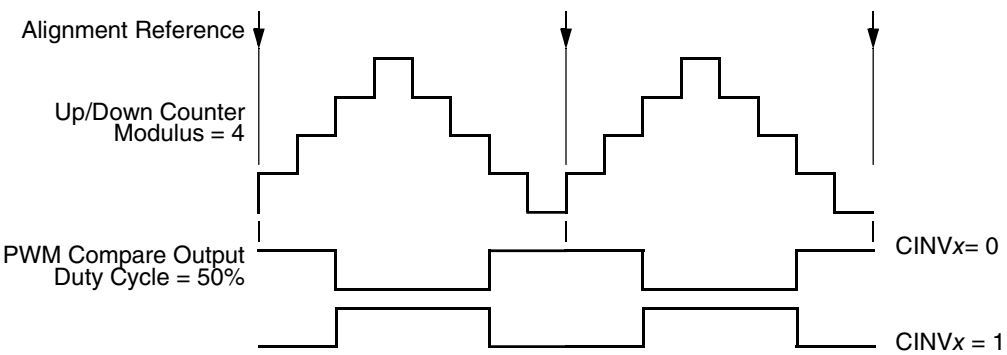


Figure 26-3. Center-Aligned PWM output

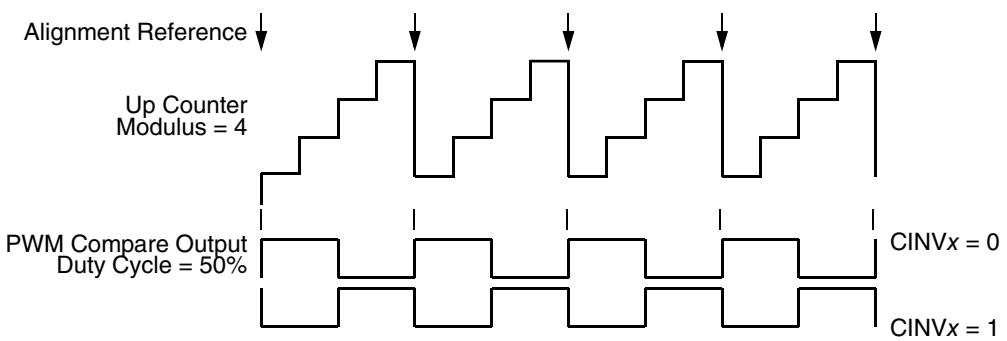


Figure 26-4. Edge-Aligned PWM output

NOTE

Because of the equals-comparator architecture of this PWM, the modulus=0 case is considered illegal. However, the deadtime constraints and fault conditions will still be guaranteed.

26.3.2.2 Period

The PWM period is determined by the value written to the counter modulo (CMOD) registers. The PWM counter is an up/down counter in a center-aligned operation. In this mode the PWM highest output resolution is two 2× system clock cycles if the PWM clock inputs from 2× system clock. The modulus is one-half of the PWM output period in PWM clock cycles.

PWM period = (PWM modulus) × (PWM clock period) × 2

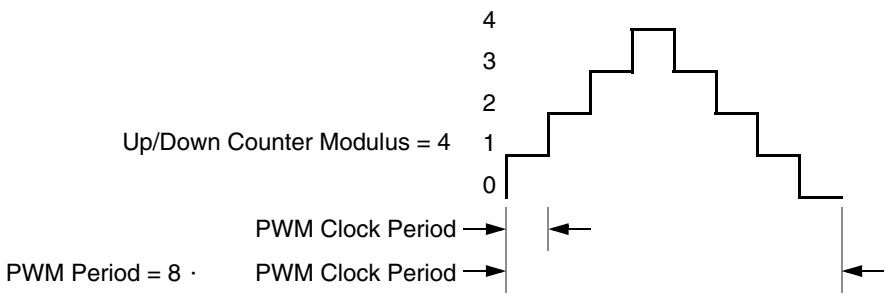


Figure 26-5. Center-Aligned PWM period

The PWM counter is an up-counter during an edge-aligned operation. In this mode, the PWM highest output resolution is one 2× system clock cycle if the PWM clock inputs from 2× system clock. The modulus is the period of the PWM output in PWM clock cycles.

PWM period = (PWM modulus) × (PWM clock period)

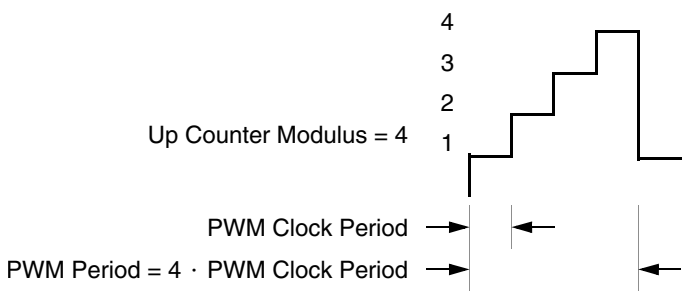


Figure 26-6. Edge-Aligned PWM period

26.3.2.3 Pulse width duty cycle

The signed 16-bit number written to the PWM value registers is the pulse width in PWM clock periods of the PWM prescaler output (or period minus the pulse width if CINVx=1).

Duty Cycle = (PWM value/Modulus)× 100

NOTE

A PWM value less than, or equal to zero deactivates the PWM output for the entire PWM period. A PWM value greater than, or equal to the modulus activates the PWM output for the entire PWM period when CINVx=0, and vice versa if CINVx=1.

Table 26-3. PWM value and underflow conditions

PWMVALn	Condition	PWM value used
0x0000–0x7FFF	Normal	Value in registers $8000–$
0x8000–0xFFFF	Underflow	0x0000

A center-aligned operation is illustrated in the following figure. The pulse width is twice the value written to the PWM value register with center-aligned output in PWM clock cycles.

PWM pulse width = (PWM value) × (PWM clock period) × 2

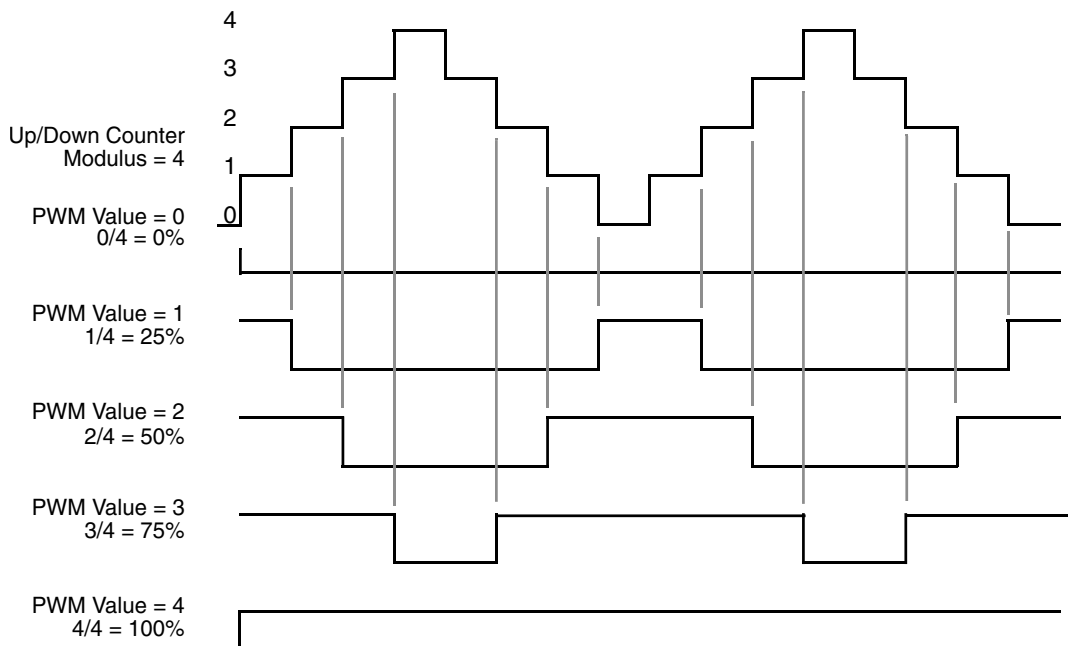


Figure 26-7. Center-Aligned PWM pulse width

An edge-aligned operation is illustrated in the following figure. The pulse width is the value written to the PWM value register with edge-aligned output in PWM clock cycles.

PWM pulse width = (PWM value) × (PWM clock period)

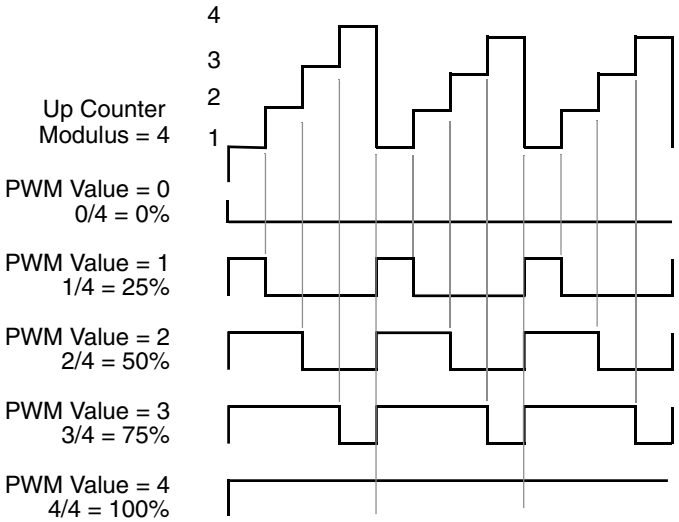


Figure 26-8. Edge-Aligned PWM pulse width

26.3.3 Independent or complementary channel operation

In the CNFG register, writing 1 to the independent (INDEPnn) or complement pair operation bit configures a pair of the PWM outputs as two independent PWM channels. Each PWM output has its own PWM value register operating independently of the other channels in independent channel operation.

Writing 0 to the INDEPnn bit configures the PWM output as a pair of complementary channels. The PWM pins are paired in complementary channel operation, illustrated in the following figure.

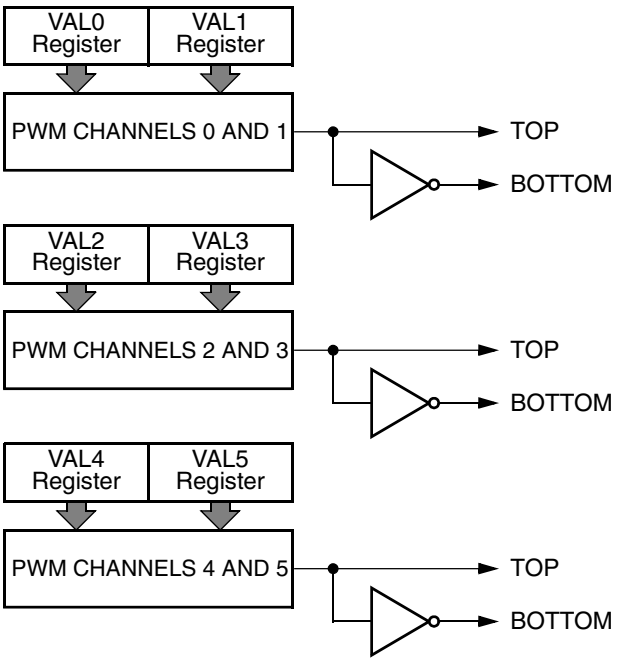


Figure 26-9. Complementary channel pairs

The complementary channel operation drives top and bottom transistors in an inverter circuit, such as the one in the following figure.

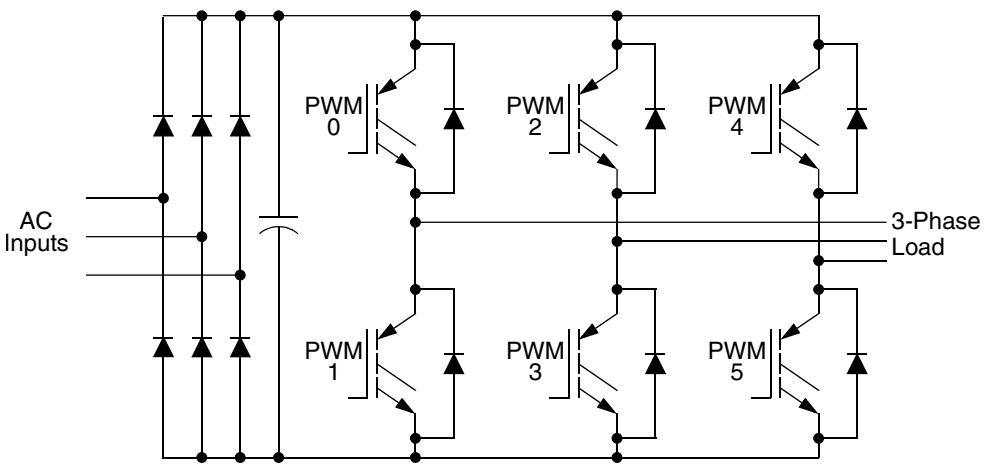


Figure 26-10. Typical 3-phase inverter

In complementary channel operation, there are three additional features:

- Deadtime insertion
- Separate top and bottom pulse width correction for distortions caused by deadtime inserted and reactive load characteristics
- Separate top and bottom output polarity control

26.3.4 Deadtime generators

While in the complementary mode, each PWM pair can be used to drive top/bottom transistors. Ideally, the PWM pairs are an inversion of each other. When the top PWM channel is active, the bottom PWM channel is inactive and vice versa.

To avoid short-circuiting between top and bottom transistor, there must be no overlap of conducting intervals between top and bottom transistor. But the transistor's characteristics make its switching-off time longer than switching-on time. To avoid the conducting overlap of top and bottom transistors, deadtime may be operationally inserted in the switching period.

Deadtime generators automatically insert software-selectable activation delays into each pair of PWM outputs during switching. The PWM deadtime (DTIM) registers specify the number of PWM clock cycles to use for deadtime delay. Every time the PWM generator output changes state, deadtime is inserted. DTIM controls deadtime during both low state to high state transitions and high state to low state transitions. Deadtime forces both PWM outputs in the pair to the inactive state.

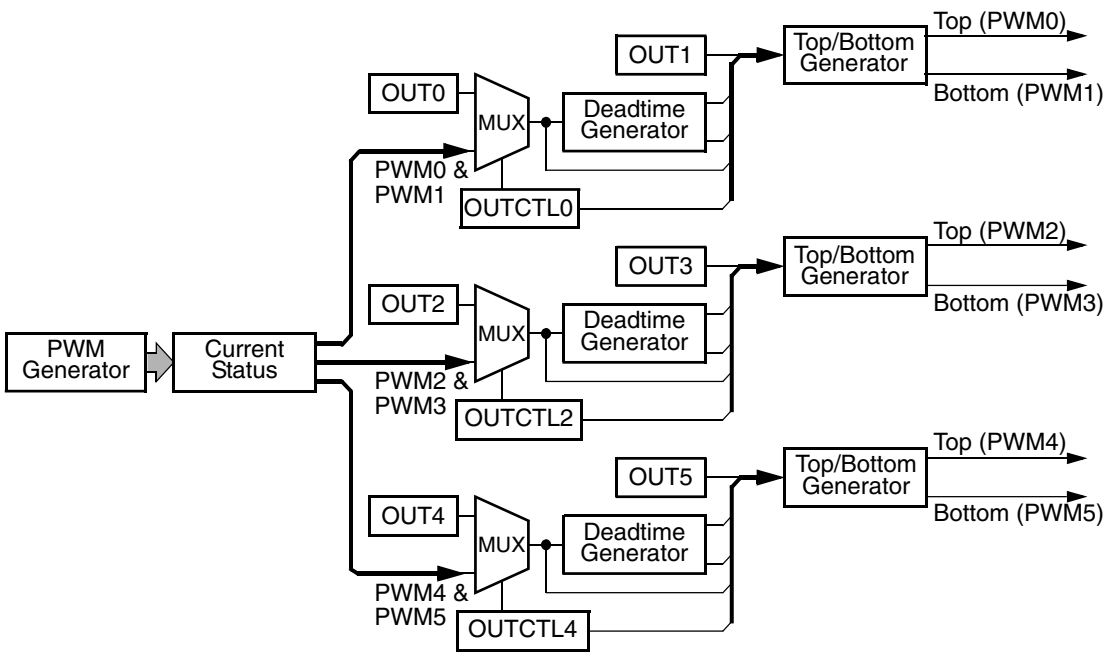


Figure 26-11. Deadtime generators

The following figures illustrate deadtime insertion in different operation conditions.

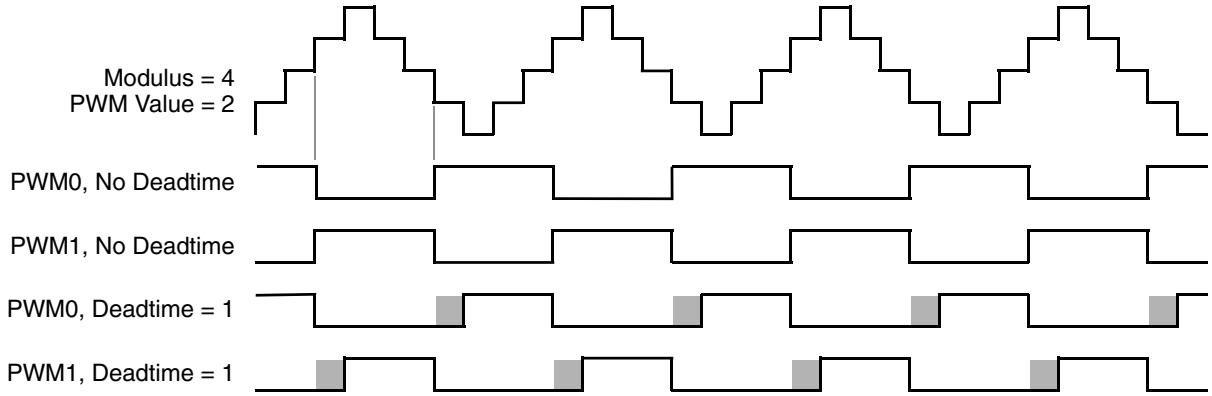


Figure 26-12. Deadtime insertion, center alignment

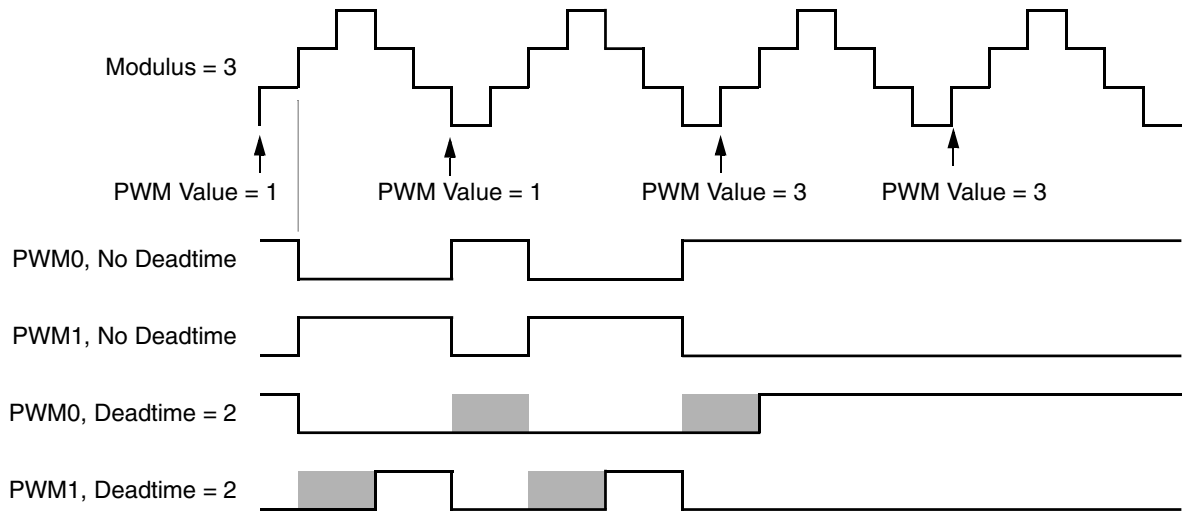


Figure 26-13. Deadtime at duty cycle boundaries

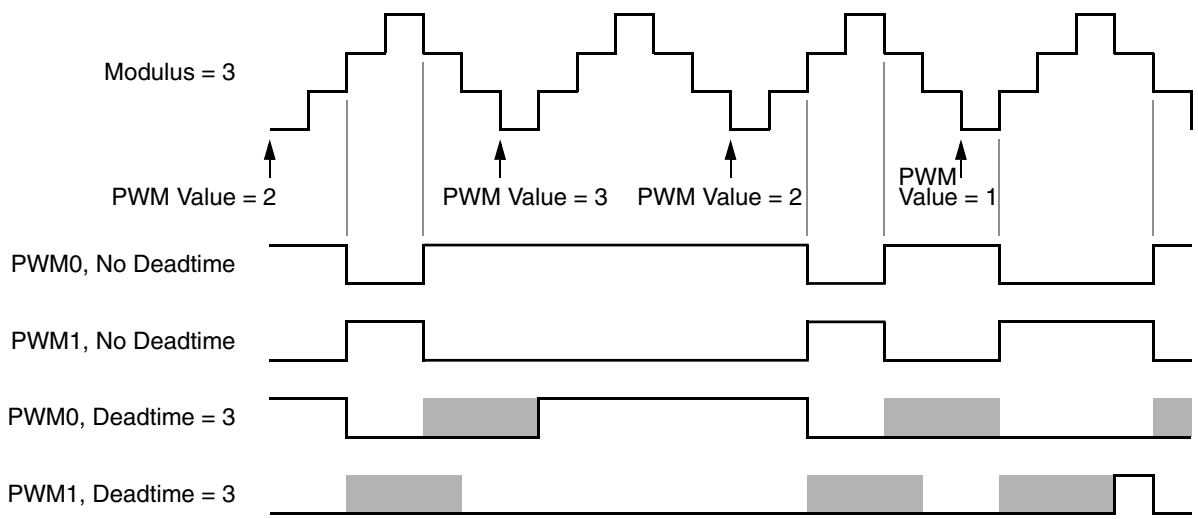


Figure 26-14. Deadtime and small pulse widths

NOTE

The waveform at the output pin is delayed by two PWM Operation Clock cycles for deadtime insertion.

26.3.5 Asymmetric PWM output

In complementary mode with center-align operation, the PWM duty cycle is able to change alternatively at every half cycle. The count direction of the PWM counter selects either the odd or the even PWM value registers to use in the PWM cycle. For counting up, select even PWM value registers to use in the PWM cycle. For counting down, select odd PWM value registers to use in the PWM cycle.

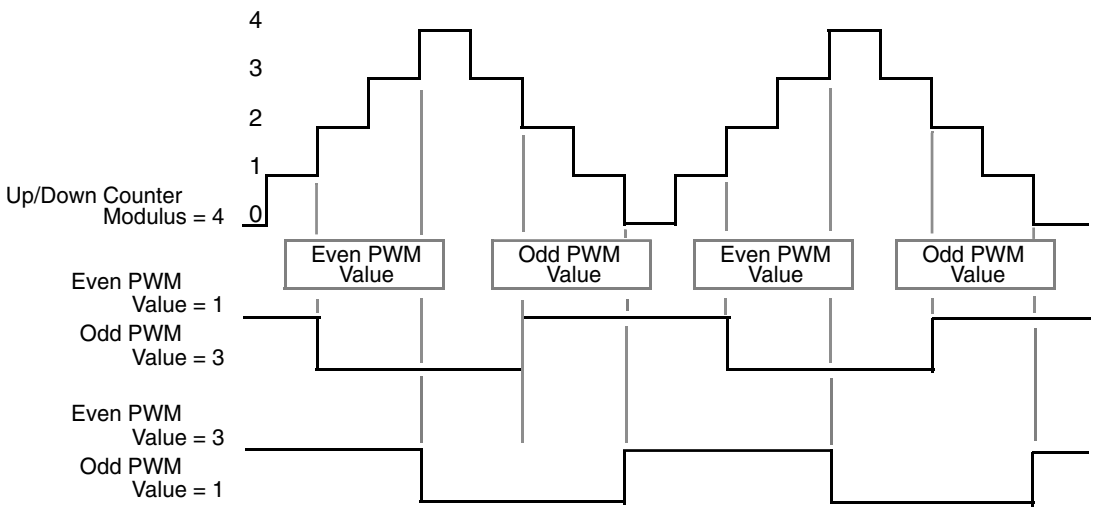


Figure 26-15. Asymmetric waveform - phase shift PWM output

26.3.6 Variable edge placement PWM output

In complementary mode with edge-aligned mode, the timing of both edges of the PWM output can be controlled using the PECn bits in the PECTRL register and the CINVn bits in the CINV register. The edge aligned pulse created by the even value register and the associated CINV bit is XORed with the pulse created by the odd value register and its associated CINV. The results of the XOR are fed into the complement and dead-time logic. In contrast to asymmetric PWM output mode, the PWM phase shift can pass the PWM cycle boundary, as shown in the following figure.

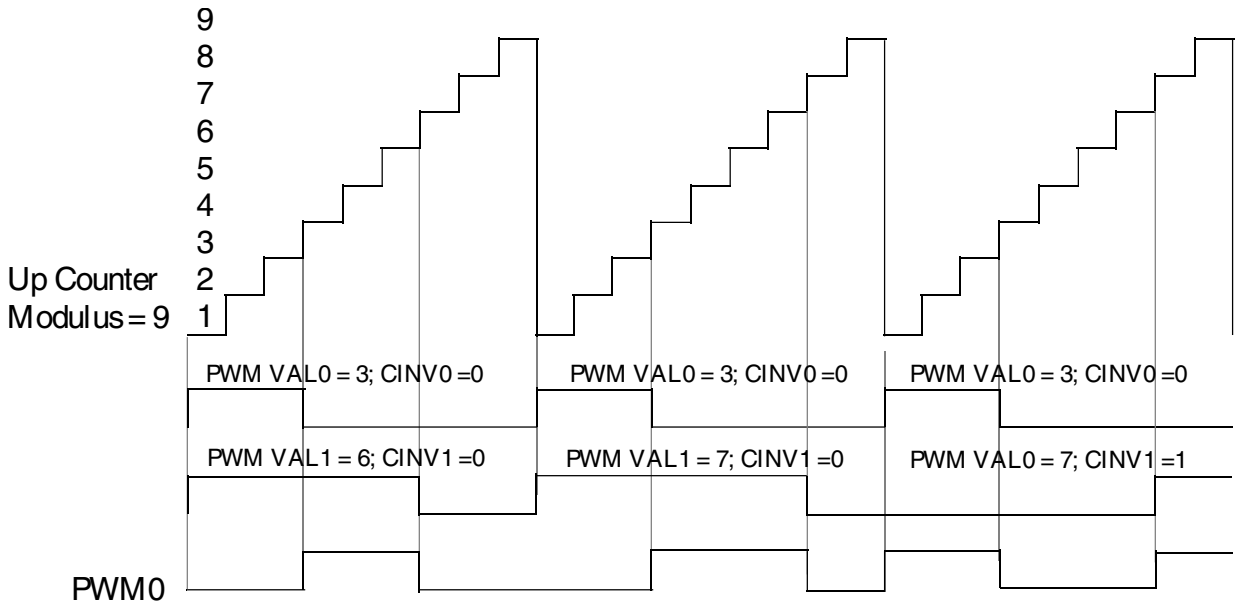


Figure 26-16. Variable edge placement waveform - phase shift PWM output

26.3.7 PWM output polarity

Positive polarity means when the PWM is active its output is high. Conversely, negative polarity means when the PWM is active its output is low.

Output polarity of the PWMs is determined by two options:

- TOPNEGnn controls the polarity of PWM0, PWM2 and PWM4 outputs, which typically drive the top transistors of the pair. When TOPNEGnn is set these outputs are active-low.
- BOTNEGnn controls the polarity of PWM1, PWM3 and PWM5 outputs, which typically drive the bottom transistors of the pair. When BOTNEGnn is set these outputs are active-low.

Both TOPNEGnn and BOTNEGnn bits are in the configure (CNFG) register. Software Output Control

Setting output control enable (OUTCTRLn) bit, the PWM outputs are driven by software rather than by the PWM generator.

In an independent mode, with OUTCTRLn=1, the output bit OUTn, controls the PWMn channel. Setting and clearing the OUTn bit activates and deactivates the corresponding PWM channel.

The OUTCTRLn and OUTn bits are in the PWM output control (OUT) register.

During software output control, TOPNEGnn and BOTNEGnn still control output polarity.

In complementary channel operation, odd and even OUTCTRLn must be identical and switched concurrently for proper operation. The even-numbered OUTn bits replace the PWM generator outputs. The deadtime generators inserts deadtime whenever an even OUTn bit toggles. Deadtime is not inserted when the odd OUTn bit toggles. The even OUTn bit controls complementary channel pairs when the odd OUTn bit is set. However, the even OUTn bit still controls complementary channel pairs with odd PWMn deactivated if the odd OUTn bit is cleared. In other words, setting the odd OUTn bit makes its corresponding PWMn the complement of its even pair, while clearing the odd OUTn bit deactivates the odd PWMn. Please refer to the following figure.

Setting the OUTCTLn bits do not disable the PWM generators. They continue to run, but no longer control the output pins. When the OUTCTLn bits are cleared, the outputs of the PWM generator takes control of PWM outputs at the beginning of the next PWM cycle. Please refer to the following figure.

Software can drive the PWM outputs, even when the PWM Enable (PWMEN) bit is set to zero.

NOTE

Avoid an unexpected deadtime insertion by clearing the OUTn bits before setting and after clearing the OUTCTLn bits.

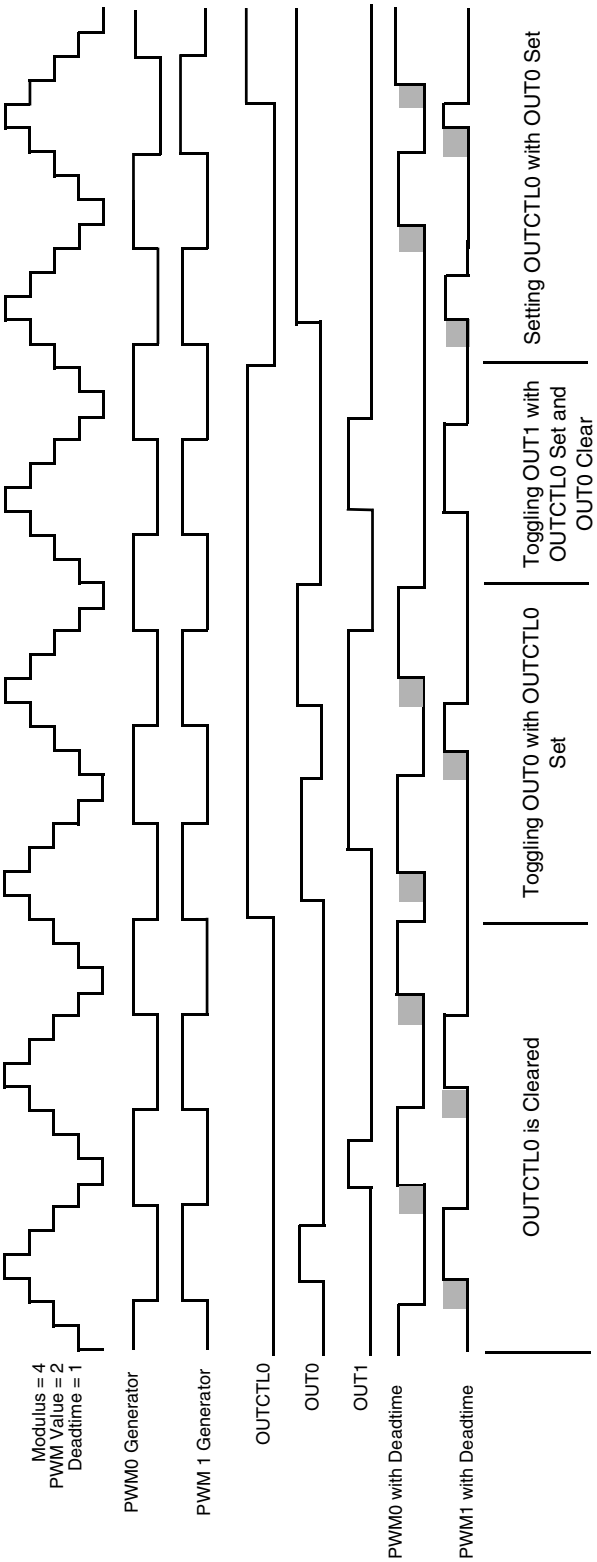


Figure 26-17. Software output control in complementary mode

26.3.8 Generator loading

26.3.8.1 Load enable

The load okay (LDOK) bit enables loading the PWM generator with:

- A prescaler divisor from the PRSC1 and PRSC0 bits in the control (CTRL) register
- A PWM period from the PWM counter modulus (CMOD) register
- A PWM pulse width from the all PWM value (VALn) registers

LDOK prevents reloading of these PWM parameters simultaneously. Setting LDOK allows the prescale bits, CMOD and VALn registers to be loaded into a set of buffers. The loaded buffers are used by the PWM generator at the beginning of the next PWM reload cycle. Set LDOK by reading it, and then writing a 1 to it. After loading, LDOK is automatically cleared.

26.3.8.2 Load frequency

The LDFQ3, LDFQ2, LDFQ1, and LDFQ0 bits in the CTRL register select an integral loading frequency of one to 16-PWM reload opportunities. The LDFQ bits take effect at every PWM reload opportunity, regardless of the state of the LDOK bit. The HALF bit in the CTRL register controls half-cycle reloads for center-aligned PWMs. If the HALF bit is set, a reload opportunity occurs at both beginning of the PWM cycle and at the PWM half cycle. If the HALF bit is not set, a reload opportunity occurs only at the beginning of the cycle. Reload opportunities can only occur at the beginning of a PWM cycle in edge-aligned mode.

NOTE

Loading a new modulus on a half cycle will force the counter to the new modulus value minus one count on the next PWM clock cycle. Half cycle reloads are only changes reload rate in center-aligned mode. Enabling or disabling half cycle reloads in edge-aligned mode will have no effect on the reload rate.

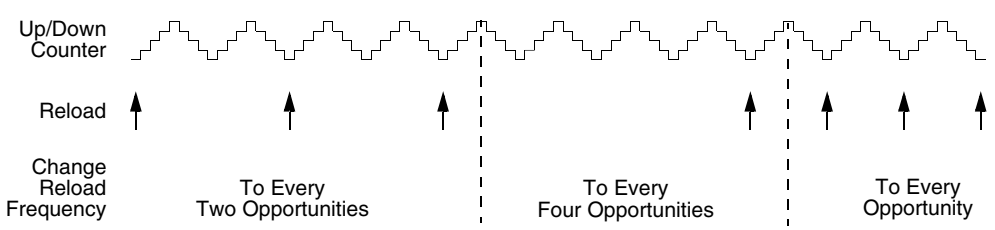


Figure 26-18. Full cycle reload frequency change

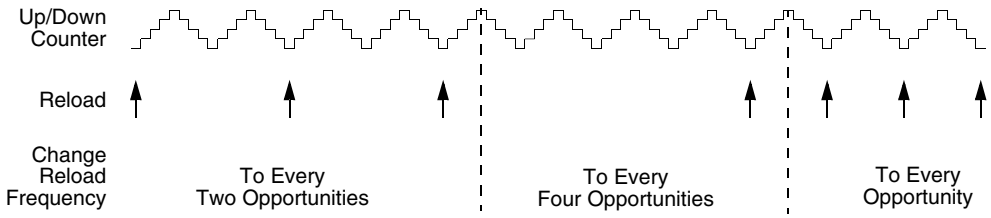


Figure 26-19. Half cycle reload frequency change

26.3.8.3 Reload flag

At every reload opportunity the PWM reload flag (PWMF) bit in the CTRL register is set regardless of the state of the LDOK bit. If the PWM reload interrupt enable (PWMRIE) bit is set, the PWMF flag generates a core interrupt request allowing software to calculate new PWM parameters in real time. When PWMRIE is not set, reloads still occur at the selected reload rate without generating interrupt requests. Clear the PWMF bit by reading it then write a 0 to it.

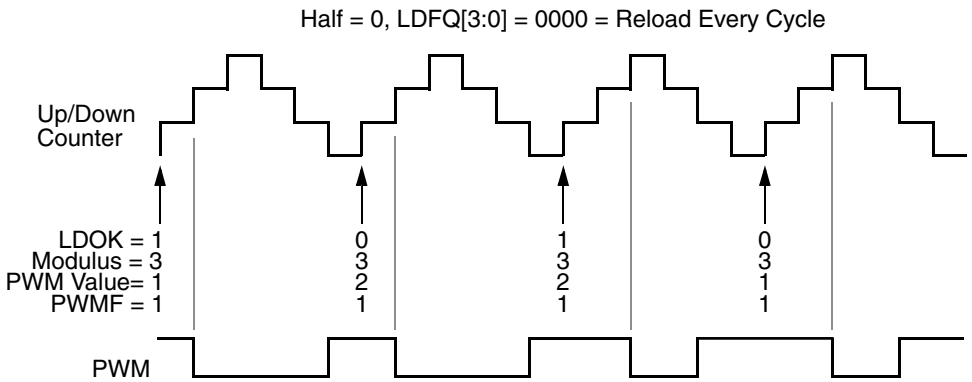


Figure 26-20. Full-Cycle center-aligned PWM value loading

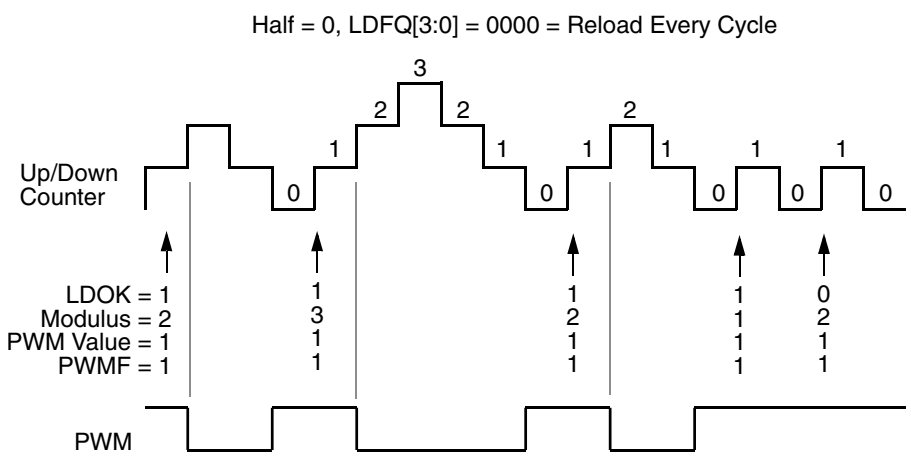


Figure 26-21. Full-Cycle center-aligned modulus loading

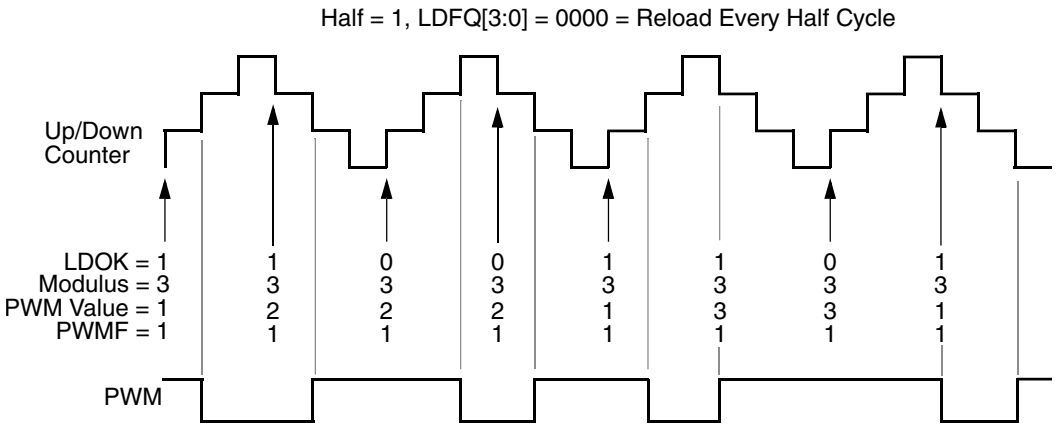


Figure 26-22. Half-Cycle center-aligned PWM value loading

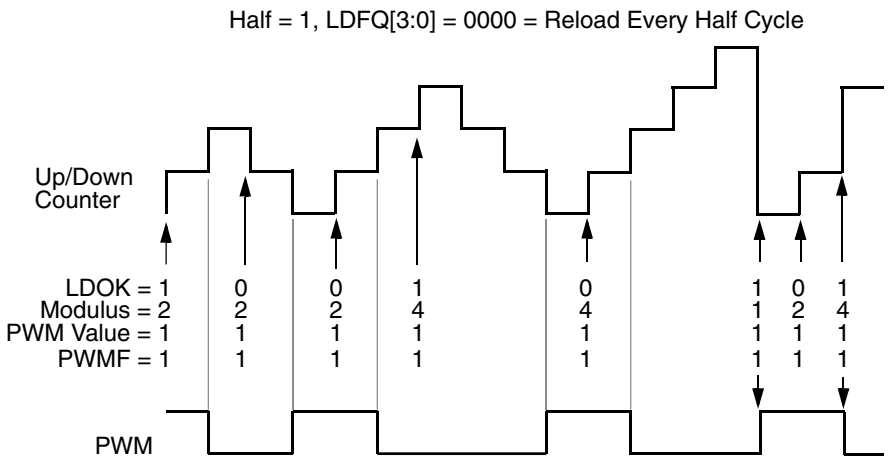


Figure 26-23. Half-Cycle center-aligned modulus loading

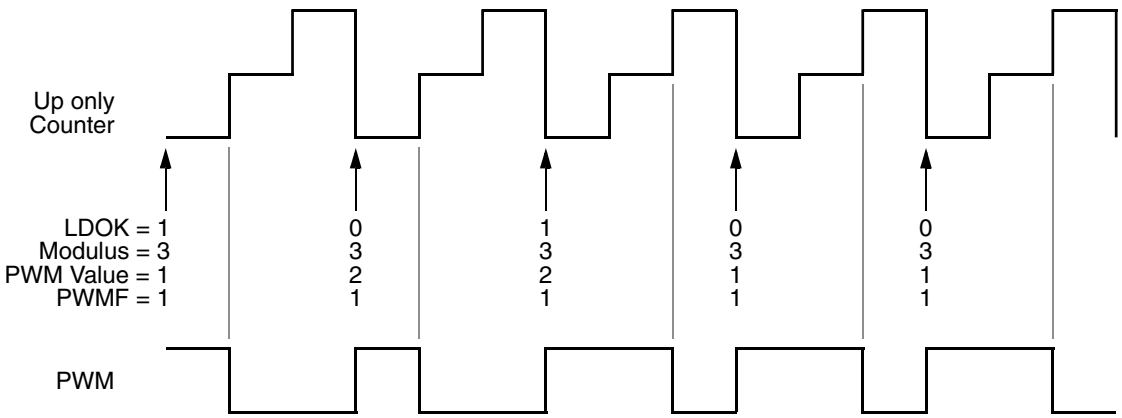


Figure 26-24. Edge-Aligned PWM value loading

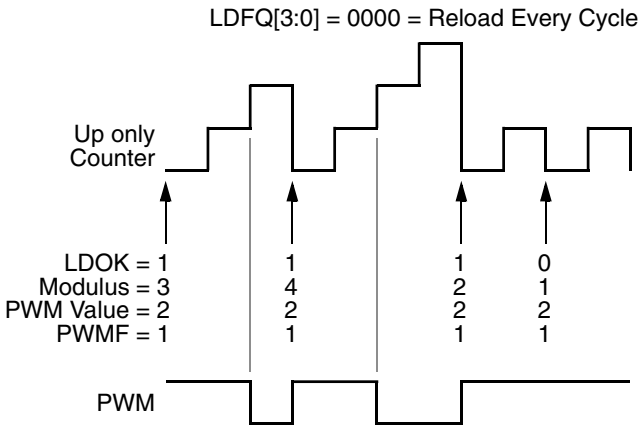


Figure 26-25. Edge-Aligned modulus loading

26.3.8.4 Initialization

Initialize all registers and set the load okay (LDOK) bit before setting the ENABLE (PWMEN) bit. With LDOK set, setting the PWMEN bit is first set, a reload will immediately occur, thereby setting the PWMF bit. The PWMF bit generates an interrupt request if the PWMRIE bit is set. In complementary channel operation, the IPOLn bits determine whether the even or odd numbered PWM value registers control the outputs for the first PWM cycle.

NOTE

Even if LDOK is not set, setting PWMEN also sets the PWMF bit. To prevent a core interrupt request, clear the PWMRIE bit before setting PWMEN bit.

Setting PWMEN bit for the first time after reset without first setting LDOK loads a prescaler divisor of one, a PWM value of $0000, and an unknown modulus. If the LDOK bit is not set after the PWMEN bit is cleared, then set (without a RESET) the value last loaded will be used in the PWM generated. If the deadtime registers are changed after PWMEN or OUTCTLn bits are set, an improper deadtime insertion will occur.

Initializing the deadtime registers after setting PWMEN or OUTCTLn can cause an improper deadtime insertion. However, the deadtime can never be shorter than the specified value.

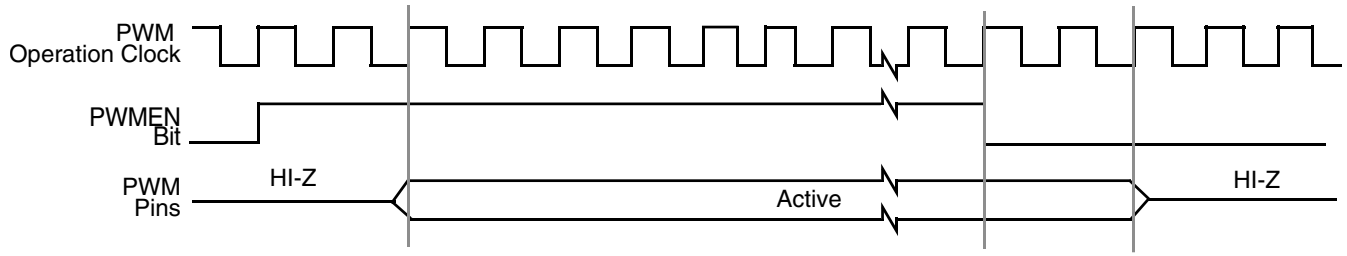


Figure 26-26. PWMEN and PWM pins in independent operation (OUTCTL0–5 = 0)

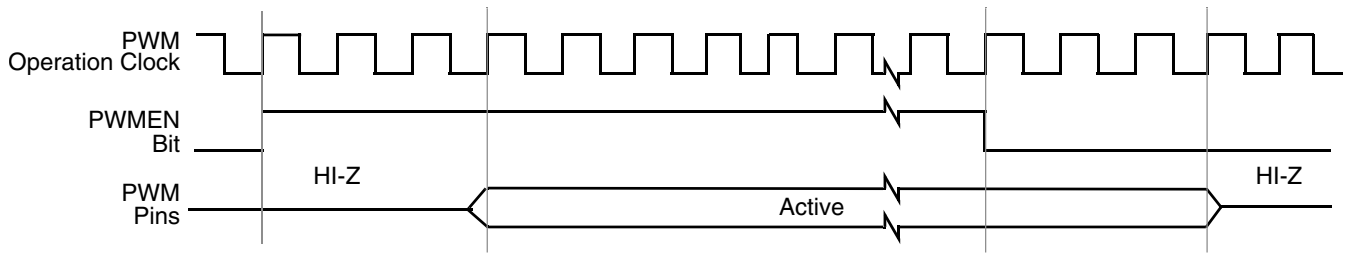


Figure 26-27. PWMEN and PWM pins in complement operation (OUTCTL0, 2, 4 = 0)

When the PWMEN bit is cleared:

- The PWMn pins will be in their inactive status unless OUTCTLn=1
- The PWM counter is cleared and does not count
- The PWM generator forces its outputs to zero
- The PWMF and pending interrupt requests are not cleared
- All fault circuitry remains active
- Software output control remains active if OUTCTLn=1
- Deadtime insertion continues during software output control

26.3.9 Fault protection

Fault protection can disable any combination of PWM pins. Faults are generated by either a 1 or 0, determined by the fault polarity control bits in the fault control (FCTRL) register on any of the FAULT pins. Each FAULT pin can be mapped arbitrarily to any of the PWM pins. When fault protection hardware disables PWM pins, the PWM generator continues to run, only the output pins are deactivated. The fault decoder disables PWM pins selected by the fault logic and the disable mapping register. Please see the following figure. Each bank of four bits in the disable mapping registers (DMAP1-2) controls the mapping for a single PWM pin. Please refer to the following table. The fault protection is enabled even when the PWM is not enabled; therefore, if a fault is latched in, it must be cleared prior to enabling the PWM to prevent an unexpected interrupt.

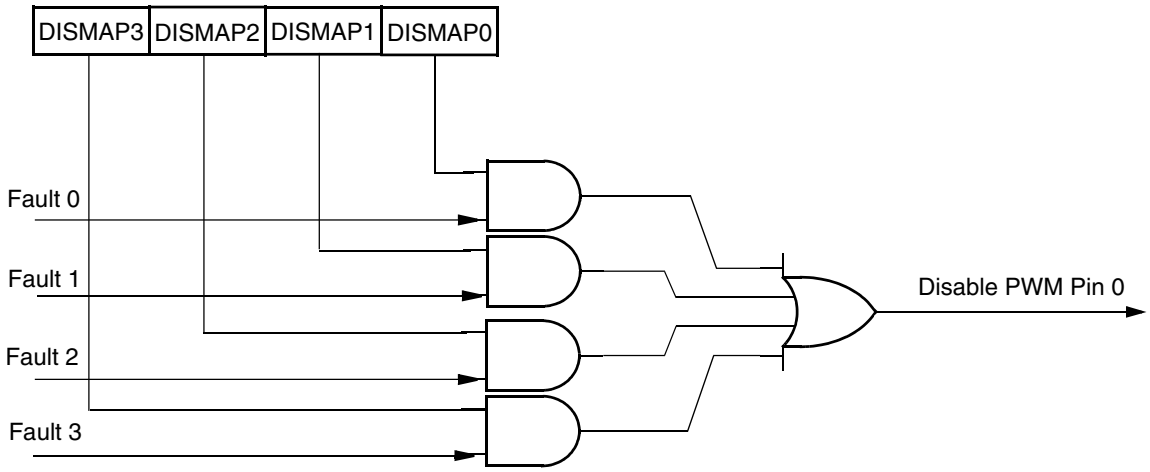


Figure 26-28. Fault decoder for PWM 0

Table 26-4. Fault mapping

PWM Pin	Controlling Register Bits
PWM0	DISMAP3–DISMAP0
PWM1	DISMAP7–DISMAP4
PWM2	DISMAP11–DISMAP8
PWM3	DISMAP15–DISMAP12
PWM4	DISMAP19–DISMAP16
PWM5	DISMAP23–DISMAP20

NOTE

For parts with less than four fault pins, the same controls apply. The unavailable DISMAP field bits should be set to zero. For example, if fault 3 is not available as an input, set DISMAP3=0.

26.3.9.1 Fault pin filter

Each fault pin has a filter to test for fault conditions. A fault input transition to a high state is not declared until the input is sampled high on two consecutive PWM operation clocks. Only then FFLAGn and FPINn are set. The FPINn bit will remain set until the fault input is detected low on two consecutive PWM operation clocks. Clear FFLAGn by writing a 1 to the corresponding fault acknowledge (FTACKn) bit. If the FIEn, FAULTn pin interrupt enable bit is set, the FFLAGn flag generates an interrupt request. The interrupt request latch remains set until one of the following actions occur:

- Software clears the FFLAGn flag by writing a 1 to the FTACKn bit
- Software clears the FIEn bit by writing a 0 to it
- A reset occurs

26.3.9.2 Automatic fault clearing

In automatic mode, when FMODEn is set, disabled PWM pins are enabled when the FAULTn pin returns to 0 and a new PWM half cycle begins. Please refer to the following figure. Clearing the FFLAGn flag does not affect disabled PWM pins when FMODEn is set.

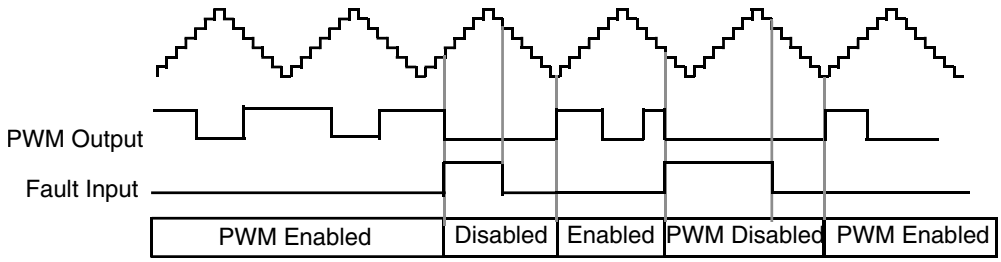


Figure 26-29. Automatic fault clearing

26.3.9.3 Manual fault clearing

In manual mode, the fault pins are grouped in pairs, each pair sharing common functionality. A fault condition on Fault pins 0 and 2 can be cleared by software clearing the corresponding FFLAG bit, allowing the PWM(s) to enable at the next PWM half cycle regardless of the logic level at the fault pin. The PWM outputs will remain enabled even if the logic level of the fault pin is still high. The fault pin must go low and then back high to register a new fault and disable the PWM outputs. Figure 1-30. A fault condition on fault pins 1 and 3 can be cleared only by software clearing corresponding FFLAGn bit, allowing the PWM(s) to enable if a logic low at the fault pin is detected at the start of the next PWM half cycle boundary. Please see Figure 1-31.

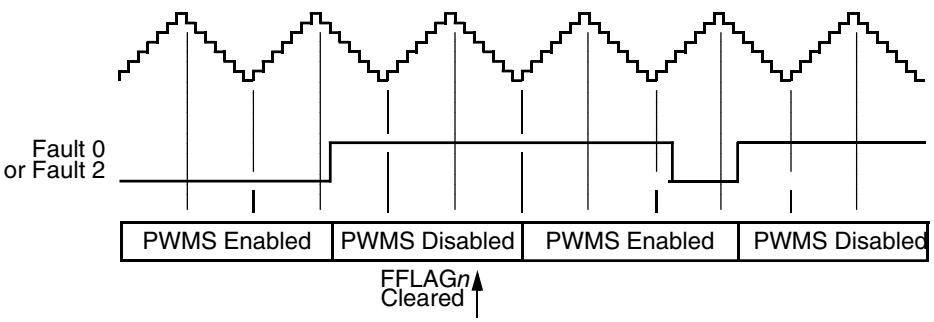


Figure 26-30. Manual fault clearing (example 1)

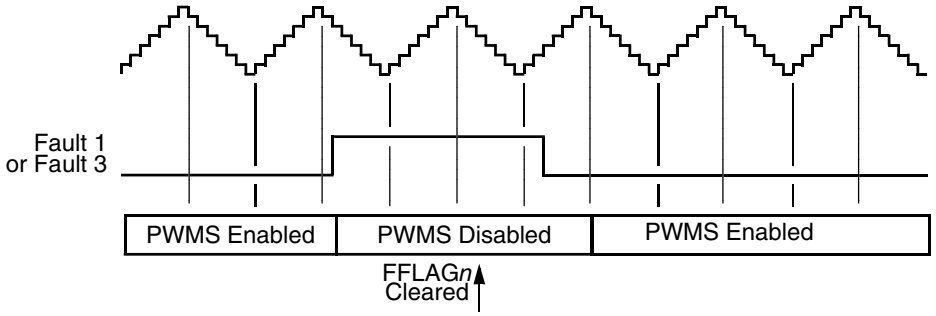


Figure 26-31. Manual fault clearing (example 2)

NOTE

PWM half-cycle boundaries occur at both the PWM cycle start and when the counter equals the modulus, so in edge-aligned operation full cycles and half cycles are equal.

Fault protection also applies during software output control when the OUTCTLn bits are set. Fault clearing still occurs at half PWM cycle boundaries while the PWM generator is engaged where PWMEN=1. However, the OUTn bits can also control the PWM pins while the PWM generator is off where PWMEN=0. Thus, fault clearing occurs at PWM operation clock cycles while the PWM generator is off and at the start of PWM cycles when the generator is engaged.

26.4 Memory Map and Register Descriptions

PWM memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
40	PWM Control Register: Low (PWM_CTRLL)	8	R/W	00h	26.4.1/506
41	PWM Control Register: High (PWM_CTRLH)	8	R/W	00h	26.4.2/507
42	PWM Fault Control Register: Low (PWM_FCTRLL)	8	R/W	00h	26.4.3/508
43	PWM Fault Control Register: High (PWM_FCTRLH)	8	R/W	00h	26.4.4/510
44	PWM Fault Status Acknowledge Register: Low (PWM_FLTACKL)	8	W	00h	26.4.5/510
45	PWM Fault Status Acknowledge Register: High (PWM_FLTACKH)	8	R	00h	26.4.6/511
46	PWM Output Control Register: Low (PWM_OUTL)	8	R/W	00h	26.4.7/513
47	PWM Output Control Register: High (PWM_OUTH)	8	R/W	00h	26.4.8/514
48	PWM Counter Register: Low (PWM_CNTRL)	8	R	00h	26.4.9/514

Table continues on the next page...

PWM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
49	PWM Counter Register: High (PWM_CNTRH)	8	R	00h	26.4.10/ 515
4A	PWM Counter Register: Low (PWM_CMODL)	8	R/W	00h	26.4.11/ 515
4B	PWM Counter Register: High (PWM_CMODH)	8	R/W	00h	26.4.12/ 516
4C	PWM Value Register: Low (PWM_VAL0L)	8	R/W	00h	26.4.13/ 516
4D	PWM Value Register: High (PWM_VAL0H)	8	R/W	00h	26.4.14/ 517
4E	PWM Value Register: Low (PWM_VAL1L)	8	R/W	00h	26.4.13/ 516
4F	PWM Value Register: High (PWM_VAL1H)	8	R/W	00h	26.4.14/ 517
50	PWM Value Register: Low (PWM_VAL2L)	8	R/W	00h	26.4.13/ 516
51	PWM Value Register: High (PWM_VAL2H)	8	R/W	00h	26.4.14/ 517
52	PWM Value Register: Low (PWM_VAL3L)	8	R/W	00h	26.4.13/ 516
53	PWM Value Register: High (PWM_VAL3H)	8	R/W	00h	26.4.14/ 517
54	PWM Value Register: Low (PWM_VAL4L)	8	R/W	00h	26.4.13/ 516
55	PWM Value Register: High (PWM_VAL4H)	8	R/W	00h	26.4.14/ 517
56	PWM Value Register: Low (PWM_VAL5L)	8	R/W	00h	26.4.13/ 516
57	PWM Value Register: High (PWM_VAL5H)	8	R/W	00h	26.4.14/ 517
58	PWM Deadtime Register: Low (PWM_DTIM0L)	8	R/W	FFh	26.4.15/ 518
59	PWM Deadtime Register: High (PWM_DTIM0H)	8	R/W	0Fh	26.4.16/ 518
5A	PWM Deadtime Register: Low (PWM_DTIM1L)	8	R/W	FFh	26.4.15/ 518
5B	PWM Deadtime Register: High (PWM_DTIM1H)	8	R/W	0Fh	26.4.16/ 518
5C	PWM Disable Mapping Registers 1: Low (PWM_DMAP1L)	8	R/W	FFh	26.4.17/ 519
5D	PWM Disable Mapping Registers 1: High (PWM_DMAP1H)	8	R/W	FFh	26.4.18/ 520
5E	PWM Disable Mapping Registers 2: Low (PWM_DMAP2L)	8	R/W	FFh	26.4.19/ 520

Table continues on the next page...

PWM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
1820	PWM Configure Register: Low (PWM_CNFGL)	8	R/W	00h	26.4.20/ 520
1821	PWM Configure Register: High (PWM_CNFGH)	8	R/W	00h	26.4.21/ 521
1822	PWM Channel Control Register: Low (PWM_CCTRLL)	8	R/W	00h	26.4.22/ 522
1823	PWM Channel Control Register: High (PWM_CCTRLH)	8	R/W	40h	26.4.23/ 524
1826	PWM Pulse Edge Control Register: Low (PWM_PECTRLL)	8	R/W	03h	26.4.24/ 524
1829	PWM Compare Invert Register: High (PWM_CINVH)	8	R/W	00h	26.4.25/ 525

26.4.1 PWM Control Register: Low (PWM_CTRLL)

Address: 40h base + 0h offset = 40h

Bit	7	6	5	4	3	2	1	0
Read	PRSC		PWMRIE	PWMF	0		LDOK	PWMEN
Write								
Reset	0	0	0	0	0	0	0	0

PWM_CTRLL field descriptions

Field	Description
7–6 PRSC	Prescaler These buffered read/write bits select the PWM clock frequency illustrated in the table below. **NOTE:** Reading the PRSCn bits reads the buffered values and not necessarily the values currently in effect. The PRSCn bits take effect at the beginning of the next PWM cycle and only when the load okay bit, LDOK, is set. For this device, the f_{IPBus} is the high speed clock HSCLK. 00 f_{IPBus} 01 $f_{IPBus}/2$ 10 $f_{IPBus}/4$ 11 $f_{IPBus}/8$
5 PWMRIE	PWM Reload Interrupt Enable This read/write bit enables the PWMF flag to generate interrupt requests. Reset clears PWMRIE. 0 PWMF interrupt requests disabled. 1 PWMF interrupt requests enabled.
4 PWMF	PWM Reload Flag

Table continues on the next page...

PWM_CTRLL field descriptions (continued)

Field	Description
	This read/write flag is set at the beginning of every reload cycle regardless of the state of the LDOK bit. Clear PWMF by reading PWM control register with PWMF set and then writing a zero to the PWMF bit. If another reload occurs before the clearing sequence is complete, writing zero to PWMF has no effect. Reset clears PWMF. **NOTE:** Clearing PWMF clears pending PWMF interrupt requests. 0 No new reload cycle since last PWMF clearing. 1 New reload cycle since last PWMF clearing.
3–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 LDOK	Load Okay This read/write bit loads the prescaler bits of CTRL and the entire PMMCM and VAL registers into a set of buffers. The buffered prescaler divisor, PWM counter modulus value, and PWM pulse width take effect at the next PWM reload. Set LDOK by writing a one to it. LDOK is automatically cleared after the new values are loaded, or can be manually cleared before a reload by writing a zero to it. Reset clears LDOK. 0 Do not load new modulus, prescaler, and PWM values. 1 Load prescaler, modulus, and PWM values.
0 PWMEN	PWM Enable This read/write bit enables the PWM generator and the PWM pins. When PWMEN equals zero, the PWM pins are in their inactive states unless OUTCTLn equals one. A reset clears PWMEN. 0 PWM generator and PWM pins disabled unless OUTCTL = 1. 1 PWM generator and PWM pins enabled.

26.4.2 PWM Control Register: High (PWM_CTRLH)

Address: 40h base + 1h offset = 41h

Bit	7	6	5	4	3	2	1	0
Read	LDFQ				HALF	0		
Write								
Reset	0	0	0	0	0	0	0	0

PWM_CTRLH field descriptions

Field	Description
7–4 LDFQ	Load Frequency Bits These buffered read/write bits select the PWM load frequency according to the table below. Reset clears the LDFQ bits, selecting loading every PWM opportunity. A PWM opportunity is determined by the half bit. **NOTE:** The LDFQn bits take effect when the current load cycle is complete, regardless of the state of the load okay bit, LDOK. Reading the LDFQn bits reads the buffered values and not necessarily the values currently in effect. 0000 Every PWM opportunity 0001 Every 2 PWM opportunities

Table continues on the next page...

PWM_CTRLH field descriptions (continued)

Field	Description
	0010 Every 3 PWM opportunities 0011 Every 4 PWM opportunities 0100 Every 5 PWM opportunity 0101 Every 6 PWM opportunities 0110 Every 7 PWM opportunities 0111 Every 8 PWM opportunities 1000 Every 9 PWM opportunity 1001 Every 10 PWM opportunities 1010 Every 11 PWM opportunities 1011 Every 12 PWM opportunities 1100 Every 13 PWM opportunity 1101 Every 14 PWM opportunities 1110 Every 15 PWM opportunities 1111 Every 16 PWM opportunities
3 HALF	Half Cycle Reload This read/write bit enables half-cycle reloads in center-aligned PWM mode. This bit has no effect on edge-aligned PWMs. 0 Half-cycle reloads disabled. 1 Half-cycle reloads enabled.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

26.4.3 PWM Fault Control Register: Low (PWM_FCTRLL)

Address: 40h base + 2h offset = 42h

Bit	7	6	5	4	3	2	1	0
Read Write	FIE3	FMODE3	FIE2	FMODE2	FIE1	FMODE1	FIE0	FMODE0
Reset	0	0	0	0	0	0	0	0

PWM_FCTRLL field descriptions

Field	Description
7 FIE3	FAULT3 Pin Interrupt Enable This read/write bit enables interrupt requests generated by the filtered FAULT3 pin. A reset clears FIE3. **NOTE:** The fault protection circuit is independent of the FIE3 bits and is always active. If a fault is detected, the PWM pins are disabled according to the PWM disable mapping register. 0 FAULT3 interrupt requests disabled 1 FAULT3 interrupt requests enabled
6 FMODE3	FAULT3 Pin Clearing Mode This read/write bit selects automatic or manual clearing of FAULT3 pin faults. A reset clears FMODE3.

Table continues on the next page...

PWM_FCTRLL field descriptions (continued)

Field	Description
	0 Manual fault clearing of FAULT3 pin faults. 1 Automatic fault clearing of FAULT3 pin faults.
5 FIE2	FAULT2 Pin Interrupt Enable This read/write bit enables interrupt requests generated by the filtered FAULT2 pin. A reset clears FIE2. **NOTE:** The fault protection circuit is independent of the FIE2 bits and is always active. If a fault is detected, the PWM pins are disabled according to the PWM disable mapping register. 0 FAULT2 interrupt requests disabled 1 FAULT2 interrupt requests enabled
4 FMODE2	FAULT2 Pin Clearing Mode This read/write bit selects automatic or manual clearing of FAULT2 pin faults. A reset clears FMODE2. 0 Manual fault clearing of FAULT2 pin faults. 1 Automatic fault clearing of FAULT2 pin faults.
3 FIE1	FAULT1 Pin Interrupt Enable This read/write bit enables interrupt requests generated by the filtered FAULT1 pin. A reset clears FIE1. **NOTE:** The fault protection circuit is independent of the FIE1 bits and is always active. If a fault is detected, the PWM pins are disabled according to the PWM disable mapping register. 0 FAULT1 interrupt requests disabled 1 FAULT1 interrupt requests enabled
2 FMODE1	FAULT1 Pin Clearing Mode This read/write bit selects automatic or manual clearing of FAULT1 pin faults. A reset clears FMODE1. 0 Manual fault clearing of FAULT1 pin faults. 1 Automatic fault clearing of FAULT1 pin faults.
1 FIE0	FAULT0 Pin Interrupt Enable This read/write bit enables interrupt requests generated by the filtered FAULT0 pin. A reset clears FIE0. **NOTE:** The fault protection circuit is independent of the FIE0 bits and is always active. If a fault is detected, the PWM pins are disabled according to the PWM disable mapping register. 0 FAULT0 interrupt requests disabled 1 FAULT0 interrupt requests enabled
0 FMODE0	FAULT0 Pin Clearing Mode This read/write bit selects automatic or manual clearing of FAULT0 pin faults. A reset clears FMODE0. 0 Manual fault clearing of FAULT0 pin faults. 1 Automatic fault clearing of FAULT0 pin faults.

26.4.4 PWM Fault Control Register: High (PWM_FCTRLH)

Address: 40h base + 3h offset = 43h

Bit	7	6	5	4	3	2	1	0
Read	0				FPOL			
Write								
Reset	0	0	0	0	0	0	0	0

PWM_FCTRLH field descriptions

Field	Description
7–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
FPOL	FAULTn Polarity Control These read/write bits control the polarity of the FAULTn pin inputs. A reset clears FPOLn. FPOL2 is also used to control the polarity of the external sync input and output. 0 A 1 on FAULTn indicates a fault condition 1 A 0 on FAULTn indicates a fault condition

26.4.5 PWM Fault Status Acknowledge Register: Low (PWM_FLTACKL)

NOTE

After enabling clock to PWM, but before enabling any PWM interrupt, clear all flags in FLTACK.

Address: 40h base + 4h offset = 44h

Bit	7	6	5	4	3	2	1	0
Read	0		0		0		0	
Write		FTACK3		FTACK2		FTACK1		FTACK0
Reset	0	0	0	0	0	0	0	0

PWM_FLTACKL field descriptions

Field	Description
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6 FTACK3	FAULT3 Pin Acknowledge

Table continues on the next page...

PWM_FLTACKL field descriptions (continued)

Field	Description
	Writing a one to FTACK3 clears FFLAG3. Writing a zero has no effect. Reset clears FTACK3. The fault protection is enabled even when the PWM is not enabled; therefore, a fault is latched in, requiring it to be cleared in order to prevent an interrupt when the PWM is enabled.
5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
4 FTACK2	FAULT2 Pin Acknowledge Writing a one to FTACK2 clears FFLAG2. Writing a zero has no effect. Reset clears FTACK2. The fault protection is enabled even when the PWM is not enabled; therefore, a fault is latched in, requiring it to be cleared in order to prevent an interrupt when the PWM is enabled.
3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 FTACK1	FAULT1 Pin Acknowledge Writing a one to FTACK1 clears FFLAG1. Writing a zero has no effect. Reset clears FTACK1. The fault protection is enabled even when the PWM is not enabled; therefore, a fault is latched in, requiring it to be cleared in order to prevent an interrupt when the PWM is enabled.
1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 FTACK0	FAULT0 Pin Acknowledge Writing a one to FTACK0 clears FFLAG0 Writing a zero has no effect. Reset clears FTACK0. The fault protection is enabled even when the PWM is not enabled; therefore, a fault is latched in, requiring it to be cleared in order to prevent an interrupt when the PWM is enabled.

26.4.6 PWM Fault Status Acknowledge Register: High (PWM_FLTACKH)

NOTE

After enabling clock to PWM, but before enabling any PWM interrupt, clear all flags in FLTACK.

Address: 40h base + 5h offset = 45h

Bit	7	6	5	4	3	2	1	0
Read	FPIN3	FFLAG3	FPIN2	FFLAG2	FPIN1	FFLAG1	FPIN0	FFLAG0
Write								
Reset	0	0	0	0	0	0	0	0

PWM_FLTACKH field descriptions

Field	Description
7 FPIN3	FAULT3 Pin This read-only bit reflects the current state of the filtered FAULT3 pin. A reset has no effect on FPIN3.

Table continues on the next page...

PWM_FLTACKH field descriptions (continued)

Field	Description
	0 an invalid fault state on the FAULT3 pin 1 a valid fault state on the FAULT3 pin
6 FFLAG3	FAULT3 Flag This read-only flag is set within two CPU cycles after a rising edge on the filtered FAULT3 pin. Clear FFLAG3 by writing a one to the FTACK3 bit in this register (FLTACK). A reset clears FFLAG3. 0 No fault on the FAULT3 pin 1 Fault on the FAULT3 pin
5 FPIN2	FAULT2 Pin This read-only bit reflects the current state of the filtered FAULT2 pin. A reset has no effect on FPIN2. 0 an invalid fault state on the FAULT2 pin 1 a valid fault state on the FAULT2 pin
4 FFLAG2	FAULT2 Flag This read-only flag is set within two CPU cycles after a rising edge on the filtered FAULT2 pin. Clear FFLAG2 by writing a one to the FTACK2 bit in this register (FLTACK). A reset clears FFLAG2. 0 No fault on the FAULT2 pin 1 Fault on the FAULT2 pin
3 FPIN1	FAULT1 Pin This read-only bit reflects the current state of the filtered FAULT1 pin. A reset has no effect on FPIN1. 0 an invalid fault state on the FAULT1 pin 1 a valid fault state on the FAULT1 pin
2 FFLAG1	FAULT1 Flag This read-only flag is set within two CPU cycles after a rising edge on the filtered FAULT1 pin. Clear FFLAG1 by writing a one to the FTACK1 bit in this register (FLTACK). A reset clears FFLAG1. 0 No fault on the FAULT1 pin 1 Fault on the FAULT1 pin
1 FPIN0	FAULT0 Pin This read-only bit reflects the current state of the filtered FAULT0 pin. A reset has no effect on FPIN0. 0 an invalid fault state on the FAULT0 pin 1 a valid fault state on the FAULT0 pin
0 FFLAG0	FAULT0 Flag This read-only flag is set within two CPU cycles after a rising edge on the filtered FAULT0 pin. Clear FFLAG0 by writing a one to the FTACK0 bit in this register (FLTACK). A reset clears FFLAG0. 0 No fault on the FAULT0 pin 1 Fault on the FAULT0 pin

26.4.7 PWM Output Control Register: Low (PWM_OUTL)

Address: 40h base + 6h offset = 46h

Bit	7	6	5	4	3	2	1	0
Read	0		OUT5	OUT4	OUT3	OUT2	OUT1	OUT0
Write								
Reset	0	0	0	0	0	0	0	0

PWM_OUTL field descriptions

Field	Description
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5 OUT5	Output 5 When the corresponding OUTCTL bit is set, these read/write bits control the PWM pins. 0 PWM5 is inactive 1 PWM5 is complement of PWM 4 (complementary channel operation); PWM5 is active (independent channel operation)
4 OUT4	Output 4 When the corresponding OUTCTL bit is set, these read/write bits control the PWM pins. 0 PWM4 is inactive 1 PWM4 is active
3 OUT3	Output 3 When the corresponding OUTCTL bit is set, these read/write bits control the PWM pins. 0 PWM3 is inactive 1 PWM3 is complement of PWM2 (complementary channel operation); PWM3 is active (independent channel operation)
2 OUT2	Output 2 When the corresponding OUTCTL bit is set, these read/write bits control the PWM pins. 0 PWM2 is inactive 1 PWM2 is active
1 OUT1	Output 1 When the corresponding OUTCTL bit is set, these read/write bits control the PWM pins. 0 PWM1 is inactive 1 PWM1 is complement of PWM0 (complementary channel operation); PWM1 is active (independent channel operation)
0 OUT0	Output 0 When the corresponding OUTCTL bit is set, these read/write bits control the PWM pins. 0 PWM0 is inactive 1 PWM0 is active

26.4.8 PWM Output Control Register: High (PWM_OUTH)

Address: 40h base + 7h offset = 47h

Bit	7	6	5	4	3	2	1	0
Read	PADEN	0	OUTCTL					
Write								
Reset	0	0	0	0	0	0	0	0

PWM_OUTH field descriptions

Field	Description
7 PADEN	Output Pad Enable The PWMn output pads can be enabled or disabled by setting the PAD_EN bit. The power-up default has the pads disabled. This bit does not affect the functionality of the PWM, so the PWM module can be energized with the output pads disabled. This enable is to power-up with a safe default value for the PWM drivers. 0 Output pads disabled 1 Output pads enabled
6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
OUTCTL	Output Control Enables These read/write bits enable software control of their corresponding PWM pin. When OUTCTLn is set, the OUTn bit activates and deactivates the PWMn output or the CINVH register is used to select an alternate control of the PWM outputs. A reset clears the OUTCTL bits. 0 Software control disabled (normal PWM operation) 1 Software control enabled

26.4.9 PWM Counter Register: Low (PWM_CNTRL)

This read-only register, together with CNTRH, displays the state of the 15-bit PWM counter. Reading the CNTRL causes an internal hold register to be updated with the CNTRH value. Reading the CNTRH reads this internal hold register. Always read the lower byte before reading the upper byte in order to guarantee a coherent 15-bit value is read.

Address: 40h base + 8h offset = 48h

Bit	7	6	5	4	3	2	1	0
Read	CNTR7_0							
Write								
Reset	0	0	0	0	0	0	0	0

PWM_CNTRL field descriptions

Field	Description
CNTR7_0	Counter 7:0

26.4.10 PWM Counter Register: High (PWM_CNTRH)

This read-only register, together with CNTRL, displays the state of the 15-bit PWM counter. Reserved bit 7, cannot be modified. It is read as zero. Reading the CNTRL causes an internal hold register to be updated with the CNTRH value. Reading the CNTRH reads this internal hold register. Always read the lower byte before reading the upper byte in order to guarantee a coherent 15-bit value is read.

Address: 40h base + 9h offset = 49h

Bit	7	6	5	4	3	2	1	0
Read	0	CNTR14_8						
Write								
Reset	0	0	0	0	0	0	0	0

PWM_CNTRH field descriptions

Field	Description
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CNTR14_8	Counter 14:8

26.4.11 PWM Counter Register: Low (PWM_CMODL)

The 15-bit unsigned value written to this buffered, read/write register defines the PWM period in PWM clock periods. Reserved bit 15 cannot be modified. It is read as zero.

NOTE

The PWM counter modulo register is buffered. The value written does not take effect until the LDOK bit is set and the next PWM load cycle begins. Reading CMOD reads the value in a buffer. It is not necessarily the value the PWM generator is currently using.

Address: 40h base + Ah offset = 4Ah

Bit	7	6	5	4	3	2	1	0
Read Write	CMOD7_0							
Reset	0	0	0	0	0	0	0	0

PWM_CMODL field descriptions

Field	Description
CMOD7_0	Counter Modulo 7:0

26.4.12 PWM Counter Register: High (PWM_CMODH)

The 15-bit unsigned value written to this buffered, read/write register defines the PWM period in PWM clock periods. Reserved bit 15 cannot be modified. It is read as zero.

NOTE

The PWM counter modulo register is buffered. The value written does not take effect until the LDOK bit is set and the next PWM load cycle begins. Reading CMOD reads the value in a buffer. It is not necessarily the value the PWM generator is currently using.

Address: 40h base + Bh offset = 4Bh

Bit	7	6	5	4	3	2	1	0
Read	0	CMOD14_8						
Write								
Reset	0	0	0	0	0	0	0	0

PWM_CMODH field descriptions

Field	Description
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CMOD14_8	Counter Modulo 14:8

26.4.13 PWM Value Register: Low (PWM_VAL*n*L)

The 16-bit signed value in these buffered, read/write registers defines the PWM pulse width in PWM clock periods for each PWM output channel.

NOTE

The PWM value registers are buffered. The value written does not take effect until the LDOK bit is set and the next PWM load cycle begins. Reading VALn reads the value in a buffer and not necessarily the value the PWM generator is currently using.

A PWM value less than or equal to zero deactivates the PWM output for the entire PWM period. A PWM value greater than, or equal to the modulus, activates the PWM output for the entire PWM period.

NOTE

The terms activate and deactivate refer to the high and low logic states of the PWM outputs.

Address: 40h base + Ch offset + (2d × i), where i=0d to 5d

Bit	7	6	5	4	3	2	1	0
Read Write	PMVAL7_0							
Reset	0	0	0	0	0	0	0	0

PWM_VAL*n*L field descriptions

Field	Description
PMVAL7_0	PWM Pulse Width Value7:0

26.4.14 PWM Value Register: High (PWM_VAL*n*H)

The 16-bit signed value in these buffered, read/write registers defines the PWM pulse width in PWM clock periods for each PWM output channel.

NOTE

The PWM value registers are buffered. The value written does not take effect until the LDOK bit is set and the next PWM load cycle begins. Reading VALn reads the value in a buffer and not necessarily the value the PWM generator is currently using.

A PWM value less than or equal to zero deactivates the PWM output for the entire PWM period. A PWM value greater than, or equal to the modulus, activates the PWM output for the entire PWM period.

NOTE

The terms activate and deactivate refer to the high and low logic states of the PWM outputs.

Address: 40h base + Dh offset + (2d × i), where i=0d to 5d

Bit	7	6	5	4	3	2	1	0
Read Write	PMVAL15_8							
Reset	0	0	0	0	0	0	0	0

PWM_VAL*n*H field descriptions

Field	Description
PMVAL15_8	PWM Pulse Width Value 15:8

26.4.15 PWM Deadtime Register: Low (PWM_DTIM*n*L)

Deadtime operation is only applicable to complementary channel operation. The 12-bit value written to this write-protected registers is in terms of PWM clock cycles. Reset sets the PWM deadtime register to a default value of 0x0FFF, selecting a deadtime of 4096-PWM clock cycles minus one PWM clock cycle. This register is write protected after the WP bit in the PWM configuration register is set. Reserved bits 15–12 cannot be modified. They are read as zero.

NOTE

Deadtime is affected by changes to the prescaler value. The deadtime duration is determined as follows: DT = P × PWMDT – 1, where DT is deadtime, P is the prescaler value, PWMDT is the programmed value of dead time. For example: if the prescaler is programmed for a divide-by-two and PWMDT is set to five, then P = 2 and the deadtime value is equal to DT = 2 × 5 – 1 = 9 PWM clock cycles. A special case exists when the P = 1, DT = PWMDT

The PWMDT field is used to control the deadtime during transitions of the even PWM output.

Address: 40h base + 18h offset + (2d × i), where i=0d to 1d

Bit	7	6	5	4	3	2	1	0
Read Write	PWMDT7_0							
Reset	1	1	1	1	1	1	1	1

PWM_DTIM*n*L field descriptions

Field	Description
PWMDT7_0	PWM Pulse Width Value7:0

26.4.16 PWM Deadtime Register: High (PWM_DTIM*n*H)

Deadtime operation is only applicable to complementary channel operation. The 12-bit value written to this write-protected registers is in terms of PWM clock cycles. Reset sets the PWM deadtime register to a default value of 0x0FFF, selecting a deadtime of 4096-PWM clock cycles minus one PWM clock cycle. This register is write protected after the WP bit in the PWM configuration register is set. Reserved bits 15–12 cannot be modified. They are read as zero.

NOTE

Deadtime is affected by changes to the prescaler value. The deadtime duration is determined as follows: DT = P × PWMDT – 1, where DT is deadtime, P is the prescaler value, PWMDT is the programmed value of dead time. For example: if the prescaler is programmed for a divide-by-two and PWMDT is set to five, then P = 2 and the deadtime value is equal to DT = 2 × 5 – 1 = 9 PWM clock cycles. A special case exists when the P = 1, DT = PWMDT

The PWMDT field is used to control the deadtime during transitions of the even PWM output.

Address: 40h base + 19h offset + (2d × i), where i=0d to 1d

Bit	7	6	5	4	3	2	1	0
Read	0				PWMDT11_8			
Write								
Reset	0	0	0	0	1	1	1	1

PWM_DTIM*n*H field descriptions

Field	Description
7–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
PWMDT11_8	PWM Pulse Width Value7:0

26.4.17 PWM Disable Mapping Registers 1: Low (PWM_DMAP1L)

These write-protectable registers determine which PWM pins are affected by the fault protection inputs. Reset sets all of the bits used in the PWM disable mapping registers. These registers are write protected after the WP bit in the PWM configure register is set. Reserved bits 15-8 in the DMAP2 register cannot be modified. The bits are read as zero.

Address: 40h base + 1Ch offset = 5Ch

Bit	7	6	5	4	3	2	1	0
Read Write	DISMAP7_0							
Reset	1	1	1	1	1	1	1	1

PWM_DMAP1L field descriptions

Field	Description
DISMAP7_0	PWM Disable Mapping7:0

26.4.18 PWM Disable Mapping Registers 1: High (PWM_DMAP1H)

These write-protectable registers determine which PWM pins are affected by the fault protection inputs. Reset sets all of the bits used in the PWM disable mapping registers. These registers are write protected after the WP bit in the PWM configure register is set. Reserved bits 15-8 in the DMAP2 register cannot be modified. The bits are read as zero.

Address: 40h base + 1Dh offset = 5Dh

Bit	7	6	5	4	3	2	1	0
Read Write	DISMAP15_8							
Reset	1	1	1	1	1	1	1	1

PWM_DMAP1H field descriptions

Field	Description
DISMAP15_8	PWM Disable Mapping 15:8

26.4.19 PWM Disable Mapping Registers 2: Low (PWM_DMAP2L)

These write-protectable registers determine which PWM pins are affected by the fault protection inputs. Reset sets all of the bits used in the PWM disable mapping registers. These registers are write protected after the WP bit in the PWM configure register is set. Reserved bits 15-8 in the DMAP2 register cannot be modified. The bits are read as zero.

Address: 40h base + 1Eh offset = 5Eh

Bit	7	6	5	4	3	2	1	0
Read Write	DISMAP23_16							
Reset	1	1	1	1	1	1	1	1

PWM_DMAP2L field descriptions

Field	Description
DISMAP23_16	PWM Disable Mapping 23:16

26.4.20 PWM Configure Register: Low (PWM_CNFGL)

Address: 40h base + 17E0h offset = 1820h

Bit	7	6	5	4	3	2	1	0
Read	0	BOTNEG			INDEP			WP
Write								
Reset	0	0	0	0	0	0	0	0

PWM_CNFGL field descriptions

Field	Description
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6–4 BOTNEG	Bottom-side PWM Polarity Bit This write-protectable bit determines the polarity for the bottom-side PWMs. **NOTE:** Each pair of PWM channels can be configured: channel zero to one, channel two to three, and channel four to five. 0 Positive bottom-side polarity 1 Negative bottom-side polarity
3–1 INDEP	Independent or Complimentary Pair Operation This write-protectable bit determines if the motor control PWM channels are independent PWMs or complementary PWM pairs. **NOTE:** Each pair of PWM channels can be configured: channel zero to one, channel two to three, and channel four to five. 0 Complementary PWM pair 1 Independent PWMs
0 WP	Write Protect This write-protectable bit enables write protection to be used for all write-protectable registers. While clear, WP allows write-protectable registers to be written. When set, WP prevents any further writes to write-protectable registers. After it is set, WP can be cleared only by a reset. Write-protectable registers include CINVH, DMAP1L, DMAP1H, DMAP2L, DTIMnL, DTIMnH, CNFGL, CNFGH, and the CCTRLH[ENHA]. The CCTRLL[VLMODE], CCTRLL[SWP01], CCTRLL[SWP23], and CCTRLL[SWP45] bits are protected when the CCTRLH[ENHA] bit is set to zero. CCTRLH[ENHA] is in turn protected by setting CNFGL[WP]. **NOTE:** The write to the CNFG register that sets the WP bit is the last write accepted to that register until the part is reset. 0 Write-protectable registers may written to 1 Write-protectable registers are read only

26.4.21 PWM Configure Register: High (PWM_CNFGH)

Address: 40h base + 17E1h offset = 1821h

Bit	7	6	5	4	3	2	1	0
Read	0	DBGEN	WAITEN	EDG	0	TOPNEG		
Write								
Reset	0	0	0	0	0	0	0	0

PWM_CNFGH field descriptions

Field	Description
7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

Table continues on the next page...

PWM_CNFGH field descriptions (continued)

Field	Description
6 DBGEN	Debug Enable When set to one, the PWM continues to run while the chip is in EOnCE debug mode. If the device enters EOnCE mode and this bit is zero, then the PWM outputs are switched to their inactive state until EOnCE mode is exited. At that point the PWM pins resume operation as programmed in the PWM registers. For certain types of motors (such as 3-phase AC), it is imperative that this bit be left in its default state (in which the PWM is disabled in EOnCE mode). Failure to do so could result in damaging the motor. For other types of motors (example: DC motors), this bit might safely be set to one, enabling the PWM in debug mode. The key point is PWM parameter updates do not occur in debug mode. Any motors requiring such updates should be disabled during EOnCE mode. If in doubt, leave this bit cleared to zero.
5 WAITEN	Wait Enable When set to one, the PWM continues to run while the chip is in wait mode. In this mode, the peripheral clock continues to run but the DSC clock does not. If the device enters wait mode and this bit is zero, then the PWM outputs are switched to their inactive state until wait mode is exited. At that point the PWM pins resume operation as programmed in the PWM registers. For certain types of motors (such as 3-phase AC), it is imperative that this bit be left in its default state (in which the PWM is disabled in wait mode). Failure to do so could result in damaging the motor. For other types of motors (example: DC motors), this bit might safely be set to one, enabling the PWM in wait mode. The key point is PWM parameter updates do not occur in this mode. Any motors requiring such updates should be disabled during wait mode. If in doubt, leave this bit set to zero.
4 EDG	Edge-Aligned or Center-Aligned PWMs This write-protectable bit determines whether all PWM channels use edge-aligned or center-aligned waveforms. 0 Center-aligned PWMs 1 Edge-aligned PWMs
3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
TOPNEG	Top-side PWM Polarity Bit This write-protectable bit determines the polarity for the top-side PWMs. **NOTE:** Each pair of PWM channels can be configured: channel zero to one, channel two to three, and channel four to five. 0 Positive top-side polarity 1 Negative top-side polarity

26.4.22 PWM Channel Control Register: Low (PWM_CCTRLL)

This write-protectable register contains the configuration bits that determine PWM modes of operation as detailed below. The ENHA bit cannot be modified after the WP bit in the CNFG register is set. ENHA in turn provides protection for the VLMODE[1:0], SWP45, SWP23 and SWP01 bits. The Mask bits are not write protectable.

NOTE

the SWAP bit only inverts the complementary signal and has no effect in independent mode.

Address: 40h base + 17E2h offset = 1822h

Bit	7	6	5	4	3	2	1	0
Read	0		VLMODE		0	SWP45	SWP23	SWP01
Write								
Reset	0	0	0	0	0	0	0	0

PWM_CCTRLL field descriptions

Field	Description
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5–4 VLMODE	Value Register Load Mode These two bits determine the way the value registers are being loaded. These bits are write protected when ENHA is zero. 00 Each value register is accessed independently 01 Writing to value register zero also writes to value registers one to five 10 Writing to value register zero also writes to value registers one to three 11 Reserved
3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 SWP45	Swap 4 and 5 This bit is write protected when ENHA is zero. 0 No swap 1 Channel four and channel five are swapped
1 SWP23	Swap 2 and 3 This bit is write protected when ENHA is zero. 0 No swap 1 Channel two and channel three are swapped
0 SWP01	Swap 0 and 1 This bit is write protected when ENHA is zero. 0 No swap 1 Channel zero and channel one are swapped

26.4.23 PWM Channel Control Register: High (PWM_CCTRLH)

This write-protectable register contains the configuration bits that determine PWM modes of operation as detailed below. The ENHA bit cannot be modified after the WP bit in the CNFG register is set. ENHA in turn provides protection for the VLMODE[1:0], SWP45, SWP23 and SWP01 bits. The Mask bits are not write protectable.

Address: 40h base + 17E3h offset = 1823h

Bit	7	6	5	4	3	2	1	0
Read	ENHA	1	MSK					
Write								
Reset	0	1	0	0	0	0	0	0

PWM_CCTRLH field descriptions

Field	Description
7 ENHA	Enable Hardware Acceleration This bit enables writing to the VLMODE[1:0], SWP45, SWP23, and SWP01 bits. The bit is write protected by the CNFG register WP bit. 0 Disable writing to VLMODE[1:0], SWP45, SWP23, and SWP01 bits 1 Enable writing to VLMODE[1:0], SWP45, SWP23, and SWP01 bits
6 Reserved	This field is reserved. This read-only field is reserved and always has the value 1.
MSK	Mask These six bits determine the mask for each of the PWM logical channels. 0 Unmasked 1 Masked, channel set to a value of zero percent duty cycle

26.4.24 PWM Pulse Edge Control Register: Low (PWM_PECTRLL)

This register is used to control PWM pulse generation for various applications, such as a power-supply phase-shifting application.

The PECn bits only apply in edge-aligned operation during complementary mode. These control bits allow the PWM pulses generated by both the odd and even VAL regs to be XORed together prior to the complementary logic and deadtime insertion.

NOTE

The PECn bits are buffered. The value written does not take effect until the LDOK bit is set and the next PWM load cycle begins. Reading PECn reads the value in a buffer and not necessarily the value the PWM generator is currently using.

Address: 40h base + 17E6h offset = 1826h

Bit	7	6	5	4	3	2	1	0
Read	0		PEC2	PEC1	PEC0	1		
Write								
Reset	0	0	0	0	0	0	1	1

PWM_PECTRLL field descriptions

Field	Description
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
5 PEC2	Pulse Edge Control 2 This bit controls PWM4/PWM5 pair. 0 Normal operation. 1 Allow one of VAL4 and VAL5 to activate the PWM pulse and the other to deactivate the pulse.
4 PEC1	Pulse Edge Control 1 This bit controls PWM2/PWM3 pair. 0 Normal operation. 1 Allow one of VAL2 and VAL3 to activate the PWM pulse and the other to deactivate the pulse.
3 PEC0	Pulse Edge Control 0 This bit controls PWM0/PWM1 pair. 0 Normal operation. 1 Allow one of VAL0 and VAL1 to activate the PWM pulse and the other to deactivate the pulse.
Reserved	This field is reserved. This read-only field is reserved and always has the value 1.

26.4.25 PWM Compare Invert Register: High (PWM_CINVH)

This register is affected by the WP bit in the CNFG register. It can only be written when that bit is clear.

Address: 40h base + 17E9h offset = 1829h

Bit	7	6	5	4	3	2	1	0
Read	0		CINV5	CINV4	CINV3	CINV2	CINV1	CINV0
Write								
Reset	0	0	0	0	0	0	0	0

PWM_CINVH field descriptions

Field	Description
7–6 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

Table continues on the next page...

PWM_CINVH field descriptions (continued)

Field	Description
5 CINV5	PWM Compare Invert 5 This bit controls the polarity of PWM compare output 5. 0 PWM output 5 is high when CNTR is less than VAL5 1 PWM output 5 is high when CNTR is greater than VAL5
4 CINV4	PWM Compare Invert 4 This bit controls the polarity of PWM compare output 4. 0 PWM output 4 is high when CNTR is less than VAL4 1 PWM output 4 is high when CNTR is greater than VAL4
3 CINV3	PWM Compare Invert 3 This bit controls the polarity of PWM compare output 3. 0 PWM output 3 is high when CNTR is less than VAL3 1 PWM output 3 is high when CNTR is greater than VAL3
2 CINV2	PWM Compare Invert 2 This bit controls the polarity of PWM compare output 2. 0 PWM output 2 is high when CNTR is less than VAL2 1 PWM output 2 is high when CNTR is greater than VAL2
1 CINV1	PWM Compare Invert 1 This bit controls the polarity of PWM compare output 1. 0 PWM output 1 is high when CNTR is less than VAL1 1 PWM output 1 is high when CNTR is greater than VAL1
0 CINV0	PWM Compare Invert 0 This bit controls the polarity of PWM compare output 0. 0 PWM output 0 is high when CNTR is less than VAL0 1 PWM output 0 is high when CNTR is greater than VAL0

26.5 Resets

All PWM registers are reset to their default values upon any system reset.

26.6 Clocks

The PWM operation clock runs at either system clock or 2 × system clock, which is selected in the SIM module.

26.7 Interrupts

PWM sources can generate CPU interrupt requests:

- Reload flag (PWMF)—PWMF is set at the beginning of every reload cycle. The reload interrupt enable bit, PWMRIE, enables PWMF to generate CPU interrupt requests. PWMF and PWMRIE are in PWM control register (CTRL)
- Fault flags (FFLAG0–FFLAG3)—The FFLAGn bit is set when a logic one occurs on the FAULTn pin. The fault pin interrupt enable bits, FIE0–FIE3, enable the FFLAGn flags to generate CPU interrupt requests. FFLAG0–FFLAG3 are in the fault status register. FIE0–FIE3 are in the fault control register

Chapter 27
Development support

27.1 Introduction

This chapter describes the single-wire background debug mode (BDM), which uses the on-chip background debug controller (BDC) module, and the independent on-chip real-time in-circuit emulation (ICE) system, which uses the on-chip debug (DBG) module.

27.1.1 Forcing active background

The method for forcing active background mode depends on the specific HCS08 derivative. For the 9S08xxxx, you can force active background after a power-on reset by holding the BKGD pin low as the device exits the reset condition. You can also force active background by driving BKGD low immediately after a serial background command that writes a one to the BDFR bit in the SBDFR register. Other causes of reset including an external pin reset or an internally generated error reset ignore the state of the BKGD pin and reset into normal user mode. If no debug pod is connected to the BKGD pin, the MCU will always reset into normal operating mode.

27.1.2 Features

Features of the BDC module include:

- Single pin for mode selection and background communications
- BDC registers are not located in the memory map
- SYNC command to determine target communications rate
- Non-intrusive commands for memory access
- Active background mode commands for CPU register access
- GO and TRACE1 commands
- BACKGROUND command can wake CPU from stop or wait modes
- One hardware address breakpoint built into BDC

- BDC clock runs in stop mode, if BDC enabled
- Watchdog disabled by default while in active background mode. It can also be enabled by proper configuration

Features of the ICE system include:

- Two trigger comparators: Two address + read/write (R/W) or one full address + data + R/W
- Flexible 8-word by 16-bit FIFO (first-in, first-out) buffer for capture information:
  - Change-of-flow addresses or
  - Event-only data
- Two types of breakpoints:
  - Tag breakpoints for instruction opcodes
  - Force breakpoints for any address access
- Nine trigger modes:
  - Basic: A-only, A OR B
  - Sequence: A then B
  - Full: A AND B data, A AND NOT B data
  - Event (store data): Event-only B, A then event-only B
  - Range: Inside range (A ≤ address ≤ B), outside range (address < A or address > B)

27.2 Background debug controller (BDC)

All MCUs in the HCS08 Family contain a single-wire background debug interface that supports in-circuit programming of on-chip nonvolatile memory and sophisticated non-intrusive debug capabilities. Unlike debug interfaces on earlier 8-bit MCUs, this system does not interfere with normal application resources. It does not use any user memory or locations in the memory map and does not share any on-chip peripherals.

BDC commands are divided into two groups:

- Active background mode commands require that the target MCU is in active background mode (the user program is not running). Active background mode commands allow the CPU registers to be read or written, and allow the user to trace one user instruction at a time, or GO to the user program from active background mode.
- Non-intrusive commands can be executed at any time even while the user's program is running. Non-intrusive commands allow a user to read or write MCU memory locations or access status and control registers within the background debug controller.

Typically, a relatively simple interface pod is used to translate commands from a host computer into commands for the custom serial interface to the single-wire background debug system. Depending on the development tool vendor, this interface pod may use a standard RS-232 serial port, a parallel printer port, or some other type of communications such as a universal serial bus (USB) to communicate between the host PC and the pod. The pod typically connects to the target system with ground, the BKGD pin, RESET, and sometimes V_{DD}. An open-drain connection to reset allows the host to force a target system reset, which is useful to regain control of a lost target system or to control startup of a target system before the on-chip nonvolatile memory has been programmed. Sometimes V_{DD} can be used to allow the pod to use power from the target system to avoid the need for a separate power supply. However, if the pod is powered separately, it can be connected to a running target system without forcing a target system reset or otherwise disturbing the running application program.

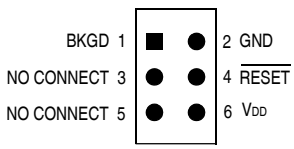


Figure 27-1. BDM tool connector

27.2.1 BKGD pin description

BKGD is the single-wire background debug interface pin. The primary function of this pin is for bidirectional serial communication of active background mode commands and data. During reset, this pin is used to select between starting in active background mode or starting the user's application program. This pin is also used to request a timed sync response pulse to allow a host development tool to determine the correct clock frequency for background debug serial communications.

BDC serial communications use a custom serial protocol first introduced on the M68HC12 Family of microcontrollers. This protocol assumes the host knows the communication clock rate that is determined by the target BDC clock rate. All communication is initiated and controlled by the host that drives a high-to-low edge to signal the beginning of each bit time. Commands and data are sent most significant bit first (MSB first). For a detailed description of the communications protocol, refer to Communication details.

If a host is attempting to communicate with a target MCU that has an unknown BDC clock rate, a SYNC command may be sent to the target MCU to request a timed sync response signal from which the host can determine the correct communication speed.

BKGD is a pseudo-open-drain pin and there is an on-chip pullup so no external pullup resistor is required. Unlike typical open-drain pins, the external RC time constant on this pin, which is influenced by external capacitance, plays almost no role in signal rise time. The custom protocol provides for brief, actively driven speedup pulses to force rapid rise times on this pin without risking harmful drive level conflicts. Refer to Communication details for more detail.

When no debugger pod is connected to the 6-pin BDM interface connector, the internal pullup on BKGD chooses normal operating mode. When a debug pod is connected to BKGD it is possible to force the MCU into active background mode after reset. The specific conditions for forcing active background depend upon the HCS08 derivative (refer to the introduction to this Development Support section). It is not necessary to reset the target MCU to communicate with it through the background debug interface.

27.2.2 Communication details

The BDC serial interface requires the external controller to generate a falling edge on the BKGD pin to indicate the start of each bit time. The external controller provides this falling edge whether data is transmitted or received.

BKGD is a pseudo-open-drain pin that can be driven either by an external controller or by the MCU. Data is transferred MSB first at 16 BDC clock cycles per bit (nominal speed). The interface times out if 512 BDC clock cycles occur between falling edges from the host. Any BDC command that was in progress when this timeout occurs is aborted without affecting the memory or operating mode of the target MCU system.

The custom serial protocol requires the debug pod to know the target BDC communication clock speed.

The clock switch (CLKSW) control bit in the BDC status and control register allows the user to select the BDC clock source. The BDC clock source can either be the MSTRCLK or the alternate BDC clock source.

The BKGD pin can receive a high or low level or transmit a high or low level. The following diagrams show timing for each of these cases. Interface timing is synchronous to clocks in the target BDC, but asynchronous to the external host. The internal BDC clock signal is shown for reference in counting cycles.

The following figure shows an external host transmitting a logic 1 or 0 to the BKGD pin of a target HCS08 MCU. The host is asynchronous to the target so there is a 0-to-1 cycle delay from the host-generated falling edge to where the target perceives the beginning of the bit time. Ten target BDC clock cycles later, the target senses the bit level on the BKGD pin. Typically, the host actively drives the pseudo-open-drain BKGD pin during

host-to-target transmissions to speed up rising edges. Because the target does not drive the BKGD pin during the host-to-target transmission period, there is no need to treat the line as an open-drain signal during this period.

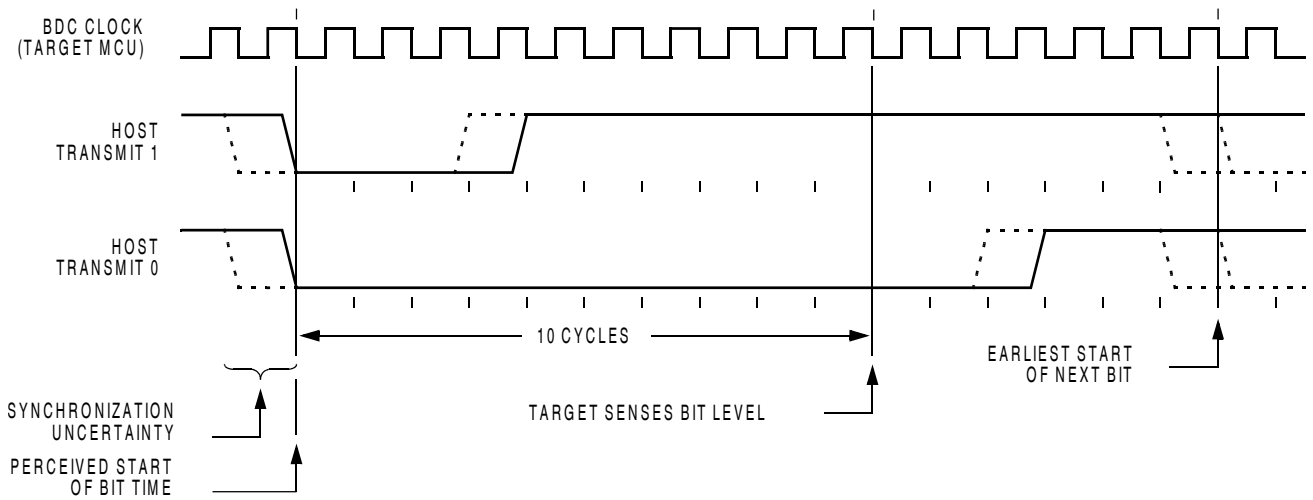


Figure 27-2. BDC host-to-target serial bit timing

The next figure shows the host receiving a logic 1 from the target HCS08 MCU. Because the host is asynchronous to the target MCU, there is a 0-to-1 cycle delay from the host-generated falling edge on BKGD to the perceived start of the bit time in the target MCU. The host holds the BKGD pin low long enough for the target to recognize it (at least two target BDC cycles). The host must release the low drive before the target MCU drives a brief active-high speedup pulse seven cycles after the perceived start of the bit time. The host should sample the bit level about 10 cycles after it started the bit time.

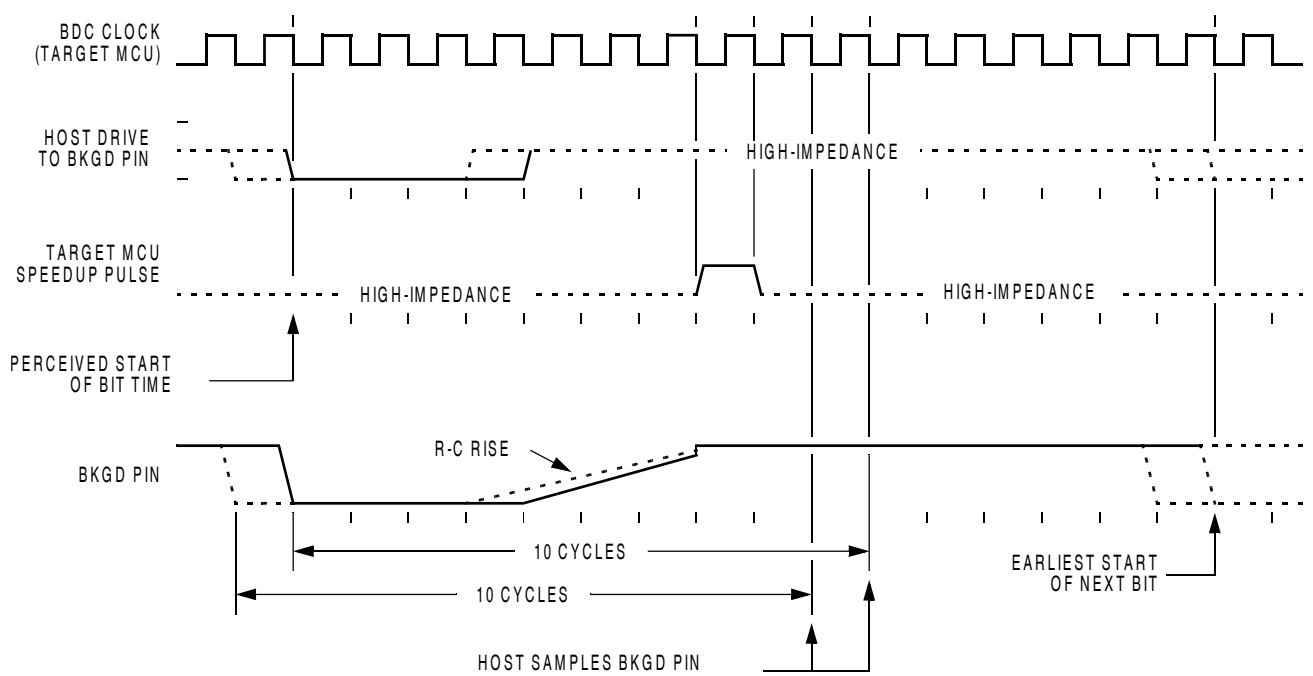


Figure 27-3. BDC target-to-host serial bit timing (logic 1)

The following figure shows the host receiving a logic 0 from the target HCS08 MCU. Because the host is asynchronous to the target MCU, there is a 0-to-1 cycle delay from the host-generated falling edge on BKGD to the start of the bit time as perceived by the target MCU. The host initiates the bit time but the target HCS08 finishes it. Because the target wants the host to receive a logic 0, it drives the BKGD pin low for 13 BDC clock cycles, then briefly drives it high to speed up the rising edge. The host samples the bit level about 10 cycles after starting the bit time.

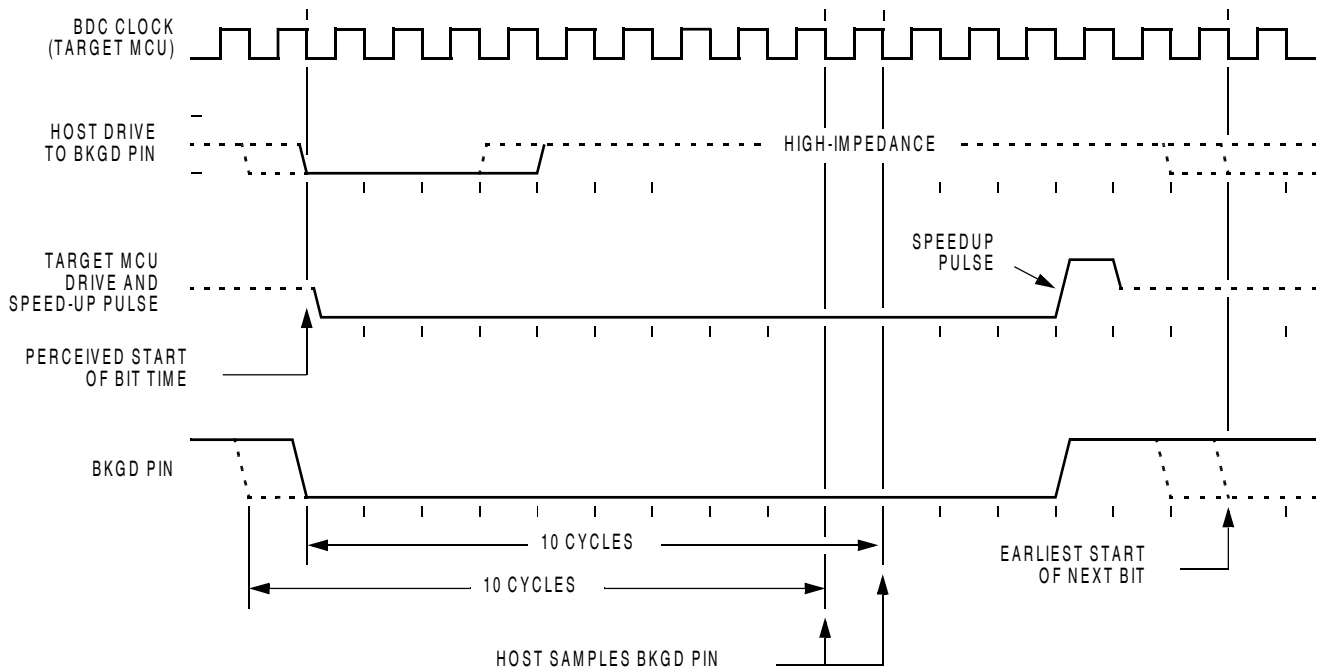


Figure 27-4. BDM target-to-host serial bit timing (logic 0)

27.2.3 BDC commands

BDC commands are sent serially from a host computer to the BKGD pin of the target HCS08 MCU. All commands and data are sent MSB-first using a custom BDC communications protocol. Active background mode commands require that the target MCU is currently in the active background mode while non-intrusive commands may be issued at any time whether the target MCU is in active background mode or running a user application program.

The following table shows all HCS08 BDC commands, a shorthand description of their coding structure, and the meaning of each command.

Coding Structure Nomenclature

This nomenclature is used in the following table to describe the coding structure of the BDC commands.

```
Commands begin with an 8-bit hexadecimal command code in the host-to-target direction (most
significant bit first)
/ =separates parts of the command
d=delay 16 target BDC clock cycles
AAAA = a 16-bit address in the host-to-target direction
RD   = 8 bits of read data in the target-to-host direction
WD   = 8 bits of write data in the host-to-target direction
RD16 = 16 bits of read data in the target-to-host direction
WD16 = 16 bits of write data in the host-to-target direction
SS   = the contents of BDCSCR in the target-to-host direction (STATUS)
CC   = 8 bits of write data for BDCSCR in the host-to-target direction (CONTROL)
RBKP = 16 bits of read data in the target-to-host direction (from BDCBKPT breakpoint
register)
WBKP = 16 bits of write data in the host-to-target direction (for BDCBKPT breakpoint
register)
```

Table 27-1. BDC command summary

Command mnemonic	Active BDM/ non-intrusive	Coding structure	Description
SYNC	Non-intrusive	N/A[1]	Request a timed reference pulse to determine target BDC communication speed
ACK_ENABLE	Non-intrusive	D5/d	Enable acknowledge protocol. Refer to NXP document order no. HCS08RMv1/D.
ACK_DISABLE	Non-intrusive	D6/d	Disable acknowledge protocol. Refer to NXP document order no. HCS08RMv1/D.
BACKGROUND	Non-intrusive	90/d	Enter active background mode if enabled (ignore if ENBDM bit equals 0)
READ_STATUS	Non-intrusive	E4/SS	Read BDC status from BDCSCR
WRITE_CONTROL	Non-intrusive	C4/CC	Write BDC controls in BDCSCR
READ_BYTE	Non-intrusive	E0/AAAA/d/RD	Read a byte from target memory
READ_BYTE_WS	Non-intrusive	E1/AAAA/d/SS/RD	Read a byte and report status
READ_LAST	Non-intrusive	E8/SS/RD	Re-read byte from address just read and report status
WRITE_BYTE	Non-intrusive	C0/AAAA/WD/d	Write a byte to target memory
WRITE_BYTE_WS	Non-intrusive	C1/AAAA/WD/d/SS	Write a byte and report status
READ_BKPT	Non-intrusive	E2/RBKP	Read BDCBKPT breakpoint register
WRITE_BKPT	Non-intrusive	C2/WBKP	Write BDCBKPT breakpoint register
GO	Active BDM	08/d	Go to execute the user application program starting at the address currently in the PC
TRACE1	Active BDM	10/d	Trace 1 user instruction at the address in the PC, then return to active background mode

Table continues on the next page...

Table 27-1. BDC command summary (continued)

Command mnemonic	Active BDM/ non-intrusive	Coding structure	Description
TAGGO	Active BDM	18/d	Same as GO but enable external tagging (HCS08 devices have no external tagging pin)
READ_A	Active BDM	68/d/RD	Read accumulator (A)
READ_CCR	Active BDM	69/d/RD	Read condition code register (CCR)
READ_PC	Active BDM	6B/d/RD16	Read program counter (PC)
READ_HX	Active BDM	6C/d/RD16	Read H and X register pair (H:X)
READ_SP	Active BDM	6F/d/RD16	Read stack pointer (SP)
READ_NEXT	Active BDM	70/d/RD	Increment H:X by one then read memory byte located at H:X
READ_NEXT_WS	Active BDM	71/d/SS/RD	Increment H:X by one then read memory byte located at H:X. Report status and data.
WRITE_A	Active BDM	48/WD/d	Write accumulator (A)
WRITE_CCR	Active BDM	49/WD/d	Write condition code register (CCR)
WRITE_PC	Active BDM	4B/WD16/d	Write program counter (PC)
WRITE_HX	Active BDM	4C/WD16/d	Write H and X register pair (H:X)
WRITE_SP	Active BDM	4F/WD16/d	Write stack pointer (SP)
WRITE_NEXT	Active BDM	50/WD/d	Increment H:X by one, then write memory byte located at H:X
WRITE_NEXT_WS	Active BDM	51/WD/d/SS	Increment H:X by one, then write memory byte located at H:X. Also report status.

1. The SYNC command is a special operation that does not have a command code.

The SYNC command is unlike other BDC commands because the host does not necessarily know the correct communications speed to use for BDC communications until after it has analyzed the response to the SYNC command.

To issue a SYNC command, the host:

- Drives the BKGD pin low for at least 128 cycles of the slowest possible BDC clock (The slowest clock is normally the reference oscillator/64 or the self-clocked rate/ 64.)
- Drives BKGD high for a brief speedup pulse to get a fast rise time (This speedup pulse is typically one cycle of the fastest clock in the system.)
- Removes all drive to the BKGD pin so it reverts to high impedance
- Monitors the BKGD pin for the sync response pulse

The target, upon detecting the SYNC request from the host (which is a much longer low time than would ever occur during normal BDC communications):

- Waits for BKGD to return to a logic high
- Delays 16 cycles to allow the host to stop driving the high speedup pulse
- Drives BKGD low for 128 BDC clock cycles
- Drives a 1-cycle high speedup pulse to force a fast rise time on BKGD
- Removes all drive to the BKGD pin so it reverts to high impedance

The host measures the low time of this 128-cycle sync response pulse and determines the correct speed for subsequent BDC communications. Typically, the host can determine the correct communication speed within a few percent of the actual target speed and the communication protocol can easily tolerate speed errors of several percent.

27.2.4 BDC hardware breakpoint

The BDC includes one relatively simple hardware breakpoint that compares the CPU address bus to a 16-bit match value in the BDCBKPT register. This breakpoint can generate a forced breakpoint or a tagged breakpoint. A forced breakpoint causes the CPU to enter active background mode at the first instruction boundary following any access to the breakpoint address. The tagged breakpoint causes the instruction opcode at the breakpoint address to be tagged so that the CPU will enter active background mode rather than executing that instruction if and when it reaches the end of the instruction queue. This implies that tagged breakpoints can be placed only at the address of an instruction opcode while forced breakpoints can be set at any address.

The breakpoint enable (BKPTEN) control bit in the BDC status and control register (BDCSCR) is used to enable the breakpoint logic (BKPTEN = 1). When BKPTEN = 0, its default value after reset, the breakpoint logic is disabled and no BDC breakpoints are requested regardless of the values in other BDC breakpoint registers and control bits. The force/tag select (FTS) control bit in BDCSCR is used to select forced (FTS = 1) or tagged (FTS = 0) type breakpoints.

The on-chip debug module (DBG) includes circuitry for two additional hardware breakpoints that are more flexible than the simple breakpoint in the BDC module.

27.3 On-chip debug system (DBG)

Because HCS08 devices do not have external address and data buses, the most important functions of an in-circuit emulator have been built onto the chip with the MCU. The debug system consists of an 8-stage FIFO that can store address or data bus information,

and a flexible trigger system to decide when to capture bus information and what information to capture. The system relies on the single-wire background debug system to access debug control registers and to read results out of the eight stage FIFO.

The debug module includes control and status registers that are accessible in the user's memory map. These registers are located in the high register space to avoid using valuable direct page memory space.

Most of the debug module's functions are used during development, and user programs rarely access any of the control and status registers for the debug module. The one exception is that the debug system can provide the means to implement a form of ROM patching. This topic is discussed in greater detail in Hardware breakpoints.

27.3.1 Comparators A and B

Two 16-bit comparators (A and B) can optionally be qualified with the R/W signal and an opcode tracking circuit. Separate control bits allow you to ignore R/W for each comparator. The opcode tracking circuitry optionally allows you to specify that a trigger will occur only if the opcode at the specified address is actually executed as opposed to only being read from memory into the instruction queue. The comparators are also capable of magnitude comparisons to support the inside range and outside range trigger modes. Comparators are disabled temporarily during all BDC accesses.

The A comparator is always associated with the 16-bit CPU address. The B comparator compares to the CPU address or the 8-bit CPU data bus, depending on the trigger mode selected. Because the CPU data bus is separated into a read data bus and a write data bus, the RWAEN and RWA control bits have an additional purpose, in full address plus data comparisons they are used to decide which of these buses to use in the comparator B data bus comparisons. If RWAEN = 1 (enabled) and RWA = 0 (write), the CPU's write data bus is used. Otherwise, the CPU's read data bus is used.

The currently selected trigger mode determines what the debugger logic does when a comparator detects a qualified match condition. A match can cause:

- Generation of a breakpoint to the CPU
- Storage of data bus values into the FIFO
- Starting to store change-of-flow addresses into the FIFO (begin type trace)
- Stopping the storage of change-of-flow addresses into the FIFO (end type trace)

27.3.2 Bus capture information and FIFO operation

The usual way to use the FIFO is to setup the trigger mode and other control options, then arm the debugger. When the FIFO has filled or the debugger has stopped storing data into the FIFO, you would read the information out of it in the order it was stored into the FIFO. Status bits indicate the number of words of valid information that are in the FIFO as data is stored into it. If a trace run is manually halted by writing 0 to ARM before the FIFO is full (CNT = 1:0:0:0), the information is shifted by one position and the host must perform ((8 - CNT) - 1) dummy reads of the FIFO to advance it to the first significant entry in the FIFO.

In most trigger modes, the information stored in the FIFO consists of 16-bit change-of-flow addresses. In these cases, read DBGFH then DBGFL to get one coherent word of information out of the FIFO. Reading DBGFL (the low-order byte of the FIFO data port) causes the FIFO to shift so the next word of information is available at the FIFO data port. In the event-only trigger modes (see Trigger modes), 8-bit data information is stored into the FIFO. In these cases, the high-order half of the FIFO (DBGFH) is not used and data is read out of the FIFO by simply reading DBGFL. Each time DBGFL is read, the FIFO is shifted so the next data value is available through the FIFO data port at DBGFL.

In trigger modes where the FIFO is storing change-of-flow addresses, there is a delay between CPU addresses and the input side of the FIFO. Because of this delay, if the trigger event itself is a change-of-flow address or a change-of-flow address appears during the next two bus cycles after a trigger event starts the FIFO, it will not be saved into the FIFO. In the case of an end-trace, if the trigger event is a change-of-flow, it will be saved as the last change-of-flow entry for that debug run.

The FIFO can also be used to generate a profile of executed instruction addresses when the debugger is not armed. When ARM = 0, reading DBGFL causes the address of the most-recently fetched opcode to be saved in the FIFO. To use the profiling feature, a host debugger would read addresses out of the FIFO by reading DBGFH then DBGFL at regular periodic intervals. The first eight values would be discarded because they correspond to the eight DBGFL reads needed to initially fill the FIFO. Additional periodic reads of DBGFH and DBGFL return delayed information about executed instructions so the host debugger can develop a profile of executed instruction addresses.

27.3.3 Change-of-flow information

To minimize the amount of information stored in the FIFO, only information related to instructions that cause a change to the normal sequential execution of instructions is stored. With knowledge of the source and object code program stored in the target system, an external debugger system can reconstruct the path of execution through many instructions from the change-of-flow information stored in the FIFO.

For conditional branch instructions where the branch is taken (branch condition was true), the source address is stored (the address of the conditional branch opcode). Because BRA and BRN instructions are not conditional, these events do not cause change-of-flow information to be stored in the FIFO.

Indirect JMP and JSR instructions use the current contents of the H:X index register pair to determine the destination address, so the debug system stores the run-time destination address for any indirect JMP or JSR. For interrupts, RTI, or RTS, the destination address is stored in the FIFO as change-of-flow information.

27.3.4 Tag vs. force breakpoints and triggers

Tagging is a term that refers to identifying an instruction opcode as it is fetched into the instruction queue, but not taking any other action until and unless that instruction is actually executed by the CPU. This distinction is important because any change-of-flow from a jump, branch, subroutine call, or interrupt causes some instructions that have been fetched into the instruction queue to be thrown away without being executed.

A force-type breakpoint waits for the current instruction to finish and then acts upon the breakpoint request. The usual action in response to a breakpoint is to go to active background mode rather than continuing to the next instruction in the user application program.

The tag vs. force terminology is used in two contexts within the debug module. The first context refers to breakpoint requests from the debug module to the CPU. The second refers to match signals from the comparators to the debugger control logic. When a tag-type break request is sent to the CPU, a signal is entered into the instruction queue along with the opcode so that if/when this opcode ever executes, the CPU will effectively replace the tagged opcode with a BGND opcode so the CPU goes to active background mode rather than executing the tagged instruction. When the TRGSEL control bit in the DBGT register is set to select tag-type operation, the output from comparator A or B is qualified by a block of logic in the debug module that tracks opcodes and produces only a trigger to the debugger if the opcode at the compare address is actually executed. There is separate opcode tracking logic for each comparator so more than one compare event can be tracked through the instruction queue at a time.

27.3.5 Trigger modes

The trigger mode controls the overall behavior of a debug run. The 4-bit TRG field in the DBGT register selects one of nine trigger modes. When TRGSEL = 1 in the DBGT register, the output of the comparator must propagate through an opcode tracking circuit before triggering FIFO actions. The BEGIN bit in DBGT chooses whether the FIFO begins storing data when the qualified trigger is detected (begin trace), or the FIFO stores data in a circular fashion from the time it is armed until the qualified trigger is detected (end trigger).

A debug run is started by writing a 1 to the ARM bit in the DBGC register, which sets the ARMF flag and clears the AF and BF flags and the CNT bits in DBGS. A begin-trace debug run ends when the FIFO gets full. An end-trace run ends when the selected trigger event occurs. Any debug run can be stopped manually by writing a 0 to ARM or DBGEN in DBGC.

In all trigger modes except event-only modes, the FIFO stores change-of-flow addresses. In event-only trigger modes, the FIFO stores data in the low-order eight bits of the FIFO.

The BEGIN control bit is ignored in event-only trigger modes and all such debug runs are begin type traces. When TRGSEL = 1 to select opcode fetch triggers, it is not necessary to use R/W in comparisons because opcode tags would apply only to opcode fetches that are always read cycles. It would also be unusual to specify TRGSEL = 1 while using a full mode trigger because the opcode value is normally known at a particular address.

The following trigger mode descriptions state only the primary comparator conditions that lead to a trigger. Either comparator can usually be further qualified with R/W by setting RWAEN (RWBEN) and the corresponding RWA (RWB) value to be matched against R/W. The signal from the comparator with optional R/W qualification is used to request a CPU breakpoint if BRKEN = 1 and TAG determines whether the CPU request will be a tag request or a force request.

A-Only – Trigger when the address matches the value in comparator A

A OR B – Trigger when the address matches either the value in comparator A or the value in comparator B

A Then B – Trigger when the address matches the value in comparator B but only after the address for another cycle matched the value in comparator A. There can be any number of cycles after the A match and before the B match.

A AND B Data (Full Mode)— This is called a full mode because address, data, and R/W (optionally) must match within the same bus cycle to cause a trigger event. Comparator A checks address, the low byte of comparator B checks data, and R/W is checked against RWA if RWAEN = 1. The high-order half of comparator B is not used.

In full trigger modes it is not useful to specify a tag-type CPU breakpoint (BRKEN = TAG = 1), but if you do, the comparator B data match is ignored for the purpose of issuing the tag request to the CPU and the CPU breakpoint is issued when the comparator A address matches.

A AND NOT B Data (Full Mode)— Address must match comparator A, data must not match the low half of comparator B, and R/W must match RWA if RWAEN = 1. All three conditions must be met within the same bus cycle to cause a trigger.

In full trigger modes it is not useful to specify a tag-type CPU breakpoint (BRKEN = TAG = 1), but if you do, the comparator B data match is ignored for the purpose of issuing the tag request to the CPU and the CPU breakpoint is issued when the comparator A address matches.

Event-Only B (Store Data)— Trigger events occur each time the address matches the value in comparator B. Trigger events cause the data to be captured into the FIFO. The debug run ends when the FIFO becomes full.

A Then Event-Only B (Store Data)— After the address has matched the value in comparator A, a trigger event occurs each time the address matches the value in comparator B. Trigger events cause the data to be captured into the FIFO. The debug run ends when the FIFO becomes full.

Inside Range (A ≤ Address ≤ B)— A trigger occurs when the address is greater than or equal to the value in comparator A and less than or equal to the value in comparator B at the same time.

Outside Range (Address < A or Address > B)— A trigger occurs when the address is either less than the value in comparator A or greater than the value in comparator B.

27.3.6 Hardware breakpoints

The BRKEN control bit in the DBGC register may be set to 1 to allow any of the trigger conditions described in Trigger modes to be used to generate a hardware breakpoint request to the CPU. TAG in DBGC controls whether the breakpoint request will be treated as a tag-type breakpoint or a force-type breakpoint. A tag breakpoint causes the current opcode to be marked as it enters the instruction queue. If a tagged opcode reaches

the end of the pipe, the CPU executes a BGND instruction to go to active background mode rather than executing the tagged opcode. A force-type breakpoint causes the CPU to finish the current instruction and then go to active background mode.

If the background mode has not been enabled (ENBDM = 1) by a serial WRITE_CONTROL command through the BKGD pin, the CPU will execute an SWI instruction instead of going to active background mode.

27.4 Memory map and register description

This section contains the descriptions of the BDCand DBG registers and control bits. Refer to the high-page register summary in the device overview chapter of this data sheet for the absolute address assignments for all DBG registers. This section refers to registers and control bits only by their names. An NXP-provided equate or header file is used to translate these names into the appropriate absolute addresses.

BDC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
0	BDC Status and Control Register (BDC_SCR)	8	R/W	00h	27.4.1/543
1	BDC Breakpoint Match Register: High (BDC_BKPTH)	8	R/W	00h	27.4.2/545
2	BDC Breakpoint Register: Low (BDC_BKPTL)	8	R/W	00h	27.4.3/546
3	System Background Debug Force Reset Register (BDC_SBDFR)	8	W (always reads 0)	00h	27.4.4/546

27.4.1 BDC Status and Control Register (BDC_SCR)

This register can be read or written by serial BDC commands (READ_STATUS and WRITE_CONTROL) but is not accessible to user programs because it is not located in the normal memory map of the MCU.

NOTE

The reset values shown in the register figure are those in the normal reset conditions. If the MCU is reset in BDM, ENBDM, BDMACT, CLKSW will be reset to 1 and others all be to 0.

Address: 0h base + 0h offset = 0h

Bit	7	6	5	4	3	2	1	0
Read	ENBDM	BDMACT	BKPTEN	FTS	CLKSW	WS	WSF	DVF
Write								
Reset	0	0	0	0	0	0	0	0

BDC_SCR field descriptions

Field	Description
7 ENBDM	Enable BDM (Permit Active Background Mode) Typically, this bit is written to 1 by the debug host shortly after the beginning of a debug session or whenever the debug host resets the target and remains 1 until a normal reset clears it. 0 BDM cannot be made active (non-intrusive commands still allowed). 1 BDM can be made active to allow active background mode commands.
6 BDMACT	Background Mode Active Status This is a read-only status bit. 0 BDM not active (user application program running). 1 BDM active and waiting for serial commands.
5 BKPTEN	BDC Breakpoint Enable If this bit is clear, the BDC breakpoint is disabled and the FTS (force tag select) control bit and BDCBKPT match register are ignored. 0 BDC breakpoint disabled. 1 BDC breakpoint enabled.
4 FTS	Force/Tag Select When FTS = 1, a breakpoint is requested whenever the CPU address bus matches the BDCBKPT match register. When FTS = 0, a match between the CPU address bus and the BDCBKPT register causes the fetched opcode to be tagged. If this tagged opcode ever reaches the end of the instruction queue, the CPU enters active background mode rather than executing the tagged opcode. 0 Tag opcode at breakpoint address and enter active background mode if CPU attempts to execute that instruction 1 Breakpoint match forces active background mode at next instruction boundary (address need not be an opcode)
3 CLKSW	Select Source for BDC Communications Clock CLKSW defaults to 0, which selects the alternate BDC clock source. 0 Alternate BDC clock source. 1 MCU MSTRCLK.
2 WS	Wait or Stop Status When the target CPU is in wait or stop mode, most BDC commands cannot function. However, the BACKGROUND command can be used to force the target CPU out of wait or stop and into active background mode where all BDC commands work. Whenever the host forces the target MCU into active background mode, the host should issue a READ_STATUS command to check that BDMACT = 1 before attempting other BDC commands.

Table continues on the next page...

BDC_SCR field descriptions (continued)

Field	Description
	0 Target CPU is running user application code or in active background mode (was not in wait or stop mode when background became active). 1 Target CPU is in wait or stop mode, or a BACKGROUND command was used to change from wait or stop to active background mode.
1 WSF	Wait or Stop Failure Status This status bit is set if a memory access command failed due to the target CPU executing a wait or stop instruction at or about the same time. The usual recovery strategy is to issue a BACKGROUND command to get out of wait or stop mode into active background mode, repeat the command that failed, then return to the user program. (Typically, the host would restore CPU registers and stack values and re-execute the wait or stop instruction.) 0 Memory access did not conflict with a wait or stop instruction. 1 Memory access command failed because the CPU entered wait or stop mode.
0 DVF	Data Valid Failure Status 0 Memory access did not conflict with a slow memory access 1 Memory access command failed because CPU was not finished with a slow memory access.

27.4.2 BDC Breakpoint Match Register: High (BDC_BKPTH)

This register, together with BDC_BKPTL, holds the address for the hardware breakpoint in the BDC. The BKPTEN and FTS control bits in BDCSCR are used to enable and configure the breakpoint logic. Dedicated serial BDC commands (READ_BKPT and WRITE_BKPT) are used to read and write the BDCBKPT register but is not accessible to user programs because it is not located in the normal memory map of the MCU. Breakpoints are normally set while the target MCU is in active background mode before running the user application program.

Address: 0h base + 1h offset = 1h

Bit	7	6	5	4	3	2	1	0
Read Write	A[15:8]							
Reset	0	0	0	0	0	0	0	0

BDC_BKPTH field descriptions

Field	Description
A[15:8]	High 8-bit of hardware breakpoint address.

27.4.3 BDC Breakpoint Register: Low (BDC_BKPTL)

BDC_BKPTH and BDC_BKPTL registers hold the address for the hardware breakpoint in the BDC. The BDC_SCR[FTS] and BDC_SCR[BKPTEN] bits are used to enable and configure the breakpoint logic. Dedicated serial BDC commands (READ_BKPT and WRITE_BKPT) are used to read and write the BDC_BKPTH and BDC_BKPTL register. Breakpoints are normally set while the target MCU is in background debug mode before running the user application program. However, since READ_BKPT and WRITE_BKPT are foreground commands, they could be executed even while the user program is running.

Address: 0h base + 2h offset = 2h

Bit	7	6	5	4	3	2	1	0
Read Write	A[7:0]							
Reset	0	0	0	0	0	0	0	0

BDC_BKPTL field descriptions

Field	Description
A[7:0]	Low 8-bit of hardware breakpoint address.

27.4.4 System Background Debug Force Reset Register (BDC_SBDFR)

This register contains a single write-only control bit. A serial background mode command such as WRITE_BYTE must be used to write to SBDFR. Attempts to write this register from a user program are ignored. Reads always return 0x00.

Address: 0h base + 3h offset = 3h

Bit	7	6	5	4	3	2	1	0
Read	0							0
Write								BDFR
Reset	0	0	0	0	0	0	0	0

BDC_SBDFR field descriptions

Field	Description
7–1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 BDFR	Background Debug Force Reset

Table continues on the next page...

BDC_SBDFR field descriptions (continued)

Field	Description
	A serial active background mode command such as WRITE_BYTE allows an external debug host to force a target system reset. Writing 1 to this bit forces an MCU reset. This bit cannot be written from a user program.

Chapter 28
Debug module (DBG)

28.1 Introduction

The DBG module implements an on-chip ICE (in-circuit emulation) system and allows non-intrusive debug of application software by providing an on-chip trace buffer with flexible triggering capability. The trigger also can provide extended breakpoint capacity. The on-chip ICE system is optimized for the S08CPUV6 8-bit architecture and supports 2 M bytes of memory space.

28.1.1 Features

The on-chip ICE system includes these distinctive features:

- Three comparators (A, B, and C) with ability to match addresses in 64 KB space
  - Dual mode, Comparators A and B used to compare addresses
  - Full mode, Comparator A compares address and Comparator B compares data
  - Can be used as triggers and/or breakpoints
  - Comparator C can be used as a normal hardware breakpoint
  - Loop1 capture mode, Comparator C is used to track most recent COF event captured into FIFO
- Tag and Force type breakpoints
- Nine trigger modes
  - A
  - A Or B
  - A then B
  - A and B, where B is data (full mode)
  - A and not B, where B is data (full mode)
  - Event only B, store data
  - A then event only B, store data
  - Inside range, A ≤ address ≤ B
  - Outside range, address < A or address > B

- FIFO for storing change of flow information and event only data
  - Source address of conditional branches taken
  - Destination address of indirect JMP and JSR instruction
  - Destination address of interrupts, RTI, and RTS instruction
  - Data associated with Event B trigger modes
- Ability to End-trace until reset and begin-trace from reset

28.1.2 Modes of operation

The on-chip ICE system can be enabled in all MCU functional modes. The DBG module is disabled if the MCU is secure. The DBG module comparators are disabled when executing a Background Debug Mode (BDM) command.

28.1.3 Block diagram

The following figure shows the structure of the DBG module.

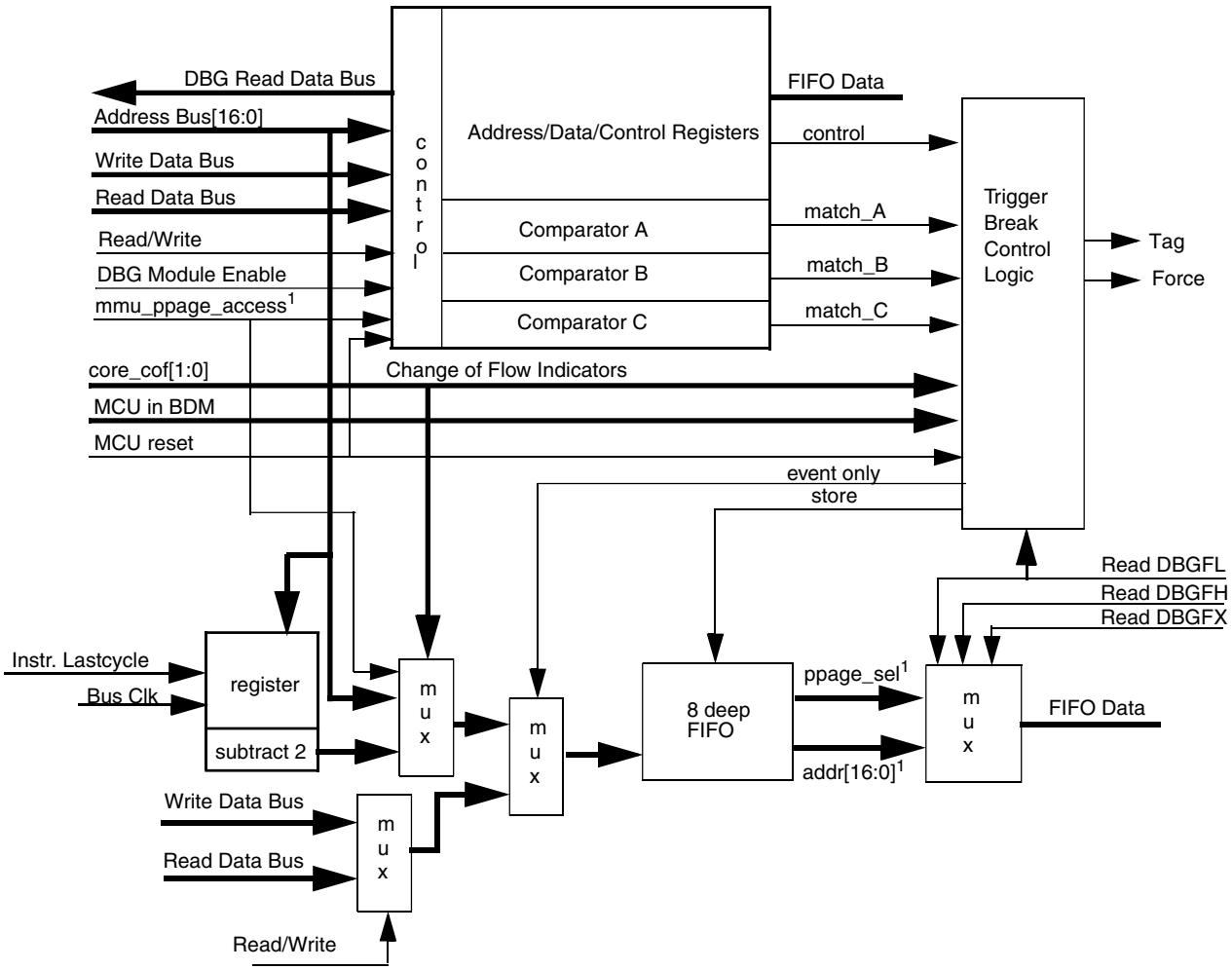


Figure 28-1. DBG block diagram

28.2 Signal description

The DBG module contains no external signals.

28.3 Memory map and registers

This section provides a detailed description of all DBG registers accessible to the end user.

DBG memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
18C0	Debug Comparator A High Register (DBG_CAH)	8	R/W	FFh	28.3.1/552
18C1	Debug Comparator A Low Register (DBG_CAL)	8	R/W	FEh	28.3.2/553
18C2	Debug Comparator B High Register (DBG_CBH)	8	R/W	00h	28.3.3/554
18C3	Debug Comparator B Low Register (DBG_CBL)	8	R/W	00h	28.3.4/554
18C4	Debug Comparator C High Register (DBG_CCH)	8	R/W	00h	28.3.5/555
18C5	Debug Comparator C Low Register (DBG_CCL)	8	R/W	00h	28.3.6/556
18C6	Debug FIFO High Register (DBG_FH)	8	R	00h	28.3.7/556
18C7	Debug FIFO Low Register (DBG_FL)	8	R	00h	28.3.8/557
18C8	Debug Comparator A Extension Register (DBG_CAX)	8	R/W	00h	28.3.9/558
18C9	Debug Comparator B Extension Register (DBG_CBX)	8	R/W	00h	28.3.10/ 559
18CA	Debug Comparator C Extension Register (DBG_CCX)	8	R/W	00h	28.3.11/ 560
18CB	Debug FIFO Extended Information Register (DBG_FX)	8	R	00h	28.3.12/ 561
18CC	Debug Control Register (DBG_C)	8	R/W	C0h	28.3.13/ 561
18CD	Debug Trigger Register (DBG_T)	8	R/W	40h	28.3.14/ 562
18CE	Debug Status Register (DBG_S)	8	R	01h	28.3.15/ 564
18CF	Debug Count Status Register (DBG_CNT)	8	R	00h	28.3.16/ 565

28.3.1 Debug Comparator A High Register (DBG_CAH)

NOTE

All the bits in this register reset to 1 in POR or non-end-run reset. The bits are undefined in end-run reset. In the case of an end-trace to reset where DBGEN = 1 and BEGIN = 0, the bits in this register do not change after reset.

Address: 18C0h base + 0h offset = 18C0h

Bit	7	6	5	4	3	2	1	0
Read Write	CA[15:8]							
Reset	1	1	1	1	1	1	1	1

DBG_CAH field descriptions

Field	Description
CA[15:8]	Comparator A High Compare Bits The Comparator A High compare bits control whether Comparator A will compare the address bus bits [15:8] to a logic 1 or logic 0. 0 Compare corresponding address bit to a logic 0. 1 Compare corresponding address bit to a logic 1.

28.3.2 Debug Comparator A Low Register (DBG_CAL)

NOTE

All the bits in this register reset to 1 in POR or non-end-run reset. The bits are undefined in end-run reset. In the case of an end-trace to reset where DBGEN = 1 and BEGIN = 0, the bits in this register do not change after reset.

Address: 18C0h base + 1h offset = 18C1h

Bit	7	6	5	4	3	2	1	0
Read Write	CA[7:0]							
Reset	1	1	1	1	1	1	1	0

DBG_CAL field descriptions

Field	Description
CA[7:0]	Comparator A Low The Comparator A Low compare bits control whether Comparator A will compare the address bus bits [7:0] to a logic 1 or logic 0. 0 Compare corresponding address bit to a logic 0. 1 Compare corresponding address bit to a logic 1.

28.3.3 Debug Comparator B High Register (DBG_CBH)

NOTE

All the bits in this register reset to 0 in POR or non-end-run reset. The bits are undefined in end-run reset. In the case of an end-trace to reset where DBGEN = 1 and BEGIN = 0, the bits in this register do not change after reset.

Address: 18C0h base + 2h offset = 18C2h

Bit	7	6	5	4	3	2	1	0
Read Write	CB[15:8]							
Reset	0	0	0	0	0	0	0	0

DBG_CBH field descriptions

Field	Description
CB[15:8]	Comparator B High Compare Bits The Comparator B High compare bits control whether Comparator B will compare the address bus bits [15:8] to a logic 1 or logic 0.Not used in full mode. 0 Compare corresponding address bit to a logic 0. 1 Compare corresponding address bit to a logic 1.

28.3.4 Debug Comparator B Low Register (DBG_CBL)

NOTE

All the bits in this register reset to 0 in POR or non-end-run reset. The bits are undefined in end-run reset. In the case of an end-trace to reset where DBGEN = 1 and BEGIN = 0, the bits in this register do not change after reset.

Address: 18C0h base + 3h offset = 18C3h

Bit	7	6	5	4	3	2	1	0
Read Write	CB[7:0]							
Reset	0	0	0	0	0	0	0	0

DBG_CBL field descriptions

Field	Description
CB[7:0]	Comparator B Low The Comparator B Low compare bits control whether Comparator B will compare the address bus bits [7:0] to a logic 1 or logic 0. 0 Compare corresponding address bit to a logic 0. 1 Compare corresponding address bit to a logic 1.

28.3.5 Debug Comparator C High Register (DBG_CCH)

NOTE

All the bits in this register reset to 0 in POR or non-end-run reset. The bits are undefined in end-run reset. In the case of an end-trace to reset where DBGEN = 1 and BEGIN = 0, the bits in this register do not change after reset.

Address: 18C0h base + 4h offset = 18C4h

Bit	7	6	5	4	3	2	1	0
Read Write	CC[15:8]							
Reset	0	0	0	0	0	0	0	0

DBG_CCH field descriptions

Field	Description
CC[15:8]	Comparator C High Compare Bits The Comparator C High compare bits control whether Comparator C will compare the address bus bits [15:8] to a logic 1 or logic 0. 0 Compare corresponding address bit to a logic 0. 1 Compare corresponding address bit to a logic 1.

28.3.6 Debug Comparator C Low Register (DBG_CCL)

NOTE

All the bits in this register reset to 0 in POR or non-end-run reset. The bits are undefined in end-run reset. In the case of an end-trace to reset where DBGEN = 1 and BEGIN = 0, the bits in this register do not change after reset.

Address: 18C0h base + 5h offset = 18C5h

Bit	7	6	5	4	3	2	1	0
Read Write	CC[7:0]							
Reset	0	0	0	0	0	0	0	0

DBG_CCL field descriptions

Field	Description
CC[7:0]	Comparator C Low The Comparator C Low compare bits control whether Comparator C will compare the address bus bits [7:0] to a logic 1 or logic 0. 0 Compare corresponding address bit to a logic 0. 1 Compare corresponding address bit to a logic 1.

28.3.7 Debug FIFO High Register (DBG_FH)

NOTE

All the bits in this register reset to 0 in POR or non-end-run reset. The bits are undefined in end-run reset. In the case of an end-trace to reset where DBGEN = 1 and BEGIN = 0, the bits in this register do not change after reset.

Address: 18C0h base + 6h offset = 18C6h

Bit	7	6	5	4	3	2	1	0
Read	F[15:8]							
Write								
Reset	0	0	0	0	0	0	0	0

DBG_FH field descriptions

Field	Description
F[15:8]	FIFO High Data Bits The FIFO High data bits provide access to bits [15:8] of data in the FIFO. This register is not used in event only modes and will read a $00 for valid FIFO words.

28.3.8 Debug FIFO Low Register (DBG_FL)

NOTE

All the bits in this register reset to 0 in POR or non-end-run reset. The bits are undefined in end-run reset. In the case of an end-trace to reset where DBGEN = 1 and BEGIN = 0, the bits in this register do not change after reset.

Address: 18C0h base + 7h offset = 18C7h

Bit	7	6	5	4	3	2	1	0
Read	F[7:0]							
Write								
Reset	0	0	0	0	0	0	0	0

DBG_FL field descriptions

Field	Description
F[7:0]	FIFO Low Data Bits The FIFO Low data bits contain the least significant byte of data in the FIFO. When reading FIFO words, read DBGFX and DBGFH before reading DBGFL because reading DBGFL causes the FIFO pointers to advance to the next FIFO location. In event-only modes, there is no useful information in DBGFX and DBGFH so it is not necessary to read them before reading DBGFL.

28.3.9 Debug Comparator A Extension Register (DBG_CAX)

NOTE

All the bits in this register reset to 0 in POR or non-end-run reset. The bits are undefined in end-run reset. In the case of an end-trace to reset where DBGEN = 1 and BEGIN = 0, the bits in this register do not change after reset.

Address: 18C0h base + 8h offset = 18C8h

Bit	7	6	5	4	3	2	1	0
Read	RWAEN	RWA	0					
Write								
Reset	0	0	0	0	0	0	0	0

DBG_CAX field descriptions

Field	Description
7 RWAEN	Read/Write Comparator A Enable Bit The RWAEN bit controls whether read or write comparison is enabled for Comparator A. 0 Read/Write is not used in comparison. 1 Read/Write is used in comparison.
6 RWA	Read/Write Comparator A Value Bit The RWA bit controls whether read or write is used in compare for Comparator A. The RWA bit is not used if RWAEN = 0. 0 Write cycle will be matched. 1 Read cycle will be matched.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

28.3.10 Debug Comparator B Extension Register (DBG_CBX)

NOTE

All the bits in this register reset to 0 in POR or non-end-run reset. The bits are undefined in end-run reset. In the case of an end-trace to reset where DBGEN = 1 and BEGIN = 0, the bits in this register do not change after reset.

Address: 18C0h base + 9h offset = 18C9h

Bit	7	6	5	4	3	2	1	0
Read	RWBEN	RWB	0					
Write								
Reset	0	0	0	0	0	0	0	0

DBG_CBX field descriptions

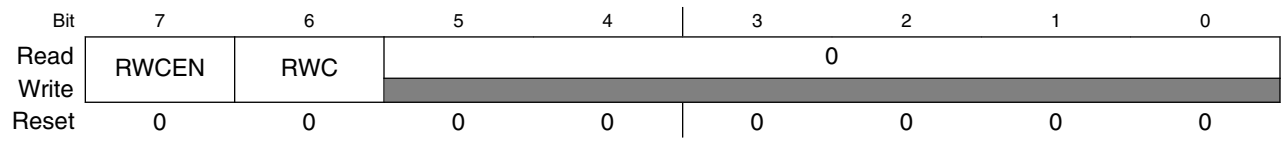
Field	Description
7 RWBEN	Read/Write Comparator B Enable Bit The RWBEN bit controls whether read or write comparison is enabled for Comparator B. In full modes, RWAEN and RWA are used to control comparison of R/W and RWBEN is ignored. 0 Read/Write is not used in comparison. 1 Read/Write is used in comparison.
6 RWB	Read/Write Comparator B Value Bit The RWB bit controls whether read or write is used in compare for Comparator B. The RWB bit is not used if RWBEN = 0.In full modes, RWAEN and RWA are used to control comparison of R/W and RWB is ignored. 0 Write cycle will be matched. 1 Read cycle will be matched.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

28.3.11 Debug Comparator C Extension Register (DBG_CCX)

NOTE

All the bits in this register reset to 0 in POR or non-end-run reset. The bits are undefined in end-run reset. In the case of an end-trace to reset where DBGEN = 1 and BEGIN = 0, the bits in this register do not change after reset.

Address: 18C0h base + Ah offset = 18CAh



DBG_CCX field descriptions

Field	Description
7 RWCEN	Read/Write Comparator C Enable Bit The RWCEN bit controls whether read or write comparison is enabled for Comparator C. 0 Read/Write is not used in comparison. 1 Read/Write is used in comparison.
6 RWC	Read/Write Comparator C Value Bit The RWC bit controls whether read or write is used in compare for Comparator C. The RWC bit is not used if RWCEN = 0. 0 Write cycle will be matched. 1 Read cycle will be matched.
Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

28.3.12 Debug FIFO Extended Information Register (DBG_FX)

NOTE

All the bits in this register reset to 0 in POR or non-end-run reset. The bits are undefined in end-run reset. In the case of an end-trace to reset where DBGEN = 1 and BEGIN = 0, the bits in this register do not change after reset.

Address: 18C0h base + Bh offset = 18CBh

Bit	7	6	5	4	3	2	1	0
Read	PPACC	0						Bit16
Write								
Reset	0	0	0	0	0	0	0	0

DBG_FX field descriptions

Field	Description
7 PPACC	PPAGE Access Indicator Bit This bit indicates whether the captured information in the current FIFO word is associated with an extended access through the PPAGE mechanism or not. This is indicated by the internal signal mmu_ppage_sel which is 1 when the access is through the PPAGE mechanism. 0 The information in the corresponding FIFO word is event-only data or an unpaged 17-bit CPU address with bit-16 = 0. 1 The information in the corresponding FIFO word is a 17-bit flash address with PPAGE[2:0] in the three most significant bits and CPU address[13:0] in the 14 least significant bits.
6–1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 Bit16	Extended Address Bit 16 This bit is the most significant bit of the 17-bit core address.

28.3.13 Debug Control Register (DBG_C)

Address: 18C0h base + Ch offset = 18CCh

Bit	7	6	5	4	3	2	1	0
Read Write	DBGEN	ARM	TAG	BRKEN	0			LOOP1
Reset	1	1	0	0	0	0	0	0

DBG_C field descriptions

Field	Description
7 DBGEN	DBG Module Enable Bit The DBGEN bit enables the DBG module. The DBGEN bit is forced to zero and cannot be set if the MCU is secure. 0 DBG not enabled. 1 DBG enabled.
6 ARM	Arm Bit The ARM bit controls whether the debugger is comparing and storing data in FIFO. 0 Debugger not armed. 1 Debugger armed.
5 TAG	Tag or Force Bit The TAG bit controls whether a debugger or comparator C breakpoint will be requested as a tag or force breakpoint to the CPU. The TAG bit is not used if BRKEN = 0. 0 Force request selected. 1 Tag request selected.
4 BRKEN	Break Enable Bit The BRKEN bit controls whether the debugger will request a breakpoint to the CPU at the end of a trace run, and whether comparator C will request a breakpoint to the CPU. 0 CPU break request not enabled. 1 CPU break request enabled.
3–1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 LOOP1	Select LOOP1 Capture Mode This bit selects either normal capture mode or LOOP1 capture mode. LOOP1 is not used in event-only modes. 0 Normal operation - capture COF events into the capture buffer FIFO. 1 LOOP1 capture mode enabled. When the conditions are met to store a COF value into the FIFO, compare the current COF address with the address in comparator C. If these addresses match, override the FIFO capture and do not increment the FIFO count. If the address does not match comparator C, capture the COF address, including the PPACC indicator, into the FIFO and into comparator C..

28.3.14 Debug Trigger Register (DBG_T)

NOTE

The figure shows the values in POR or non-end-run reset. All the bits are undefined in end-run reset. In the case of an end-trace to reset where DBGEN=1 and BEGIN=0, the ARM and BRKEN bits are cleared but the remaining control bits in this register do not change after reset.

NOTE

The DBG trigger register (DBGT) can not be changed unless ARM=0.

Address: 18C0h base + Dh offset = 18CDh

Bit	7	6	5	4	3	2	1	0
Read	TRGSEL	BEGIN	0		TRG			
Write								
Reset	0	1	0	0	0	0	0	0

DBG_T field descriptions

Field	Description
7 TRGSEL	Trigger Selection Bit The TRGSEL bit controls the triggering condition for the comparators. 0 Trigger on any compare address access. 1 Trigger if opcode at compare address is execute.
6 BEGIN	Begin/End Trigger Bit The BEGIN bit controls whether the trigger begins or ends storing of data in FIFO. 0 Trigger at end of stored data. 1 Trigger before storing data.
5–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
TRG	Trigger Mode Bits The TRG bits select the trigger mode of the DBG module. 0000 A only. 0001 A or B. 0010 A then B. 0011 Event only B. 0100 A then event only B. 0101 A and B (full mode). 0110 A and not B (full mode). 0111 Inside range. 1000 Outside range. 1001-1111 No trigger.

28.3.15 Debug Status Register (DBG_S)

NOTE

The figure shows the values in POR or non-end-run reset. The bits of AF, BF and CF are undefined and ARMF is reset to 0 in end-run reset. In the case of an end-trace to reset where DBGEN=1 and BEGIN=0, ARMF gets cleared by reset but AF, BF, and CF do not change after reset.

Address: 18C0h base + Eh offset = 18CEh

Bit	7	6	5	4	3	2	1	0
Read	AF	BF	CF	0				ARMF
Write								
Reset	0	0	0	0	0	0	0	1

DBG_S field descriptions

Field	Description
7 AF	Trigger A Match Bit The AF bit indicates if Trigger A match condition was met since arming. 0 Comparator A did not match. 1 Comparator A match.
6 BF	Trigger B Match Bit The BF bit indicates if Trigger B match condition was met since arming. 0 Comparator B did not match. 1 Comparator B match.
5 CF	Trigger C Match Bit The CF bit indicates if Trigger C match condition was met since arming. 0 Comparator C did not match. 1 Comparator C match.
4–1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 ARMF	Arm Flag Bit The ARMF bit indicates whether the debugger is waiting for trigger or waiting for the FIFO to fill. While DBGEN = 1, this status bit is a read-only image of the ARM bit in DBGC. 0 Debugger not armed. 1 Debugger armed.

28.3.16 Debug Count Status Register (DBG_CNT)

NOTE

All the bits in this register reset to 0 in POR or non-end-run reset. The bits are undefined in end-run reset. In the case of an end-trace to reset where DBGEN = 1 and BEGIN = 0, the CNT[3:0] bits do not change after reset.

Address: 18C0h base + Fh offset = 18CFh

Bit	7	6	5	4	3	2	1	0
Read	0				CNT			
Write								
Reset	0	0	0	0	0	0	0	0

DBG_CNT field descriptions

Field	Description
7–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
CNT	FIFO Valid Count Bits The CNT bits indicate the amount of valid data stored in the FIFO. Table 1-20 shows the correlation between the CNT bits and the amount of valid data in FIFO. The CNT will stop after a count to eight even if more data is being stored in the FIFO. The CNT bits are cleared when the DBG module is armed, and the count is incremented each time a new word is captured into the FIFO. The host development system is responsible for checking the value in CNT[3:0] and reading the correct number of words from the FIFO because the count does not decrement as data is read out of the FIFO at the end of a trace run. 0000 No data valid. 0001 1 word valid. 0010 2 words valid. 0011 3 words valid. 0100 4 words valid. 0101 5 words valid. 0110 6 words valid. 0111 7 words valid. 1000 8 words valid.

28.4 Functional description

This section provides a complete functional description of the on-chip ICE system. The DBG module is enabled by setting the DBG_C[DBGEN] bit. Enabling the module allows the arming, triggering and storing of data in the FIFO. The DBG module is made up of three main blocks, the comparators, trigger break control logic and the FIFO.

28.4.1 Comparator

The DBG module contains three comparators, A, B, and C. Comparator A compares the core address bus with the address stored in the DBG_CAH and DBG_CAL registers. Comparator B compares the core address bus with the address stored in the DBG_CBH and DBG_CBL registers except in full mode, where it compares the data buses to the data stored in the DBG_CBL register. Comparator C compares the core address bus with the address stored in the DBG_CCH and DBG_CCL registers. Matches on comparators A, B, and C are signaled to the trigger break control (TBC) block.

28.4.1.1 RWA and RWAEN in full modes

In full modes ("A And B" and "A And Not B") DBG_CAX[RWAEN and DBG_CAX[RWA] are used to select read or write comparisons for both comparators A and B. To select write comparisons and the write data bus in Full Modes set DBG_CAX[RWAEN] = 1 and DBG_CAX[RWA] = 0, otherwise read comparisons and the read data bus will be selected. The DBG_CBX[RWBEN] and DBG_CBX[RWB] bits are not used and will be ignored in full modes.

28.4.1.2 Comparator C in loop1 capture mode

Normally comparator C is used as a third hardware breakpoint and is not involved in the trigger logic for the on-chip ICE system. In this mode, it compares the core address bus with the address stored in the DBG_CCX, DBG_CCH, and DBG_CCL registers. However, in loop1 capture mode, comparator C is managed by logic in the DBG module to track the address of the most recent change-of-flow event that was captured into the FIFO buffer. In loop1 capture mode, comparator C is not available for use as a normal hardware breakpoint.

When the DBG_C[ARM] and DBG_C[DBGEN] bits are set to one in loop1 capture mode, comparator C value registers are cleared to prevent the previous contents of these registers from interfering with the loop1 capture mode operation. When a COF event is detected, the address of the event is compared to the contents of the DBG_CCH and DBG_CCL registers to determine whether it is the same as the previous COF entry in the capture FIFO. If the values match, the capture is inhibited to prevent the FIFO from filling up with duplicate entries. If the values do not match, the COF event is captured into the FIFO and the DBG_CCH and DBG_CCL registers are updated to reflect the address of the captured COF event.

28.4.2 Breakpoints

A breakpoint request to the CPU at the end of a trace run can be created if the DBG_C[BRKEN] bit is set. The value of the DBG_T[BEGIN] bit determines when the breakpoint request to the CPU will occur. If the DBG_T[BEGIN] bit is set, begin-trigger is selected and the breakpoint request will not occur until the FIFO is filled with 8 words. If the DBG_T[BEGIN] bit is cleared, end-trigger is selected and the breakpoint request will occur immediately at the trigger cycle.

When traditional hardware breakpoints from comparators A or B are desired, set DBG_T[BEGIN] = 0 to select an end-trace run and set the trigger mode to either 0x0 (A-only) or 0x1 (A OR B) mode.

There are two types of breakpoint requests supported by the DBG module, tag-type and force-type. Tagged breakpoints are associated with opcode addresses and allow breaking just before a specific instruction executes. Force breakpoints are not associated with opcode addresses. The DBG_C[TAG] bit determines whether CPU breakpoint requests will be a tag-type or force-type breakpoints. When DBG_C[TAG] = 0, a force-type breakpoint is requested and it will take effect at the next instruction boundary after the request. When DBG_C[TAG] = 1, a tag-type breakpoint is registered into the instruction queue and the CPU will break if/when this tag reaches the head of the instruction queue and the tagged instruction is about to be executed.

28.4.2.1 Hardware breakpoints

Comparators A, B, and C can be used as three traditional hardware breakpoints whether the on-chip ICE real-time capture function is required or not. To use any breakpoint or trace run capture functions set DBG_C[DBGEN] = 1. DBG_C[BRKEN] and DBG_C[TAG] affect all three comparators. When DBG_C[BRKEN] = 0, no CPU breakpoints are enabled. When DBG_C[BRKEN] = 1, CPU breakpoints are enabled and the DBG_C[TAG] bit determines whether the breakpoints will be tag-type or force-type

breakpoints. To use comparators A and B as hardware breakpoints, set DBG_T = 0x81 for tag-type breakpoints and 0x01 for force-type breakpoints. This sets up an end-type trace with trigger mode "A OR B".

Comparator C is not involved in the trigger logic for the on-chip ICE system.

28.4.3 Trigger selection

The DBG_T[TRGSEL] bit is used to determine the triggering condition of the on-chip ICE system. DBG_T[TRGSEL] applies to both trigger A and B except in the event only trigger modes. By setting the DBG_T[TRGSEL] bit, the comparators will qualify a match with the output of opcode tracking logic. The opcode tracking logic is internal to each comparator and determines whether the CPU executed the opcode at the compare address. With the DBG_T[TRGSEL] bit cleared a comparator match is all that is necessary for a trigger condition to be met.

NOTE

If the DBG_T[TRGSEL] is set, the address stored in the comparator match address registers must be an opcode address for the trigger to occur.

28.4.4 Trigger break control (TBC)

The TBC is the main controller for the DBG module. Its function is to decide whether data should be stored in the FIFO based on the trigger mode and the match signals from the comparator. The TBC also determines whether a request to break the CPU should occur.

The DBG_C[TAG] bit controls whether CPU breakpoints are treated as tag-type or force-type breakpoints. The DBG_T[TRGSEL] bit controls whether a comparator A or B match is further qualified by opcode tracking logic. Each comparator has a separate circuit to track opcodes because the comparators could correspond to separate instructions that could be propagating through the instruction queue at the same time.

In end-type trace runs (DBG_T[BEGIN] = 0), when the comparator registers match, including the optional R/W match, this signal goes to the CPU break logic where DBG_C[BRKEN] determines whether a CPU break is requested and the DBG_C[TAG] control bit determines whether the CPU break will be a tag-type or force-type breakpoint. When DBG_T[TRGSEL] is set, the R/W qualified comparator match signal also passes through the opcode tracking logic. If/when it propagates through this logic, it will cause a

trigger to the ICE logic to begin or end capturing information into the FIFO. In the case of an end-type (DBG_T[BEGIN] = 0) trace run, the qualified comparator signal stops the FIFO from capturing any more information.

If a CPU breakpoint is also enabled, you would want DBG_C[TAG] and DBG_T[TRGSEL] to agree so that the CPU break occurs at the same place in the application program as the FIFO stopped capturing information. If DBG_T[TRGSEL] was 0 and DBG_C[TAG] was 1 in an end-type trace run, the FIFO would stop capturing as soon as the comparator address matched, but the CPU would continue running until a TAG signal could propagate through the CPUs instruction queue, which could take a long time in the case where changes of flow caused the instruction queue to be flushed. If DBG_T[TRGSEL] was one and DBG_C[TAG] was zero in an end-type trace run, the CPU would break before the comparator match signal could propagate through the opcode tracking logic to end the trace run.

In begin-type trace runs (DBG_T[BEGIN] = 1), the start of FIFO capturing is triggered by the qualified comparator signals, and the CPU breakpoint (if enabled by DBG_C[BRKEN]=1) is triggered when the FIFO becomes full. Since this FIFO full condition does not correspond to the execution of a tagged instruction, it would not make sense to use DBG_C[TAG] = 1 for a begin-type trace run.

28.4.4.1 Begin- and end-trigger

The definition of begin- and end-trigger as used in the DBG module are as follows:

- Begin-trigger: storage in FIFO occurs after the trigger and continues until 8 locations are filled.
- End-trigger: storage in FIFO occurs until the trigger with the least recent data falling out of the FIFO if more than 8 words are collected.

28.4.4.2 Arming the DBG module

Arming occurs by enabling the DBG module by setting the DBG_C[DBGEN] bit and by setting the DBG_C[ARM] bit. The DBG_C[ARM] and DBG_S[ARMF] bits are cleared when the trigger condition is met in end-trigger mode or when the FIFO is filled in begin-trigger mode. In the case of an end-trace where DBG_C[DBGEN] = 1 and DBG_T[BEGIN] = 0, DBG_C[ARM] and DBG_S[ARMF] are cleared by any reset to end the trace run that was in progress. The DBG_S[ARMF] bit is also cleared if DBG_C[ARM] is written to zero or when the DBG_C[DBGEN] bit is low. The TBC logic determines whether a trigger condition has been met based on the trigger mode and the trigger selection.

28.4.4.3 Trigger modes

The on-chip ICE system supports nine trigger modes. The trigger mode is used as a qualifier for either starting or ending the storing of data in the FIFO. When the match condition is met, the appropriate flag AF or BF is set in DBG_S register. Arming the DBG module clears the DBG_S[AF], DBG_S[BF], and DBG_S[CF] flags. In all trigger modes except for the event only modes change of flow addresses are stored in the FIFO. In the event only modes only the value on the data bus at the trigger event B comparator match address will be stored.

28.4.4.3.1 A only

In the A only trigger mode, if the match condition for A is met, the DBG_S[AF] flag is set.

28.4.4.3.2 A or B

In the A or B trigger mode, if the match condition for A or B is met, the corresponding flag(s) in the DBG_S register are set.

28.4.4.3.3 A then B

In the A then B trigger mode, the match condition for A must be met before the match condition for B is compared. When the match condition for A or B is met, the corresponding flag in the DBG_S register is set.

28.4.4.3.4 Event only B

In the event only B trigger mode, if the match condition for B is met, the DBG_S[BF] flag is set. The event only B trigger mode is considered a begin-trigger type and the DBG_T[BEGIN] bit is ignored.

28.4.4.3.5 A then event only B

In the A then event only B trigger mode, the match condition for A must be met before the match condition for B is compared. When the match condition for A or B is met, the corresponding flag in the DBG_S register is set. The A then event only B trigger mode is considered a begin-trigger type and the DBG_T[BEGIN] bit is ignored.

28.4.4.3.6 A and B (full mode)

In the A and B trigger mode, comparator A compares to the address bus and comparator B compares to the data bus. In the A and B trigger mode, if the match condition for A and B happen on the same bus cycle, both the DBG_S[AF] and DBG_S[BF] flags are set. If a match condition on only A or only B happens, no flags are set.

For breakpoint tagging operation with an end-trigger type trace, only matches from comparator A will be used to determine if the Breakpoint conditions are met and comparator B matches will be ignored.

28.4.4.3.7 A and not B (full mode)

In the A and not B trigger mode, comparator A compares to the address bus and comparator B compares to the data bus. In the A and not B trigger mode, if the match condition for A and not B happen on the same bus cycle, both the DBG_S[AF] and DBG_S[BF] flags are set. If a match condition on only A or only not B occur no flags are set.

For breakpoint tagging operation with an end-trigger type trace, only matches from comparator A will be used to determine if the breakpoint conditions are met and comparator B matches will be ignored.

28.4.4.3.8 Inside range, A ≤ address ≤ B

In the inside range trigger mode, if the match condition for A and B happen on the same bus cycle, both the DBG_S[AF] and DBG_S[BF] flags are set. If a match condition on only A or only B occur no flags are set.

28.4.4.3.9 Outside range, address < A or address > B

In the outside range trigger mode, if the match condition for A or B is met, the corresponding flag in the DBGS register is set.

The four control bits DBG_T[BEGIN] and DBG_T[TRGSEL], and DBG_C[BRKEN] and DBG_C[TAG], determine the basic type of debug run as shown in the following table. Some of the 16 possible combinations are not used (refer to the notes at the end of the table).

Table 28-1. Basic types of debug runs

BEGIN	TRGSEL	BRKEN	TAG	Type of debug run
0	0	0	x	Fill FIFO until trigger address (no CPU breakpoint - keep running)
0	0	1	0	Fill FIFO until trigger address, then force CPU breakpoint
0	0	1	1	Do not use
0	1	0	x	Fill FIFO until trigger opcode about to execute (no CPU breakpoint - keep running)
0	1	1	0	
0	1	1	1	Fill FIFO until trigger opcode about to execute (trigger causes CPU breakpoint)
1	0	0	x	Start FIFO at trigger address (No CPU breakpoint - keep running)
1	0	1	0	Start FIFO at trigger address, force CPU breakpoint when FIFO full
1	0	1	1	
1	1	0	x	Start FIFO at trigger opcode (No CPU breakpoint - keep running)
1	1	1	0	Start FIFO at trigger opcode, force CPU breakpoint when FIFO full
1	1	1	1	

28.4.5 FIFO

The FIFO is an eight word deep FIFO. In all trigger modes except for event only, the data stored in the FIFO will be change of flow addresses. In the event only trigger modes only the data bus value corresponding to the event is stored. In event only trigger modes, the high byte of the valid data from the FIFO will always read a 0x00.

28.4.5.1 Storing data in FIFO

In all trigger modes except for the event only modes, the address stored in the FIFO will be determined by the change of flow indicators from the core. The signal core_cof[1] indicates the current core address is the destination address of an indirect JSR or JMP instruction, or a RTS or RTI instruction or interrupt vector and the destination address should be stored. The signal core_cof[0] indicates that a conditional branch was taken and that the source address of the conditional branch should be stored.

28.4.5.2 Storing with begin-trigger

Storing with begin-trigger can be used in all trigger modes. Once the DBG module is enabled and armed in the begin-trigger mode, data is not stored in the FIFO until the trigger condition is met. Once the trigger condition is met the DBG module will remain armed until 8 words are stored in the FIFO. If the core_cof[1] signal becomes asserted, the current address is stored in the FIFO. If the core_cof[0] signal becomes asserted, the address registered during the previous last cycle is decremented by two and stored in the FIFO.

28.4.5.3 Storing with end-trigger

Storing with end-trigger cannot be used in event-only trigger modes. After the DBG module is enabled and armed in the end-trigger mode, data is stored in the FIFO until the trigger condition is met. If the core_cof[1] signal becomes asserted, the current address is stored in the FIFO. If the core_cof[0] signal becomes asserted, the address registered during the previous last cycle is decremented by two and stored in the FIFO. When the trigger condition is met, the DBG_C[ARM] and DBG_S[ARMF] will be cleared and no more data will be stored. In non-event only end-trigger modes, if the trigger is at a change of flow address the trigger event will be stored in the FIFO.

28.4.5.4 Reading data from FIFO

The data stored in the FIFO can be read using BDM commands provided the DBG module is enabled and not armed (DBG_C[DBGEN] = 1 and DBG_C[ARM] = 0). The FIFO data is read out first-in-first-out. By reading the DBG_CNT[CNT] bits at the end of a trace run, the number of valid words can be determined. The FIFO data is read by optionally reading the DBG_FH register followed by the DBG_FL register. Each time the DBG_FL register is read, the FIFO is shifted to allow reading of the next word, however, the count does not decrement. In event-only trigger modes where the FIFO will contain only the data bus values stored, to read the FIFO only DBG_FL needs to be accessed.

The FIFO is normally read only while DBG_C[ARM] = 0 and DBG_S[ARMF] = 0, however, reading the FIFO while the DBG module is armed will return the data value in the oldest location of the FIFO and the TBC will not allow the FIFO to shift. This action could cause a valid entry to be lost because the unexpected read blocked the FIFO advance.

If the DBG module is not armed and the DBG_FL register is read, the TBC will store the current opcode address. Through periodic reads of the DBG_FH and DBG_FL registers while the DBG module is not armed, host software can provide a histogram of program execution This is called profile mode.

28.4.6 Interrupt priority

When DBG_T[TRGSEL] is set and the DBG module is armed to trigger on begin- or end-trigger types, a trigger is not detected in the condition where a pending interrupt occurs at the same time that a target address reaches the top of the instruction pipe. In these conditions, the pending interrupt has higher priority and code execution switches to the interrupt service routine.

When DBG_T[TRGSEL] is clear and the DBG module is armed to trigger on end-trigger types, the trigger event is detected on a program fetch of the target address, even when an interrupt becomes pending on the same cycle. In these conditions, the pending interrupt has higher priority, the exception is processed by the core and the interrupt vector is fetched. Code execution is halted before the first instruction of the interrupt service routine is executed. In this scenario, the DBG module will have cleared DBG_C[ARM] without having recorded the change-of-flow that occurred as part of the interrupt exception. Note that the stack will hold the return addresses and can be used to reconstruct execution flow in this scenario.

When DBG_T[TRGSEL] is clear and the DBG module is armed to trigger on begin-trigger types, the trigger event is detected on a program fetch of the target address, even when an interrupt becomes pending on the same cycle. In this scenario, the FIFO captures

the change of flow event. Because the system is configured for begin-trigger, the DBG remains armed and does not break until the FIFO has been filled by subsequent change of flow events.

28.5 Resets

The DBG module cannot cause an MCU reset.

There are two different ways this module will respond to reset depending upon the conditions before the reset event. If the DBG module was setup for an end trace run with DBG_C[DBGEN] = 1 and DBG_T[BEGIN] = 0, DBG_C[ARM], DBG_S[ARMF], and DBG_C[BRKEN] are cleared but the reset function on most DBG control and status bits is overridden so a host development system can read out the results of the trace run after the MCU has been reset. In all other cases including POR, the DBG module controls are initialized to start a begin trace run starting from when the reset vector is fetched. The conditions for the default begin trace run are:

- DBG_CAX = 0x00, DBG_CAH=0xFF, DBG_CAL=0xFE so comparator A is set to match when the 16-bit CPU address 0xFFFE appears during the reset vector fetch
- DBG_C = 0xC0 to enable and arm the DBG module
- DBG_T = 0x40 to select a force-type trigger, a BEGIN trigger, and A-only trigger mode

Appendix A
Changes between revision 5 and 4

Table A-1. Changes between revision 5 and 4

Chapter	Description
Memory	• Updated the register names in the Register addresses assignments
Cyclic redundancy check (CRC)	• Corrected CRC_PL1[PH1] to CRC_PL1[PL1] • Updated the registers descriptions in CRC_PH0, CRC_PH1, CRC_PL0 and CRC_PL1

Appendix B
Changes between revision 4 and 3

Table B-1. Changes between revision 4 and 3

Chapter	Description
Throughout the book	• Added new part of MC9S08SU8VFK.

Appendix C
Changes between revision 3 and 2

Table C-1. Changes between revision 3 and 2

Chapter	Description
Chapter 2 Introduction	• Updated block diagram in MCU block diagram.
Chapter 3 Memory	• Updated the random access memory ranges in Memory map. • Updated the PORT_PTxDD[n] descriptions. • Updated System device identification (SDID)
Chapter 14 Power Management Controller (PMC)	• Removed the note in the PMC_CTRL register.
Chapter 17 Analog-to-digital converter (ADC)	• Updated V_{REFH} descriptions in the ADC analog supply and reference connections.
Chapter 25 Gate Drive Unit (GDU)	• Updated Functional description • Added a note to the GDU_SIGBIAS[BIASSEL].

How to Reach Us:

Home Page:
nxp.com

Web Support:
nxp.com/support

Information in this document is provided solely to enable system and software implementers to use NXP products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits based on the information in this document. NXP reserves the right to make changes without further notice to any products herein.

NXP makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does NXP assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. “Typical” parameters that may be provided in NXP data sheets and/or specifications can and do vary in different applications, and actual performance may vary over time. All operating parameters, including “typicals,” must be validated for each customer application by customer’s technical experts. NXP does not convey any license under its patent rights nor the rights of others. NXP sells products pursuant to standard terms and conditions of sale, which can be found at the following address: nxp.com/SalesTermsandConditions.

NXP, the NXP logo, Freescale, and the Freescale logo are trademarks of NXP B.V. All other product or service names are the property of their respective owners. All rights reserved.

© 2016-2017 NXP B.V.

Document Number MC9S08SU16RM
Revision 5, 4/2017