EPSON
EXCEED YOUR VISION

S2R72C05***
Technical Manual

SEIKO EPSON CORPORATION

Rev.1.00

NOTICE

No part of this material may be reproduced or duplicated in any form or by any means without the written permission of Seiko Epson. Seiko Epson reserves the right to make changes to this material without notice. Seiko Epson does not assume any liability of any kind arising out of any inaccuracies contained in this material or due to its application or use in any product or circuit and, further, there is no representation that this material is applicable to products requiring high level reliability, such as, medical products. Moreover, no license to any intellectual property rights is granted by implication or otherwise, and there is no representation or warranty that anything made in accordance with this material will be free from any patent or copyright infringement of a third party. This material or portions thereof may contain technology or the subject relating to strategic products under the control of the Foreign Exchange and Foreign Trade Law of Japan and may require an export license from the Ministry of Economy, Trade and Industry or other approval from another government agency.

All other product names mentioned herein are trademarks and/or registered trademarks of their respective companies.

Scope

This document applies to the S2R72C05 USB2.0 Device Host Controller LSI.

Table of Contents

Scope ..... **iii**

1 DESCRIPTION OF FUNCTIONS **1**

1.1 Device/Host Register Map Selection and Function Selection 1
1.1.1 Register Map Selection Procedure 1
1.1.2 Device/Host Function Selection Procedure 2
1.1.3 Port State Change Detection Status 2
1.1.3.1 Using VBUS_B Terminal Change Status 2
1.1.3.2 Using Signal Line Change Status 3
1.2 USB Device Control 5
1.2.1 Endpoints 5
1.2.2 Transaction 6
1.2.2.1 SETUP Transactions 8
1.2.2.2 Bulk/Interrupt OUT Transactions 9
1.2.2.3 Isochronous OUT Transactions 10
1.2.2.4 Bulk/Interrupt IN Transactions 10
1.2.2.5 Isochronous IN Transaction 12
1.2.2.6 PING Transactions 12
1.2.3 Control Transfer 13
1.2.3.1 Setup Stage 15
1.2.3.2 Data Stage/Status Stage 15
1.2.3.3 Automatic Address Setup Function 15
1.2.3.4 Descriptor Return Function 16
1.2.4 Bulk Transfer/Interrupt Transfer/Isochronous Transfer 16
1.2.5 Data Flow 16
1.2.5.1 OUT Transfer 16
1.2.5.2 IN Transfer 17
1.2.6 Bulk Only Support 18
1.2.6.1 CBW Support 18
1.2.6.2 CSW Support 19
1.2.7 Auto Negotiation Function 19
1.2.7.1 DISABLE 21
1.2.7.2 IDLE 21
1.2.7.3 WAIT_TIM3US 21
1.2.7.4 WAIT_CHIRP 21
1.2.7.5 WAIT_RSTEND 22
1.2.7.6 DET_SUSPEND 22
1.2.7.7 IN_SUSPEND 22
1.2.7.8 CHK_EVENT 22
1.2.7.9 WAIT_RESTORE 22
1.2.7.10 ERR 23
1.2.7.11 Description of Individual Negotiation Functions 23

1.2.7.11.1 Suspend Detection (HS Mode) 23
1.2.7.11.2 Suspend Detection (FS Mode) 25
1.2.7.11.3 Reset Detection (HS Mode) 27
1.2.7.11.4 Reset Detection (FS Mode) 28
1.2.7.11.5 HS Detection Handshake 29
1.2.7.11.5.1 When Connected to the FS Host Downstream Port 29
1.2.7.11.5.2 When Connected to the HS Host Downstream Port 32
1.2.7.11.5.3 If Reset During SNOOZE 35
1.2.7.11.6 Resume Issue 37
1.2.7.11.7 Resume Detection 40
1.2.7.11.8 Cable Attachment 42
1.3 USB Host Control 44
1.3.1 Channels 44
1.3.1.1 Channel Outline 44
1.3.1.2 Dedicated Control Channel 46
1.3.1.3 General Channels 47
1.3.1.4 Channel Usage Examples 48
1.3.1.4.1 With One Storage Device Connected 48
1.3.1.4.2 With One Communication Device Connected 49
1.3.1.4.3 With One Human Interface Device Connected 50
1.3.1.4.4 With Two Storage Devices Connected via a Hub 51
1.3.2 Scheduling 52
1.3.3 Transactions 53
1.3.3.1 SETUP Transaction 54
1.3.3.2 Bulk OUT Transaction 55
1.3.3.3 Interrupt OUT Transaction 56
1.3.3.4 Isochronous OUT Transaction 57
1.3.3.5 Bulk IN Transaction 58
1.3.3.6 Interrupt IN Transaction 60
1.3.3.7 Isochronous IN Transaction 61
1.3.3.8 PING Transaction 63
1.3.3.9 Low-speed (LS) Transaction 64
1.3.3.10 Split Transaction 66
1.3.4 Control Transfer 67
1.3.4.1 Setup Stage 68
1.3.4.2 Data Stage/Status Stage 68
1.3.4.3 Control Transfer Support Function 69
1.3.5 Bulk Transfer/Interrupt Transfer/Isochronous Transfer 72
1.3.6 Data Flow 72
1.3.6.1 OUT Transfer 72
1.3.6.2 IN Transfer 73
1.3.7 Zero Length Packet Automatic Issuing Function 74
1.3.7.1 Bulk/Interrupt OUT Transfer Zero Length Packet Automatic Issuing Function 74
1.3.8 Bulk-only Support Function 75
1.3.9 Audio Class Assist Function 80
1.3.10 Host State Management Support Function 81

1.3.10.1 Host State 81
1.3.10.1.1 IDLE 83
1.3.10.1.2 WAIT_CONNECT 83
1.3.10.1.3 DISABLED 84
1.3.10.1.4 RESET 85
1.3.10.1.5 OPERATIONAL 86
1.3.10.1.6 SUSPEND 86
1.3.10.1.7 RESUME 87
1.3.10.2 Detection Functions 88
1.3.10.2.1 VBUS Error Detection 88
1.3.10.2.2 Disconnect Detection 90
1.3.10.2.2.1 If HS Device is Disconnected 90
1.3.10.2.2.2 If FS or LS Device is Disconnected 91
1.3.10.2.3 Remote Wakeup Detection 92
1.3.10.2.3.1 If HS Device Is Connected 92
1.3.10.2.3.2 If FS Device Is Connected 94
1.3.10.2.3.3 If LS Device Is Connected 95
1.3.10.2.4 Device Chirp Detection Function 96
1.3.10.2.4.1 If a Correct Device Chirp Is Detected 96
1.3.10.2.4.2 If a Device Error Chirp Is Detected 98
1.3.10.2.5 Port Error Detection 99
1.3.10.3 Individual Host State Management Support Function Explanations 100
1.3.10.3.1 GoIDLE 100
1.3.10.3.2 GoWAIT_CONNECT 102
1.3.10.3.2.1 If an FS Device Is Connected 102
1.3.10.3.2.2 If an LS Device Is Connected 104
1.3.10.3.3 GoDISABLED 106
1.3.10.3.3.1 If an HS Device Is Connected 106
1.3.10.3.3.2 If an FS Device Is Connected 108
1.3.10.3.3.3 If an LS Device Is Connected 110
1.3.10.3.4 GoRESET 112
1.3.10.3.4.1 HResetting an HS Device 112
1.3.10.3.4.2 Device Error Chirp Detection 115
1.3.10.3.4.2.1 If Chirp Completion Disable (H_NegoControl_1.DisChirpFinish) Setting Is 0 115
1.3.10.3.4.2.2 If Chirp Completion Disable (H_NegoControl_1.DisChirpFinish) Setting is 1 117
1.3.10.3.4.3 Resetting an FS Device 120
1.3.10.3.4.4 Resetting an LS Device 122
1.3.10.3.5 GoOPERATIONAL 123
1.3.10.3.6 GoSUSPEND 124
1.3.10.3.6.1 If HS Device Is Connected 124
1.3.10.3.6.2 If FS Device Is Connected 126
1.3.10.3.6.3 If LS Device Is Connected 128
1.3.10.3.7 GoRESUME 130
1.3.10.3.7.1 If an HS Device Is Connected 130
1.3.10.3.7.2 If FS Device Is Connected 132
1.3.10.3.7.3 If an LS Device Is Connected 134

1.3.10.3.8 GoWAIT_CONNECTtoDIS 136
1.3.10.3.9 GoWAIT_CONNECTtoOP 137
1.3.10.3.9.1 If HS Device Is Connected 137
1.3.10.3.9.2 If FS or LS Device Is Connected 139
1.3.10.3.10 GoRESETtoOP 141
1.3.10.3.10.1 If HS Device Is Connected 141
1.3.10.3.10.2 If FS or LS Device Is Connected 142
1.3.10.3.11 GoSUSPENDtoOP 143
1.3.10.3.12 GoRESUMEtoOP 145
1.4 Media Data Transfer Function 146
1.4.1 Media Data 146
1.4.2 Media Data Transfers 146
1.4.3 Reduced Power Consumption 147
1.5 Power Management Functions 149
1.5.1 2-port Mode (ClkSelect.Port1x2 Is Cleared to 0) 149
1.5.1.1 SLEEP 150
1.5.1.2 SNOOZE 151
1.5.1.3 ACTIVE60 152
1.5.1.4 ACT_DEVICE 152
1.5.1.5 ACT_HOST 153
1.5.1.6 ACT_ALL 153
1.5.2 1-port Mode (ClkSelect.Port1x2 Set to 1) 154
1.5.2.1 SLEEP 155
1.5.2.2 SNOOZE 155
1.5.2.3 ACTIVE60 156
1.5.2.4 ACT_DEVICE 156
1.5.2.5 ACT_HOST 156
1.5.2.6 ACT_ALL 156
1.6 FIFO Management 157
1.6.1 FIFO Memory Map 157
1.6.2 Descriptor Area 158
1.6.2.1 Writing Data to Descriptor Area 158
1.6.2.2 Executing Data Stage (IN) with Descriptor Area 159
1.6.3 CBW Area 159
1.6.3.1 CBW Area (for USB device) 159
1.6.3.2 CBW Area (for USB host) 159
1.6.4 CSW Area 160
1.6.4.1 CSW Area (for USB device) 160
1.6.4.2 CSW Area (for USB Host) 160
1.6.5 FIFO Access Methods 160
1.6.5.1 RAM Access Methods (RAM_Rd) 160
1.6.5.2 RAM Access Methods (RAM_WrDoor) 161
1.6.5.3 FIFO Access Methods (Register Access) 161
1.6.5.4 FIFO Access Methods (DMA) 161
1.6.5.5 FIFO Access Methods (IDE) 162
1.6.5.6 FIFO Access Restrictions 162

1.7 Media FIFO Management 163
1.7.1 Media FIFO 163
1.7.2 Media FIFO Access Methods 163
1.7.2.1 Media FIFO Access Methods (Register Access) 163
1.7.2.2 Media FIFO Access Methods (DMA) 163
1.7.2.3 Media FIFO Access Methods (IDE) 164
1.7.2.4 Media FIFO Access Restrictions 164
1.8 CPUIF 165
1.8.1 CPUIF Mode 165
1.8.2 CPUIF Mode Setup 166
1.8.3 Block Configuration 167
1.8.3.1 REG (S2R72C05 Registers) 168
1.8.3.1.1 Synchronous Register Access (Write) 168
1.8.3.1.2 Synchronous Register Access (Read) 168
1.8.3.1.3 FIFO Access (Write) 168
1.8.3.1.4 FIFO Access (Read) 169
1.8.3.1.5 FIFO Access Fractional Number Processing 169
1.8.3.1.6 RAM_Rd Access 172
1.8.3.1.7 Asynchronous Register Access (Writing) 172
1.8.3.1.8 Asynchronous Register Access (Reading) 172
1.8.3.2 DMA0/DMA1(DMA ch.0/ch.1) 173
1.8.3.2.1 Basic Functions 173
1.8.3.2.2 Terminal Setup 175
1.8.3.2.3 Count mode (Write) 175
1.8.3.2.4 Count mode (Read) 177
1.8.3.2.5 Free-run Mode (Write) 178
1.8.3.2.6 Free-run Mode (Read) 179
1.8.3.2.7 REQ Assert Count Option (Write) 179
1.8.3.2.8 REQ Assert Count Option (Read) 180
1.8.3.2.9 DMA FIFO Access Fractional Number Processing 181
1.9 Timer 182
1.9.1 Operating Mode 182
1.9.2 Operation Start/Stop 182
1.10 IDE I/F 183
1.10.1 IDE Task File Register Access 183
1.10.1.1 Reading from IDE Task File Register 183
1.10.1.2 Writing to IDE Task File Register 183
1.10.1.3 Sequential Writing to IDE Task File Register 184
1.10.1.4 Auto Status Register Reading from IDE Task File Register 184
1.10.2 PIO Access 185
1.10.2.1 PIO Read DMA 185
1.10.2.2 PIO Write DMA 185
1.10.3 Multi-Word DMA 186
1.10.3.1 Multi-Word DMA Read 186
1.10.3.2 Multi-Word DMA Write 186
1.10.4 Ultra DMA 187

1.10.4.1 Ultra DMA Read 187
1.10.4.2 Ultra DMA Write 187
1.10.5 IDE Transfer Mode Settings 189
1.11 Boundary Scan (JTAG) 191
1.11.1 Supported Instructions 191
1.11.2 DEVICE_CODE 191
1.11.3 Terminals Excluded from Boundary Scan 191

2 REGISTERS 192

2.1 Initial Register Map 192
2.2 Device/Host Common Register Map 193
2.3 Device Register Map 201
2.4 Host Register Map 207
2.5 Initial Register Details 213
2.5.1 xxxh *CPUIF_MODE (CPU I/F Mode)* 213
2.6 Device/Host Register Details 214
2.6.1 000h *MainIntStat (Main Interrupt Status)* 214
2.6.2 001h *DeviceIntStat (Device Interrupt Status)* 216
2.6.3 002h *HostIntStat (Host Interrupt Status)* 218
2.6.4 003h CPU_IntStat (CPU Interrupt Status) 220
2.6.5 004h IDE_IntStat (IDE Interrupt Status) 222
2.6.6 005h MediaFIFO_IntStat (Media FIFO Interrupt Status) 224
2.6.7 010h *MainIntEnb (Main Interrupt Enable)* 225
2.6.8 011h *DeviceIntEnb (Device Interrupt Enable)* 226
2.6.9 012h *HostIntEnb (Host Interrupt Enable)* 227
2.6.10 013h CPU_IntEnb (CPU Interrupt Enable) 228
2.6.11 014h IDE_IntEnb (IDE Interrupt Enable) 229
2.6.12 015h MediaFIFO_IntEnb (Media FIFO Interrupt Enable) 230
2.6.13 020h *RevisionNum (Revision Number)* 231
2.6.14 021h *ChipReset (Chip Reset)* 232
2.6.15 022h *PM_Control_0 (Power Management Control 0)* 233
2.6.16 023h *PM_Control_1 (Power Management Control 1)* 238
2.6.17 024h *WakeupTim_H (Wakeup Time High)* 239
2.6.18 025h *WakeupTim_L (Wakeup Time Low)* 239
2.6.19 026h *H_USB_Control (Host USB Control)* 240
2.6.20 027h *H_XcvrControl (Host Xcvr Control)* 241
2.6.21 028h *D_USB_Status (Device USB Status)* 243
2.6.22 029h *H_USB_Status (Host USB Status)* 244
2.6.23 030h FIFO_Rd_0 (FIFO Read 0) 245
2.6.24 031h FIFO_Rd_1 (FIFO Read 1) 245
2.6.25 032h FIFO_Wr_0(FIFO Write 0) 246
2.6.26 033h FIFO_Wr_1(FIFO Write 1) 246
2.6.27 034h FIFO_RdRemain_H (FIFO Read Remain High) 247
2.6.28 035h FIFO_RdRemain_L (FIFO Read Remain Low) 247
2.6.29 036h FIFO_WrRemain_H (FIFO Write Remain High) 248
2.6.30 037h FIFO_WrRemain_L (FIFO Write Remain Low) 248

2.6.31 038h FIFO_ByteRd(FIFO Byte Read) 249
2.6.32 03Ah FIFO_ByteWr(FIFO Byte Write) 250
2.6.33 040h RAM_RdAdrs_H (RAM Read Address High) 251
2.6.34 041h RAM_RdAdrs_L (RAM Read Address Low) 251
2.6.35 042h RAM_RdControl (RAM Read Control) 252
2.6.36 043h RAM_RdCount (RAM Read Counter) 253
2.6.37 044h RAM_WrAdrs_H (RAM Write Address High) 254
2.6.38 045h RAM_WrAdrs_L (RAM Write Address Low) 254
2.6.39 046h RAM_WrDoor_0 (RAM Write Door 0) 255
2.6.40 047h RAM_WrDoor_1 (RAM Write Door 1) 255
2.6.41 048h MediaFIFO_Control (Media FIFO Control) 256
2.6.42 049h ClrAllMediaFIFO_Join (Clear All Media FIFO Join) 257
2.6.43 04Ah MediaFIFO_Join (Media FIFO Join) 258
2.6.44 050h RAM_Rd_00 (RAM Read 00) 259
2.6.45 051h RAM_Rd_01 (RAM Read 01) 259
2.6.46 052h RAM_Rd_02 (RAM Read 02) 259
2.6.47 053h RAM_Rd_03 (RAM Read 03) 259
2.6.48 054h RAM_Rd_04 (RAM Read 04) 259
2.6.49 055h RAM_Rd_05 (RAM Read 05) 259
2.6.50 056h RAM_Rd_06 (RAM Read 06) 259
2.6.51 057h RAM_Rd_07 (RAM Read 07) 259
2.6.52 058h RAM_Rd_08 (RAM Read 08) 259
2.6.53 059h RAM_Rd_09 (RAM Read 09) 259
2.6.54 05Ah RAM_Rd_0A (RAM Read 0A) 259
2.6.55 05Bh RAM_Rd_0B (RAM Read 0B) 259
2.6.56 05Ch RAM_Rd_0C (RAM Read 0C) 259
2.6.57 05Dh RAM_Rd_0D (RAM Read 0D) 259
2.6.58 05Eh RAM_Rd_0E (RAM Read 0E) 259
2.6.59 05Fh RAM_Rd_0F (RAM Read 0F) 259
2.6.60 060h RAM_Rd_10 (RAM Read 10) 259
2.6.61 061h RAM_Rd_11 (RAM Read 11) 259
2.6.62 062h RAM_Rd_12 (RAM Read 12) 259
2.6.63 063h RAM_Rd_13 (RAM Read 13) 259
2.6.64 064h RAM_Rd_14 (RAM Read 14) 259
2.6.65 065h RAM_Rd_15 (RAM Read 15) 259
2.6.66 066h RAM_Rd_16 (RAM Read 16) 259
2.6.67 067h RAM_Rd_17 (RAM Read 17) 259
2.6.68 068h RAM_Rd_18 (RAM Read 18) 259
2.6.69 069h RAM_Rd_19 (RAM Read 19) 259
2.6.70 06Ah RAM_Rd_1A (RAM Read 1A) 259
2.6.71 06Bh RAM_Rd_1B (RAM Read 1B) 259
2.6.72 06Ch RAM_Rd_1C (RAM Read 1C) 259
2.6.73 06Dh RAM_Rd_1D (RAM Read 1D) 259
2.6.74 06Eh RAM_Rd_1E (RAM Read 1E) 259
2.6.75 06Fh RAM_Rd_1F (RAM Read 1F) 259
2.6.76 071h DMA0_Config (DMA0 Config) 261

2.6.77 072h DMA0_Control (DMA0 Control) 263
2.6.78 074h DMA0_Remain_H (DMA0 FIFO Remain High) 264
2.6.79 075h DMA0_Remain_L (DMA0 FIFO Remain Low) 264
2.6.80 078h DMA0_Count_HH (DMA0 Transfer Byte Counter High/High) 265
2.6.81 079h DMA0_Count_HL (DMA0 Transfer Byte Counter High/Low) 265
2.6.82 07Ah DMA0_Count_LH (DMA0 Transfer Byte Counter Low/High) 265
2.6.83 07Bh DMA0_Count_LL (DMA0 Transfer Byte Counter Low/Low) 265
2.6.84 07Ch DMA0_RdData_0 (DMA0 Read Data 0) 267
2.6.85 07Dh DMA0_RdData_1 (DMA0 Read Data 1) 267
2.6.86 07Eh DMA0_WrData_0 (DMA0 Write Data 0) 268
2.6.87 07Fh DMA0_WrData_1 (DMA0 Write Data 1) 268
2.6.88 081h DMA1_Config (DMA1 Config) 269
2.6.89 082h DMA1_Control (DMA1 Control) 271
2.6.90 084h DMA1_Remain_H (DMA1 FIFO Remain High) 272
2.6.91 085h DMA1_Remain_L (DMA1 FIFO Remain Low) 272
2.6.92 088h DMA1_Count_HH (DMA1 Transfer Byte Counter High/High) 273
2.6.93 089h DMA1_Count_HL (DMA1 Transfer Byte Counter High/Low) 273
2.6.94 08Ah DMA1_Count_LH (DMA1 Transfer Byte Counter Low/High) 273
2.6.95 08Bh DMA1_Count_LL (DMA1 Transfer Byte Counter Low/Low) 273
2.6.96 08Ch DMA1_RdData_0 (DMA1 Read Data 0) 275
2.6.97 08Dh DMA1_RdData_1 (DMA1 Read Data 1) 275
2.6.98 08Eh DMA1_WrData_0 (DMA1 Write Data 0) 276
2.6.99 08Fh DMA1_WrData_1 (DMA1 Write Data 1) 276
2.6.100 090h IDE_Status (IDE Status) 277
2.6.101 091h IDE_Control (IDE Control) 278
2.6.102 092h IDE_Config_0 (IDE Configuration 0) 279
2.6.103 093h IDE_Config_1 (IDE Configuration 1) 280
2.6.104 094h IDE_Rmod (IDE Register Mode) 281
2.6.105 095h IDE_Tmod (IDE Transfer Mode) 282
2.6.106 096h IDE_Umod (IDE Ultra-DMA Transfer Mode) 283
2.6.107 09Ah IDE_CRC_H (IDE CRC High) 284
2.6.108 09Bh IDE_CRC_L (IDE CRC Low) 284
2.6.109 09Dh IDE_Count_H (IDE Transfer Byte Counter High) 285
2.6.110 09Eh IDE_Count_M (IDE Transfer Byte Counter Middle) 285
2.6.111 09Fh IDE_Count_L (IDE Transfer Byte Counter Low) 285
2.6.112 0A0h IDE_RegAdrs (IDE Register Address) 286
2.6.113 0A2h IDE_RdRegValue_0 (IDE Register Read Value 0) 287
2.6.114 0A3h IDE_RdRegValue_1 (IDE Register Read Value 1) 287
2.6.115 0A4h IDE_WrRegValue_0 (IDE Register Write Value 0) 288
2.6.116 0A5h IDE_WrRegValue_1 (IDE Register Write Value 1) 288
2.6.117 0A6h IDE_SeqWrRegControl (IDE Sequential Register Write Control) 289
2.6.118 0A7h IDE_SeqWrRegCnt (IDE Sequential Register Write Counter) 290
2.6.119 0A8h IDE_SeqWrRegAdrs (IDE Sequential Register Write Address FIFO) 291
2.6.120 0A9h IDE_SeqWrRegValue (IDE Sequential Register Write Value FIFO) 292
2.6.121 0ACh IDE_RegConfig (IDE Register Configuration) 293
2.6.122 0B1h *HostDeviceSel (Host Device Select)* 294

2.6.123 0B3h *ModeProtect (Mode Protection)* 295
2.6.124 0B5h *ClkSelect (Clock Select)* 296
2.6.125 0B7h *ChipConfig (Chip Configuration)* 298
2.6.126 0BDh TimerConfig (Timer Configuration) 300
2.6.127 0BEh TimerSet_H (Timer Set High) 301
2.6.128 0BFh TimerSet_L (Timer Set Low) 301
2.6.129 0C0h AREAnIntStat (AREAn Interrupt Status) 302
2.6.130 0C1h AREA0IntStat (AREA0 Interrupt Status) 304
2.6.131 0C2h AREA1IntStat (AREA1 Interrupt Status) 304
2.6.132 0C3h AREA2IntStat (AREA2 Interrupt Status) 304
2.6.133 0C4h AREA3IntStat (AREA3 Interrupt Status) 304
2.6.134 0C5h AREA4IntStat (AREA4 Interrupt Status) 304
2.6.135 0C6h AREA5IntStat (AREA5 Interrupt Status) 304
2.6.136 0C8h AREAnIntEnb (AREAn Interrupt Enable) 305
2.6.137 0C9h AREA0IntEnb (AREA0 Interrupt Enable) 306
2.6.138 0CAh AREA1IntEnb (AREA1 Interrupt Enable) 306
2.6.139 0CBh AREA2IntEnb (AREA2 Interrupt Enable) 306
2.6.140 0CCh AREA3IntEnb (AREA3 Interrupt Enable) 306
2.6.141 0CDh AREA4IntEnb (AREA4 Interrupt Enable) 306
2.6.142 0CEh AREA5IntEnb (AREA5 Interrupt Enable) 306
2.6.143 0D0h AREA0Join_0 (AREA 0 Join 0) 307
2.6.144 0D2h AREA1Join_0 (AREA 1 Join 0) 307
2.6.145 0D4h AREA2Join_0 (AREA 2 Join 0) 307
2.6.146 0D6h AREA3Join_0 (AREA3 Join 0) 307
2.6.147 0D8h AREA4Join_0 (AREA 4 Join 0) 307
2.6.148 0DAh AREA5Join_0 (AREA 5 Join 0) 307
2.6.149 0D1h AREA0Join_1 (AREA 0 Join 1) 309
2.6.150 0D3h AREA1Join_1 (AREA 1 Join 1) 309
2.6.151 0D5h AREA2Join_1 (AREA 2 Join 1) 309
2.6.152 0D7h AREA3Join_1 (AREA 3 Join 1) 309
2.6.153 0D9h AREA4Join_1 (AREA 4 Join 1) 309
2.6.154 0DBh AREA5Join_1 (AREA 5 Join 1) 309
2.6.155 0DEh ClrAREAnJoin_0 (Clear AREA n Join 0) 311
2.6.156 0DFh ClrAREAnJoin_1 (Clear AREA 1 Join 1) 312
2.6.157 180h AREA0StartAdrs_H (AREA 0 Start Address High) 313
2.6.158 181h AREA0StartAdrs_L (AREA 0 Start Address Low) 313
2.6.159 184h AREA1StartAdrs_H (AREA 1 Start Address High) 313
2.6.160 185h AREA1StartAdrs_L (AREA 1 Start Address Low) 313
2.6.161 188h AREA2StartAdrs_H (AREA2 Start Address High) 313
2.6.162 189h AREA2StartAdrs_L (AREA2 Start Address Low) 313
2.6.163 18Ch AREA3StartAdrs_H (AREA3 Start Address High) 313
2.6.164 18Dh AREA3StartAdrs_L (AREA3 Start Address Low) 313
2.6.165 190h AREA4StartAdrs_H (AREA4 Start Address High) 313
2.6.166 191h AREA4StartAdrs_L (AREA4 Start Address Low) 313
2.6.167 194h AREA5StartAdrs_H (AREA5 Start Address High) 313
2.6.168 195h AREA5StartAdrs_L (AREA5 Start Address Low) 313

2.6.169 182h AREA0EndAdrs_H (AREA 0 End Address High) ... 315
2.6.170 183h AREA0EndAdrs_L (AREA 0 End Address Low) ... 315
2.6.171 186h AREA1EndAdrs_H (AREA1 End Address High) ... 315
2.6.172 187h AREA1EndAdrs_L (AREA1 End Address Low) ... 315
2.6.173 18Ah AREA2EndAdrs_H (AREA2 End Address High) ... 315
2.6.174 18Bh AREA2EndAdrs_L (AREA2 End Address Low) ... 315
2.6.175 18Eh AREA3EndAdrs_H (AREA3 End Address High) ... 315
2.6.176 18Fh AREA3EndAdrs_L (AREA3 End Address Low) ... 315
2.6.177 192h AREA4EndAdrs_H (AREA4 End Address High) ... 315
2.6.178 193h AREA4EndAdrs_L (AREA4 End Address Low) ... 315
2.6.179 196h AREA5EndAdrs_H (AREA5 End Address High) ... 315
2.6.180 197h AREA5EndAdrs_L (AREA5 End Address Low) ... 315
2.6.181 19Fh AREAnFIFO_Clr (AREAn FIFO Clear) ... 317
2.7 Device Register Details ... 318
2.7.1 0E0h *D_SIE_IntStat (Device SIE Interrupt Status)* ... 318
2.7.2 0E2h D_FIFO_IntStat (Device FIFO Interrupt Status) ... 321
2.7.3 0E3h D_BulkIntStat (Device Bulk Interrupt Status) ... 323
2.7.4 0E4h D_EPrIntStat (Device EPr Interrupt Status) ... 324
2.7.5 0E5h D_EP0IntStat (Device EP0 Interrupt Status) ... 325
2.7.6 0E6h D_EPaIntStat (Device EPa Interrupt Status) ... 325
2.7.7 0E7h D_EPbIntStat (Device EPb Interrupt Status) ... 325
2.7.8 0E8h D_EPcIntStat (Device EPc Interrupt Status) ... 325
2.7.9 0E9h D_EPdIntStat (Device EPd Interrupt Status) ... 325
2.7.10 0EAh D_EPeIntStat (Device EPe Interrupt Status) ... 325
2.7.11 0F0h *D_SIE_IntEnb (Device SIE Interrupt Enable)* ... 327
2.7.12 0F2h D_FIFO_IntEnb (Device FIFO Interrupt Enable) ... 328
2.7.13 0F3h D_BulkIntEnb (Device Bulk Interrupt Enable) ... 329
2.7.14 0F4h D_EPrIntEnb (Device EPr Interrupt Enable) ... 330
2.7.15 0F5h D_EP0IntEnb (Device EP0 Interrupt Enable) ... 331
2.7.16 0F6h D_EPaIntEnb (Device EPa Interrupt Enable) ... 331
2.7.17 0F7h D_EPbIntEnb (Device EPb Interrupt Enable) ... 331
2.7.18 0F8h D_EPcIntEnb (Device EPc Interrupt Enable) ... 331
2.7.19 0F9h D_EPdIntEnb (Device EPd Interrupt Enable) ... 331
2.7.20 0FAh D_EPeIntEnb (Device EPe Interrupt Enable) ... 331
2.7.21 100h *D_Reset (Device Reset)* ... 332
2.7.22 102h D_NegoControl (Device Negotiation Control) ... 333
2.7.23 105h D_XcvrControl (Device Xcvr Control) ... 335
2.7.24 106h D_USB_Test (Device USB_Test) ... 336
2.7.25 108h D_EPnControl (Device Endpoint Control) ... 338
2.7.26 10Ah D_BulkOnlyControl (Device BulkOnly Control) ... 339
2.7.27 10Bh D_BulkOnlyConfig (Device BulkOnly Configuration) ... 340
2.7.28 110h D_EP0SETUP_0 (Device EP0 SETUP 0) ... 342
2.7.29 111h D_EP0SETUP_1 (Device EP0 SETUP 1) ... 342
2.7.30 112h D_EP0SETUP_2 (Device EP0 SETUP 2) ... 342
2.7.31 113h D_EP0SETUP_3 (Device EP0 SETUP 3) ... 342
2.7.32 114h D_EP0SETUP_4 (Device EP0 SETUP 4) ... 342

2.7.33 115h D_EP0SETUP_5 (Device EP0 SETUP 5) ... 342
2.7.34 116h D_EP0SETUP_6 (Device EP0 SETUP 6) ... 342
2.7.35 117h D_EP0SETUP_7 (Device EP0 SETUP 7) ... 342
2.7.36 118h D_USB_Address (Device USB Address) ... 343
2.7.37 11Ah D_SETUP_Control(Device SETUP Control) ... 344
2.7.38 11Eh D_FrameNumber_H (Device FrameNumber High) ... 345
2.7.39 11Fh D_FrameNumber_L (Device FrameNumber Low) ... 345
2.7.40 120h D_EP0MaxSize (Device EP0 Max Packet Size) ... 346
2.7.41 121h D_EP0Control (Device EP0 Control) ... 347
2.7.42 122h D_EP0ControlIN (Device EP0 Control IN) ... 349
2.7.43 123h D_EP0ControlOUT (Device EP0 Control OUT) ... 351
2.7.44 130h D_EPaMaxSize_H (Device EPa Max Packet Size High) ... 353
2.7.45 131h D_EPaMaxSize_L (Device EPa Max Packet Size Low) ... 353
2.7.46 140h D_EPbMaxSize_H (Device EPb Max Packet Size High) ... 353
2.7.47 141h D_EPbMaxSize_L (Device EPb Max Packet Size Low) ... 353
2.7.48 150h D_EPcMaxSize_H (Device EPc Max Packet Size High) ... 353
2.7.49 151h D_EPcMaxSize_L (Device EPc Max Packet Size Low) ... 353
2.7.50 1A0h D_EPdMaxSize_H (Device EPd Max Packet Size High) ... 353
2.7.51 1A1h D_EPdMaxSize_L (Device EPd Max Packet Size Low) ... 353
2.7.52 1B0h D_EPeMaxSize_H (Device EPe Max Packet Size High) ... 353
2.7.53 1B1h D_EPeMaxSize_L (Device EPe Max Packet Size Low) ... 353
2.7.54 132h D_EPaConfig_0 (Device EPa Configuration 0) ... 355
2.7.55 142h D_EPbConfig_0 (Devie EPb Configuration 0) ... 355
2.7.56 152h D_EPcConfig_0 (Device EPc Configuration 0) ... 355
2.7.57 1A2h D_EPdConfig_0 (Device EPd Configuration 0) ... 355
2.7.58 1B2h D_EPeConfig_0 (Device EPe Configuration 0) ... 355
2.7.59 134h D_EPaControl (Device EPa Control) ... 357
2.7.60 144h D_EPbControl (Device EPb Control) ... 357
2.7.61 154h D_EPcControl (Device EPc Control) ... 357
2.7.62 1A4h D_EPdControl (Device EPd Control) ... 357
2.7.63 1B4h D_EPeControl (Device EPe Control) ... 357
2.7.64 160h D_DescAdrs_H (Device Descriptor Address High) ... 359
2.7.65 161h D_DescAdrs_L (Device Descriptor Address Low) ... 359
2.7.66 162h D_DescSize_H (Device Descriptor Size High) ... 360
2.7.67 163h D_DescSize_L (Device Descriptor Size Low) ... 360
2.7.68 170h D_DMA0_FIFO_Control (Device DMA0 FIFO Control) ... 361
2.7.69 172h D_DMA1_FIFO_Control (Device DMA1 FIFO Control) ... 362
2.7.70 1BEh *DTM_Config (Device Transceiver Macro Config)* ... 363
2.7.71 1E1h D_ModeControl (Address Mode Control) ... 364
2.8 Host Register Details ... 365
2.8.1 0E0h H_SIE_IntStat_0 (Host SIE Interrupt Status 0) ... 365
2.8.2 0E1h H_SIE_IntStat_1 (SIE Host Interrupt Status 1) ... 367
2.8.3 0E2h H_FIFO_IntStat (Host FIFO Interrupt Status) ... 369
2.8.4 0E3h H_FrameIntStat (Host Frame Interrupt Status) ... 371
2.8.5 0E4h H_CHrIntStat (Host CHr Interrupt Status) ... 372
2.8.6 0E5h H_CH0IntStat (Host CH0 Interrupt Status) ... 373

2.8.7 0E6h H_CHaIntStat (Host CHa Interrupt Status) 375
2.8.8 0E7h H_CHbIntStat (Host CHb Interrupt Status) 377
2.8.9 0E8h H_CHcIntStat (Host CHc Interrupt Status) 377
2.8.10 0E9h H_CHdIntStat (Host CHd Interrupt Status) 377
2.8.11 0EAh H_CHeIntStat (Host CHe Interrupt Status) 377
2.8.12 0F0h H_SIE_IntEnb_0 (Host SIE Interrupt Enable) 379
2.8.13 0F1h H_SIE_IntEnb_1 (SIE Host Interrupt Enable 1) 380
2.8.14 0F2h H_FIFO_IntEnb (Host FIFO Interrupt Enable) 381
2.8.15 0F3h H_FrameIntEnb (Host Frame Interrupt Enable) 382
2.8.16 0F4h H_CHrIntEnb (Host CHr Interrupt Enable) 383
2.8.17 0F5h H_CH0IntEnb (Host CH0 Interrupt Enable) 384
2.8.18 0F6h H_CHaIntEnb (Host CHa Interrupt Enable) 385
2.8.19 0F7h H_CHbIntEnb (Host CHb Interrupt Enable) 386
2.8.20 0F8h H_CHcIntEnb (Host CHc Interrupt Enable) 386
2.8.21 0F9h H_CHdIntEnb (Host CHd Interrupt Enable) 386
2.8.22 0FAh H_CHeIntEnb (Host CHe Interrupt Enable) 386
2.8.23 100h *H_Reset (Host Reset)* 387
2.8.24 102h H_NegoControl_0 (Host NegoControl 0) 388
2.8.25 104h H_NegoControl_1 (Host NegoControl 1) 390
2.8.26 106h H_USB_Test (Host USB_Test) 391
2.8.27 110h H_CH0SETUP_0 (Host CH0 SETUP 0) 393
2.8.28 111h H_CH0SETUP_1 (Host CH0 SETUP 1) 393
2.8.29 112h H_CH0SETUP_2 (Host CH0 SETUP 2) 393
2.8.30 113h H_CH0SETUP_3 (Host CH0 SETUP 3) 393
2.8.31 114h H_CH0SETUP_4 (Host CH0 SETUP 4) 393
2.8.32 115h H_CH0SETUP_5 (Host CH0 SETUP 5) 393
2.8.33 116h H_CH0SETUP_6 (Host CH0 SETUP 6) 393
2.8.34 117h H_CH0SETUP_7 (Host CH0 SETUP 7) 393
2.8.35 11Eh H_FrameNumber_H (Host FrameNumber High) 394
2.8.36 11Fh H_FrameNumber_L (Host FrameNumber Low) 394
2.8.37 120h H_CH0Config_0 (Host Channel 0 Configuration0) 395
2.8.38 121h H_CH0Config_1 (Host Channel 0 Configuration1) 397
2.8.39 123h H_CH0MaxPktSize (Host Channel 0 Max Packet Size) 398
2.8.40 126h H_CH0TotalSize_H (Host Channel 0 Total Size High) 399
2.8.41 127h H_CH0TotalSize_L (Host Channel 0 Total Size Low) 399
2.8.42 128h H_CH0HubAdrs (Host Channel 0 Hub Address) 400
2.8.43 129h H_CH0FuncAdrs (Host Channel 0 Function Address) 401
2.8.44 12Bh H_CTL_SupportControl (Host ControlTransfer Support Control) 402
2.8.45 12Eh H_CH0ConditionCode (Host Channel 0 Condition Code) 404
2.8.46 130h H_CHaConfig_0 (Host Channel a Configuration0) 405
2.8.47 131h H_CHaConfig_1 (Host Channel a Configuration1) 407
2.8.48 132h H_CHaMaxPktSize_H (Host Channel a Max Packet Size High) 408
2.8.49 133h H_CHaMaxPktSize_L (Host Channel a Max Packet Size Low) 408
2.8.50 134h H_CHaTotalSize_HH (Host Channel a Total Size High-High) 409
2.8.51 135h H_CHaTotalSize_HL (Host Channel a Total Size High-Low) 409
2.8.52 136h H_CHaTotalSize_LH (Host Channel a Total Size Low-High) 409

2.8.53 137h H_CHaTotalSize_LL (Host Channel a Total Size Low-Low) 409
2.8.54 138h H_CHaHubAdrs (Host Channel a Hub Address) 411
2.8.55 139h H_CHaFuncAdrs (Host Channel a Function Address) 412
2.8.56 13Ah H_CHaBO_SupporotControl (Host Bulk Only Transfer Supporot Control) 413
2.8.57 13Bh H_CHaBO_CSW_RcvDataSize (Host CSW Receive Data Size) 415
2.8.58 13Ch H_CHaBO_OUT_EP_Control (Host OUT Endpoint Control) 416
2.8.59 13Dh H_CHaBO_IN_EP_Control (Host IN Endpoint Control) 417
2.8.60 13Eh H_CHaConditionCode (Host Channel a Condition Code) 418
2.8.61 140h H_CHbConfig_0 (Host Channel b Configuration0) 419
2.8.62 150h H_CHcConfig_0 (Host Channel c Configuration0) 419
2.8.63 160h H_CHdConfig_0 (Host Channel d Configuration0) 419
2.8.64 170h H_CHeConfig_0 (Host Channel e Configuration0) 419
2.8.65 141h H_CHbConfig_1 (Host Channel b Configuration1) 421
2.8.66 151h H_CHcConfig_1 (Host Channel c Configuration1) 421
2.8.67 161h H_CHdConfig_1 (Host Channel d Configuration1) 421
2.8.68 171h H_CHeConfig_1 (Host Channel e Configuration1) 421
2.8.69 142h H_CHbMaxPktSize_H (Host Channel b Max Packet Size High) 423
2.8.70 143h H_CHbMaxPktSize_L (Host Channel b Max Packet Size Low) 423
2.8.71 152h H_CHcMaxPktSize_H (Host Channel c Max Packet Size High) 423
2.8.72 153h H_CHcMaxPktSize_L (Host Channel c Max Packet Size Low) 423
2.8.73 162h H_CHdMaxPktSize_H (Host Channel d Max Packet Size High) 423
2.8.74 163h H_CHdMaxPktSize_L (Host Channel d Max Packet Size Low) 423
2.8.75 172h H_CHeMaxPktSize_H (Host Channel e Max Packet Size High) 423
2.8.76 173h H_CHeMaxPktSize_L (Host Channel e Max Packet Size Low) 423
2.8.77 144h H_CHbTotalSize_HH (Host Channel b Total Size High-High) 425
2.8.78 145h H_CHbTotalSize_HL (Host Channel b Total Size High-Low) 425
2.8.79 146h H_CHbTotalSize_LH (Host Channel b Total Size Low-High) 425
2.8.80 147h H_CHbTotalSize_LL (Host Channel b Total Size Low-Low) 425
2.8.81 154h H_CHcTotalSize_HH (Host Channel c Total Size High-High) 425
2.8.82 155h H_CHcTotalSize_HL (Host Channel c Total Size High-Low) 425
2.8.83 156h H_CHcTotalSize_LH (Host Channel c Total Size Low-High) 425
2.8.84 157h H_CHcTotalSize_LL (Host Channel c Total Size Low-Low) 425
2.8.85 164h H_CHdTotalSize_HH (Host Channel d Total Size High-High) 425
2.8.86 165h H_CHdTotalSize_HL (Host Channel d Total Size High-Low) 425
2.8.87 166h H_CHdTotalSize_LH (Host Channel d Total Size Low-High) 425
2.8.88 167h H_CHdTotalSize_LL (Host Channel d Total Size Low-Low) 425
2.8.89 174h H_CHeTotalSize_HH (Host Channel e Total Size High-High) 425
2.8.90 175h H_CHeTotalSize_HL (Host Channel e Total Size High-Low) 425
2.8.91 176h H_CHeTotalSize_LH (Host Channel e Total Size Low-High) 425
2.8.92 177h H_CHeTotalSize_LL (Host Channel e Total Size Low-Low) 425
2.8.93 148h H_CHbHubAdrs (Host Channel b Hub Address) 428
2.8.94 158h H_CHcHubAdrs (Host Channel c Hub Address) 428
2.8.95 168h H_CHdHubAdrs (Host Channel d Hub Address) 428
2.8.96 178h H_CHeHubAdrs (Host Channel e Hub Address) 428
2.8.97 149h H_CHbFuncAdrs (Host Channel b Function Address) 429
2.8.98 159h H_CHcFuncAdrs (Host Channel c Function Address) 429

2.8.99 169h H_CHdFuncAdrs (Host Channel d Function Address) 429
2.8.100 179h H_CHeFuncAdrs (Host Channel e Function Address) 429
2.8.101 14Ah H_CHbInterval_H (Host Channel b Interval High) 430
2.8.102 14Bh H_CHbInterval_L (Host Channel b Interval Low) 430
2.8.103 15Ah H_CHcInterval_H (Host Channel c Interval High) 430
2.8.104 15Bh H_CHcInterval_L (Host Channel c Interval Low) 430
2.8.105 16Ah H_CHdInterval_H (Host Channel d Interval High) 430
2.8.106 16Bh H_CHdInterval_L (Host Channel d Interval Low) 430
2.8.107 17Ah H_CHeInterval_H (Host Channel e Interval High) 430
2.8.108 17Bh H_CHeInterval_L (Host Channel e Interval Low) 430
2.8.109 14Ch H_CHbTranPause(Host Channel b Transaction Pause) 432
2.8.110 15Ch H_CHcTranPause(Host Channel c Transaction Pause) 432
2.8.111 16Ch H_CHdTranPause(Host Channel d Transaction Pause) 432
2.8.112 17Ch H_CHeTranPause(Host Channel e Transaction Pause) 432
2.8.113 14Eh H_CHbConditionCode (Host Channel b Condition Code) 433
2.8.114 15Eh H_CHcConditionCode (Host Channel c Condition Code) 433
2.8.115 16Eh H_CHdConditionCode (Host Channel d Condition Code) 433
2.8.116 17Eh H_CHeConditionCode (Host Channel e Condition Code) 433
2.8.117 1B0h H_TriggerFrameNum_H (Host Trigger Frame Number High) 434
2.8.118 1B1h H_TriggerFrameNum_L (Host Trigger Frame Number Low) 434
2.8.119 1BEh *HTM_Config (Host Transceiver Macro Config)* 435
2.8.120 1F5h H_Protect (Host Protect) 436

Appendix A IDE_Config_1.Swap Bit Settings 437

Appendix B Connection to Little-endian CPU 438

Appendix C 1-Port Mode 439

Appendix D SetAddress Request Response (for TS only) 441

1 DESCRIPTION OF FUNCTIONS

This section describes the operations of this LSI.

The registers are described according to the following rules.

- Notation for a single address register:

  Register name + register

  E.g.: “MainIntStat register”

- Notation when referring to individual bits:

  Register name.bit name + bit or bit name + bit

  E.g.: “MainIntStat.CPU_IntStat bit”

- Register for each endpoint:

  D_EPx{x=... } …register

  E.g.: “D_EPx{x=0,a-e}IntStat register”

- Register for each channel:

  H_CHx{x=...} …register

  E.g.: “H_CHx{x=0,a-e}IntStat register”

- Register for each DMA channel:

  DMAx{x=...} …register

  E.g.: “DMAx{x=0,1}Config register”

1.1 Device/Host Register Map Selection and Function Selection

When using USB with this LSI, select the register map and function for either the device register or host register.

The device/host combined register and device register bit can be accessed when the device register map is selected (device map selected). The device/host combined register and host register bit can be accessed when the host register map is selected (host map selected).

1.1.1 Register Map Selection Procedure

The register map selection for the device register and host register depend on the register bit settings.

Register map selection can be set in all power management states (refer to “1.5 Power Management Functions”): SLEEP, SNOOZE, ACTIVE60, ACT_DEVICE, ACT_HOST, and ACT_ALL. Table 1-1 shows the device/host register map selection settings.

Table 1-1 Device mode/host mode selection settings

Item	Register/bit	Description
Device/host register map selection	HostDeviceSel.HOSTxDEVICE	Either the device or host register map can be selected, and the selected register can be accessed. 1: Host map selected 0: Device map selected

1.1.2 Device/Host Function Selection Procedure

The device and host function selection procedures for this LSI differ, depending on the ClkSelect.PORT1x2 bit setting.

If the ClkSelect.PORT1x2 bit has been cleared (for 2-port mode), the functions available vary depending on the power management state linked to the power management function. For example, the device function can be used if switched to the ACT_DEVICE state by GoActDevice. Similarly, the host function can be used if switched to the ACT_HOST state by GoActHost. The PM_Control_1.MonActFunc bit for the function currently available for use here can be checked. The device function can be used if the MonActFunc bit has been reset to zero; the host function can be used if it has been set to 1. If switched to the ACT_ALL state by GoActAllDev or GoActAllHost, the device or host function available for use is indicated by the MonActFunc bit.

If the ClkSelect.PORT1x2 bit has been set (for 1-port mode), the functions can be selected in accordance with the above-mentioned register map selection. In other words, the device function is available when the HostDeviceSel.HOSTxDEVICE bit has been cleared, while the host function is available when the bit is set.

1.1.3 Port State Change Detection Status

This LSI features a USB port state detection function.

This function can be used in all power management states (refer to “1.5 Power Management Functions”): SLEEP, SNOOZE, ACTIVE60, ACT_DEVICE, ACT_HOST, and ACT_ALL. It can be used regardless of device/host mode, allowing detection of downstream port (host port) state changes in device mode and upstream port (device mode) state changes in host mode (when in 2-port mode).

Changes in host or device port state changes can be used as mode selection triggers.

1.1.3.1 Using VBUS_B Terminal Change Status

This status indicates a change has occurred in the device port VBUS_B terminal state.

Table 1-2 lists registers related to VBUS_B terminal change status.

Table 1-2 Registers related to VBUS_B terminal change status

Item	Register/bit	Description
VBUS_B terminal change status	DeviceIntStat.VBUS_Changed	Indicates a change in the device port VBUS_B terminal status.
VBUS_B terminal change status interrupt enable	DeviceIntEnb.EnVBUS_Changed	Permits or prevents MainIntStat.DeviceIntStat bit assert, depending on VBUS_B terminal change status.
VBUS_B terminal status	D_USB_Status.VBUS	Indicates the VBUS_B terminal status.

When using the VBUS_B terminal change status, the firmware uses processes (1), (2), and (4) to (7).

(1) Clears the VBUS_B terminal change status.

(2) Sets the VBUS_B terminal change status interrupt enable and sets the MainIntEnb.EnDeviceIntStat bit (when using interrupt).

(3) A VBUS_B terminal change status is issued when a change occurs in the device port VBUS_B terminal status.

(4) Confirms the VBUS_B terminal change status.

(5) Clears the VBUS_B terminal change status.

(6) Clears the VBUS_B terminal change status interrupt enable.

(7) Confirms the VBUS_B terminal status. If the VBUS_B terminal status bit is "1," VBUS is deemed to be supplied (a host or hub is connected to the device port).

1.1.3.2 Using Signal Line Change Status

This status indicates a change in the host port DP terminal or DM terminal status.

Table 1-3 lists the registers related to signal line change status.

Table 1-3 Registers related to signal line change status

Item	Register/bit	Description
Signal line change status	HostIntStat.LineStateChanged	Indicates a change in the host port DP terminal or DM terminal status.
Signal line change status interrupt enable	HostIntEnb.EnLineStateChanged	Permits or prevents MainIntStat.HostIntStat bit assert, depending on signal line change status.
VBUS error detection status	HostIntStat.VBUS_Err	Monitors the VBUSFLG_A terminal and indicates any VBUS error.
VBUS error detection status interrupt enable	HostIntEnb.EnVBUS_Err	Permits or prevents MainIntStat.HostIntStat bit assert, depending on VBUS error detection status.
Operating mode	H_XcvrControl.OpMode[1:0]	Sets the host transceiver macro (HTM) operating mode.
VBUS enable	H_USB_Control.VBUS_Enb	Outputs a signal from the VBUSEN_A terminal to enable the external USB power switch.
Host port VBUS status	H_USB_Status.VBUS_State	Indicates the host port VBUSFLG_A terminal status (normal/error).
Host port signal line status	H_USB_Status.LineState[1:0]	Indicates the USB cable signal status.
Host state transition execute	H_NegoControl_1.AutoMode[3:0]	Sets the host state for transition.

Unlike the connect detection status operating as a USB host, the signal line change status is a special status for detecting host port signal line changes when the USB host function is not used, i.e. in SLEEP, SNOOZE, ACTIVE60, ACT_DEVICE, or ACT_ALL states (refer to "1.5 Power Management Functions"). This status is asserted frequently if HostIntEnb.EnLineStateChanged is enabled when operating in USB host mode. For this reason, disable HostIntEnb.EnLineStateChanged when operating in USB host mode.

VBUS enable should be turned off and VBUS drive stopped immediately if a VBUS error detection status is detected in any mode other than USB host mode.

When using the signal line change status, the firmware uses processes (1) to (4) and (6) to (9).

(1) Clears the signal line change status and VBUS error detection status.

(2) Sets signal line change status enable and VBUS error detection status enable. Also sets the MainIntEnb.EnHostIntStat bit (when using interrupt).

(3) Sets 0x93 for the H_XcvrControl register.

(4) Turns on VBUS enable.

(5) If a device is connected to the host port, the line state changes from SE0, and a signal line change status is issued.

(6) Confirms the signal line change status.

(7) Clears the signal line change status.

(8) Clears the signal line change status enable.

(9) Confirms the host port signal line state. A device is deemed connected to the host port if "01" or "10" is indicated.

The H_XcvrControl register and VBUS_Enb bit maintain the values set here when host mode is subsequently selected. These settings are set automatically by the hardware to suit the host state set when the firmware sets the host state transition registers to the appropriate code settings. For detailed information, refer to "1.3.10 Host State Management Support Function."

1.2 USB Device Control

This section describes the USB device control functions for this LSI.

1.2.1 Endpoints

This LSI features one endpoint (EP0) for control transfer and five general endpoints (EPa, EPb, EPc, EPd, and EPe). The endpoints EPa, EPb, EPc, EPd, and EPe can be used as bulk, interrupt, or isochronous transfer endpoints.

The LSI hardware provides endpoints to control transactions but does not provide functions for managing interfaces defined by USB (USB-defined interfaces). USB-defined interfaces should be installed as firmware. The endpoints should be set and combined appropriately to form USB-defined interfaces in accordance with the device-specific descriptor definition.

Each endpoint includes fixed basic settings determined by the USB-defined interface and variable control items and status for controlling each transfer. Make the basic settings at times such as when the chip is initialized or the USB-defined interface is switched.

Table 1-4 indicates the basic settings for endpoint EP0 (default control pipe).

Endpoint EP0 shares register sets and FIFO areas for the IN and OUT directions. The appropriate data transaction direction for the endpoint EP0 data stage and status stage should be set on execution by the firmware.

The transaction can be executed by linking to a FIFO area (described later). Before setting AREA0Join_1.JoinEP0CH0, reserve the FIFO area first with AREA0StartAdrs_H,L and AREA0EndAdrs_H,L, then initialize with AREA0FIFO_Clr. Data transfers using this FIFO area are not possible until linked via this join process.

Table 1-4 Endpoint EP0 basic settings

Item	Register/bit	Description
Max packet size	D_EP0MaxSize	Sets the maximum packet size to 8, 16, 32, or 64 for FS operation and to 64 for HS operation. Use with endpoint EP0 assigned to FIFO AREA0.
FIFO area	AREA0StartAdrs_H,L AREA0EndAdrs_H,L	Specifies the FIFO address corresponding to the AREA0 area. The FIFO area should be at least as large as the max packet size.
FIFO linking	AREA0Join_1.JoinEP0CH0	Links endpoint EP0 to the AREA0 area to permit data transfers for EP0.

Table 1-5 shows the basic settings for the general endpoints (EPa, EPb, EPc, EPd, and EPe). The transaction direction and endpoint number can be set as desired for endpoints EPa, EPb, EPc, EPd, and EPe, enabling independent use of up to five endpoints. They should be set appropriately to suit the specifics of the USB-defined interface definition and enabled to form a USB-defined interface.

Transactions can be executed by linking endpoints EPa, EPb, EPc, EPd, and EPe to the corresponding FIFO areas. FIFO areas should be reserved using the AREAx{x=1-5}StartAdrs_H,L and

AREAx{x=1-5}EndAdrs_H,L registers and initialized using AREAnFIFO_Clr before setting the AREAx{x=1-5}Join_1 register as required. Transactions cannot be executed for endpoints for which this join processing has not been performed. For this reason, avoid joining endpoints that will not be used.

Table 1-5 General endpoint basic settings

Item	Register/bit	Description
Transaction direction	D_EPx{x=a-e}Config_0.INxOUT	Sets the transfer direction for each endpoint.
Max packet size	D_EPx{x=a-e}MaxSize_H, D_EPx{x=a-e}MaxSize_L	Sets the maximum packet size for each endpoint to 8, 16, 32, 64, or 512 bytes. However, endpoints used for bulk transfers should be set to 8, 16, 32, or 64 bytes for FS mode or 512 bytes for HS mode. Endpoints used for isochronous transfers should be set to any of 1 to 1,023 bytes for FS mode or any of 1 to 1,024 bytes for HS mode.
Endpoint number	D_EPx{x=a-e}Config_0.EndpointNumber	Sets the desired endpoint numbers within the range 0x1 to 0xF.
Toggle mode	D_EPx{x=a-e}Config_0.IntEP_Mode	Sets the operating mode for interrupt mode. Endpoints for bulk transfer should be set to “0” regardless of direction. Set the toggle sequence mode for IN-direction endpoints. Set to “1” for OUT-direction endpoints with interrupt transfer.
Isochronous mode	D_EPx{x=a-e}Config_0.ISO	Set to “1” for endpoints used for isochronous transfers.
FIFO area	AREAx{x=1-5}StartAdrs_H,L AREAx{x=1-5}EndAdrs_H,L	Specify and reserve the area address. The areas should be at least as large as the linked endpoint maximum packet sizes. FIFO size may affect transfer throughput.
FIFO linking	AREAx{x=1-5}Join_1	Links the FIFO areas to endpoints. The endpoints to be used should be linked only as shown by the combinations below. Endpoint EPa: AREA1 Endpoint EPb: AREA2 Endpoint EPc: AREA3 Endpoint EPd: AREA4 Endpoint EPe: AREA5

1.2.2 Transaction

This LSI implements transaction execution functions via hardware and interfaces to execute transactions with the firmware. The interface with the firmware is installed as an interrupt signal asserted by the control register and status register or status. For detailed information on the settings that assert interrupts based on status, refer to “2 REGISTERS.”

This LSI issues status information to the firmware for each transaction. However, the firmware need not control each transaction. The LSI refers to the FIFO when responding to a transaction and performs processing automatically using the data quantity or free space size to determine whether data transfer is possible.

For an OUT endpoint, for example, the firmware reads data from the FIFO using the CPU interface (DMA read or register read) or IDE interface (IDE write). Creating free space in the FIFO enables continuous automatic execution of the OUT transaction. For an IN endpoint, the firmware writes data to the FIFO using the CPU interface (DMA write or register write) or IDE interface (IDE read); creating active data in the FIFO enables continuous automatic execution of the IN transaction.

Table 1-6 shows the control items and status for endpoint EP0 transaction control.

Table 1-6 Endpoint EP0 control items and status

Item	Register/bit	Description
Transaction direction	D_EP0Control.INxOUT	Sets the transfer direction for the data and status stages.
Descriptor reply enable	D_EP0Control.ReplyDescriptor	Activates the descriptor automatic response.
Descriptor reply address	D_DescAdrs_H, DescAdrs_L	Specifies the initial FIFO address for data returned based on the descriptor automatic response.
Descriptor size	D_DescSize_H, DescSize_L	Specifies the data size returned based on the descriptor automatic response.
Control prohibit	D_SETUP_Control.ProtectEP0	Setting this bit prevents access to the EP0ControlIN and EP0ControlOUT register ForceNAK and ForceSTALL bits. This bit is set by the LSI hardware when the RcvEP0SETUP status is established and can be cleared by CPU register access.
Short packet send enable	D_EP0ControlIN.EnShortPkt	Enables transmission of short packets of smaller than the max packet size. Cleared upon completion of the IN transaction sending the short packet.
Toggle sequence bit	D_EP0ControlIN.ToggleStat, D_EP0ControlOUT.ToggleStat	Indicates the toggle sequence bit status. Initialized automatically by the SETUP stage.
Toggle set	D_EP0ControlIN.ToggleSet, D_EP0ControlOUT.ToggleSet	Sets the toggle sequence bit.
Toggle clear	D_EP0ControlIN.ToggleClr, D_EP0ControlOUT.ToggleClr	Clears the toggle sequence bit.
Forced NAK response	D_EP0ControlIN.ForceNAK, D_EP0ControlOUT.ForceNAK	Returns an NAK to IN or OUT transactions (including PING) regardless of FIFO data or free space size.
STALL response	D_EP0ControlIN.ForceSTALL, D_EP0ControlOUT.ForceSTALL	Returns a STALL to IN or OUT transactions (including PING).
Automatic ForceNAK set	D_EP0ControlOUT.AutoForceNAK	Sets the D_EP0ControlOUT.ForceNAK bit upon completion of each OUT transaction.
SETUP receipt status	DeviceIntStat.RcvEP0SETUP	Indicates that a SETUP transaction has been performed.
Transaction status	D_EP0IntStat.OUT_ShortACK, D_EP0IntStat.IN_TranACK, D_EP0IntStat.OUT_TranACK, D_EP0IntStat.IN_TranNAK, D_EP0IntStat.OUT_TranNAK, D_EP0IntStat.IN_TranErr, D_EP0IntStat.OUT_TranErr	Indicates transaction results.
Descriptor reply data stage completion status	D_FIFO_IntStat.DescriptorCmp	Indicates the completion of a descriptor automatic response data stage.

Table 1-7 shows the control items and status for general endpoint EPa, EPb, EPc, EPd, and EPe transaction processing.

Table 1-7 General endpoint control items and status

Item	Register/bit	Description
Automatic ForceNAK set	D_EPx{x=a-e}Control.AutoForceNAK	Sets the endpoint D_EPx{x=a-e}Control.ForceNAK bit upon completion of each OUT transaction.
Short packet send enable	D_EPx{x=a-e}Control.EnShortPkt	Enables the transmission of short packets of smaller than the max packet size for IN transactions. Cleared upon completion of the IN transaction sending the short packet.
Automatic ForceNAK setting prohibition using short packet receipt	D_EPx{x=a-e}Control.DisAF_NAK_Short	Prohibits the function (*) from automatically setting the endpoint D_EPx{x=a-e}Control.ForceNAK bit on receiving a short packet in the OUT transaction. *Active if not prohibited by this bit.
Toggle sequence bit	D_EPx{x=a-e}Control.ToggleStat	Indicates the toggle sequence bit status.
Toggle set	D_EPx{x=a-e}Control.ToggleSet	Sets the toggle sequence bit.
Toggle clear	D_EPx{x=a-e}Control.ToggleClr	Clears the toggle sequence bit.
Forced NAK response	D_EPx{x=a-e}Control.ForceNAK	Returns an NAK to transactions regardless of FIFO data or free space size.
STALL response	D_EPx{x=a-e}Control.ForceSTALL	Returns a STALL to transactions.
Transaction status	D_EPx{x=a-e}IntStat.OUT_ShortACK, D_EPx{x=a-e}IntStat.IN_TranACK, D_EPx{x=a-e}IntStat.OUT_TranACK, D_EPx{x=a-e}IntStat.IN_TranNAK, D_EPx{x=a-e}IntStat.OUT_TranNAK, D_EPx{x=a-e}IntStat.IN_TranErr, D_EPx{x=a-e}I~tStat.OUT_TranErr	Indicates transaction results.

1.2.2.1 SETUP Transactions

This LSI unconditionally executes SETUP transactions addressed to endpoint EP0 on its own node. (The USB function must be enabled by the D_NegoControl.ActiveUSB bit.)

When a SETUP transaction is issued, the contents of all the data packets (8 bytes) are placed in registers D_EP0SETUP_0 to D_EP0SETUP_7 and an ACK response is returned. RcvEP0SETUP status is issued to the firmware, except for SetAddress() requests.

This LSI does not respond nor issue the status in the event of an error during a SETUP transaction.

On completion of a SETUP transaction, the LSI sets the D_EP0ControlIN and D_EP0ControlOUT registers ForceNAK bits and clears the ForceSTALL bit. The ToggleStat bit and D_SETUP_Control.ProtectEP0 bit are also set. When the firmware has completed the endpoint EP0 settings and is ready to proceed to the next stage, the SETUP_Control.ProtectEP0 bit should be cleared and the corresponding direction ForceNAK bits cleared for the D_EP0ControlIN and D_EP0ControlOUT registers.

Figure 1-1 illustrates the SETUP transaction. The host (a) issues a SETUP token addressed to endpoint 0 on this node. The host (b) then sends an 8-byte data packet. The LSI writes this data to

registers D_EP0SETUP_0 to D_EP0SETUP_7. The LSI (c) automatically returns an ACK response. The register set is set automatically and a status indication is issued to the firmware.

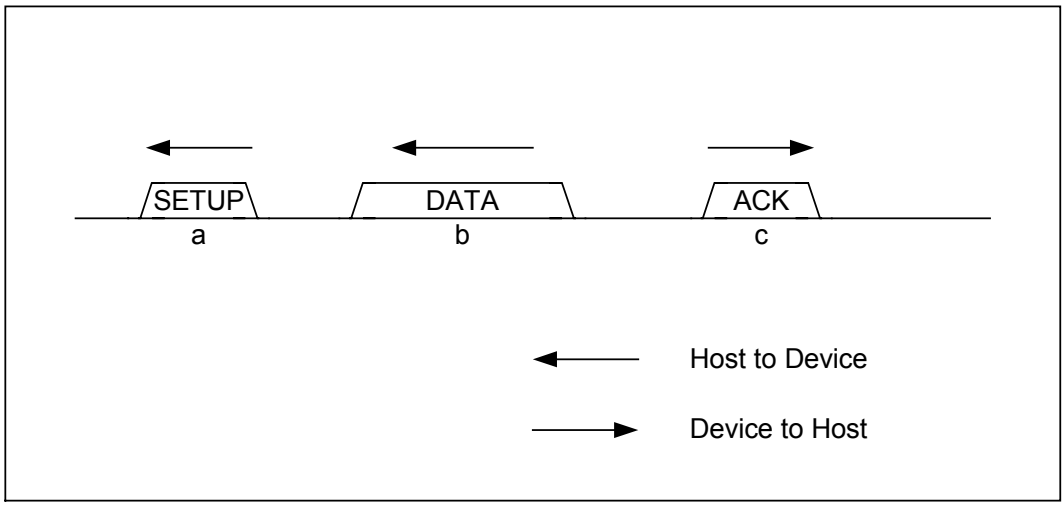


Figure 1-1 SETUP Transaction

1.2.2.2 Bulk/Interrupt OUT Transactions

For bulk or interrupt OUT transactions, data receipt begins if FIFO free space exceeds max packet size.

The LSI completes the transaction and returns an ACK or NYET response once all data has been correctly received for bulk or interrupt OUT transactions. The corresponding endpoint OUT_TranACK status (D_EPx{x=0,a-c}IntStat.OUT_TranACK bit) is then issued to the firmware, after which the FIFO is updated and the area reserved with the data received.

For bulk or interrupt OUT transactions, the OUT_ShortACK status (D_EPx{x=0,a-e}IntStat.OUT_ShortACK bit) is also issued upon receipt of all short packet data in addition to the transaction completion processing described above.

The endpoint D_EPx{x=a-e}Control.ForceNAK bit is set once the D_EPx{x=a-e}Control.DisAF_NAK_Short bit is cleared.

If a toggle mismatch occurs for bulk or interrupt OUT transactions, an ACK response is returned for the transaction but no status indication is issued. The FIFO is not updated.

If an error occurs in bulk or interrupt OUT transactions, no response is returned for the transaction, and an OUT_TranErr status (D_EPx{x=0,a-e}IntStat.OUT_TranErr bit) is issued. The FIFO is not updated.

If not all data was received for bulk or interrupt OUT transactions, a NAK response is returned for the transaction. An OUT_TranNAK status (D_EPx{x=0,a-e}IntStat.OUT_TranNAK bit) is issued. The FIFO is not updated.

Figure 1-2 shows the procedure for a bulk or interrupt OUT transaction when completed. The host (a) issues an OUT token addressed to the OUT-direction endpoint in this node. The host (b) then

sends a data packet within the max packet size. The LSI writes this data to the corresponding endpoint FIFO. The LSI (c) automatically returns an ACK response on receiving the data. The automatically set register is set and a status indication is issued to the firmware.

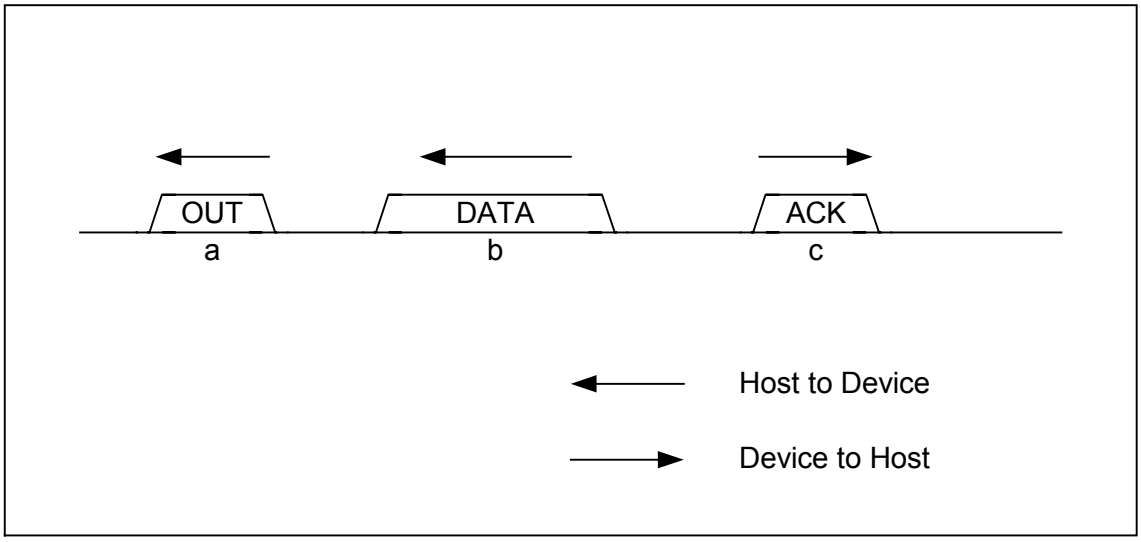


Figure 1-2 OUT transaction

1.2.2.3 Isochronous OUT Transactions

For isochronous OUT transactions, data receipt begins if the FIFO free space exceeds the max packet size. Satisfactory throughput is provided by assigning approximately twice as much as FIFO as the max packet size to enable receipt while creating free space by reading FIFO data using register reading or DMA reading via the CPU interface.

The corresponding endpoint OUT_TranACK status (EPx{x=a-e}IntStat.OUT_TranACK bit) is issued to the firmware when all data is correctly received for an isochronous OUT transaction. The FIFO is then updated and the area reserved with the data received.

For isochronous OUT transactions, the OUT_ShortACK status (EPx{x=a-e}IntStat.OUT_ShortACK bit) is also issued upon receipt of all data of less than max packet size and completion of the transaction processing described above. Clearing the EPx{x=a-e}Control.DisAF_NAK_Short bit sets the EPx{x=a-e}ForceNAK bit for that endpoint.

If an error occurs for isochronous OUT transactions, the data is not received, and the FIFO is not updated. OUT_TranErr status (EPx{x=a-e}IntStat.OUT_TranErr bit) is issued.

OUT_TranNAK status (EPx{x=a-e}IntStat.OUT_TranNAK bit) is issued if the data for one packet is not fully received for isochronous OUT transactions. The FIFO is not updated.

1.2.2.4 Bulk/Interrupt IN Transactions

For IN-direction bulk and interrupt endpoints, a data packet is returned in response to the IN transaction if the FIFO contains data corresponding to the max packet size or if the firmware permits short packet transmission.

Short packet (including data length zero packets) transmission permission is granted by setting the D_EP0ControlIN.EnShortPkt bit or D_EPx{x=a-e}Control.EnShortPkt bit. When a short packet is transmitted, no new data should be written to that endpoint FIFO from the point at which transmission permission is granted until the completion of the transaction.

The D_EP0ControlIN.ForceNAK bit is set for endpoint EP0 upon completion of the IN transaction sending the short packet.

When an ACK is received for the IN transaction returning data, the transaction is completed and the IN_TranACK status (D_EPx{x=0,a-e}IntStat.IN_TranACK bit) is issued to the firmware. The FIFO is then updated and the data transmitted is cleared, freeing the area.

If an ACK is not received for the IN transaction returning data, a transaction failure is assumed, and the IN_TranErr status (D_EPx{x=0,a-e}IntStat.IN_TranErr bit) is issued to the firmware. The FIFO is not updated, and the area is not freed.

If the FIFO does not contain data corresponding to the max packet size and short packet transmission is not permitted for bulk or interrupt IN-direction endpoints, an NAK response is returned to the IN transaction and IN_TranNAK status (D_EPx{x=0,a-e}IntStat.IN_TranNAK bit) is issued to the firmware. The FIFO is not updated, and the area is not freed.

Figure 1-3 shows the procedure for a bulk or interrupt IN transaction when completed. The host (a) issues an IN token addressed to the IN-direction endpoint in this node. The LSI (b) then sends a data packet within the max packet size if a response to this IN transaction is possible. The host (c) returns an ACK response. The LSI automatically sets the register set on receiving the ACK and issues a status indication to the firmware.

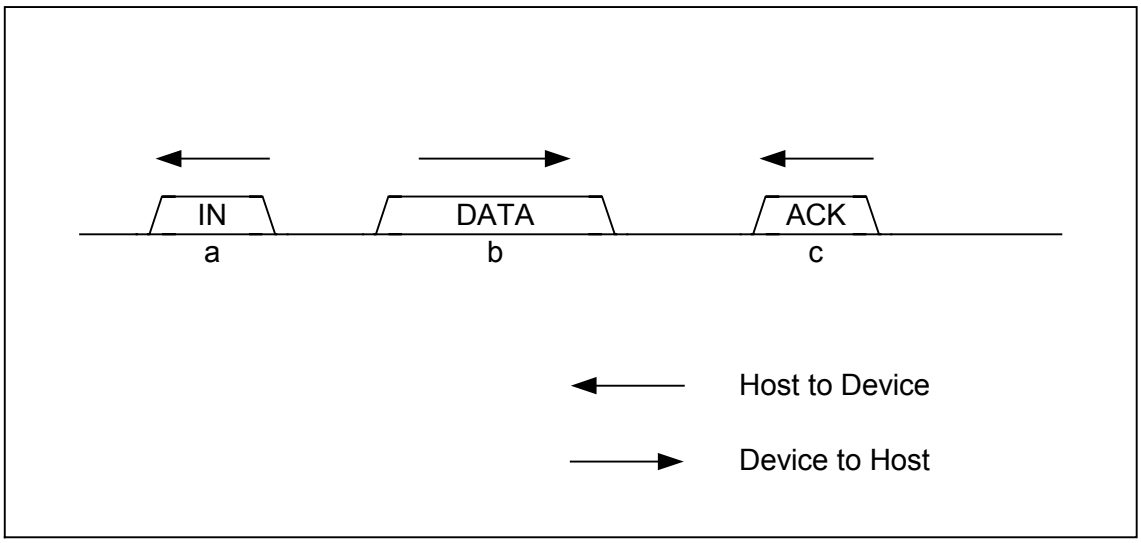


Figure 1-3 IN transaction

1.2.2.5 Isochronous IN Transaction

For IN-direction isochronous endpoints, a data packet is returned in response to the IN transaction if the FIFO contains data corresponding to the max packet size or if the firmware permits short packet transmission.

Short packet (including data length zero packets) transmission permission is granted by setting the EPx{x=a-e}Control.EnShortPkt bit.

When a short packet is transmitted, no new data should be written to that endpoint FIFO from the point at which transmission permission is granted until completion of the transaction.

When the data packet is returned to the isochronous IN transaction, the transaction is completed and the IN_TranACK status (EPx{x=a-e}IntStat.IN_TranACK bit) is issued to the firmware. The FIFO is then updated and the data transmitted is cleared, freeing the area.

If the FIFO does not contain data corresponding to the max packet size and short packet transmission is not permitted for isochronous IN-direction endpoints, a response is returned to the IN transaction with a zero length data packet, and IN_TranNAK status (EPx{x=a-e}IntStat.IN_TranNAK bit) is issued to the firmware. The FIFO is not updated, and the area is not freed.

1.2.2.6 PING Transactions

For bulk OUT-direction endpoints, PING transactions are executed during HS operation.

If the corresponding endpoint FIFO free space exceeds the max packet size, an ACK response is returned to the PING transaction. No status indication is issued to the firmware.

If the corresponding endpoint FIFO free space is smaller than the max packet size, a NAK response is returned to the PING transaction. An OUT_TranNAK status (D_EPx{x=0,a-e}IntStat.OUT_TranNAK bit) indication is issued to the firmware.

The FIFO is never updated for PING transactions.

Figure 1-4 illustrates the ACK response to the PING transaction. The host (a) issues a PING token addressed to the OUT-direction endpoint in this node. The LSI (b) returns an ACK response to the PING transaction if the FIFO includes free space equivalent to the max packet size. A status indication is issued to the firmware.

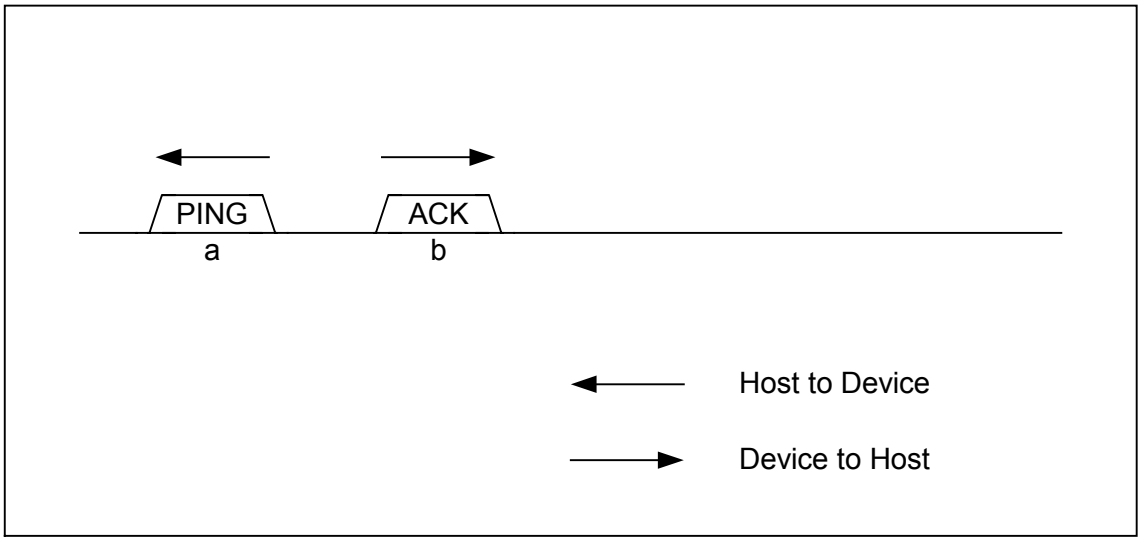


Figure 1-4 PING transaction

1.2.3 Control Transfer

Control transfer for endpoint EP0 controls in conjunction with individual transactions, with the exception of SetAddress() requests. SetAddress() requests are processed automatically using an automatic address setup function (described later).

Figure 1-5 illustrates a control transfer when the data stage is in the OUT direction. The host (a) initiates control transfer depending on the SETUP transaction. The device firmware analyzes the request details and prepares a response to the data stage. The host (b) issues an OUT transaction, the data stage is performed, and the device receives the data. The host (c) issues an IN transaction, the data stage is performed, and the device returns a data length zero packet.

Control transfers without a data stage are performed without the data stage shown in this example.

The transition to the status stage is triggered by the host issuing a transaction in the direction opposite to the data stage. The firmware should be used to monitor the IN_TranNAK status (D_EP0IntStat.IN_TranNAK bit) and should serve as the trigger for switching from data stage to status stage.

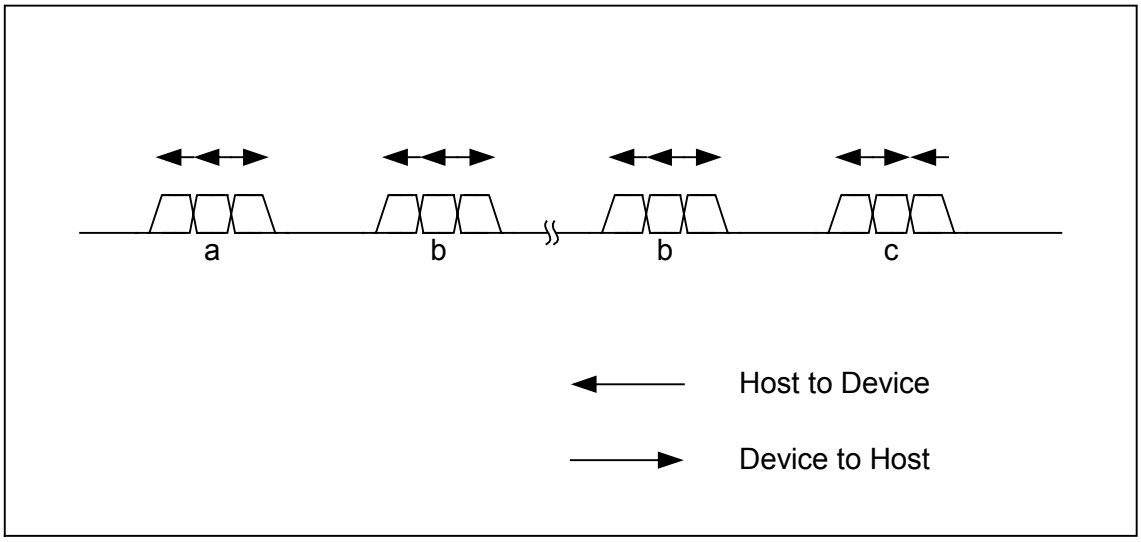


Figure 1-5 Control transfer when data stage is OUT

Figure 1-6 illustrates a control transfer when the data stage is in the IN direction. The host (a) initiates control transfer depending on the SETUP transaction. The device firmware analyzes the request details and prepares to respond to the data stage. The host (b) issues an IN transaction, the data stage is performed, and the device sends the data. The host (c) issues an OUT transaction, the status stage is performed, and the device returns an ACK response.

The transition to the status stage is triggered by the host issuing a transaction in the direction opposite to the data stage. The firmware should be used to monitor the OUT_TranNAK status (D_EP0IntStat.OUT_TranNAK bit) and should serve as the trigger for switching from data stage to status stage.

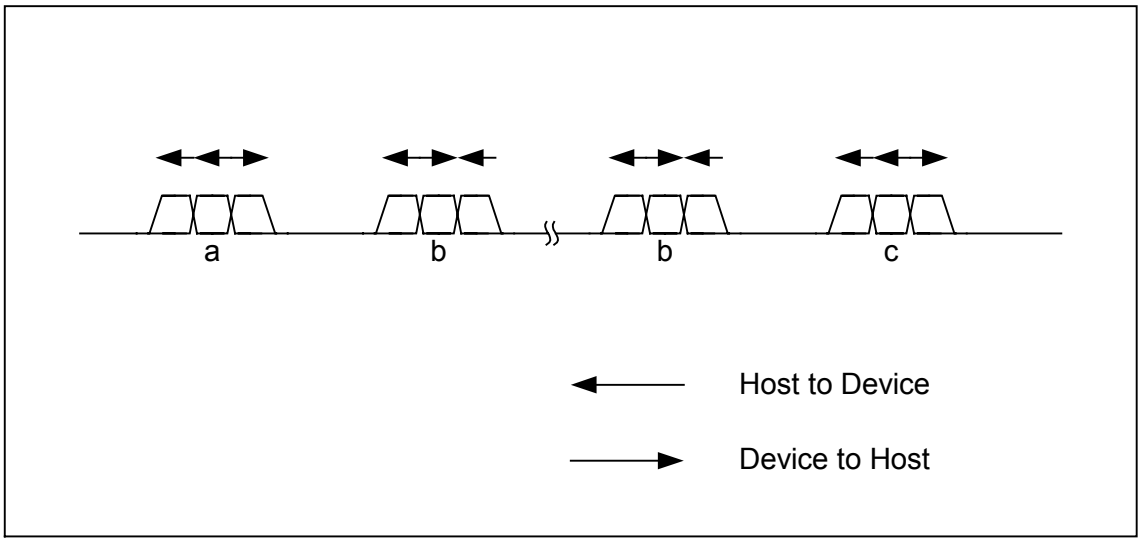


Figure 1-6 Control transfer when data stage is IN

The control transfer data stage and status stage perform normal IN and OUT transactions, so flow control is enabled using NAK. The device is allowed to prepare a response within the timeframe stipulated.

1.2.3.1 Setup Stage

A setup transaction is executed automatically upon receipt of a SETUP token addressed to the same node.

The firmware should monitor the RcvEP0SETUP status and analyze requests from registers D_EP0SETUP_0 to D_EP0SETUP_7 to control transfers.

If a request received includes a data stage in the OUT direction, the D_EP0Control register INxOUT bit should be cleared and endpoint EP0 set to the OUT direction to change to the data stage.

If the request received includes a data stage in the IN direction, the D_EP0Control register INxOUT bit should be set and endpoint EP0 set to the IN direction to change to the data stage.

If the request received has no data stage, the D_EP0Control register INxOUT bit should be set and endpoint EP0 set to the IN direction to change to the status stage.

1.2.3.2 Data Stage/Status Stage

Read registers D_EP0SETUP_0 to D_EP0SETUP_7 and move to the next stage, depending on the details of the request analyzed.

If the stage is in the OUT direction, clear the D_EP0Control register INxOUT and set to the OUT direction. Then set the D_EP0ControlOUT register appropriately to control the stage. The ForceNAK bit will be set when the SETUP stage ends. The D_SETUP_Control.ProtectEP0 bit is also set.

If the stage is in the IN direction, set the D_EP0Control register INxOUT to the IN direction, then set the D_EP0ControlIN register appropriately to control the stage. The ForceNAK bit will be set when the SETUP stage ends. The D_SETUP_Control.ProtectEP0 bit is also set.

1.2.3.3 Automatic Address Setup Function

This LSI features a function to automate the SetAddress() request processing for endpoint EP0 control transfers.

The LSI hardware checks the request details from the registers D_EP0SETUP_0 to D_EP0SETUP_7, proceeding to request status stage processing without notifying the firmware if the SetAddress() request is valid.

Once the status stage ends, the USB_Address register address is set, and a SetAddressCmp status (D_SIE_IntStat.SetAddressCmp bit) is issued to the firmware.

The firmware monitors the SetAddressCmp status, enabling the address to be checked using the USB_Address register when it has been issued. For detailed information on the automatic address setup function, also refer to "Appendix D: SetAddress Request Response (for TS only)."

1.2.3.4 Descriptor Return Function

This LSI features a descriptor return function, which is useful for multiple requests for identical data, such as GetDescriptor() for endpoint EP0 control transfer.

The firmware can use this function for IN-direction data stage requests.

Clear the D_EP0ControlIN.ForceNAK bit, set the initial address of the data to be returned within the FIFO descriptor area in the D_DescAdrs_H,L register and the total number of data bytes to be returned to the D_DescSize_H,L register before starting the response to the data stage, and then set the D_EP0Control.ReplyDescriptor bit.

The descriptor return function executes the IN transaction by returning data packets in response to the data stage IN transaction until the set quantity of data has been returned. A NAK response is returned if an IN transaction is issued after the set quantity of data has been sent. Provided the data quantity is less than the max packet size, the descriptor return function sets D_EP0ControlIN.EnShortPkt to allow response to IN transactions until all data has been returned.

If an OUT token is received and a transition to the status stage is detected, the D_EP0Control.ReplyDescriptor bit is cleared and DescriptorCmp status (D_FIFO_IntStat.DescriptorCmp bit) is issued to the firmware. The firmware should perform the status stage if DescriptorCmp status is detected.

For detailed information on the descriptor area, refer to "1.6 FIFO Management."

1.2.4 Bulk Transfer/Interrupt Transfer/Isochronous Transfer

Bulk, interrupt, and isochronous transfers for general endpoints EPa, EPb, EPc, EPd, and EPe can be controlled as data flows (refer to "1.2.5 Data Flow") or continuous individual transactions (refer to "1.2.2 Transaction").

1.2.5 Data Flow

This section describes general data flow control for OUT and IN transfer.

1.2.5.1 OUT Transfer

Data received in OUT transfers is written to the FIFO linked to each endpoint. Data can be read from the FIFO by CPU interface register reading, CPU interface DMA reading, or write transfer to the IDE.

To read FIFO data using CPU interface register reading, select a single FIFO area using the AREAx{x=0-5}Join_0.JoinCPU_Rd bit. The FIFO area selected can be read in the sequence received from the FIFO_Rd or FIFO_ByteRd registers. The FIFO data quantity that can be read off can be obtained from the FIFO_RdRemain_H,L register. Since empty FIFO areas cannot be read,

data quantity must always be confirmed from the FIFO_RdRemain_H,L register to ensure that reading does not exceed this quantity.

To read FIFO data using CPU interface DMA reading, select a single FIFO area for each DMA channel using the AREAx{x=0-5}Join_0.JoinDMAx{x=0,1} bit and set the DMAx{x=0,1}_Control.Dir bit to "1." The FIFO area selected can be read off in the sequence received by running the DMA sequence in the CPU interface. The quantity of the remaining FIFO data can be checked using the DMAx{x=0,1}_Remain_H,L register. The CPU interface automatically pauses the DMA for flow control once the FIFO becomes empty.

To read FIFO data using IDE interface write transfer, select a single FIFO area using the AREAx{x=0-5}Join_0.JoinIDE bit and set the IDE_Control.Dir bit to "1." The FIFO area selected can be read off in the sequence received by running the IDE transfer using IDE_Control.IDE_Go. The IDE interface automatically pauses writing transfer for flow control once the FIFO becomes empty.

If the FIFO contains free space to receive data packets, a response is automatically returned to the OUT transaction, and data can be received. OUT transfers can be performed by firmware without controlling individual transactions. However, the endpoint D_EPx{x=a-e}Control.ForceNAK bit is set if a short packet (including data length zero packets) is received when the D_EPx{x=a-e}Control.DisAF_NAK_Short bit is cleared (initial value). The D_EPx{x=a-e}Control.ForceNAK bit should be cleared as soon as the next data transfer is ready.

1.2.5.2 IN Transfer

Data sent in IN transfers should be written to the FIFO linked to each endpoint. Data can be written to the FIFO by CPU interface register writing, CPU interface DMA writing, or read transfer from the IDE.

To write data to the FIFO using CPU interface register writing, select a single FIFO area using the AREAx{x=0-5}Join_0.JoinCPU_Wr bit. The FIFO area selected can be written to using the FIFO_Wr register, enabling transmission of data packets in the sequence written. The FIFO empty space can be checked from the FIFO_WrRemain_H,L register. FIFO areas cannot be written to when full, and so the empty space must always be confirmed from the FIFO_WrRemain_H,L register to ensure that writing does not exceed the space available.

To write data to the FIFO using CPU interface DMA writing, select a single FIFO area for each DMA channel using the AREAx{x=0-5}Join.JoinDMAx{x=0,1} bit and set the DMAx{x=0,1}_Control.Dir bit to "0." The FIFO area selected can be written to by the CPU interface using the DMA sequence, enabling transmission of data packets in the sequence written. The CPU interface automatically pauses the DMA for flow control once the FIFO fills up.

To write data to the FIFO using IDE interface read transfer, select a single FIFO area using the AREAx{x=0-5}Join_0.JoinIDE bit and set the IDE_Control.Dir bit to "0." The FIFO area selected can be written to by executing IDE transfer using IDE_Control.IDE_Go and writing in the

sequence read from the IDE, enabling transmission of data packets in the sequence written. The IDE interface automatically pauses the read transfer for flow control once the FIFO fills up.

If the data in the FIFO exceeds the max packet size, a response is returned to the IN transaction, allowing data transmission. Thus, IN transfers can be performed by the firmware without controlling individual transactions, but the EnShortPkt bit must be set if a short packet must be sent after the data transfer. This bit is cleared once the IN transaction is completed for the short packet sent. It can be set once data writing to the FIFO is complete. The endpoint EnShortPkt bit is set automatically if the FIFO contains fractional data of less than the max bit size when CPU interface DMA writing ends with the DMAx{x=0,1}_FIFO_Control.AutoEnShort bit set.

1.2.6 Bulk Only Support

This LSI features a bulk only support function, which assists Command Block Wrapper (CBW) receipt and Command Status Wrapper (CSW) transmission specific to USB Mass Storage Class (BulkOnly Transport Protocol) for endpoint EPa, EPb, EPc, EPd, and EPe bulk transfer.

Setting the D_BulkOnlyConfig.EPx{x=a-e}BulkOnly bit enables the bulk only support function for the corresponding endpoint.

While the bulk only support function is enabled and CBW or CSW support is running, packets are received (CBW) or sent (CSW) using the area assigned as the CBW or CSW area rather than the FIFO area normally assigned to the endpoint.

1.2.6.1 CBW Support

The firmware can use the CBW support for BulkOnly Transport Protocol command transport. Setting the D_BunlOnlyConfig.EPx{x=a-e}BulkOnly bit enables CBW support for the corresponding OUT endpoint. CBW support should be set so that it is enabled for only one endpoint. Setting the D_BulkOnlyContol.GoCBW_Mode bit while CBW support is enabled will run CBW support and treat data received in the OUT transaction for the corresponding endpoint as CBW.

If the data packet data length is the 31 bytes expected for CBW, the data is saved in the CBW area and a CBW completion status (D_BulkIntStat.CBW_Cmp bit) is issued to the firmware. The D_BulkOnlyControl.GoCBW_Mode bit is automatically cleared, and CBW support ends. If set, the D_BulkOnlyControl.GoCSW_Mode bit is also cleared simultaneously.

If the data packet data length is less than or more than 31 bytes, the data is not received, and a CBW data length error status (D_BulkIntStat.CBW_LengthErr bit) is issued to the firmware. The D_BulkOnlyControl.GoCBW_Mode bit is automatically cleared, and CBW support ends. If set, D_BulkOnlyControl.GoCSW_Mode bit is cleared simultaneously. The issuing of a CBW_Err status indicates a phase mismatch with the BulkOnly Transport Protocol, and the transfer must be reset by the firmware – for example, by using STALL for the endpoint.

If D_EPx{x=a-e}Control.ForceSTALL is set for the corresponding endpoint and a STALL response was returned to the OUT transaction, a CBW error status (D_BulkIntStat.CBW_Err bit) is

issued to the firmware, the D_BulkOnlyControl.GoCBW_Mode bit is cleared, and CBW support ends. If set, the D_BulkOnlyControl.GoCSW_Mode bit is cleared simultaneously.

If a transaction error such as CRC error occurs in the OUT transaction, the data is not received, and a CBW transaction error status (D_BulkIntStat.CBW_TranErr bit) is issued to the firmware. In this case, the D_BulkOnlyControl.GoCBW_Mode bit is not cleared, and CBW support continues. Even if set, the D_BulkOnlyControl.GoCSW_Mode bit is not cleared.

The data received at the CBW area can be read out using the RAM_Rd function.

1.2.6.2 CSW Support

The firmware can use the CSW support for BulkOnly Transport Protocol status transport. Setting the D_BulkOnlyConfig.EPx{x=a-e}BulkOnly bit enables CSW support for the corresponding IN endpoint. CSW support should be controlled so that it is enabled for a single endpoint. Setting the D_BulkOnlyContol.GoCSW_Mode bit while CSW support is enabled will run CSW support and send 13 bytes of data from the CSW area, treating the IN transaction for the corresponding endpoint as CSW.

If an ACK is received from the host and the transaction completed after 13 bytes of CSW data are returned to the host for an IN transaction, a CSW completion status (D_BulkIntStat.CSW_Cmp bit) is issued to the firmware. The D_BulkOnlyControl.GoCSW_Mode bit is automatically cleared, and CSW support ends. The D_BulkOnlyControl.GoCBW_Mode bit is set simultaneously and initiates CBW support.

If an ACK is not received from the host after 13 bytes of data are returned to the host for an IN transaction, a CSW error status (D_BulkIntStat.CSW_Err bit) is issued to the firmware. Here, the D_BulkOnlyControl.GoCSW_Mode bit is not cleared, and CSW support continues. The hardware sets the D_BulkOnlyControl.GoCBW_Mode bit simultaneously and initiates CBW support. In other words, running CSW support here will also simultaneously run CBW support. If an error occurs because the host cannot receive CSW, the CSW will be retried, but a response is possible since CSW support is running. If an error occurs because the device does not receive the ACK, the next CBW will be run, but a response is possible since CBW support is running, and CSW support ends because the CBW support is running.

Data can be written to the CSW area using the RAM_WrDoor function.

1.2.7 Auto Negotiation Function

This performs suspend detection, reset detection, HS Detection Handshake, resume detection, and restore automatically, while checking the USB status sequentially. The actual function that was run can be checked via the individual interrupts (DetectRESET, DetectSUSPEND, ChirpCmp, RestoreCmp).

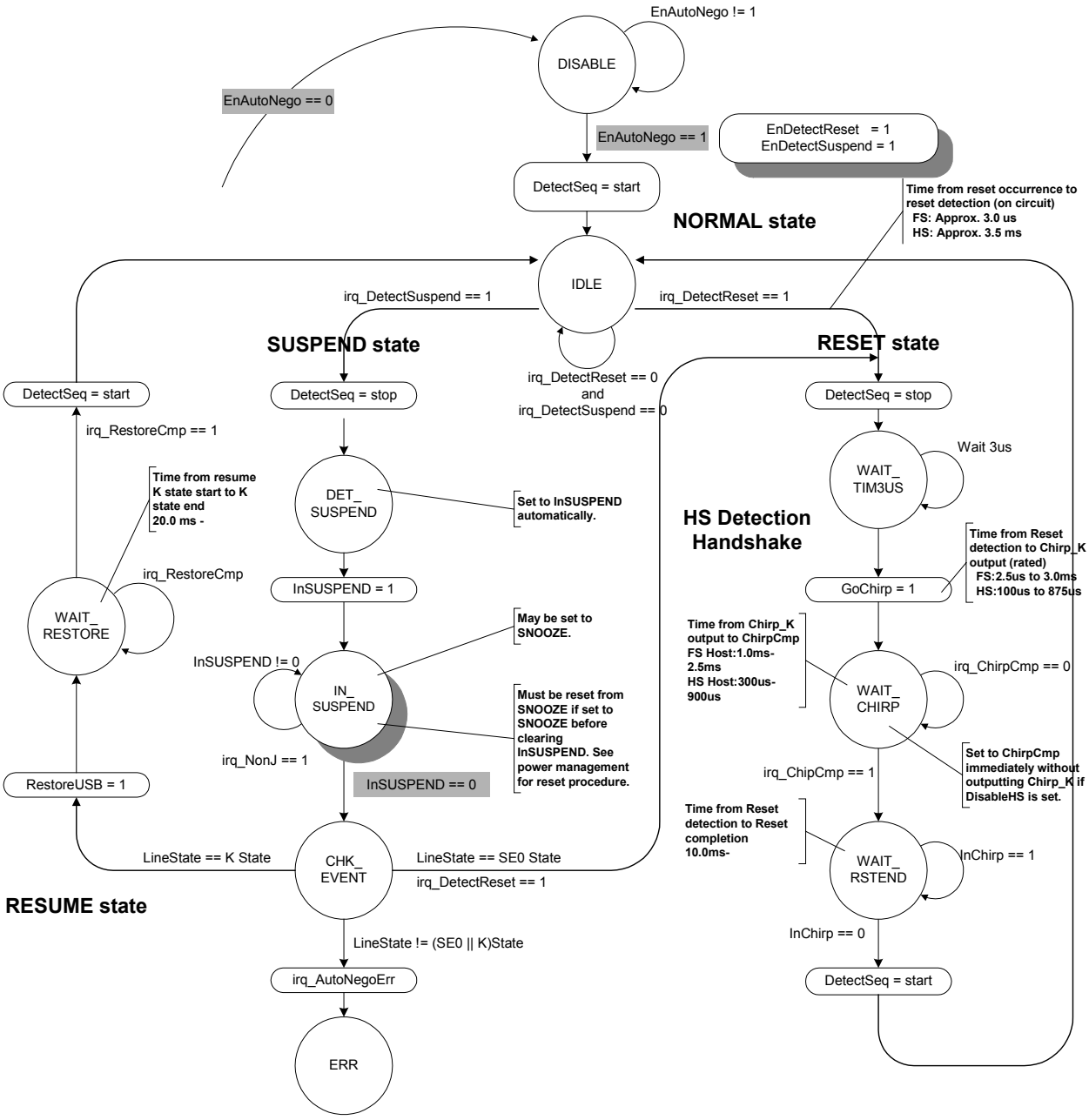


Figure 1-7 Auto negotiation

1.2.7.1 DISABLE

Clearing D_NegoControl.EnAutoNego initiates this state. To enable the auto negotiation function, set the reset detection interrupt permission bit (D_SIE_IntEnb.EnDetectRESET) and suspend detection interrupt permission bit (D_SIE_IntEnb.EnDetectSUSPEND) and permit both event detection interrupts before setting the D_NegoControl.EnAutoNego bit.

Enabling the auto negotiation function enables the internal event detection function. Never set the D_NegoControl.DisBusDetect bit with the auto negotiation function enabled.

1.2.7.2 IDLE

This state awaits reset detection and suspend detection.

FS termination is enabled if no bus activity is detected on the USB within 3 ms while the current USB speed is HS. A suspend is determined if FS-J is detected, and a reset is determined if SE0 is detected. A reset is determined if SE0 is detected for more than 2.5 µs while the current speed is FS, and a suspend is determined if no bus activity is detected within 3 ms. Reset detection or suspend detection interrupts occur simultaneously with these judgments, and the D_SIE_IntStat.DetectRESET or D_SIE_IntStat.DetectSUSPEND bits are set.

The event detection function is suspended if a suspend is determined, and the system enters a DET_SUSPEND state.

The event detection function is suspended if a reset is determined, and the system enters a WAIT_TIM3US state.

1.2.7.3 WAIT_TIM3US

This adjusts the time taken to run the HS Detection Handshake after reset detection. The system enters a WAIT_CHIRP state after a preset interval (approx. 3 µs) elapses.

1.2.7.4 WAIT_CHIRP

This automatically sets the D_NegoControl.GoChirp bit and runs the HS Detection Handshake. The Chirp completion interrupt status (D_SIE_IntStat.ChirpCmp) is set once the HS Detection Handshake ends, and the system enters a WAIT_RSTEND state. See “1.2.7.11.5 HS Detection Handshake” for detailed information on HS Detection Handshake.

If the D_NegoControl.DisableHS bit is set, the Chirp completion interrupt status (D_SIE_IntStat.ChirpCmp) is set without running the HS Detection Handshake, and the system enters a WAIT_RSTEND state.

Operations will occur at the transfer speed set in the D_USB_Status.FSxHS bit after this state expires. If it is necessary to detect changes in transfer speed, set the D_SIE_IntEnb.EnChirpCmp bit to enable the above-mentioned Chirp completion interrupt.

1.2.7.5 WAIT_RSTEND

Waits at this state for the end of the reset period. A chirp transmission from the host (receipt by this IC) is used to determine the end of the reset period for HS, and switching from SE0 to J is used for FS.

The event detection function is enabled after the reset period ends, and the system returns to an IDLE state.

1.2.7.6 DET_SUSPEND

If a suspend is determined, the D_NegoControl.InSUSPEND bit is automatically set, and the system enters the IN_SUSPEND state. The D_NegoControl.InSUSPEND bit enables a function that detects the bus transition from FS-J, allowing detection of resume or resetting by the host.

The application in question will determine whether the SUSPEND state actually reduces current consumption. This LSI includes two levels of current consumption reduction (SNOOZE and SLEEP). See “1.5 Power Management Functions” for detailed information on control methods.

Set the D_SIE_IntEnb.EnNonJ bit via the firmware and permit NonJ interrupt to detect the resume (FS-K), the suspend-end command here.

1.2.7.7 IN_SUSPEND

The system enters the CHK_EVENT state if a reset from a suspend command is determined and the D_NegoControl.InSUSPEND bit is cleared by the firmware when the NonJ interrupt status (D_SIE_IntStat.NonJ) is set.

To resume automatically from a suspend with applications for which the remote wakeup function enabled, set the D_NegoControl.SendWakeup bit while in this state and output FS-K for between 1 ms and 15 ms.

1.2.7.8 CHK_EVENT

This checks the USB cable and determines a resume if FS-K is detected or a reset if SE0 is detected. If a resume is determined, the D_NegoControl.RestoreUSB bit is set, and the transfer speed is returned to the speed in effect before the suspend (in accordance with the D_USB_Status.FSxHS value). If a reset is determined, the event detection function is suspended in the same way as for the transition from IDLE state, and the system enters the WAIT_TIM3US state.

If a state other than FS-K or SE0 is detected, the auto negotiation error interrupt status (D_SIE_IntStat.AutoNegoErr) bit is set, and an ERR state is assumed.

1.2.7.9 WAIT_RESTORE

Setting the D_SIE_IntStat.RestoreCmp bit enables event detection function and switches to the IDLE state.

1.2.7.10 ERR

The only way to reset from this state is to abort the auto negotiation function. This state is not part of the USB standards.

No means exists to determine whether the USB cable is disconnected in any state. The auto negotiation function must be aborted immediately if the USB cable is disconnected.

1.2.7.11 Description of Individual Negotiation Functions

This section describes the bus event processing used for auto negotiation. Since the auto negotiation function effects control via the LSI register interface, the processing performed automatically for auto negotiation can normally be performed through firmware.

1.2.7.11.1 Suspend Detection (HS Mode)

This LSI switches to FS mode automatically if no transmissions are sent or received (T1) within 3 ms while operating in HS mode (HS termination is disabled and FS termination (Rpu) is enabled). This operation sets DP to "H," allowing "J" to be checked via the D_USB_Status.LineState[1:0] bit (note that it is reset (see later) if "SE0" is detected). If "J" is still detected at subsequent point T2, the D_SIE_IntStat.DetectSUSPEND bit is set and a USB SUSPEND state is determined.

The XINT signal is asserted simultaneously if the D_SIE_IntEnb.EnDetectSUSPEND bit and DeviceIntEnb.EnD_SIE_IntStat bit are set and MainIntEnb.EnDeviceIntStat is set. The figure below illustrates SNOOZE operations during USB suspend.

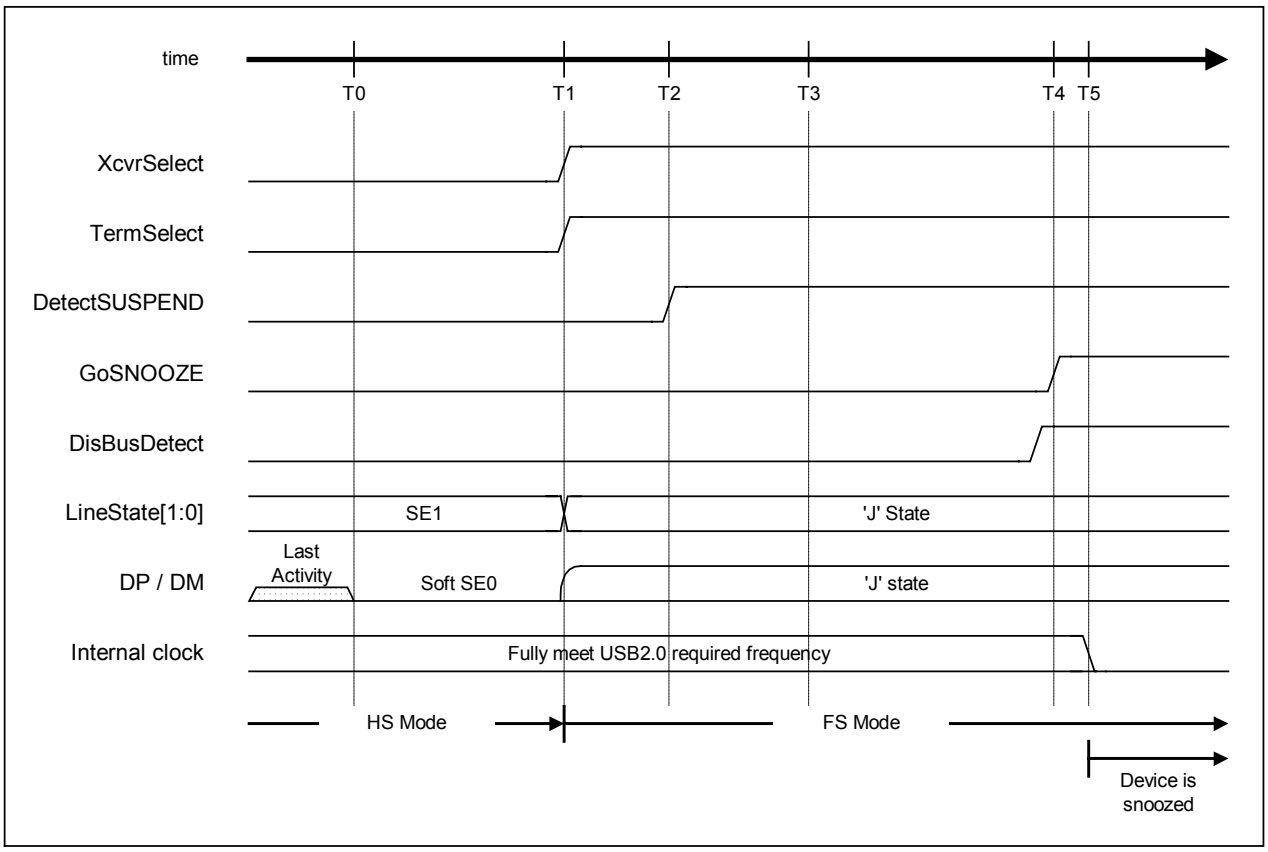


Figure 1-8 Suspend timing (HS mode)

Table 1-8 Suspend timing values (HS mode)

Timing Parameter	Description	Value
T0	Most recent bus activity	0 (reference)
T1	Sets XcvrSelect and TermSelect to “1” and switches from HS mode to FS mode if no bus activity continues at this point.	HS Reset T0 + 3.0ms < T1 {T_{WTREV}} < HS Reset T0 + 3.125ms
T2	Samples LineState[1:0]. If “J” is detected here, DetectSUSPEND is set to “1,” and a USB SUSPEND state is determined.	T1 + 100us < T2 {$T_{WTWRSTHS}$} < T1 + 875us
T3	RESUME must not be issued before this point.	HS Reset T0 + 5ms {T_{WTRSM}}
T4	Sets GoSNOOZE to “1” and moves entirely to SNOOZE. The current drawn from the VBUS hereafter must not exceed the suspend current specified for USB. (Set DisBusDetect to “1” before switching to SNOOZE.)	HS Reset T0 + 10ms {T_{2SUSP}}
T5	The internal clock stops completely.	T5 < T4 + 10us

Note: Brackets {} indicate terms defined under the USB 2.0 standards.

1.2.7.11.2 Suspend Detection (FS Mode)

This LSI determines a USB SUSPEND state and sets the SIE_IntStat.DetectSUSPEND bit if no transmissions are sent or received within 3 ms while operating in FS mode, or if "J" is detected in the D_USB_Status.LineState[1:0] bit and is still detected at T1 and is still detected at point T2.

The XINT signal is asserted simultaneously if the D_SIE_IntEnb.EnDetectSUSPEND bit and DeviceIntEnb.EnD_SIE_IntStat bit are set and MainIntEnb.EnDeviceIntStat is set. The figure below illustrates SNOOZE operations during USB suspend.

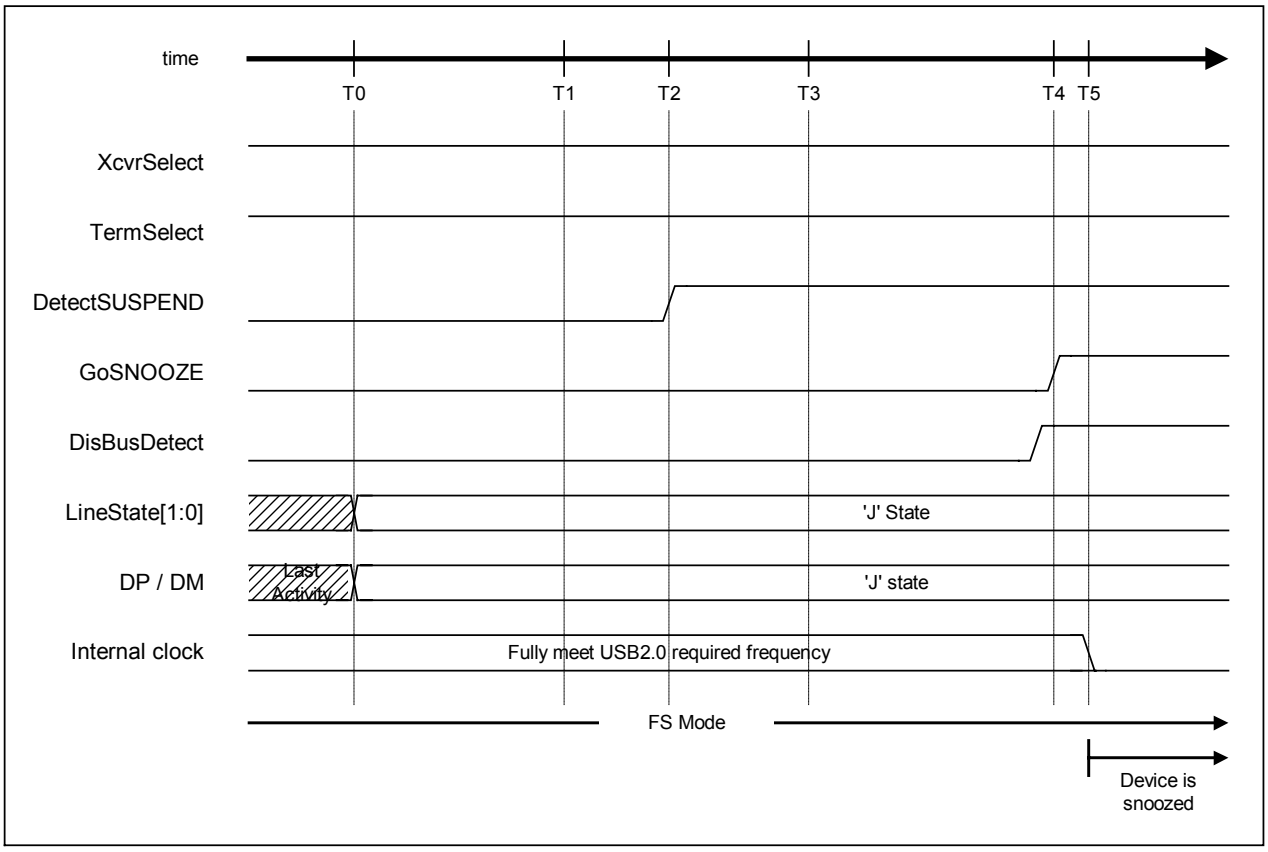


Figure 1-9 Suspend timing (FS mode)

Table 1-9 Suspend timing values (FS mode)

Timing Parameter	Description	Value
T0	Most recent bus activity	0 (reference)
T1	No bus activity at this point.	T0 + 3.0ms < T1 {T_{WTREV}} < T0 + 3.125ms
T2	Samples LineState[1:0]. If "J" is detected here, DetectSUSPEND is set to "1," and a USB SUSPEND state is determined.	T1 + 100us < T2 {$T_{WTWRSTHS}$} < T1 + 875us
T3	RESUME must not be issued before this point.	T0 + 5ms {T_{WTRSM}}
T4	Sets GoSNOOZE to "1" and moves entirely to SNOOZE. The current drawn from the VBUS hereafter must not exceed the suspend current specified for USB. (Set DisBusDetect to "1" before switching to SNOOZE.)	T0 + 10ms {T_{2SUSP}}
T5	The internal clock stops completely.	T5 < T4 + 10us

Note: Brackets {} indicate terms defined under the USB 2.0 standards.

1.2.7.11.3 Reset Detection (HS Mode)

This LSI switches to FS mode automatically if no transmissions are sent or received within 3 ms while operating in HS mode (HS termination is disabled and FS termination (Rpu) enabled). If reset, the DP line remains at "L" even when this operation is performed, allowing detection of "SE0" even for the D_USB_Status.LineState[1:0] bit. The D_SIE_IntStat.DetectRESET bit is set if "SE0" is still detected at point T2.

The XINT signal is asserted simultaneously if the D_SIE_IntEnb.EnDetectRESET bit and DeviceIntEnb.EnD_SIE_IntStat bit are set and MainIntEnb.EnDeviceIntStat is set.

The D_NegoControl.DisBusDetect bit is then set before performing the HS Detection Handshake (described later).

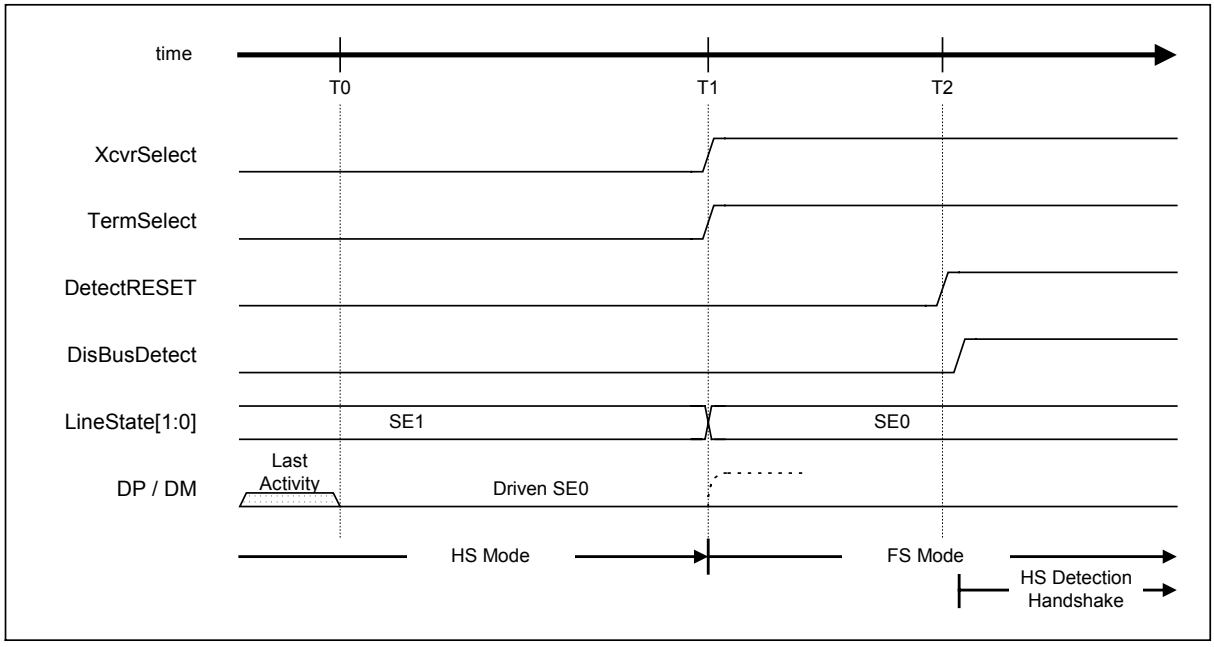


Figure 1-10 Reset timing (HS mode)

Table 1-10 Reset timing values (HS mode)

Timing Parameter	Description	Value
T0	Most recent bus activity	0 (reference)
T1	XcvrSelect and TermSelect are set to "1," and the LSI switches from HS mode to LS mode if there is still no bus activity at this point.	HS Reset T0 + 3.0ms < T1 {T_{WTREV}} < HS Reset T0 + 3.125ms
T2	Samples LineState[1:0]. If "SE0" is detected here, DetectRESET is set to "1," and the LSI determines a transition to reset. Sets DisBusDetect to "1" after detecting a reset command, then performs the HS Detection Handshake.	T1 + 100us < T2 {$T_{WTWRSTHS}$} < T1 + 875us

Note: Brackets {} indicate terms defined under the USB 2.0 standards.

1.2.7.11.4 Reset Detection (FS Mode)

The D_SIE_IntStat.DetectRESET bit is set if the D_USB_Status.LineState[1:0] bit is detected as "SE0" for more than 2.5 µs (T1) while operating in FS mode.

The XINT signal is asserted simultaneously if the D_SIE_IntEnb.EnDetectRESET bit and DeviceIntEnb.EnD_SIE_IntStat bit are set and MainIntEnb.EnDeviceIntStat is set.

The D_NegoControl.DisBusDetect bit is then set before performing the HS Detection Handshake (described later).

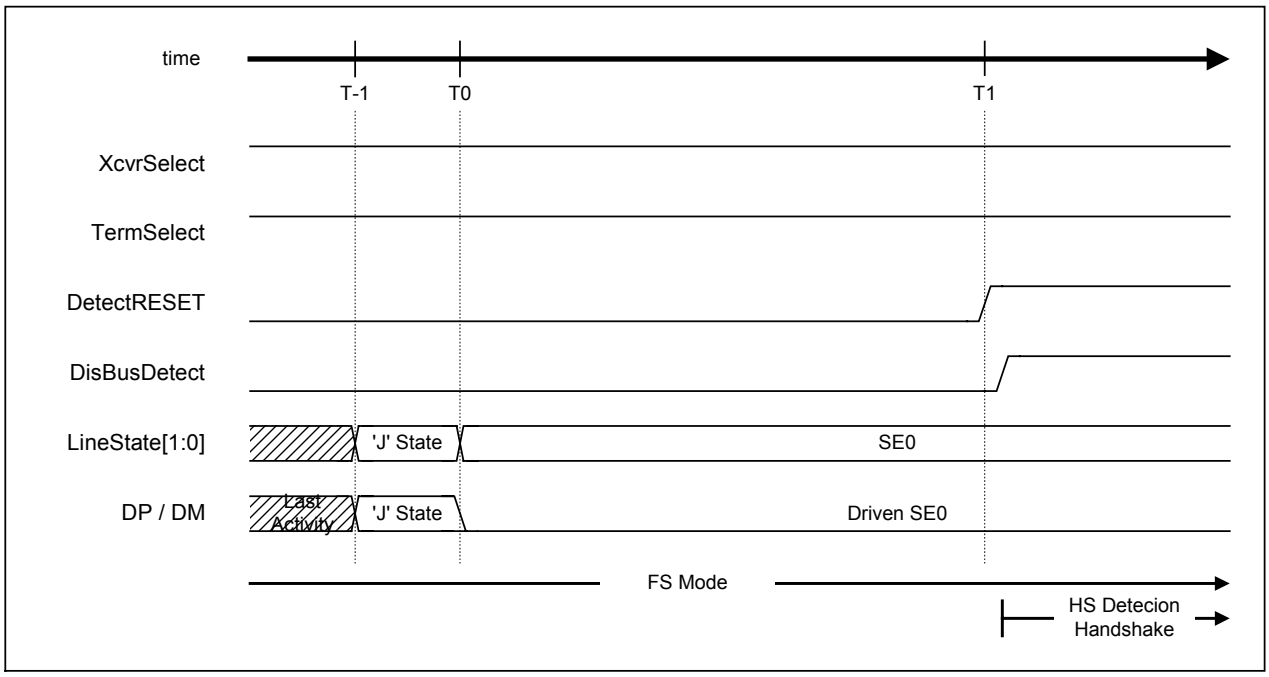


Figure 1-11 Reset timing (FS mode)

Table 1-11 Reset timing values (FS mode)

Timing Parameter	Description	Value
T-1	Most recent bus activity	
T0	Issues the reset command from the host downstream port.	0 (reference)
T1	If "SE0" continues, DetectRESET is set to "1," and the LSI determines a transition to reset. Sets DisBusDetect to "1" after detecting a reset command, then performs the HS Detection Handshake.	HS Reset T0 + 2.5us < T1 {T_{WTREV}}

Note: Brackets {} indicate terms defined under the USB 2.0 standards.

1.2.7.11.5 HS Detection Handshake

HS Detection Handshake is initiated by an "SE0" assert from the host downstream port during Suspend, FS, or HS states (at the start of resetting from these states). Refer to the USB 2.0 standards for more information.

This section describes the procedure for switching to HS Detection Handshake from the three states above.

In SUSPEND state, this LSI switches to HS Detection Handshake immediately after detecting "SE0" on the bus.

When operating in FS mode, the LSI switches to HS Detection Handshake after detecting "SE0" for more than 2.5 µs.

When operating in HS mode, the LSI first switches to FS mode after detecting "SE0" for more than 3.0 ms, since it must determine whether the state is USB Suspend or Reset. In this case, both the D_XcvrControl.XcvrSelect and D_XcvrControl.TermSelect bits are switched to FS mode, HS termination is disabled, and FS termination is enabled. Switching from these modes is performed within 3.125 ms by the hardware. The hardware checks the D_USB_Status.LineState[1:0] bit between 100 µs and 875 µs after switching modes and determines a USB SUSPEND state if "J" and a Reset if "SE0." The LSI proceeds to an HS Detection Handshake if a Reset is determined here.

A reset exists for at least 10 ms in either case, but the exact timing will vary slightly, depending on the previous state (HS or FS). The steps from "HS Reset T0" are described below with the Reset start time defined as "HS Reset T0."

The internal clock is sufficiently stable during this operation and should present no problems, but the internal clock is not output when Reset is detected if SLEEP or SNOOZE are issued during Suspend. The PM_Control_0.GoActAllDev or PM_Control_0.GoActDevice bits must always be set to "1" to operate the internal clock for the HS Detection Handshake. For detailed information on this operation, refer to "1.5 Power Management Functions."

1.2.7.11.5.1 When Connected to the FS Host Downstream Port

This section indicates the operation when the LSI is connected to a host downstream port that does not support HS. Both D_XcvrControl.XcvrSelect and D_XcvrControl.TermSelect bits must be in FS mode when the HS Detection Handshake starts (T0) (i.e., FS termination, namely DP pull-up resistance (Rpu), must be enabled and HS termination disabled).

The D_NegoControl.GoChirp bit is set first. The D_XcvrControl.OpMode[1:0] bit then switches to "Disable Bit Stuffing and NRZI encoding," and data consisting entirely of "0" is prepared (T1). This issues "HS K" (Device Chirp) over the bus. The D_XcvrControl.XcvrSelect bit is concurrently set to HS mode. Setting it to enable

transmission allows “HS K” (Device Chirp) to be sent to the host downstream port. After transmission, the LSI awaits a Host Chirp from the host downstream port (T2). Host downstream ports supporting HS normally send “HS J” and “HS K” continuously, after T3 (described later), but the Host Chirp will not be sent even by point T4 if the host downstream port does not support HS (as in this example). The D_XcvrControl.XcvrSelect bit is automatically switched to FS mode, the D_NegoControl.GoChirp bit is cleared, and the D_USB_Status.FSxHS and D_SIE_IntStat.ChirpCmp bits are set.

The XINT signal is asserted simultaneously if the D_SIE_IntEnb.EnChirpCmp and DeviceIntEnb.EnD_SIE_IntStat bits are set and MainIntEnb.EnDeviceIntStat is set. Use this to determine that the HS Detection Handshake is complete.

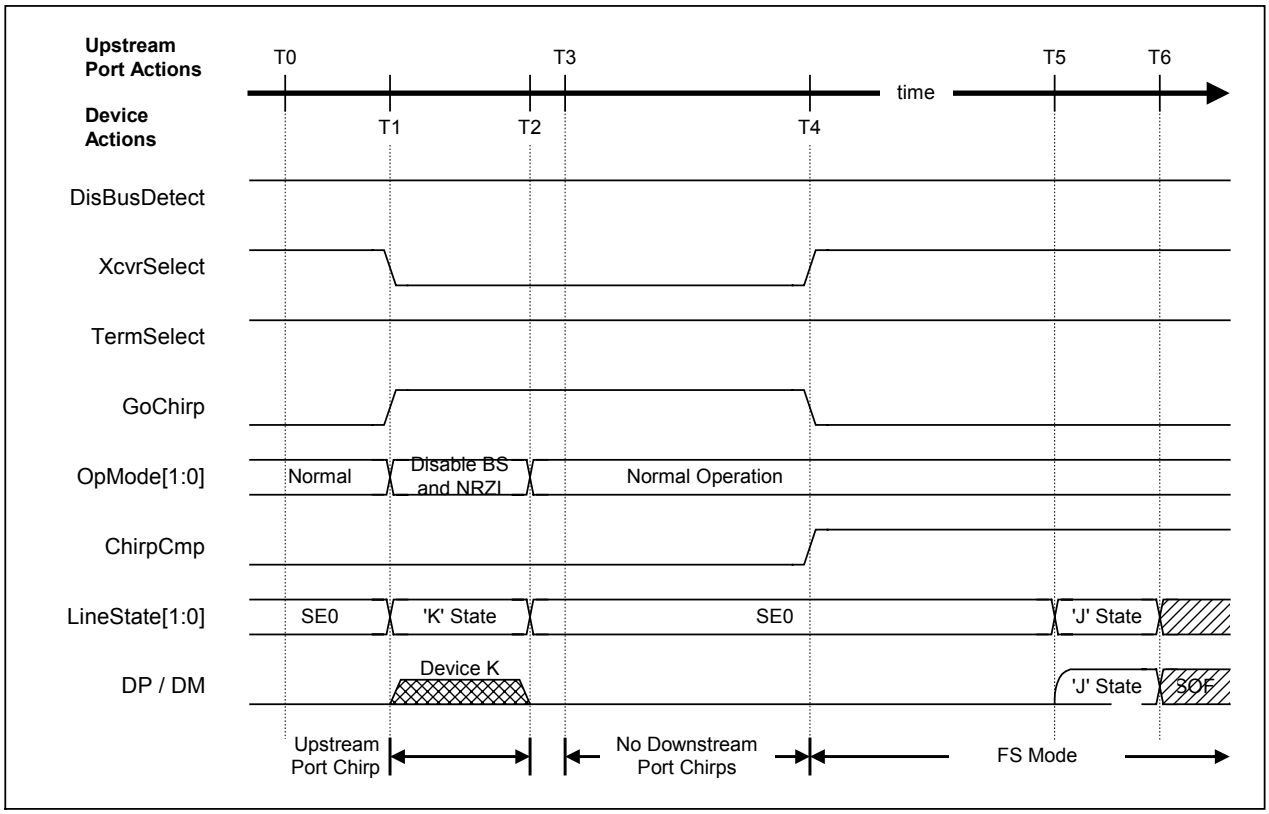


Figure 1-12 HS detection handshake timing (FS mode)

Table 1-12 HS detection handshake timing values (FS mode)

Timing Parameter	Description	Value
T0	HS detection handshake starts.	0 (reference)
T1	Enables the HS transceiver, sets GoChirp to “1,” and begins sending Chirp K.	T0 < T1 < HS Reset T0 + 6.0ms
T2	Chirp K transmission ends. Must be sent for at least 1 ms.	T1 + 1.0ms {T_{UCH}} < T2 < HS Reset T0 + 7.0ms {T_{UCHEND}}
T3	Chirp K transmission starts here if the host downstream port supports HS.	T2 < T3 < T2 + 100us {T_{WTDCH}}
T4	Returns to FS mode here if no chirp is detected. ChirpCmp is set to “1,” and the LSI awaits the end of the reset sequence.	T2 + 1.0ms < T4 {T_{WTFS}} < T2 + 2.5ms
T5	Reset sequence ends.	HS Reset T0 + 10ms {T_{DRST} (Min)}
T6	Normal operation in FS mode.	T6

Note: Brackets {} indicate terms defined under the USB 2.0 standards.

Note: Determines at 66000 cycles (internal clock: 60 MHz) to generate minimum 1 ms Chirp K.

1.2.7.11.5.2 When Connected to the HS Host Downstream Port

This section indicates the operation when the LSI is connected to a host downstream port that supports HS. Both D_XcvrControl.XcvrSelect and D_XcvrControl.TermSelect bits must be in FS mode when the HS Detection Handshake starts (T0) (i.e., FS termination, namely DP pull-up resistance (Rpu), must be enabled, and HS termination must be disabled).

The D_NegoControl.GoChirp bit is set first. The D_XcvrControl.OpMode[1:0] bit then switches to "Disable Bit Stuffing and NRZI encoding," and data consisting entirely of "0" is prepared (T1). This is to send "HS K" (Device Chirp) over the bus. The D_XcvrControl.XcvrSelect bit is concurrently set to HS mode. Setting it to enable transmission allows "HS K" (Device Chirp) to be sent to the downstream port. After transmission, the LSI awaits a Host Chirp from the downstream port (T2). The downstream port supports HS here, and "HS K" (Chirp K) and "HS J" (Chirp J) are sent continuously and alternately (T3). The XcvrControl.TermSelect bit is switched automatically to HS mode (T7) once Chirp K-J-K-J-K-J has been detected at least six times via the USB_Status.LineState[1:0] bit in this state (T6), switching entirely to HS mode. At the same time, the D_NegoControl.GoChirp bit is cleared, the D_NegoStatus.FSxHS bit is cleared, and D_SIE_IntStat.ChirpCmp is set.

The XINT signal is asserted simultaneously if the D_SIE_IntEnb.EnChirpCmp and DeviceIntEnb.EnD_SIE_IntStat bits are set and MainIntEnb.EnDeviceIntStat is set. Use this to determine that the HS Detection Handshake is complete.

Chirp K and Chirp J from this host downstream port must be recognized as bus activity to prevent the determination as a USB SUSPEND state. Chirp K and Chirp J are detected sequentially in HS mode and incorporated into the internal Suspend Timer.

The USB_Status.LineState[1:0] bit is used to detect Chirp K-J-K-J-K-J. The D_USB_Status.LineState[1:0] bit can be used since Chirp K and Chirp J are extremely slow compared to normal HS packets. However, since loading the bus signal onto the D_USB_Status.LineState[1:0] bit when receiving regular packets will result in significant noise levels, the D_USB_Status.LineState[1:0] bit outputs "J" when bus activity is detected and "SE0" when no bus activity is detected when D_XcvrControl.TermSelect bit is in HS mode.

The figure below illustrates how device-side HS termination using the D_XcvrControl.TermSelect bit works when the Chirp height is changing from point T6. The Chirp is normally around 800 mV when D_XcvrControl.TermSelect is in FS mode, and is approximately 400 mV when D_XcvrControl. TermSelect bit is in HS mode (same as for normal packets sent and received).

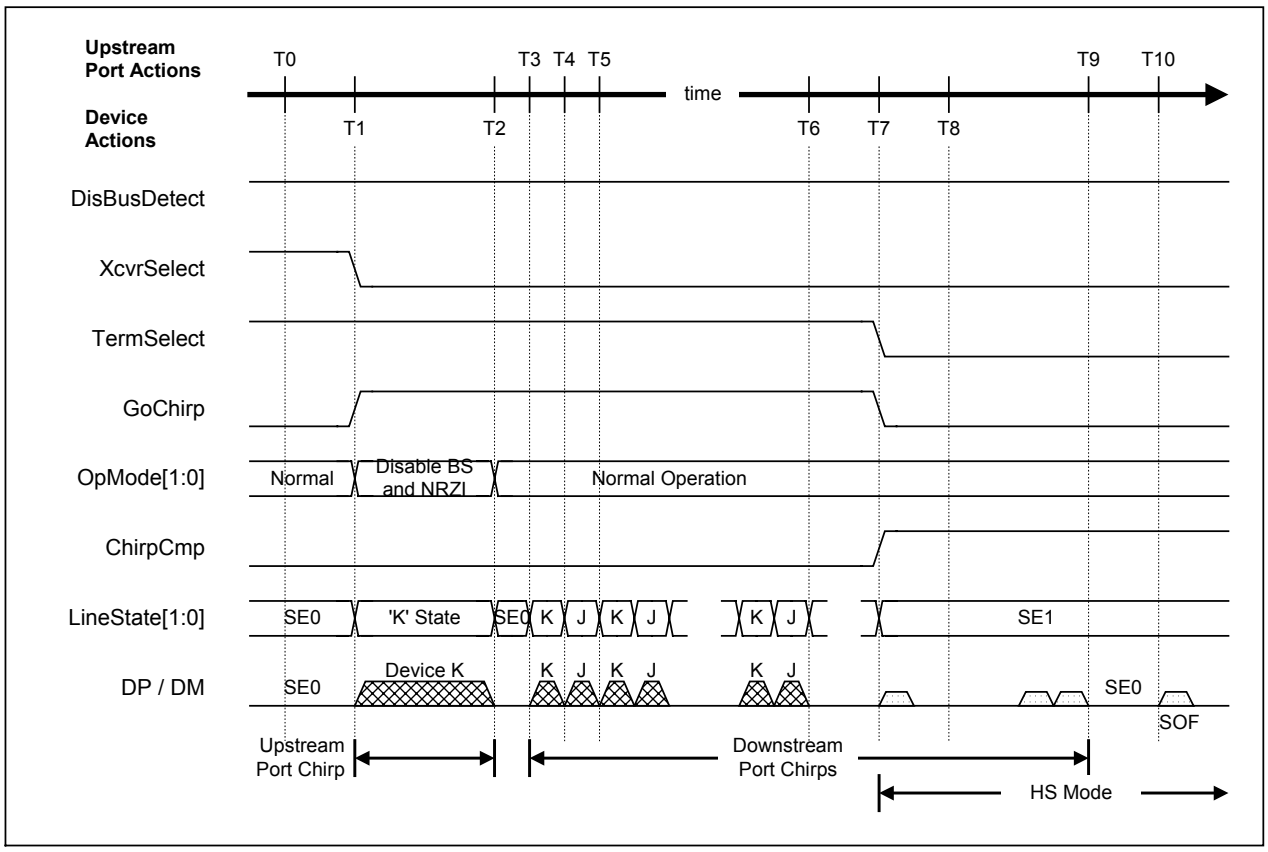


Figure 1-13 HS detection handshake timing (HS mode)

Table 1-13 HS detection handshake timing values (HS mode)

Timing Parameter	Description	Value
T0	HS detection handshake starts.	0 (reference)
T1	Enables the HS transceiver, sets GoChirp to “1,” and begins sending Chirp K.	T0 < T1 < HS Reset T0 + 6.0ms
T2	Chirp K transmission ends. Must be sent for at least 1 ms.	T1 + 1.0ms {T_{UCH}} < T2 < HS Reset T0 + 7.0ms {T_{UCHEND}}
T3	Host downstream port sends the first Chirp K to the bus.	T2 < T3 < T2 + 100us {T_{WTDCH}}
T4	Host downstream port sends a switch from Chirp K to Chirp J.	T3 + 40us {T_{DCHBIT} (Min)} < T4 < T3 + 60us {T_{DCHBIT} (Max)}
T5	Host downstream port sends a switch from Chirp J to Chirp K.	T4 + 40us {T_{DCHBIT} (Min)} < T5 < T4 + 60us {T_{DCHBIT} (Max)}
T6	Detects Chirp K-J-K-J-K-J.	T6
T7	Disables FS termination and enables HS termination on detecting Chirp K-J-K-J-K-J. ChirpCmp is set to “1,” and the LSI awaits the end of the reset sequence.	T6 < T7 < T6 + 500us
T8	Recognized as bus activity from Chirp K and Chirp J. However, not recognized as packet receipt, since SYNC is not detected.	T8
T9	Chirp K and Chirp J transmissions end from host downstream port.	T10 - 500us {T_{DCHSE0} (Max)} < T9 < T10 - 100us{T_{DCHSE0} (Min)}
T10	Reset sequence ends.	HS Reset T0 + 10ms {T_{DRST} (Min)}

Note: Brackets {} indicate terms defined under the USB 2.0 standards.

Note: Determines at 66000 cycles (internal clock: 60 MHz) to generate minimum 1 ms Chirp K.

1.2.7.11.5.3 If Reset During SNOOZE

The LSI internal clock does not produce output using PLL in the SNOOZE state. This section discusses resetting from the SNOOZE state.

If a reset is detected in SNOOZE state (T0), the D_SIE_IntStat.NonJ bit is set. A XINT signal is asserted simultaneously if the D_SIE_IntEnb.EnNonJ and DeviceIntEnb.EnD_SIE_IntStat bits are set, and the MainIntEnb.EnDeviceIntStat bit is set. The PM_Control_0.GoActDevice or PM_Control_0.GoActAllDev bits should be set to "1" by the firmware to enable the LSI to reset immediately from SNOOZE and proceed to the reset sequence (T1). PM_Control_1.PM_State[3:0] switches to "ACT_ALL" or "ACT_DEVICE," and internal clock output starts after the PLL power-up time has elapsed (T2). The HS Detection Handshake (described above) is then performed.

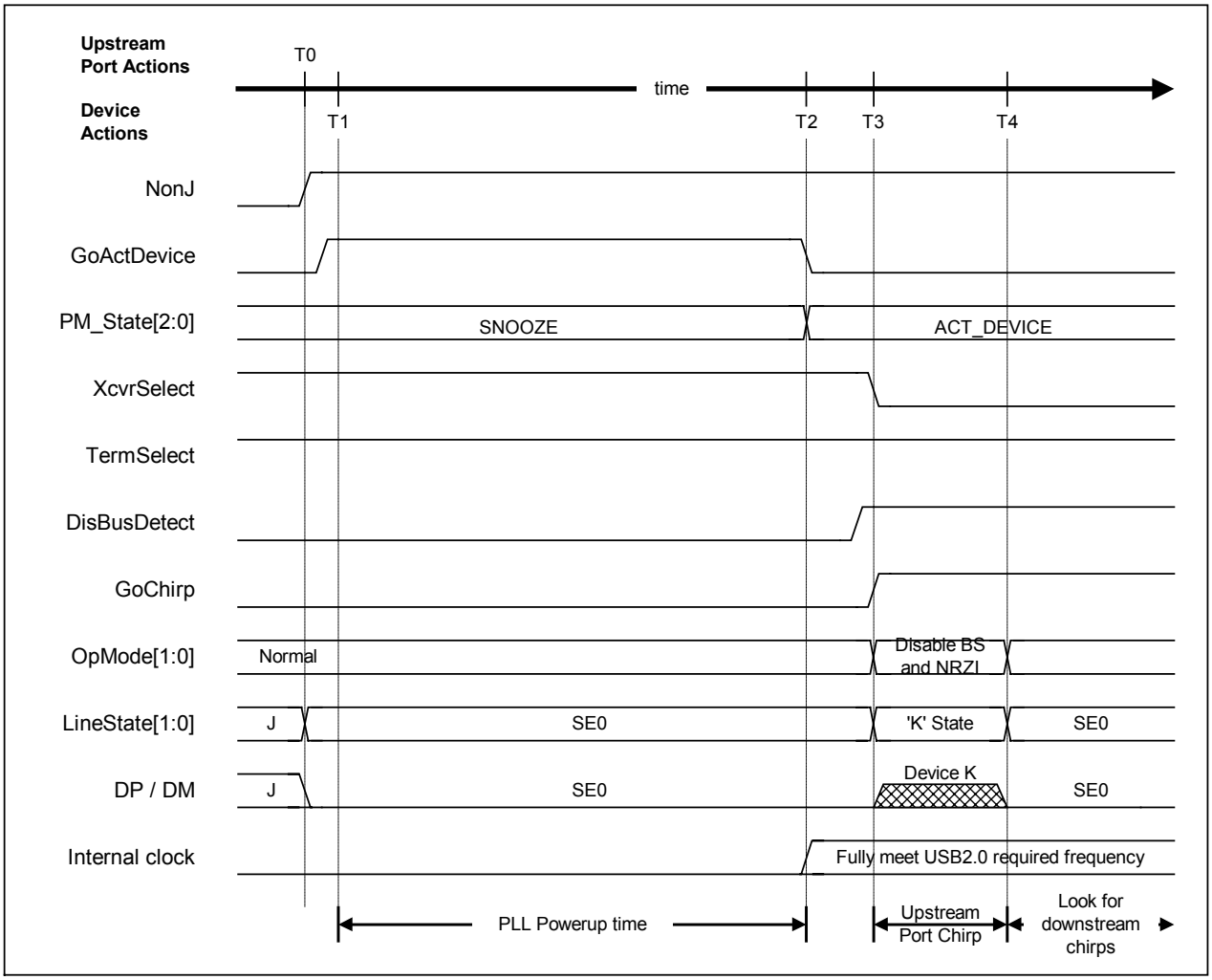


Figure 1-14 HS detection handshake timing from suspend

Table 1-14 HS detection handshake timing values from suspend

Timing Parameter	Description	Value
T0	A reset during SNOOZE is detected when NonJ is set to “1” and “LineState[1:0]” is confirmed as “SE0.”	0 (HS Reset T0)
T1	GoActDevice is set to “1” after reset is detected.	T1
T2	PM_State switches to “ACT_DEVICE.” The internal clock output stabilizes.	T1 + 250us < T2
T3	GoChirp is set to “1,” and Chirp K is sent to the bus. (DisBusDetect is set to “1” before sending Chirp K.)	T2 < T3 < HS Reset T0 + 5.8ms
T4	Chirp K transmission ends.	T3 + 1.0ms {T_{UCH}} < T4 < HS Reset T0 + 7.0ms {T_{UCHEND}}

Note: Brackets {} indicate terms defined under the USB 2.0 standards.

Note: Determines at 66000 cycles (internal clock: 60 MHz) to generate minimum 1 ms Chirp K.

Note: The situation in which the oscillator circuit is aborted (SLEEP state) is described later (PLL and OSC power-up time is required).

1.2.7.11.6 Resume Issue

This section describes methods for resuming automatically for whatever reason when Remote wakeup is supported and is enabled by the host. Note that Remote wakeup must be permitted no earlier than 5 ms after the bus has switched to idle. Current before entering USB SUSPEND state cannot be consumed from the VBUS until at least 10 ms after the Resume signal has been output.

The device first resets from SLEEP or SNOOZE for the Remote wakeup. The D_SIE_IntEnb.EnNonJ bit is cleared, and the PM_Control_0.GoActDevice or PM_Control_0.GoActAllDev bits are set (T0). Internal clock output starts once the PM_Control_1.PM_State[3:0] bit switches to “ACT_DEVICE” or “ACT_ALL” after the PLL power-up time has elapsed (T1).

The D_NegoControl.SendWakeup bit is then set and the Resume signal sent (T2). D_XcvrControl.OpMode[1:0] is set to “Disable Bit Stuffing and NRZI encoding” internally here, “0” is prepared as the data to be sent, the LSI switches to the packet transmission state, and “K” (Resume signal) is sent. The host downstream port returns “K” (Resume signal) over the bus when it detects this Resume signal (T3).

Clearing the D_NegoControl.SendWakeup bit approximately 1 ms after the start of Resume signal transmission aborts the Resume signal being sent over the bus (T4). However, the host downstream port keeps the same bus at the Resume signal here.

The D_NegoControl.RestoreUSB bit is then set. The host downstream port aborts Resume signal transmission after a preset period has elapsed (T5), 2-bit LS-EOP(2*SE0) is sent, and the LSI switches to the speed mode in effect prior to USB Suspend. Both the D_XcvrControl.XcvrSelect and D_XcvrControl.TermSelect bits are switched to the required mode (HS mode here) when this is detected (no longer “K”), and the D_NegoControl.RestoreUSB bit is then cleared and the D_SIE_IntStat.RestoreCmp bit is set. The XINT signal is asserted simultaneously if the D_SIE_IntEnb.EnRestoreCmp and DeviceIntEnb.EnD_SIE_IntStat bits are set and the MainIntEnb.EnDeviceIntStat bit is set.

The speed mode (HS or FS) is stored in the USB_Status.FSxHS bit on the start of USB Suspend, and resetting using Resume returns to the mode indicated by the D_USB_Status.FSxHS bit. The HS Detection Handshake is not performed for each resume here. Note that the explanation here describes only the situation for when the mode in effect prior to the USB Suspend was HS mode. If it was FS mode, the normal FS mode is used after T5, and there are no major differences in sequence.

There is no internal clock output for this LSI in the SNOOZE state (i.e., when the PM_Control_1.PM_State[3:0] bit is “SNOOZE”). The operations described here assume that the oscillator circuit is operating (SNOOZE state rather than SLEEP).

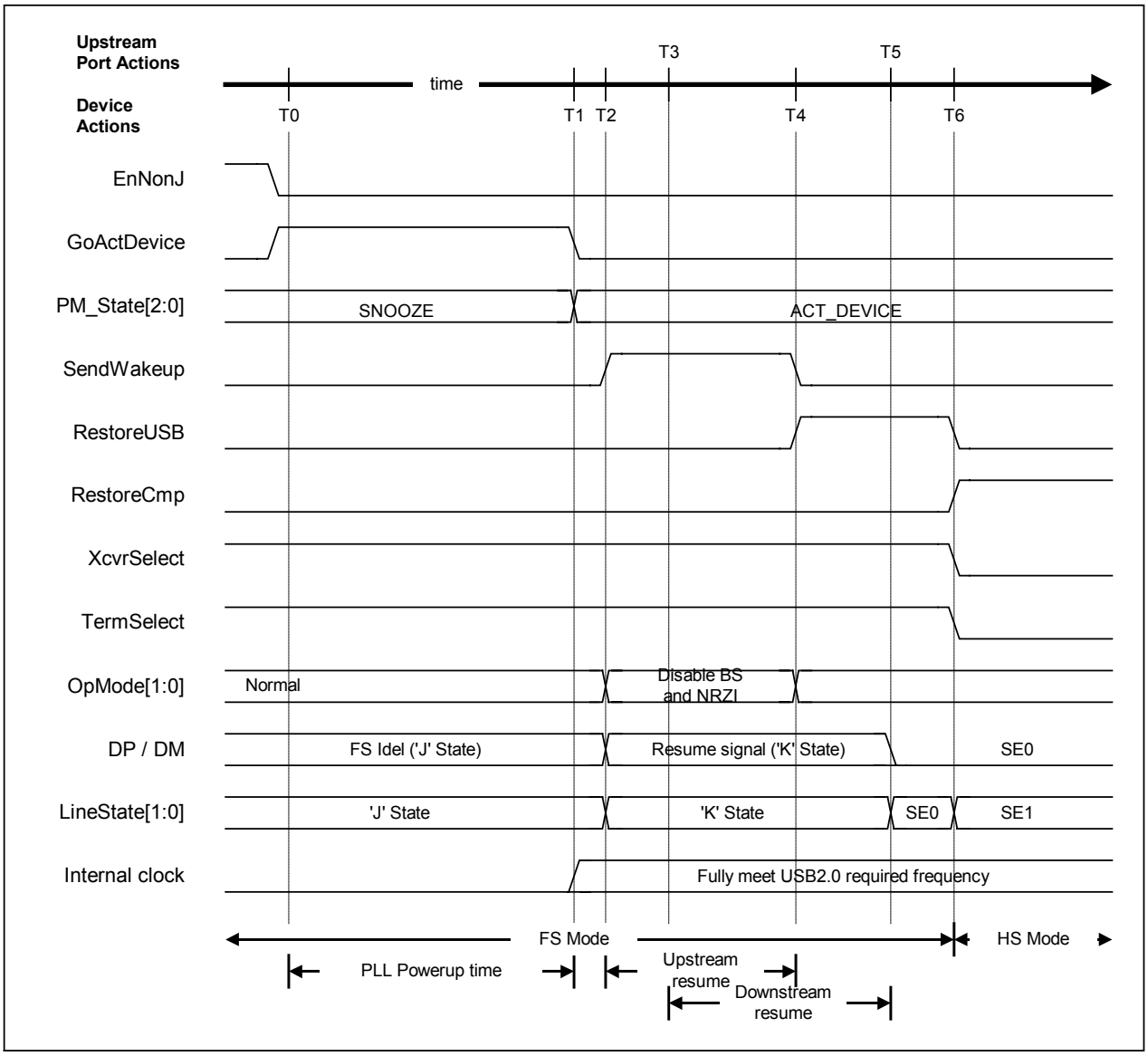


Figure 1-15 Assert resume timing (HS mode)

Table 1-15 Assert resume timing values (HS mode)

Timing Parameter	Description	Value
T0	Resume starts. GoActDevice is set to “1.” (EnNonJ must be cleared to “0” before Resume starts.)	0 (reference)
T1	PM_State switches to “ACT_DEVICE.” The internal clock output stabilizes.	T0 + 250us < T1
T2	SendWakeup is set to “1,” and FS “K” transmission starts. Current prior to USB Suspend must not be drawn for at least 10 ms here.	T0 < T2 < T0 + 10ms
T3	The host downstream port returns FS “K.”	T2 < T3 < T2 + 1.0ms
T4	SendWakeup is cleared to “0,” and FS “K” transmission ends. RestoreUSB is set to “1” after confirming “K” using LineState[1:0].	T2 + 1.0ms {T_{DRSMUP} (Min)} < T4 < T2 + 15ms {T_{DRSMUP} (Max)}
T5	The host downstream port ends FS “K” transmission.	T2 + 20ms {T_{DRSMDN}}
T6	RestoreCmp is set to “1.” The LSI automatically switches to HS mode if the mode in effect prior to USB Suspend was HS mode.	T5 + 1.33us {2 Low-speed bit times}

Note: Brackets {} indicate terms defined under the USB 2.0 standards.

1.2.7.11.7 Resume Detection

"J" (D_USB_Status.LineState[1:0] is "J") is observed on the bus while the LSI is in SUSPEND state. Observing "K" on the bus indicates that a Wake-up command (Resume command) has been received from the host downstream port (T0). The SIE_IntStat.NonJ bit is set here. The XINT signal is asserted simultaneously if the D_SIE_IntEnb.EnNonJ and DeviceIntEnb.EnD_SIE_IntStat bits are set and the MainIntEnb.EnDeviceIntStat bit is set.

The PM_Control_0.GoActDevice or PM_Control_0.GoActAllDev bits should be first set to "1" (T1). The internal clock output then starts concurrently with PM_Control_1. PM_State[3:0] switches to "ACT_DEVICE" or "ACT_ALL" after the PLL power-up time has elapsed (T2).

D_NegoControl.RestoreUSB bit is then set. The host downstream port aborts Resume signal transmission after a preset period has elapsed (T3), and the LSI switches to the speed mode in effect prior to the USB Suspend. Both the D_XcvrControl.XcvrSelect and D_XcvrControl.TermSelect bits are switched to the required mode (HS mode here) when this is detected (no longer "K"), and the D_NegoControl.RestoreUSB bit is then cleared and the D_SIE_IntStat.RestoreCmp bit is set. The XINT signal is asserted simultaneously if the D_SIE_IntEnb.EnRestoreCmp and DeviceIntEnb.EnD_SIE_IntStat bits are set and the MainIntEnb.EnDeviceIntStat bit is set.

The operations described here assume that the oscillator circuit is operating (SNOOZE state rather than SLEEP).

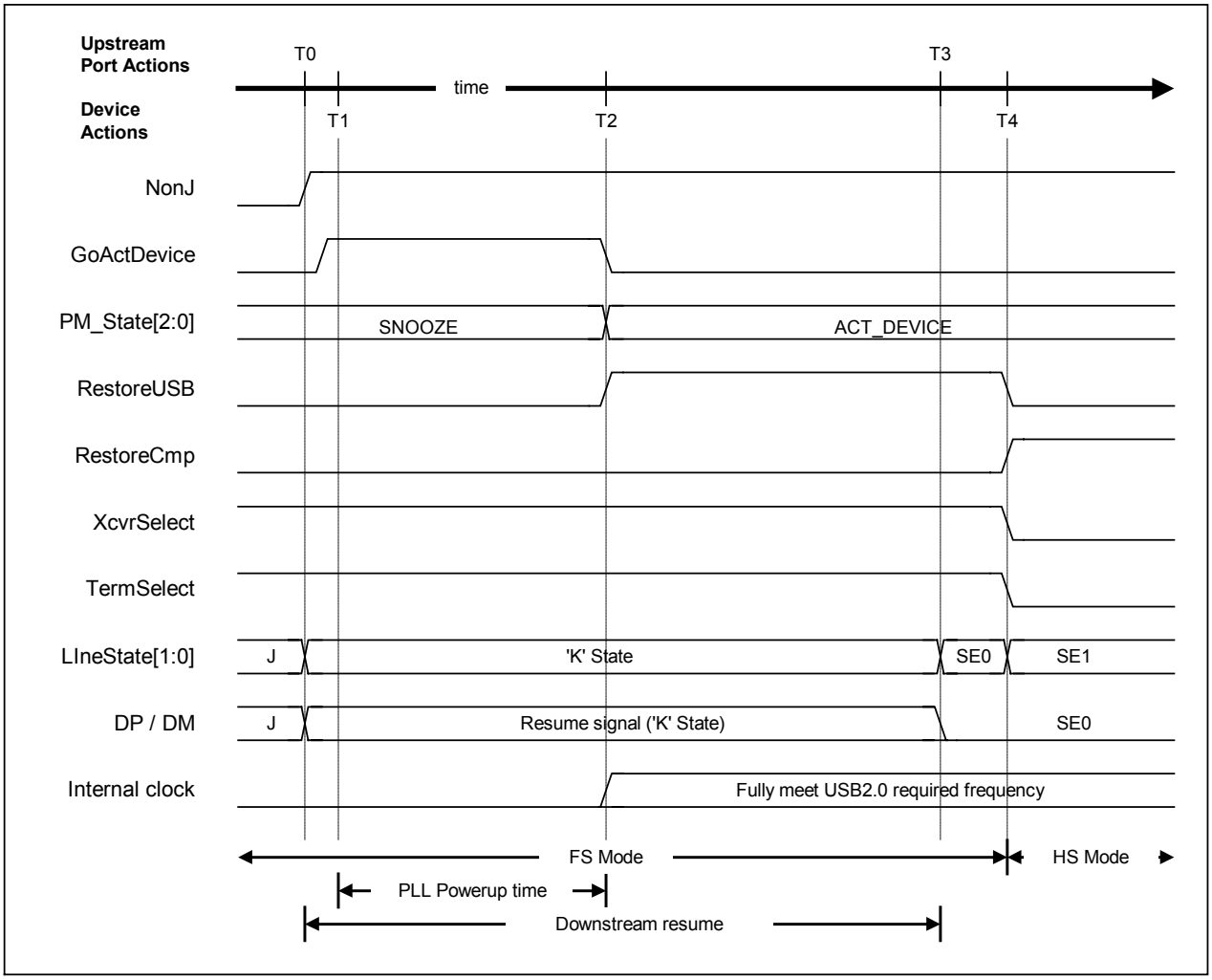


Figure 1-16 Detect resume timing (HS mode)

Table 1-16 Detect resume timing (HS mode)

Timing Parameter	Description	Value
T0	The host downstream port outputs FS “K.” NonJ is set to “1.”	0 (reference)
T1	GoActDevice is set to “1.”	T1
T2	PM_State is set to “ACT_DEVICE.” The internal clock output stabilizes. RestoreUSB is set to “1” after “K” is confirmed in LineState[1:0].	T1 + 250us < T2
T3	The host downstream port ends FS “K” transmission. At the same time, the host downstream port switches to the HS mode in effect prior to USB Suspend.	T2 + 20ms {T_{DRSMDN}}
T4	The LSI switches automatically to HS mode if the mode in effect prior to USB Suspend was HS mode.	T5 + 1.33us {2 Low-speed bit times}

Note: Brackets {} indicate terms defined under the USB 2.0 standards.

1.2.7.11.8 Cable Attachment

This section addresses the situation wherein the LSI is connected to a hub or host – i.e., when a cable is attached. When the cable is disconnected, the default values for the D_XcvrControl.XcvrSelect and D_XcvrControl.TermSelect bits should be FS mode and HS mode, respectively.

If the cable is connected when not initially connected (T0), VBUS is set to "H" and the D_USB_Status.VBUS bit is set at the same time (T1). If the LSI is in the SNOOZE state at this point, for example, the PM_Control_0.GoActDevice or PM_Control_0.GoActAllDev bit is set to "1" (T2), and internal clock output starts concurrently with PM_Control_1. PM_State[3:0] is set to "ACT_DEVICE" or "ACT_ALL" after the PLL power-up period has elapsed (T3). The D_XcvrControl.TermSelect bit should be set to FS mode (T4) to switch to FS mode temporarily, as it must initially be assumed that a FS device was connected.

The host downstream port then sends a Reset (T5), and the HS Detection Handshake starts.

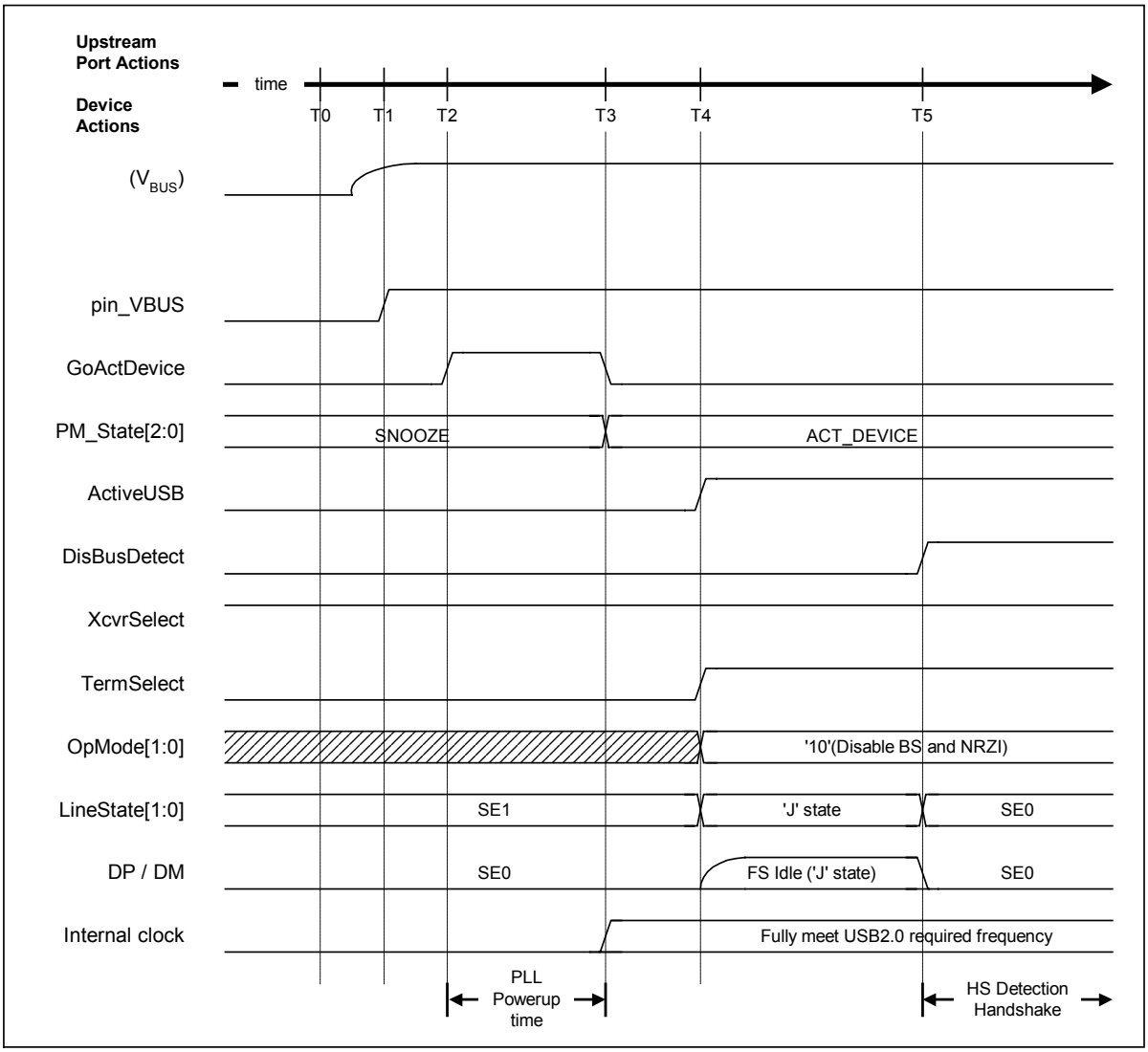


Figure 1-17 Device attach timing

Table 1-17 Device attach timing values

Timing Parameter	Description	Value
T0	No cable is attached.	0 (reference)
T1	Cable is attached, and input pin VBUS_B switches to H.	T1
T2	GoActDevice is set to “1.”	T2
T3	PM_State is set to “ACT_DEVICE.” The internal clock output stabilizes.	T2 + 250us < T3
T4	ActiveUSB is set to “1.” TermSelect is set to “1.” OpMode[1:0] is set to “00.” Switches to FS mode. FS termination is enabled.	T1 + 100ms {T_{SIGATT}} < T4
T5	A Reset is sent from the host downstream port. DisBusDetect is set to “1.”	T4 + 100ms {T_{ATTDB}} < T5

Note: Brackets {} indicate terms defined under the USB 2.0 standards.

1.3 USB Host Control

1.3.1 Channels

1.3.1.1 Channel Outline

This LSI includes channels, which refer to the host buffers corresponding to individual pipes and the setting registers used for transfers via the buffers.

Transfer information is set to the channels for individual IRPs (I/O Request Packets). FIFO areas are also assigned as buffers for the channels.

The channels divide the IRPs automatically into multiple transactions based on the information set. The channel settings can be switched for individual IRPs, enabling a single channel to support multiple endpoints.

Figure 1-18 illustrates the channel configuration.

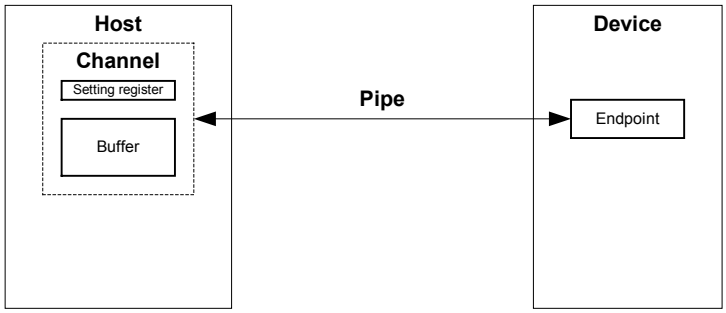


Figure 1-18 Channel outline

The firmware sets the transfer execution (H_CHx{x=0,a-e}Config_0.TranGo) after setting the buffer and transfer information. After setting the transfer execution, the firmware processing writes data to the buffer (for OUT transfer) and reads data from the buffer (for IN transfer) until all IRP data has been processed.

The hardware (channel) automatically divides IRPs into multiple transactions. Once transfer is complete, it issues an interrupt to notify the firmware.

Figure 1-19 illustrates the basic procedures involved in transfers.

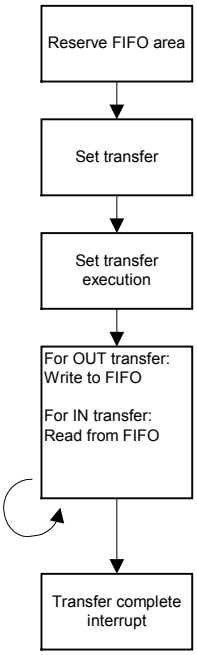


Figure 1-19 Basic channel transfer procedures

The LSI has a total of six channels, including a channel for control transfer only (CH0), a channel for bulk transfer only (CHa), and four channels for bulk/interrupt/isochronous transfer(CHb, CHc, CHd, CHe). Here, channel CH0 is referred to as the dedicated control channel, while CHa, CHb, CHc, CHd, and CHe are referred to as general channels.

Each channel includes fixed basic settings determined by the USB-defined interface, variable control for controlling each transfer, and status. The basic settings should be set when initializing the chip or when switching the USB-defined interface.

Up to four interrupt and isochronous transfers can be set at any one time.

Table 1-18 shows the transfer types supported by each channel.

Table 1-18 Supported transfer types

Channel	Supported transfer type	Remarks
CH0	Control transfer	Control transfer support function (1.3.4.3) can be used.
CHa	Bulk transfer	Bulk-only support function (1.3.8) can be used.
CHb, CHc CHd, CHe	Bulk transfer Interrupt transfer Isochronous transfer	Audio class assist function (1.3.9) can be used.

1.3.1.2 Dedicated Control Channel

This LSI uses a dedicated control channel (CH0) for control transfers. CH0 is time-multiplexed for control transfers to multiple endpoints.

Table 1-19 shows the basic settings for the dedicated control channel (CH0).

Table 1-19 Dedicated control channel basic settings

Item	Register/bit	Description
Transfer speed	H_CH0Config_0.SpeedMode	Sets the transfer speed (HS/FS/LS) for the endpoints corresponding to channel CH0.
Toggle sequence bit	H_CH0Config_0.Toggle	Sets the toggle sequence bit initial values when initiating a transaction. Also indicates the toggle sequence bit status during or after a transaction has been completed.
Transaction type	H_CH0Config_1.TID	Sets the transaction type (SETUP/IN/OUT) issued by channel CH0.
Max packet size	H_CH0MaxPktSize	Sets the maximum packet size to 8 for LS operations and 8, 16, 32, or 64 for FS operations. The maximum size set for HS operations is 64.
USB address	H_CH0FuncAdrs.FuncAdrs	Sets the USB address forming the destination for the IRP to be executed by channel CH0.
Endpoint number	H_CH0FuncAdrs.EP_Number	Sets the endpoint number forming the destination for the IRP to be executed by channel CH0.
Hub address	H_CH0HubAdrs.HubAdrs	Sets the USB address for the hub used for split transactions.
Port number	H_CH0HubAdrs.Port	Sets the port number for the hub used for split transactions.
IRP data quantity	H_CH0TotalSize_H, H_CH0TotalSize _L	Sets the data volume in bytes for the IRP to be executed by channel CH0.
FIFO area	AREAn{n=0-5}StartAdrs_H, AREAn{n=0-5}StartAdrs_L, AREAn{n=0-5}EndAdrs_H, AREAn{n=0-5}EndAdrs_L,	Sets the FIFO address for the area assigned to channel CH0. The FIFO area should be at least as large as the max packet size for channel CH0. The FIFO area size affects the data transfer throughput. Refer to “1.6 FIFO Management” for detailed information on FIFO area assignments.
FIFO area join	AREAn{n=0-5}Join_1.JoinEP0CH0	Joins channel CH0 to the FIFO area. Join channel CH0 to AREA0 when using the control transfer support function.
Setup data	H_CH0SETUP_x(x=0-7)	Sets the 8-byte data to be sent by the setup transaction.

1.3.1.3 General Channels

General channels can be used to set the transaction direction and USB address or endpoint number as required. They can support up to five endpoints simultaneously on a one-to-one basis. Implemented in the same way for the dedicated control channel, time-multiplexing for each individual IRP allows transfers to and from more than five endpoints.

Each channel includes fixed basic settings determined by the USB-defined interface, variable control for controlling each transfer, and status. The basic settings should be set when initializing the chip or when switching the USB-defined interface.

Table 1-20 shows the basic settings for the general channels. The USB-defined interface should be configured by making the appropriate settings in conjunction with the specifics of the USB-defined interface definition and enabled status.

Table 1-20 Basic settings for general channels

Item	Register/bit	Description
Transfer speed	H_CHx{x=a-e}Config_0.SpeedMode	Sets the transfer speed (HS/FS/LS) for endpoints corresponding to each channel.
Toggle sequence bit	H_CHx{x=a-e}Config_0.Toggle	Sets the toggle sequence bit initial values when initiating a transaction. Also indicates toggle sequence bit status during or after a transaction has been completed.
Transaction type	H_CHx{x=a-e}Config_1.TID	Sets the transaction type (IN/OUT) issued by each channel.
Transfer type	H_CHx{x=b-e}Config_1.TranType	Sets the transfer type (bulk/interrupt/isochronous) issued by each channel.
Max packet size	H_CHx{x=a-e}MaxPktSize_H, H_CHx{x=a-e}MaxPktSize_L	Sets the maximum packet size to any value between 1 and 1,024 bytes for transactions executed using each channel.
Hub address	H_CHx{x=a-e}HubAdrs.HubAdrs	Sets the USB address for the hub used for split transactions.
Port number	H_CHx{x=a-e}HubAdrs.Port	Sets the port number for the hub used for split transactions.
USB address	H_CHx{x=a-e}FuncAdrs.FuncAdrs	Sets the USB address forming the destination for the IRP to be executed by each channel.
Endpoint number	H_CHx{x=a-e}FuncAdrs.EP_Number	Sets the endpoint number forming the destination for the IRP to be executed by each channel.
IRP data quantity	H_CHx{x=a-e}TotalSize_HH, H_CHx{x=a-e}TotalSize_HL, H_CHx{x=a-e}TotalSize_LH, H_CHx{x=a-e}TotalSize_LL	Sets the data volume in bytes for the IRP to be executed by each channel.
Token issue interval	H_CHx{x=b-e}Interval_H, H_CHx{x=b-e}Interval_L	Sets the interval (cycle) for issuing tokens for interrupts and isochronous transfers.
FIFO area	AREAn{n=0-5}StartAdrs_H, AREAn{n=0-5}StartAdrs_L, AREAn{n=0-5}EndAdrs_H, AREAn{n=0-5}EndAdrs_L,	Sets the FIFO address for the area assigned to each channel. The FIFO area should be at least as large as the max packet size for each channel. The FIFO area size affects the data transfer throughput. Refer to “1.6 FIFO Management.” for detailed information on FIFO area assignment.
FIFO area join	AREAn{n=0-5}Join_1.JoinEPxCHx{x=a-e}	Joins each channel to the FIFO area. Join channel CHa to AREA1 when using the bulk-only support function.

1.3.1.4 Channel Usage Examples

1.3.1.4.1 With One Storage Device Connected

Figure 1-20 shows a typical channel configuration with a USB Mass Storage Class (BulkOnly Transport Protocol) storage device (e.g., hard disk drive) connected. Devices supporting this class use control, bulk IN, and bulk OUT transfers.

CH0 is used for control transfers.

General channels are used for bulk IN and bulk OUT transfers, but CHa is used when using the bulk-only support function with this LSI.

Included in CHa, the bulk-only support function manages USB Mass Storage Class (BulkOnly Transport Protocol) command transport (CBW), data transport, and status transport (CSW) automatically (see "1.3.8 Bulk-only Support Function").

If the bulk-only support function is not used, bulk IN and bulk OUT transfers can be assigned individually to general channels (e.g., CHb and CHc). In this case, transport must be managed via software.

The hardware executes transactions by scheduling transfers to and from channels (see "1.3.2 Scheduling").

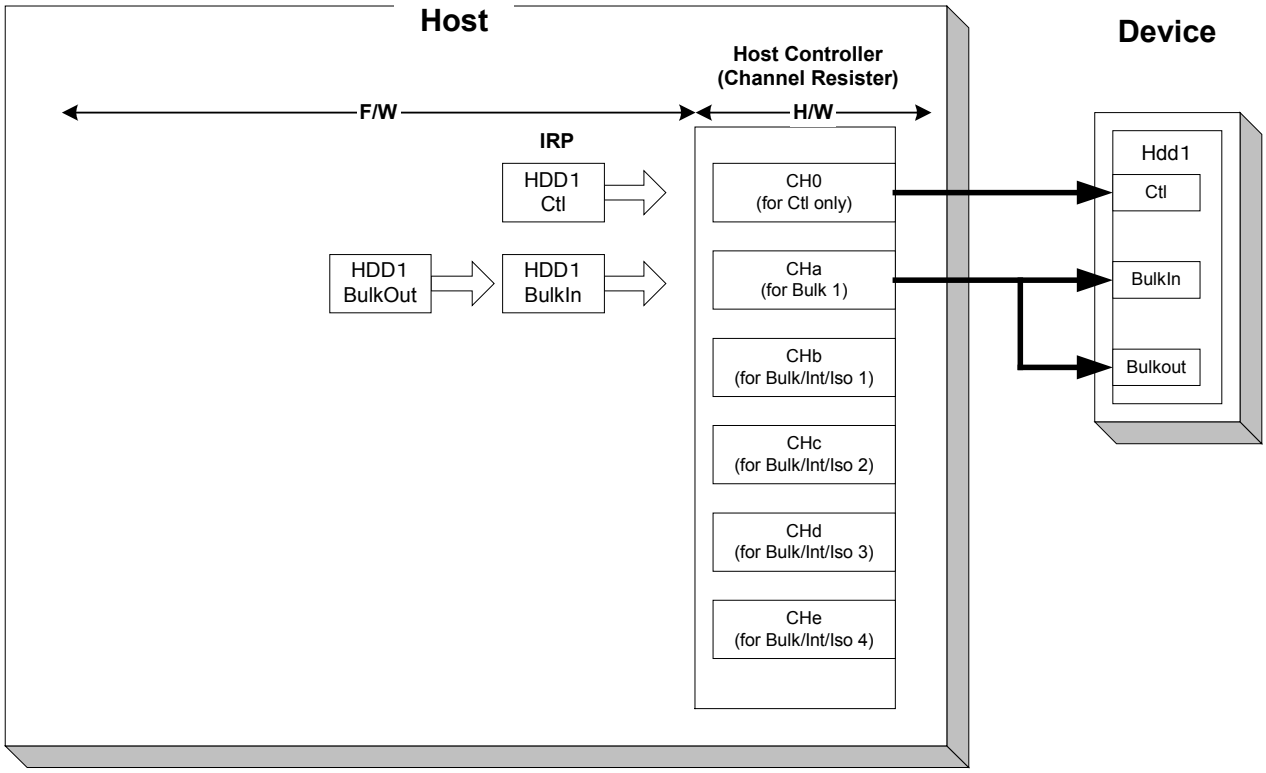


Figure 1-20 Channel usage example (with one storage device connected)

1.3.1.4.2 With One Communication Device Connected

Figure 1-21 shows a typical channel configuration with a USB Communication Device Class communication device (e.g., wireless LAN adapter) connected. Devices supporting this class use control, bulk IN, bulk OUT, and interrupt IN transfers.

CH0 is used for control transfers.

General channels (e.g., CHb and CHc) are assigned separately for the respective transfers, since bulk IN and bulk OUT transfers are performed in parallel for this class.

A general channel (e.g., CHd) is also assigned individually for interrupt IN transfer in the same way.

The hardware executes transactions by scheduling transfers to and from channels (see "1.3.2 Scheduling").

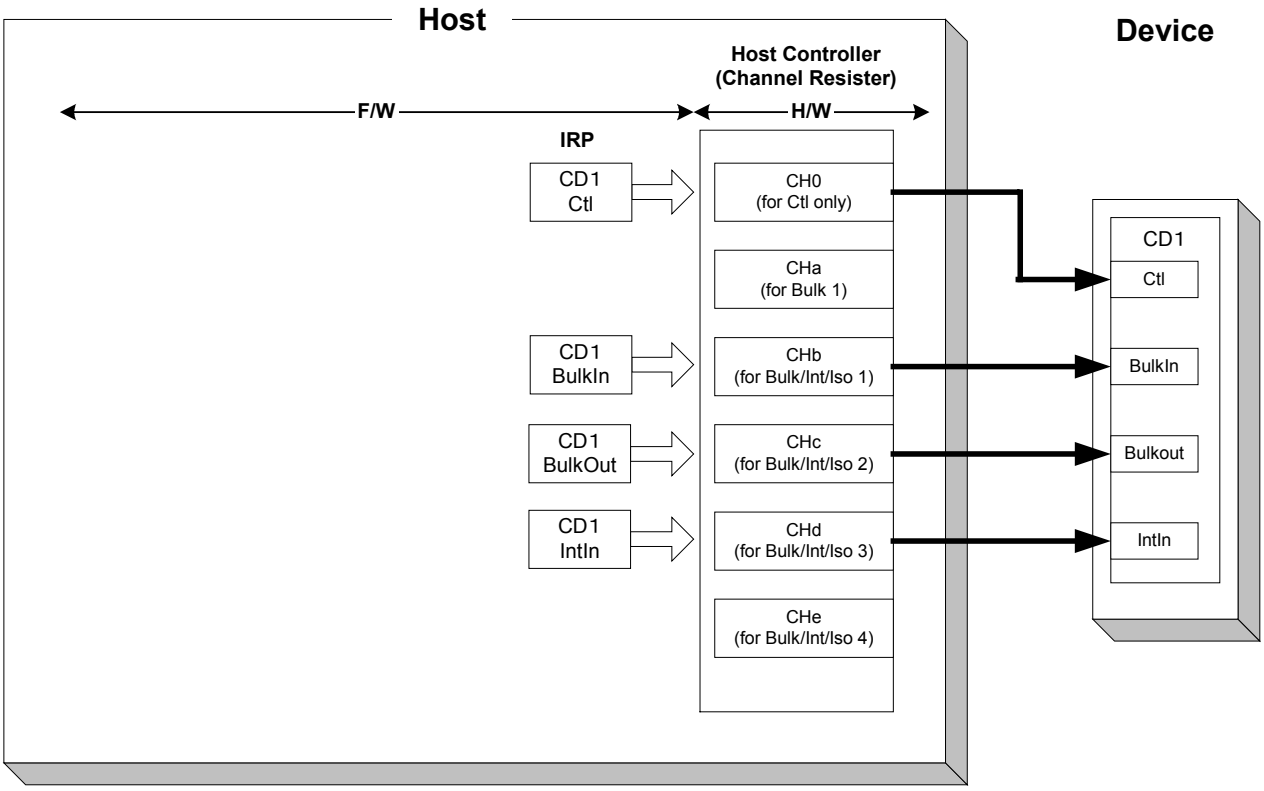


Figure 1-21 Channel usage example (with one communication device connected)

1.3.1.4.3 With One Human Interface Device Connected

Figure 1-22 shows a typical channel configuration with a USB Human Interface Device Class device (e.g., mouse) connected. Devices supporting this class use control and interrupt IN transfers.

CH0 is used for control transfers.

A general channel (e.g., CHd) is assigned for interrupt IN transfers.

The hardware executes transactions by scheduling transfers to and from channels (see "1.3.2 Scheduling").

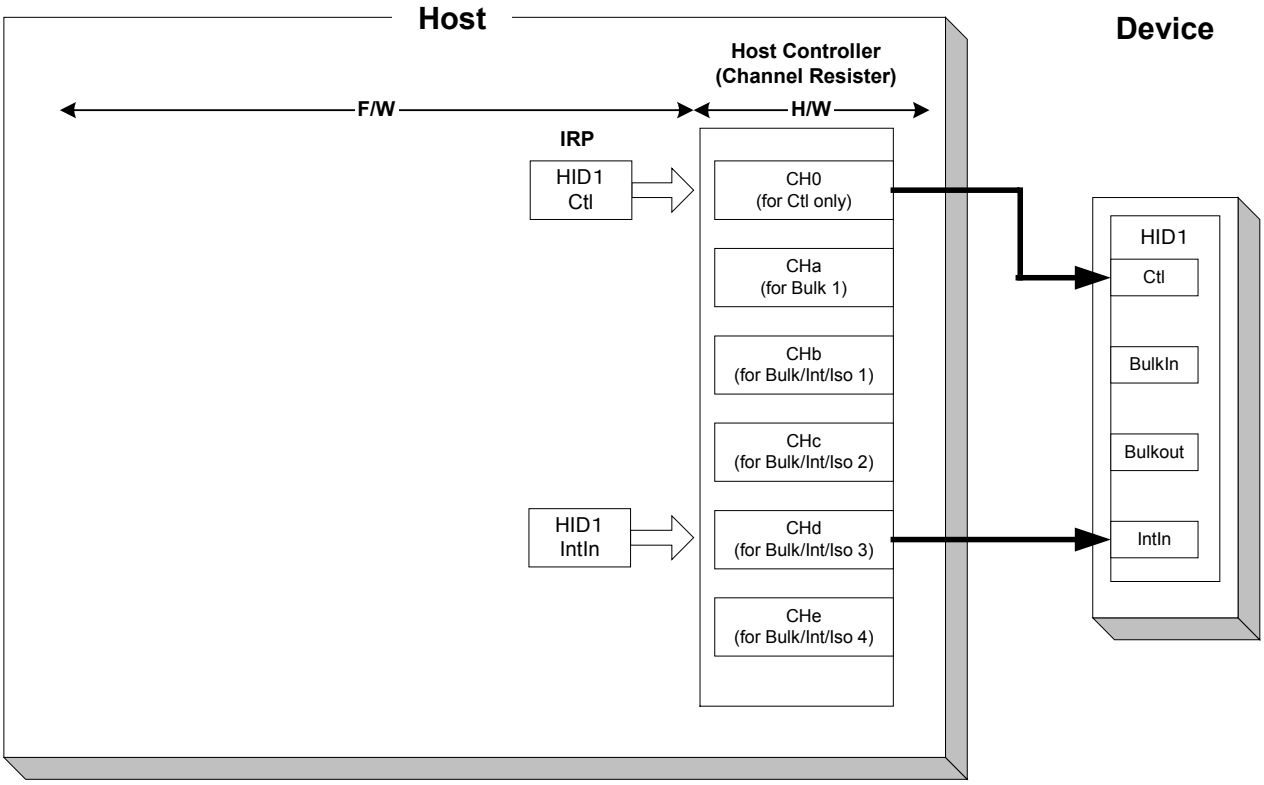


Figure 1-22 Channel usage example (with one human interface device connected)

1.3.1.4.4 With Two Storage Devices Connected via a Hub

Fgiure 1-23 shows a typical channel configuration with two USB Mass Storage Class (BulkOnly Transport Protocol) storage devices (e.g., hard disk drive or USB memory) connected via a hub. Devices supporting USB Mass Storage class use bulk IN and bulk OUT transfers.

Control and interrupt IN transfers are used for the hub.

The USB memory in this example is assumed to have a built-in hub.

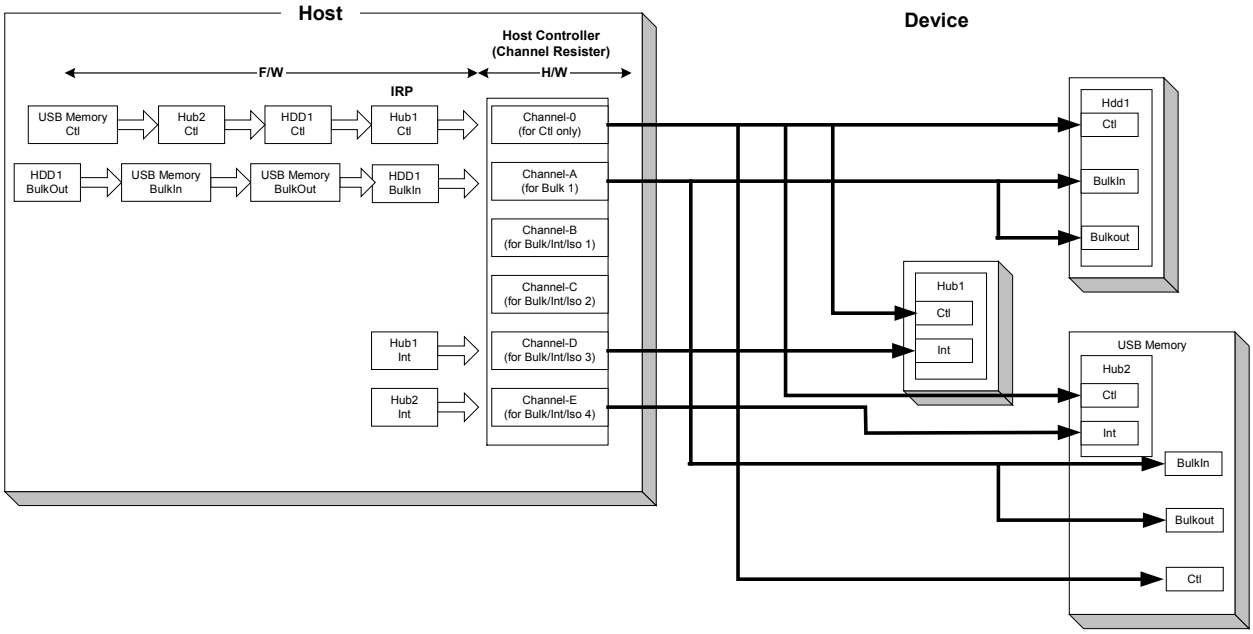
CH0 is time-multiplexed for control transfer to and from all devices.

General channels (e.g., CHd and CHe) are individually assigned for hub and USB memory interrupt IN transfers.

CHa is used when using the bulk-only support function with this LSI for bulk IN and bulk OUT transfers to and from a hard disk or USB memory. The bulk-only support function manages USB Mass Storage Class (BulkOnly Transport Protocol) command transport (CBW), data transport, and status transport (CSW) automatically (see "1.3.8 Bulk-only Support Function") and is included in CHa. If the bulk-only support function is used, the transfer devices (hard disk drive and USB memory in this example) are used by switching individually between command transport and status transport.

If the bulk-only support function is not used, bulk IN and bulk OUT transfers can be assigned individually to general channels. In this case, transport must be managed via software.

The hardware executes transactions by scheduling transfers to and from channels (see "1.3.2 Scheduling").



Fgiure 1-23 Channel usage example (with two storage devices connected via a hub)

1.3.2 Scheduling

The hardware selects one of the channels for which transfer execution (H_CHx{x=0,a-e}Config_0.TranGo) has been set, then determines whether the transfer set for that channel can be executed. If a transfer is determined to be possible, the transaction is executed in accordance with the settings. Once the transaction is completed, the hardware selects another channel, determines whether execution is possible, and executes the transaction in the same way.

The hardware performs transfers with multiple endpoints using multiple channels by repeating this procedure of selecting a channel, determining execution feasibility, and executing the transaction.

Table 1-21 shows the control items for channel CH0 scheduling control.

Table 1-21 Channel CH0 scheduling settings

Item	Register/bit	Description
Execute transfer	H_CH0Config_0.TranGo	Sets transfer execution for channel CH0. Performs the transfer in accordance with the channel settings.

Table 1-22 shows the control items for general channel (CHa, CHb, CHc, CHd, CHe) scheduling control.

Table 1-22 General channel scheduling settings

Item	Register/bit	Description
Execute transfer	H_CHx{x=a-e}Config_0.TranGo	Sets transfer execution for each channel. Performs the transfer in accordance with the channel settings.

1.3.3 Transactions

This LSI provides transaction execution functions via hardware and interfaces to execute transactions to and from the firmware. The interfaces for the firmware consist of control and status registers and interrupt signals asserted by status. Refer to "2 REGISTERS." for detailed information on the settings asserting interrupts by status.

The hardware selects a channel, determines whether transfer is possible, then executes the transaction in accordance with the channel settings, if a transfer is determined to be possible. The LSI issues the status to the firmware for each transaction. However, the firmware need not manage each transaction.

For example, when using the IN channel, the firmware can execute IN transactions automatically and continuously by reading data from the FIFO to create free space in the FIFO using the CPU interface (DMA reading or register reading). If using the OUT channel, the firmware can execute OUT transactions automatically and continuously by writing data to the FIFO to create valid data in the FIFO using the CPU interface (DMA writing or register writing)

Table 1-23 shows the control items and status involved in channel CH0 transaction control.

Table 1-23 Channel CH0 control items and status

Item	Register/bit	Description
Transaction status	H_CH0IntStat.TotalSizeCmp, H_CH0IntStat.TranACK, H_CH0IntStat.TranErr, H_CH0IntStat.ChangeCondition	Indicates the transaction results.
Transaction condition code	H_CH0ConditionCode	Indicates the transaction result details.

Table 1-24 shows the control items and status involved in general channel (CHa, CHb, CHc, CHd, CHe) transaction control.

Table 1-24 General channel control items and status

Item	Register/bit	Description
Transaction status	H_CHx{x=a-e}IntStat.TotalSizeCmp, H_CHx{x=a-e}IntStat.TranACK, H_CHx{x=a-e}IntStat.TranErr, H_CHx{x=a-e}IntStat.ChangeCondition	Indicates the transaction results.
Transaction condition code	H_CHx{x=a-e}ConditionCode	Indicates the transaction result details.

1.3.3.1 SETUP Transaction

The transaction type (H_CH0Config_1.TID) is set to "SETUP" for the CH0 basic setting register. The other basic settings are set appropriately, setup data (8 bytes) is written to the H_CH0SETUP_0 to 7 registers, and transfer execution (H_CH0Config_0.TranGo) is set, subjecting channel CH0 to USB transfer scheduling. When scheduling selects the corresponding channel, the remaining frame time is determined, and the SETUP transaction is executed.

The data in registers H_CH0SETUP_0 to 7 is used for SETUP transactions, and the data packet data length is 8 bytes.

On receiving an ACK for the SETUP transaction, ACK status (H_CH0IntStat.TranACK bit) is issued to the firmware.

If a normal response is not received for the SETUP transaction, the condition code (H_CH0ConditionCode) is set to "RetryError," and a TranErr status (H_CH0IntStat.TranErr bit) is issued to the firmware. Retry processing is performed, but H_CH0Control.TranGo is automatically cleared to end the transfer if the error occurs three times in succession, after which a ChangeCondition status (H_CH0IntStat.ChangeCondition bit) is issued.

Figure 1-24 shows the SETUP transaction arrangement. (a) The LSI issues a SETUP token addressed to the endpoint 0 existing at the destination node. (b) The LSI then sends an 8-byte data packet. (c) On receiving the ACK, the LSI automatically sets the corresponding register and issues the status to the firmware.

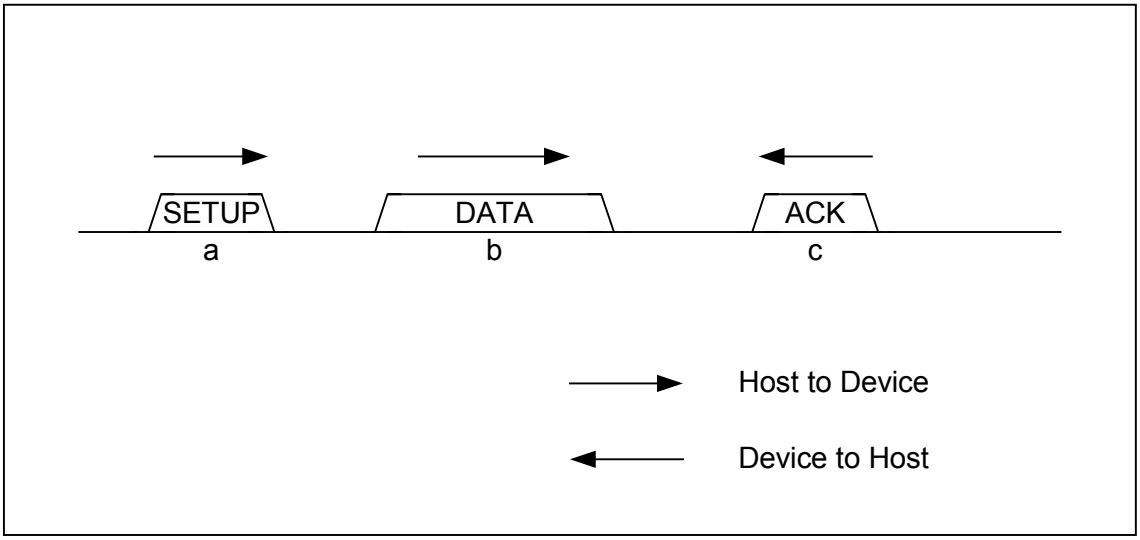


Figure 1-24 SETUP transaction

1.3.3.2 Bulk OUT Transaction

The transfer type (H_CHx{x=b-e}Config_1.TranType) is set to “Bulk” and the transaction type (H_CHx{x=a-e}Config_1.TID) is set to “OUT” for the CHx{x=a-e} basic setting register. The other basic settings are set appropriately. Transfer execution (H_CHx{x=a-e}Config_0.TranGo) is set, subjecting the channel to USB transfer scheduling by hardware. When scheduling selects the corresponding channel, the valid FIFO data quantity and remaining frame time is determined, and a bulk OUT transaction is executed. Channel CHa is dedicated for bulk transfers; the basic setting registers do not include a transfer type.

The data packet size of the individual data packets is the smaller of H_CHx{x=a-e}MaxPacketSize_H,L and H_CHx{x=a-e}TotalSize_HH,HL,LH,LL.

An ACK status (H_CHx{x=a-e}IntStat.TranACK bit) is issued to the firmware on receipt of an ACK for the bulk OUT transaction. The FIFO is then updated, freeing the area by treating the data sent as already sent.

If an NAK is received for the bulk OUT transaction, the FIFO is not updated, and the area is not freed. Selecting the corresponding channel again executes the same transaction.

If a STALL is received for the bulk OUT transaction, H_CHx{x=a-e}Config_0.TranGo is automatically cleared, ending the transfer, and the condition code (H_CHx{x=a-e}ConditionCode) is set to “STALL.” A ChangeCondition status (H_CHx{x=a-e}IntStat.ChangeCondition bit) is then issued to the firmware. The FIFO is not updated, and the area is not freed.

If a normal response is not received for the bulk OUT transaction, the FIFO is not updated, and the area is not freed. The condition code (H_CHx{x=a-e}ConditionCode) is set to “RetryError,” and a TranErr status (H_CHx{x=a-e}IntStat.TranErr bit) is issued to the firmware. Retry processing is performed, but H_CHx{x=a-e}Control.TranGo is automatically cleared to end the transfer if the error occurs three times in succession, after which a ChangeCondition status (H_CHx{x=a-e}IntStat.ChangeCondition bit) is issued to the firmware.

Figure 1-25 shows the procedure for a bulk OUT transaction when completed. (a) The LSI issues an OUT token addressed to the OUT-direction endpoint existing at the destination node. (b) The LSI then sends a data packet no larger than the maximum packet size. (c) On receiving the ACK, the LSI automatically sets the corresponding register and issues the status to the firmware.

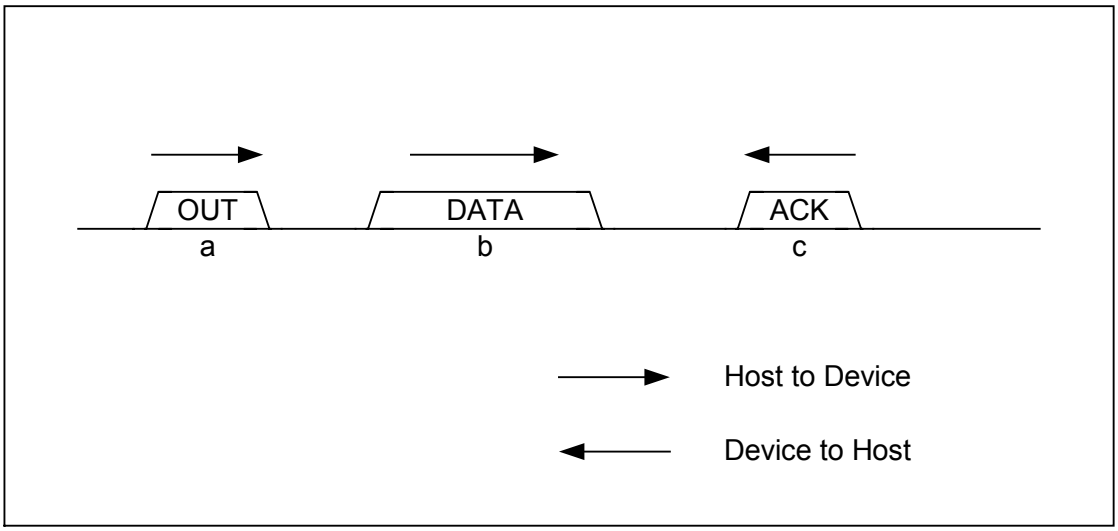


Figure 1-25 OUT transaction

1.3.3.3 Interrupt OUT Transaction

The transfer type (H_CHx{x=b-e}Config_1.TranType) is set to "Interrupt," and the transaction type (H_CHx{x=b-e}Config_1.TID) is set to "OUT" for the CHx{x=b-e} basic setting register. The token-issuing interval (H_CHx{x=b-e}Interval_H,L) is set, other basic settings are set appropriately, and transfer execution (H_CHx{x=b-e}Config_0.TranGo) is set, subjecting the channel to USB transfer scheduling by hardware. When scheduling selects the corresponding channel, the token-issuing interval (H_CHx{x=b-e}Interval_H,L), valid FIFO data quantity and remaining frame time are determined, and an interrupt OUT transaction is executed.

The data packet size is the smaller of H_CHx{x=b-e}MaxPacketSize_H,L and H_CHx{x=b-e}TotalSize_HH,HL,LH,LL.

ACK status (H_CHx{x=b-e}IntStat.TranACK bit) is issued to the firmware on receipt of an ACK for the interrupt OUT transaction. The FIFO is then updated, freeing the area by treating the data sent as already sent.

If an NAK is received for the interrupt OUT transaction, the FIFO is not updated, and the area is not freed. Selecting the corresponding channel again executes the same transaction.

If a STALL is received for the interrupt OUT transaction, H_CHx{x=b-e}Config_0.TranGo is automatically cleared, ending the transfer, and the condition code (H_CHx{x=b-e}ConditionCode) is set to "STALL." A ChangeCondition status (H_CHx{x=b-e}IntStat.ChangeCondition bit) is then issued to the firmware. The FIFO is not updated, and the area is not freed.

If a normal response is not received for the interrupt OUT transaction, the FIFO is not updated, and the area is not freed. The condition code (H_CHx{x=b-e}ConditionCode) is set to "RetryError," and a TranErr status (H_CHx{x=b-e}IntStat.TranErr bit) is issued to the firmware. Retry processing is performed, but H_CHx{x=b-e}Control.TranGo is automatically cleared to end the transfer if the error occurs three times in succession, after which a ChangeCondition status (H_CHx{x=b-e}IntStat.ChangeCondition bit) is issued to the firmware.

1.3.3.4 Isochronous OUT Transaction

The transfer type (H_CHx{x=b-e}Config_1.TranType) is set to “Isochronous” and transaction type (H_CHx{x=b-e}Config_1.TID) set to “OUT” for the CHx{x=b-e} basic setting register. The token-issuing interval (H_CHx{x=b-e}Interval_H,L) is set, other basic settings are set appropriately, and transfer execution (H_CHx{x=b-e}Config_0.TranGo) is set, subjecting the channel to USB transfer scheduling by hardware. When scheduling selects the corresponding channel, the token-issuing interval (H_CHx{x=b-e}Interval_H,L), valid FIFO data quantity and remaining frame time are determined, and an isochronous OUT transaction is executed.

The data packet size is the smaller of H_CHx{x=b-e}MaxPacketSize_H,L and H_CHx{x=b-e}TotalSize_HH,HL,LH,LL.

ACK status (H_CHx{x=b-e}IntStat.TranACK bit) is issued to the firmware once the isochronous OUT transaction ends. The FIFO is then updated, freeing the area by treating the data sent as already sent.

A transaction is not issued if the valid FIFO data quantity is less than the data packet data length. In this case, the FIFO is not updated, and the area is not freed. A TranErr status is issued to the firmware and the condition code (H_CHx{x=b-e}ConditionCode) is set to “BufferUnderrun.”

Figure 1-26 shows the procedure for an isochronous OUT transaction when completed. (a) The LSI issues an OUT token addressed to the OUT-direction endpoint existing at the destination node. (b) The LSI then sends a data packet no larger than the maximum packet size. The LSI automatically sets the corresponding register after sending the data packet and issues the status to the firmware.

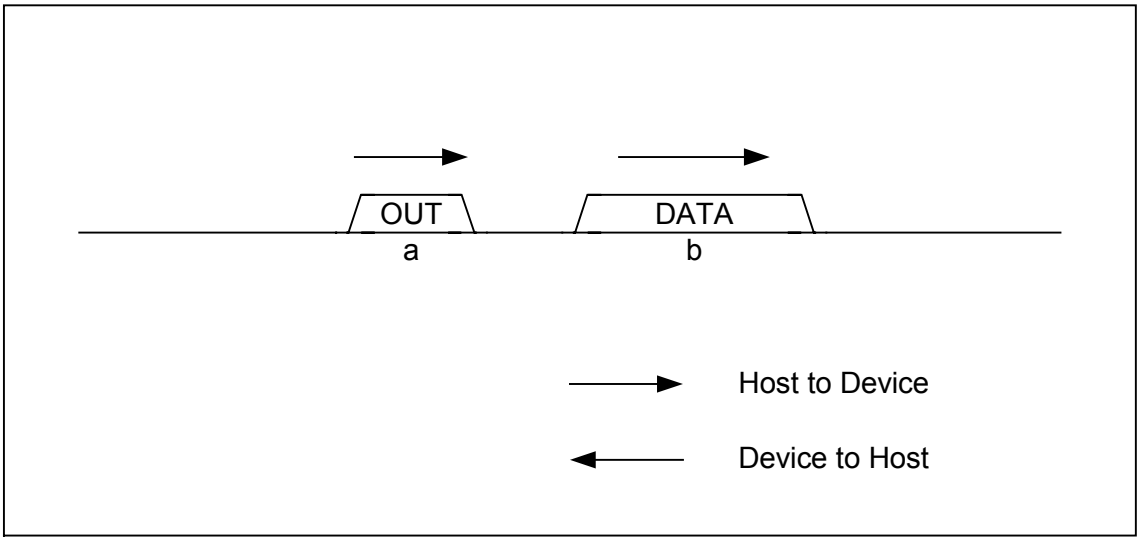


Figure 1-26 Isochronous OUT transaction

1.3.3.5 Bulk IN Transaction

The transfer type (H_CHx{x=b-e}Config_1.TranType) is set to “Bulk” and the transaction type (H_CHx{x=a-e}Config_1.TID) is set to “IN” for the CHx{x=a-e} basic setting register. The other basic settings are set appropriately, and transfer execution (H_CHx{x=a-e}Config_0.TranGo) is set, subjecting the channel to USB transfer scheduling by hardware. When scheduling selects the corresponding channel, the FIFO free space and remaining frame time is determined, and a bulk IN transaction is executed. Channel CHa is dedicated for bulk transfer, and so the basic setting registers do not include a transfer type.

The expected data length of the data packets to be received is the smaller of H_CHx{x=a-e}MaxPacketSize_H,L and H_CHx{x=a-e}TotalSize_HH,HL,LH,LL.

If all data is received normally in the bulk IN transaction, an ACK response is returned, and the transaction ends. An ACK status (H_CHx{x=a-e}IntStat.TranACK bit) is also issued to the firmware. The FIFO is then updated, reserving space by treating the data as already received.

If the data length received is shorter than the expected data length for the bulk IN transaction, H_CHx{x=a-e}Config_0.TranGo is automatically cleared, ending the transfer and returning an ACK response. The condition code (H_CHx{x=a-e}ConditionCode) is set to “DataUnderrun.” A ChangeCondition status (H_CHx{x=a-e}IntStat.ChangeCondition bit) is then issued to the firmware. The FIFO is updated and the area reserved by treating the data as already received.

If an NAK is received for the bulk IN transaction, the status is not issued, and the FIFO is not updated.

If a STALL is received for the bulk IN transaction, H_CHx{x=a-e}Config_0.TranGo is automatically cleared, ending the transfer, and the condition code (H_CHx{x=a-e}ConditionCode) is set to “STALL.” A ChangeCondition status (H_CHx{x=a-e}IntStat.ChangeCondition bit) is then issued to the firmware. The FIFO is not updated.

If the data length received is longer than the expected data length for the bulk IN transaction, H_CHx{x=a-e}Config_0.TranGo is automatically cleared, ending the transfer. No response is returned. The condition code (H_CHx{x=a-e}ConditionCode) is set to “DataOverrun” and a ChangeCondition status (H_CHx{x=a-e}IntStat.ChangeCondition bit) issued to the firmware. The FIFO is not updated.

If a toggle mismatch occurs for the bulk IN transaction, an ACK response is returned. The condition code (H_CHx{x=a-e}ConditionCode) is set to “RetryError,” and a TranErr status (H_CHx{x=a-e}IntStat.TranErr bit) is issued to the firmware. The FIFO is not updated.

If a timeout error, CRC error, bit stuffing error, or PID error (including unforeseen PID) occurs in the bulk IN transaction, no response is returned. The condition code (H_CHx{x=a-e}ConditionCode) is set to “RetryError,” and a TranErr status (H_CHx{x=a-e}IntStat.TranErr bit) is issued to the firmware. The FIFO is not updated.

If an error occurs for which the condition code (H_CHx{x=a-e}ConditionCode) is set to "RetryError," retry processing is performed. If the error occurs three times in succession, H_CHx{x=a-e}Control.TranGo is automatically cleared to end the transfer, and a ChangeCondition status (H_CHx{x=a-e}IntStat.ChangeCondition bit) is issued to the firmware.

Figure 1-27 shows the procedure for a bulk IN transaction when completed. (a) The LSI issues an IN token addressed to the IN-direction endpoint existing at the destination node. (b) The endpoint sends a data packet no larger than the maximum packet size if a response to the IN transaction is possible. (c) The LSI returns an ACK response, automatically sets the corresponding register, and issues the status to the firmware.

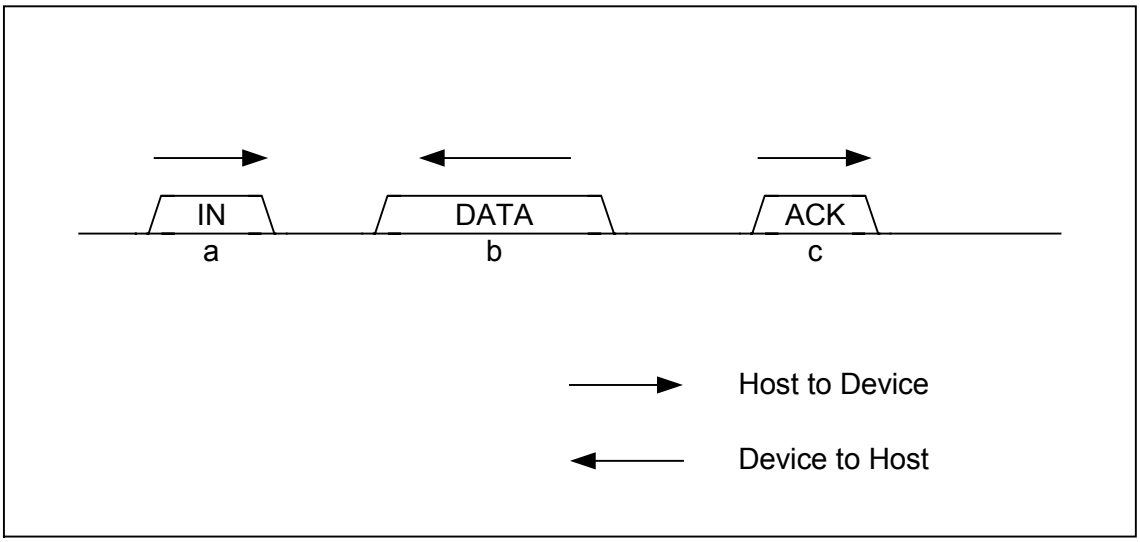


Figure 1-27 IN transaction

1.3.3.6 Interrupt IN Transaction

The transfer type (H_CHx{x=b-e}Config_1.TranType) is set to “Interrupt,” and the transaction type (H_CHx{x=b-e}Config_1.TID) is set to “IN” for the CHx{x=b-e} basic setting register. The token-issuing interval (H_CHx{x=b-e}Interval_H,L) is set, other basic settings are set appropriately, and transfer execution (H_CHx{x=b-e}Config_0.TranGo) is set, subjecting the channel to USB transfer scheduling by hardware. When scheduling selects the corresponding channel, the token-issuing interval (H_CHx{x=b-e}Interval_H,L), FIFO free space and remaining frame time are determined, and an interrupt IN transaction is executed.

The expected data length of the data packets to be received is the smaller of H_CHx{x=b-e}MaxPacketSize_H,L and H_CHx{x=b-e}TotalSize_HH,HL,LH,LL.

If all data is received normally in the interrupt IN transaction, an ACK response is returned, and the transaction ends. An ACK status (H_CHx{x=b-e}IntStat.TranACK bit) is also issued to the firmware. The FIFO is then updated, reserving space by treating the data as already received.

If the data length received is shorter than the expected data length for the interrupt IN transaction, H_CHx{x=b-e}Config_0.TranGo is automatically cleared, ending the transfer and returning an ACK response. The condition code (H_CHx{x=b-e}ConditionCode) is set to “DataUnderrun.” A ChangeCondition status (H_CHx{x=b-e}IntStat.ChangeCondition bit) is then issued to the firmware. The FIFO is updated, and the area is reserved by treating the data as already received.

If an NAK is received for the interrupt IN transaction, the status is not issued, and the FIFO is not updated. The next transaction is performed in the subsequent cycle.

If a STALL is received for the interrupt IN transaction, H_CHx{x=b-e}Config_0.TranGo is automatically cleared, ending the transfer, and the condition code (H_CHx{x=b-e}ConditionCode) is set to “STALL.” A ChangeCondition status (H_CHx{x=b-e}IntStat.ChangeCondition bit) is then issued to the firmware. The FIFO is not updated.

If the data length received is longer than the expected data length for the bulk IN transaction, H_CHx{x=b-e}Config_0.TranGo is automatically cleared, ending the transfer. No response is returned. The condition code (H_CHx{x=b-e}ConditionCode) is set to “DataOverrun.” A ChangeCondition status (H_CHx{x=b-e}IntStat.ChangeCondition bit) is then issued to the firmware. The FIFO is not updated.

If a toggle mismatch occurs for the interrupt IN transaction, an ACK response is returned. The condition code (H_CHx{x=b-e}ConditionCode) is set to “RetryError,” and a TranErr status (H_CHx{x=b-e}IntStat.TranErr bit) is issued to the firmware. The FIFO is not updated.

If a timeout error, CRC error, bit stuffing error, or PID error (including unforeseen PID) occurs in the interrupt IN transaction, no response is returned. The condition code (H_CHx{x=b-e}ConditionCode) is set to “RetryError,” and a TranErr status (H_CHx{x=b-e}IntStat.TranErr bit) is issued to the firmware. The FIFO is not updated.

If an error occurs for which the condition code (H_CHx{x=b-e}ConditionCode) is set to “RetryError,” retry processing is performed at the subsequent cycle. If the error occurs three times in succession, H_CHx{x=b-e}Control.TranGo is automatically cleared to end the transfer, and a ChangeCondition status (H_CHx{x=b-e}IntStat.ChangeCondition bit) is issued to the firmware.

1.3.3.7 Isochronous IN Transaction

The transfer type (H_CHx{x=b-e}Config_1.TranType) is set to “Isochronous” and the transaction type (H_CHx{x=b-e}Config_1.TID) is set to “IN” for the CHx{x=b-e} basic setting register. The token-issuing interval (H_CHx{x=b-e}Interval_H,L) is set, other basic settings are set appropriately, and transfer execution (H_CHx{x=b-e}Config_0.TranGo) is set, subjecting the channel to USB transfer scheduling by hardware. When scheduling selects the corresponding channel, the token-issuing interval (H_CHx{x=b-e}Interval_H,L), FIFO free space and remaining frame time are determined, and an isochronous IN transaction is executed.

The expected data length of the data packets to be received is the smaller of H_CHx{x=b-e}MaxPacketSize_H,L and H_CHx{x=b-e}TotalSize_HH,HL,LH,LL.

If all data is received normally in the isochronous IN transaction, the transaction ends. An ACK status (H_CHx{x=b-e}IntStat.TranACK bit) is issued to the firmware. The FIFO is then updated, reserving space by treating the data as already received.

If the data length received is shorter than the expected data length for the isochronous IN transaction, an ACK status (H_CHx{x=b-e}IntStat.TranACK bit) and ChangeCondition status (H_CHx{x=b-e}IntStat.ChangeCondition bit) are issued to the firmware. The condition code (H_CHx{x=b-e}ConditionCode) is set to “DataUnderrun.” The FIFO is updated, and the area is reserved by treating the data as already received.

If the data length received is longer than the expected data length for the isochronous IN transaction, a ChangeCondition status (H_CHx{x=b-e}IntStat.ChangeCondition bit) is issued to the firmware. The condition code (H_CHx{x=b-e}ConditionCode) is set to “DataOverrun,” and the FIFO is not updated.

If a timeout error, CRC error, bit stuffing error, or PID error (including unforeseen PID) occurs in the isochronous IN transaction, a TranErr status (H_CHx{x=b-e}IntStat.TranErr bit) is issued to the firmware. The condition code (H_CHx{x=b-e}ConditionCode) is set to “RetryError,” and the FIFO is not updated.

Transactions are not issued if the FIFO free space is less than the value indicated by H_CHx{x=b-e}MaxPacketSize_H,L. A TranErr status is issued to the firmware, and the condition code (H_CHx{x=b-e}ConditionCode) is set to “BufferOverrun.”

Figure 1-28 shows the procedure for an isochronous IN transaction when completed. (a) The LSI issues an IN token addressed to the IN-direction endpoint existing at the destination node. (b) The endpoint sends a data packet no larger than the maximum packet size if a response to this IN transaction is possible. The LSI automatically sets the corresponding register after receiving the data packet and issues the status to the firmware.

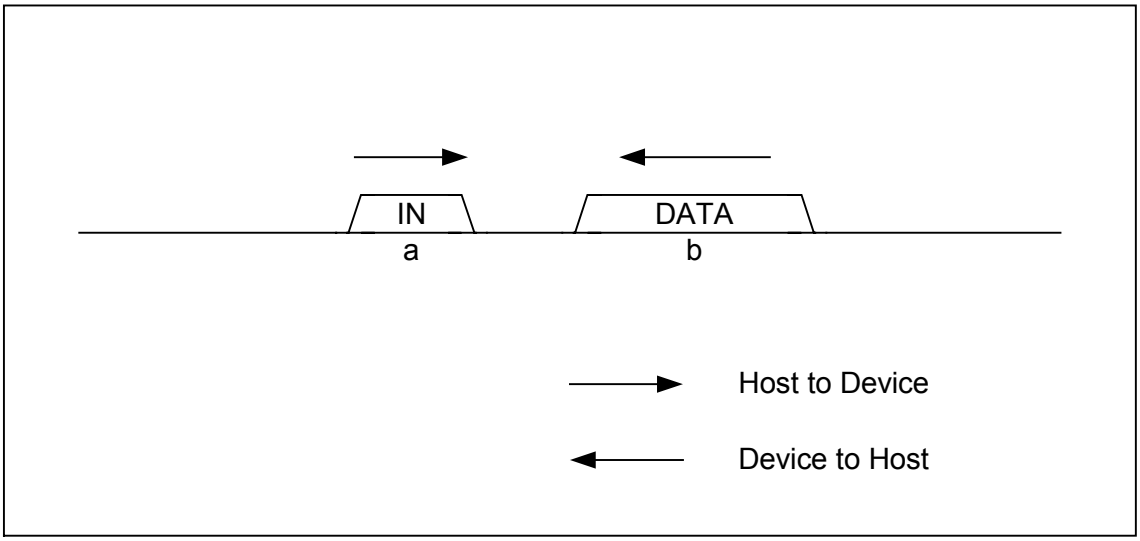


Figure 1-28 Isochronous IN transaction

1.3.3.8 PING Transaction

The LSI switches to allow execution of PING transactions under the following conditions for HS operations with channels performing bulk OUT or control OUT transactions. When the channel is in PING transaction execution mode, a PING transaction is executed when selected by scheduling.

If a NYET, NAK, or no response is received for the OUT transaction, the channel switches to a mode allowing execution of PING transactions.

If an NAK is received for the PING transaction, the channel continues to enable PING transaction execution. No status is issued to the firmware.

If an ACK is received for the PING transaction, the channel switches back from the mode enabling PING transaction execution to the mode enabling OUT transaction execution. No status is issued to the firmware.

If a STALL is received for the PING transaction, H_CHx{x=0,a-e}Config_0.TranGo is automatically cleared to end the transfer, and the condition code (H_CHx{x=0,a-e}ConditionCode) is set to "STALL." A ChangeCondition status (H_CHx{x=0,a-e}IntStat.ChangeCondition bit) is then issued to the firmware.

If a normal response is not received for the PING transaction, the condition code (H_CHx{x=0,a-e}ConditionCode) is set to "RetryError." A TranErr status (H_CHx{x=0,a-e}IntStat.TranErr bit) is issued to the firmware. Retry processing is performed, but if the error occurs three times in succession, H_CHx{x=0,a-e}Control.TranGo is automatically cleared to end the transfer, and a ChangeCondition status (H_CHx{x=0,a-e}IntStat.ChangeCondition bit) is issued to the firmware.

The FIFO is not updated for PING transactions.

Figure 1-29 shows the procedure for an ACK response to a PING transaction. (a) The LSI issues a PING token addressed to the OUT-direction endpoint existing in the node. (b) The device returns an ACK response to the PING transaction if space equivalent to the maximum packet size is found at the endpoint.

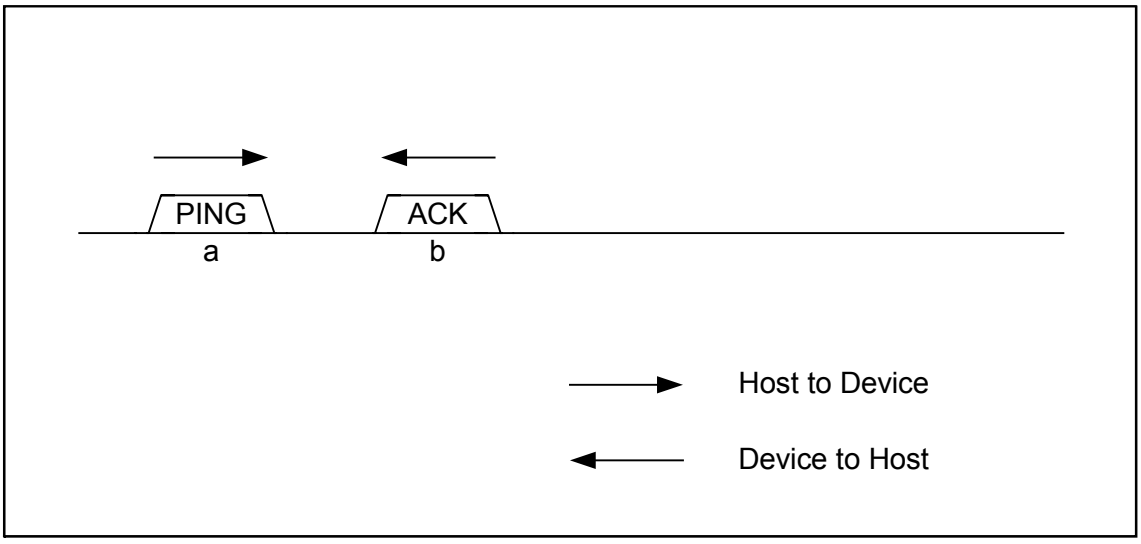


Figure 1-29 PING transaction

1.3.3.9 Low-speed (LS) Transaction

Transfer to and from LS devices uses control and interrupt transfers.

The host operates in LS mode when an LS device is connected to the downstream port. The transfer speed (H_CHx{x=0,a-e}Config_0.SpeedMode) is set to "LS" for the channel used, executing the transaction in LS bit time.

If a full-speed (FS) hub is connected to the downstream port and an LS device is connected to the hub's downstream port, the host operates in FS mode. Setting the transfer speed (H_CHx{x=0,a-e}Config_0.SpeedMode) to LS for the channel used sends downstream packets to the corresponding endpoint with preamble attached at the start of every such packet. The preamble is sent using FS bit time, while the subsequent downstream packets are sent using LS bit time.

Figure 1-30 shows the procedure for an interrupt OUT transaction when completed. (a) The LSI issues an OUT token with preamble attached at the start addressed to the OUT-direction endpoint existing at the destination node. (b) The LSI then sends a data packet no larger than the maximum packet size with preamble attached at the start. (c) The LSI automatically sets the corresponding register after receiving ACK, and issues the status to the firmware.

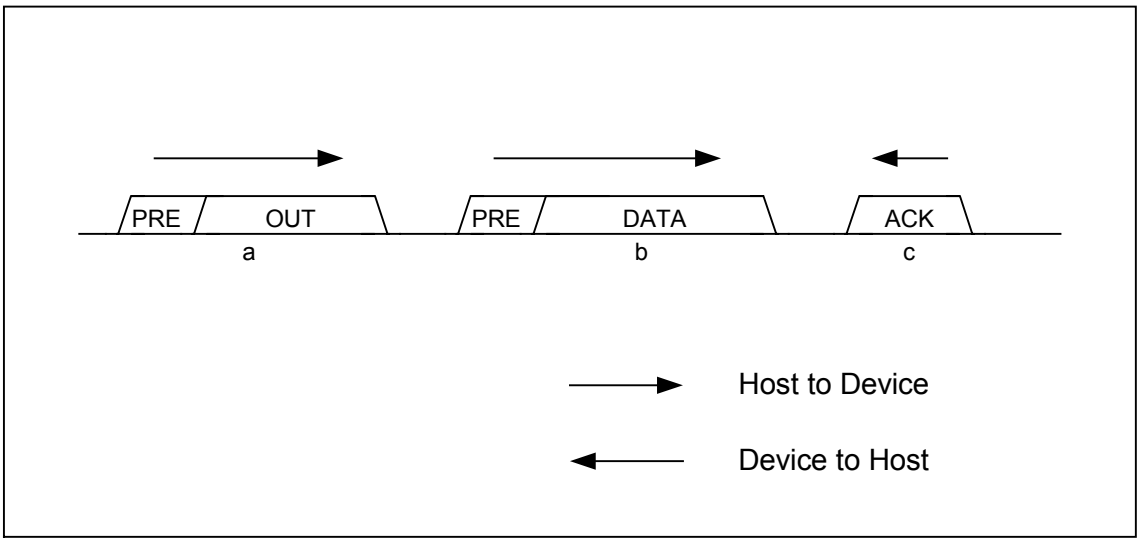


Figure 1-30 OUT transaction with preamble attached

Figure 1-31 shows the procedure for an interrupt IN transaction when completed. (a) The LSI issues an IN token with preamble attached at the start addressed to the IN-direction endpoint existing at the destination node. (b) The device sends a data packet no larger than the maximum packet size. The LSI writes this data to the corresponding channel FIFO. (c) The LSI attaches a preamble with an ACK response at the start after receiving the data, then automatically sets the corresponding register and issues the status to the firmware.

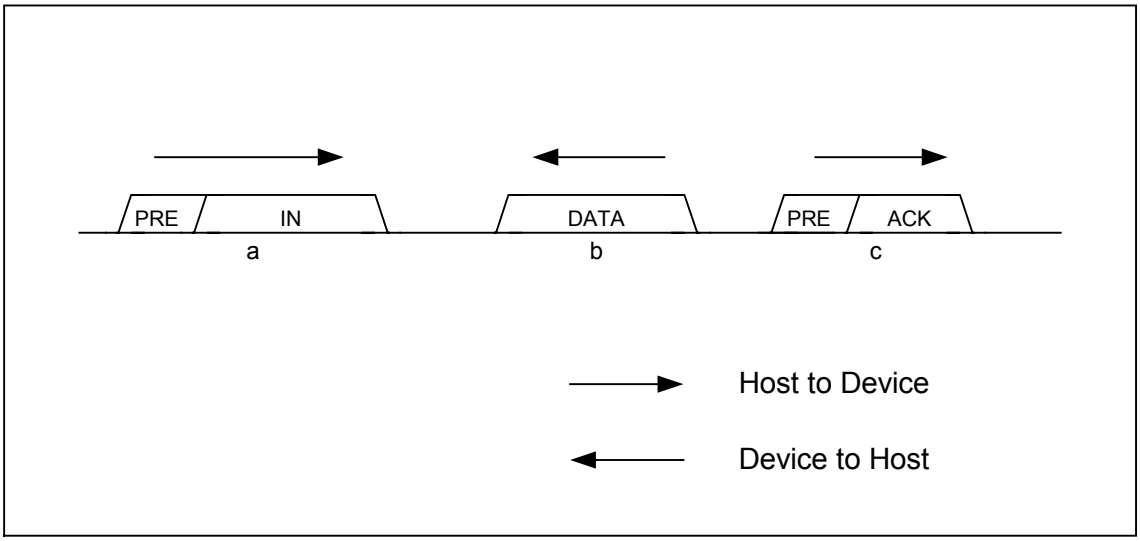


Figure 1-31 IN transaction with preamble attached

1.3.3.10 Split Transaction

If a high-speed (HS) hub is connected to the downstream port and an FS or LS device is connected to the hub's downstream port, the host operates in HS mode. Setting the transfer speed (H_CHx{x=0,a-e}Config_0.SpeedMode) to FS or LS for the channel used executes the transaction with the hub to the corresponding endpoint using a split transaction.

The hub address (H_CHx{x=0,a-e}HubAdrs.HubAdrs) and port number (H_CHx{x=0,a-e}HubAdrs.Port) is set appropriately for the corresponding channel.

The complete split transaction sequence from the start split transaction is controlled by hardware for the split transaction. This means the firmware need not recognize each transaction of the split transaction.

If the final complete split transaction is completed normally for the complete split transaction sequence from the start split transaction for control transfer, bulk transfer, interrupt transfer, or IN-direction isochronous transfer, ACK status (H_CHx{x=0,a-e}IntStat.TranACK bit) is issued to the firmware, and the FIFO is updated.

If the final start split transaction is completed normally for OUT-direction isochronous transfer, ACK status (H_CHx{x=b-e}IntStat.TranACK bit) is issued to the firmware, and the FIFO is updated.

If the start split transaction or complete split transaction is completed normally for transfers other than those described above, the status is not issued to the firmware.

If an error occurs in individual split transactions for the complete split transaction sequence from the start split transaction, the condition code (H_CHx{x=0,a-e}ConditionCode) is set to "RetryError," and a TranErr status (H_CHx{x=0,a-e}IntStat.TranErr bit) is issued to the firmware. The FIFO is not updated. Retry processing is performed. If the error occurs three times in succession for control, bulk, or interrupt transfers, H_CHx{x=0,a-e}Control.TranGo is automatically cleared to end the transfer, and a ChangeCondition status (H_CHx{x=0,a-e}IntStat.ChangeCondition bit) is issued to the firmware.

1.3.4 Control Transfer

Control transfer is either controlled as individual transactions for each stage or is performed automatically using the control transfer support function described later ("1.3.4.3 Control Transfer Support Function").

Figure 1-32 shows the procedure for controlling control transfers. The firmware sets the SETUP, DATA, and STATUS stages appropriately, and control transfer is executed by hardware.

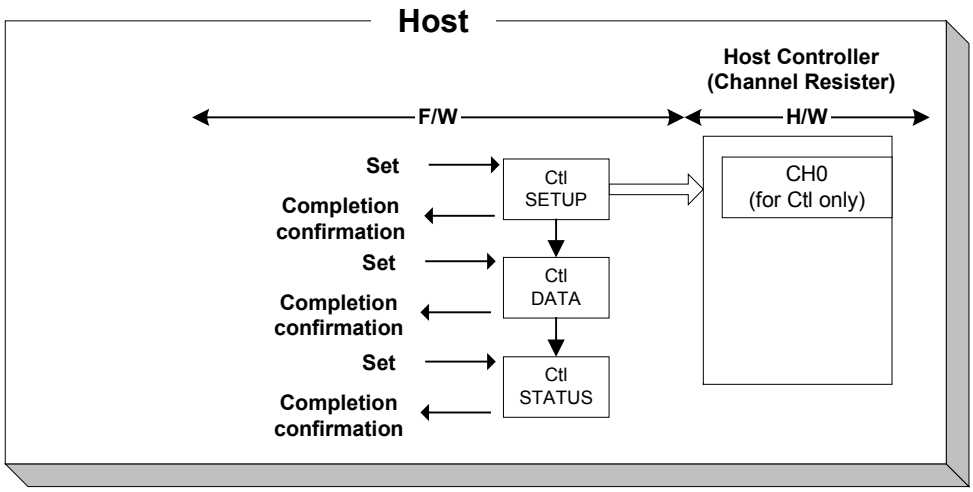


Figure 1-32 Control transfer control

Figure 1-33 shows the procedure for control transfer when the data stage is in the OUT direction. (a) The host starts the control transfer using the SETUP transaction. (b) The host issues an OUT transaction and performs the data stage. (c) The host issues an IN transaction and performs the status stage.

Control transfers without a data stage are performed without the data stage shown in this example.

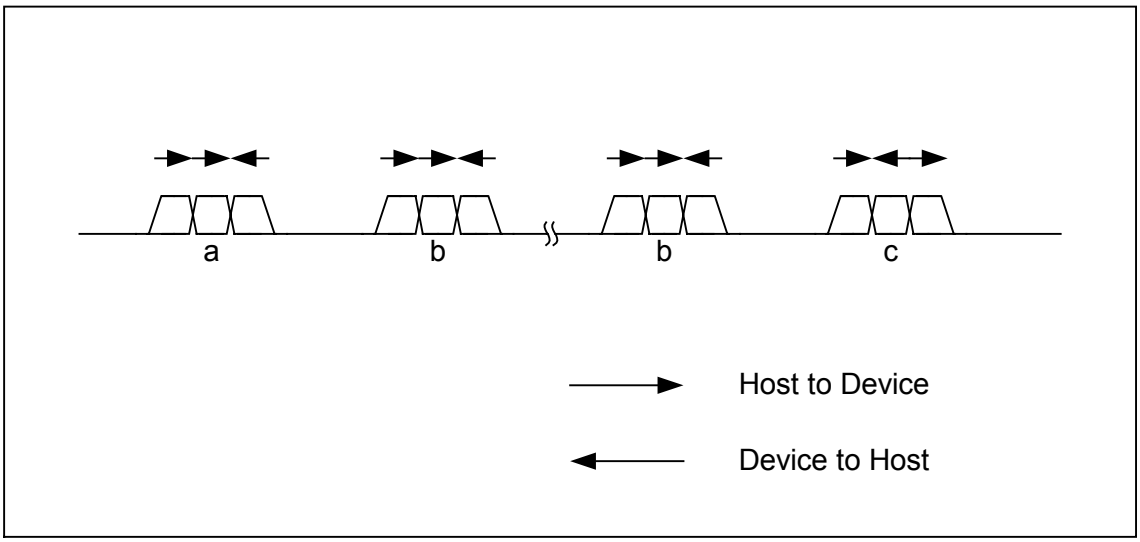


Figure 1-33 Control transfer with data stage in OUT direction

Figure 1-34 shows the procedure for control transfer when the data stage is in the IN direction. (a) The host starts the control transfer using the SETUP transaction. (b) The host issues an IN transaction and performs the data stage. (c) The host issues an OUT transaction and performs the status stage.

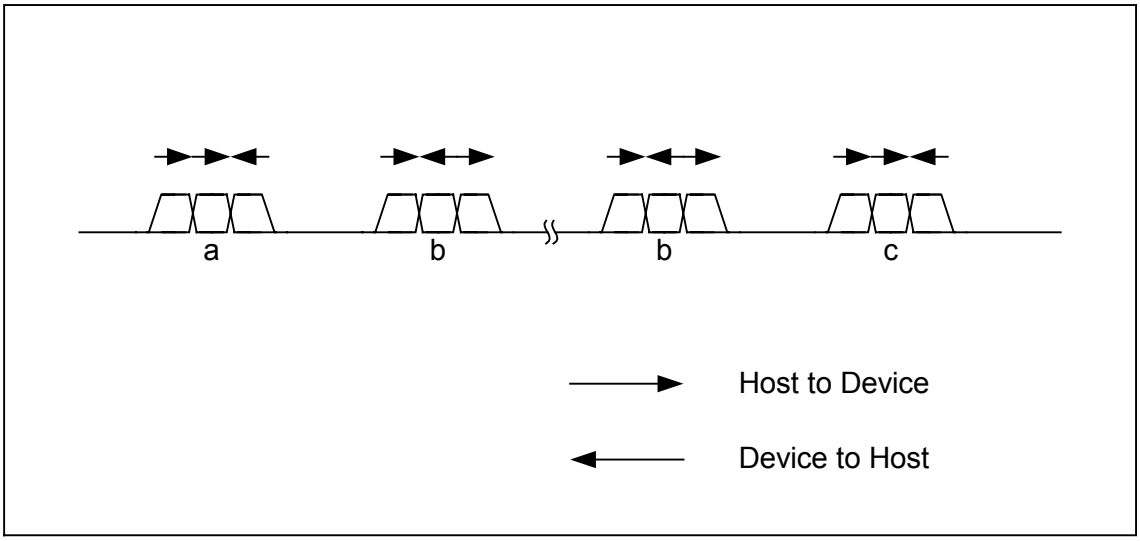


Figure 1-34 Control transfer with data stage in the IN direction

1.3.4.1 Setup Stage

The setup stage is performed using the SETUP transaction. Refer to "1.3.3.1 SETUP Transaction" for detailed information on SETUP transactions.

1.3.4.2 Data Stage/Status Stage

Proceed to the next stage once the setup stage is complete.

If the stage is in the IN direction, set the transaction type (H_CH0Config_1.TID) to "IN," set the other basic setting registers appropriately, and execute the transaction.

If the stage is in the OUT direction, set the transaction type (H_CH0Config_1.TID) to "OUT," set the other basic setting registers appropriately, and execute the transaction.

For the status stage, the IRP data quantity (H_CH0TotalSize_H,L) should be set to "0x0" before executing the transaction.

1.3.4.3 Control Transfer Support Function

This LSI includes a function for stage managing a series of control transfers automatically. This function eliminates the need to manage each stage as individual transactions via firmware.

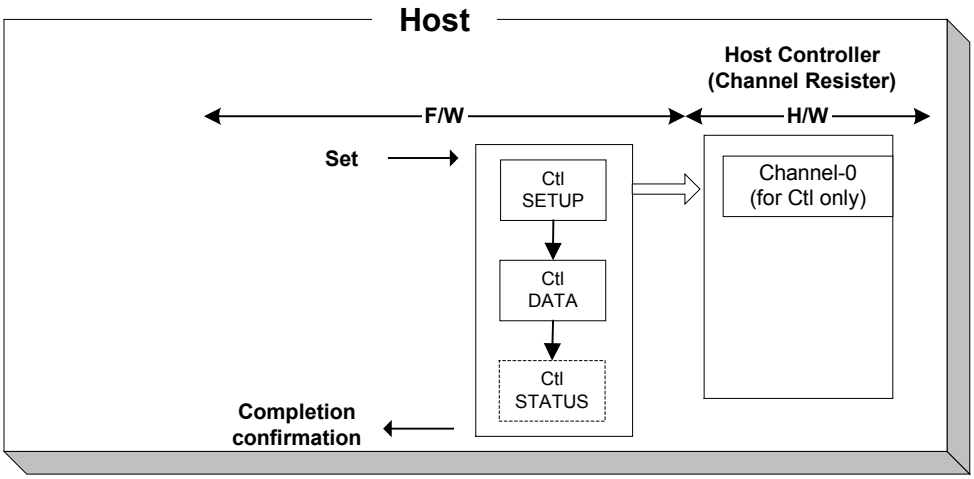


Figure 1-35 Control transfer support function control

The control transfer support function is enabled for channel CH0. When using this function, select AREA0 as the FIFO area to be joined to channel CH0. Shown below is a control transfer using this function. The firmware performs the processing in steps (1) to (4) and (7).

(1) Appropriately set the following channel CH0 basic setting registers.

Transfer speed (H_CH0Config_0.SpeedMode), maximum packet size (H_CH0MaxPktSize), USB address (H_CH0FuncAdrs.FuncAdrs), endpoint number (H_CH0FuncAdrs.EP_Number), FIFO area (AREA0StartAdrs_H,L,AREA0EndAdrs_H,L), FIFO area join (AREA0Join_1.JoinEP0CH0)

(2) Write the setup data (8 bytes) to the setup registers (H_CH0SETUP_0 to 7).

(3) If the data stage is in the OUT direction, write the data to be sent to the FIFO area joined to CH0. If the data stage is in the IN direction, clear the FIFO area joined to CH0.

(4) Set the control transfer support execute (H_CTL_SupportControl.CTL_SupportGo). The control transfer stage (H_CTL_SupportControl.CTL_SupportState) value is written to the H_CTL_SupportControl register here as "Idle(00b)."

(5) Execute the SETUP transaction using the SETUP register data (8 bytes) (SETUP stage).

(6) Execute the data stage based on the SETUP data.

- If the SETUP data bmRequestType Bit7 is 0, the data in the FIFO area joined to CH0 is sent using an OUT transaction (OUT-direction data stage). The OUT-direction data stage ends when the data quantity indicated by the SETUP data wLength is sent by the OUT transaction.

- If the SETUP data bmRequestType Bit7 is 1, an IN transaction is issued, and the data received is written to the FIFO area joined to CH0 (IN-direction data stage). The IN-direction data stage ends when the data quantity indicated by the SETUP data wLength is received by the IN transaction. The IN-direction data stage also ends when a short packet is received by the IN transaction.
- The data stage is not executed if the SETUP data wLength value is 0x000.

(7) If the FIFO area joined to CH0 is smaller than the value indicated by SETUP data wLength, the firmware must divide the data stage data for processing.

- If the data stage is in the OUT direction, transactions are no longer issued once all data has been sent to the FIFO area joined to CH0. The firmware must write the remaining data to be sent to the FIFO in sequence while checking for FIFO free space.
- If the data stage is in the IN direction, transactions are no longer issued if no more free space is found in the FIFO area joined to CH0. The firmware must read the remaining data received from the FIFO in sequence while checking the valid FIFO data quantity to create FIFO free space.

(8) Execute the status stage based on the SETUP data.

- An IN transaction is issued once the OUT-direction data stage ends (IN-direction status stage).
- A zero-length packet OUT transaction is issued once the IN-direction data stage ends, all received data in the FIFO area joined to CH0 has been read, and the FIFO emptied (OUT-direction status stage).

(9) If the control transfer ends normally, control transfer support execute (H_CTL_SupportControl.CTL_SupportGo) is automatically cleared, and a control transfer completion status (H_CH0IntStat.CTL_SupportCmp) is issued.

(10) If a transaction error occurs during a control transfer, control transfer support execution (CTL_SupportControl.CTL_SupportGo) is automatically cleared, aborting the control transfer and issuing control transfer stop status (H_CH0IntStat.CTL_SupportStop). The control transfer stage (H_CTL_SupportControl.CTL_SupportState) indicates the stage in which the error occurred. The condition code (H_CH0ConditionCode) is set to an appropriate value, and a ChangeCondition status (H_CH0IntStat.ChangeCondition bit) is issued.

If control transfer is aborted, control transfer support execution (H_CTL_SupportControl.CTL_SupportGo) is cleared. The status is issued once the control transfer abort processing is completed.

If the control transfer is completed up to the status stage and abort processing is completed, control transfer completion status (H_CH0IntStat.CTL_SupportCmp) is issued.

However, if abort processing is completed but control transfer is not completed up to the status stage, control transfer stop status (H_CH0IntStat.CTL_SupportStop) is issued.

The stage for which control transfer was aborted is indicated by the control transfer stage (H_CTL_SupportControl.CTL_SupportState).

If control transfer resumes from the aborted stage, the control transfer stage (H_CTL_SupportControl.CTL_SupportState) is set in the stage resumed (i.e. settings are retained in the stage aborted), and control transfer support execution (H_CTL_SupportControl.CTL_SupportGo) is set.

However, if a new control transfer is performed, the control transfer stage (H_CTL_SupportControl.CTL_SupportState) is set to "Idle(00b)," and control transfer support execution (H_CTL_SupportControl.CTL_SupportGo) is set.

The transfer execution bit (H_CH0Config_0.TranGo), toggle sequence bit (H_CH0Config_0.Toggle), transaction type (H_CH0Config_1.TID), and IRP data quantity (H_CH0TotalSize_H,L) are set or updated by hardware during execution of the control transfer support function, so data must not be written to these bits.

Refer to "1.3.3 Transactions" for detailed information on transaction errors.

Table 1-25 shows the settings and status for the control transfer support function.

Table 1-25 Control transfer support function control items and status

Item	Register/bit	Description
Control transfer support execution	H_CTL_SupportControl.CTL_SupportGo	Manages the control transfer stage automatically. Refer to "1.3.4.3 Control Transfer Support Function" for particulars.
Control transfer stage	H_CTL_SupportControl.CTL_SupportState	Indicates the stage being executed for the control transfer support function. Indicates the stage in which the error occurred if control transfer was aborted due to an error.
Control transfer execution results	H_CH0IntStat.CTL_SupportCmp H_CH0IntStat.CTL_SupportStop	Indicates the control transfer execution results using the control transfer support function.
Transaction status	H_CH0IntStat.TotalSizeCmp, H_CH0IntStat.TranACK, H_CH0IntStat.TranErr, H_CH0IntStat.ChangeCondition	Indicates the transaction results.
Transaction condition code	H_CH0ConditionCode.ConditionCode	Indicates transaction result details.

1.3.5 Bulk Transfer/Interrupt Transfer/Isochronous Transfer

Bulk transfer for CHa and bulk transfer, interrupt transfer or isochronous transfer for CHb, CHc, CHd, or CHe can be controlled as a sequence of individual transactions (see “1.3.3 Transactions”) even as data flows (see “1.3.6 Data Flow”).

1.3.6 Data Flow

This section describes general data flow control for OUT and IN transfers.

1.3.6.1 OUT Transfer

Set the OUT transfer total data quantity to H_CH0TotalSize_H,L or H_CHx{x=a-e}TotalSize_HH,HL,LH,LL and write the data to be sent by the OUT transfer to the FIFO area joined to each channel. Data can be written to the FIFO using CPU interface register writing, CPU interface DMA writing, or read transfer from the IDE.

To write data to the FIFO using CPU interface register writing, select one of the FIFO areas joined to the channel by the AREAn{n=0-5}Join_0.JoinCPU_Wr bit. Data can be written to the selected FIFO area by the FIFO_Wr_0,1 or FIFO_ByteWr register and is sent as data packets in the order written. The FIFO free space can be checked using the FIFO_WrRemain_H,L register. Data cannot be written to the FIFO if it is full. Always check the free space using the FIFO_WrRemain_H,L register and ensure that the data volume written does not exceed the free space.

To write data to the FIFO using IDE interface read transfer, select one channel using the AREAn{n=0-5}Join_0.JoinIDE bit and set the IDE_Control.Dir bit to 0. Data can be written to the selected channel FIFO by executing IDE transfers using IDE_Control.IDE_Go. The data is sent as data packets in the order read from the IDE. If the FIFO becomes full, the read transfer is paused automatically by the IDE interface to control flow.

The data packet size sent using OUT transaction is the smaller of H_CH0TotalSize_H,L and H_CH0MaxPktSize for CH0, and the smaller of H_CHx{x=a-e}TotalSize_HH,HL,LH,LL and H_CHx{x=a-e}MaxPktSize_H,L for channel CHx{x=a-e}.

If the data in the FIFO exceeds the data packet data size, an OUT transaction is executed, and the data is sent. H_CH0TotalSize_H,L or H_CHx{x=a-e}TotalSize_HH,HL,LH,LL is reduced by the amount of the data size sent. When TotalSize reaches zero, H_CHx{x=0,a-e}Config_0.TranGo is automatically cleared to end the transfer, and TotalSizeCmp status (H_CHx{x=0,a-e}IntStat.TotalSizeCmp bit) is issued to the firmware.

OUT transfer can be controlled in this way without controlling individual transactions.

1.3.6.2 IN Transfer

Set the IN transfer total data quantity to H_CH0TotalSize_H,L or H_CHx{x=a-e}TotalSize_HH,HL,LH,LL.

The expected size for the data packets to be received in the IN transaction is the smaller of H_CH0TotalSize_H,L and H_CH0MaxPktSize for CH0, and the smaller of H_CHx{x=a-e}TotalSize_HH,HL,LH,LL and H_CHx{x=a-e}MaxPktSize_H,L for channel CHx{x=a-e}. If FIFO free space exceeds maximum packet size, the IN transaction is executed, and data is received. H_CH0TotalSize_H,L or H_CHx{x=0,a-e}TotalSize_HH,HL,LH,LL is reduced by the amount of the data size received. When TotalSize reaches zero, H_CHx{x=0,a-e}Config_0.TranGo is automatically cleared to end the transfer, and TotalSizeCmp status (H_CHx{x=0,a-e}IntStat.TotalSizeCmp bit) is issued to the firmware.

If the received data size exceeds the expected data length, the condition code (H_CHx{x=0,a-e}ConditionCode) is set to "DataOverrun." A ChangeCondition status (H_CHx{x=0,a-e}IntStat.ChangeCondition bit) is issued to the firmware. The FIFO is not updated. For control, bulk, or interrupt transfers, H_CHx{x=0,a-e}Config_0.TranGo is automatically cleared to end the transfer for that channel.

If the received data size is smaller than the expected data packet size, the condition code (H_CHx{x=0,a-e}ConditionCode) is set to "DataUnderrun," and ChangeCondition status (H_CHx{x=0,a-e}IntStat.ChangeCondition bit) is issued to the firmware. The FIFO is updated and space is reserved by treating the data as already received. For control, bulk, or interrupt transfers, H_CHx{x=0,a-e}Config_0.TranGo is automatically cleared to end the transfer for that channel.

IN transfer can be performed in this way without need for firmware control of individual transactions.

Data received by IN transfer is written to the FIFO area joined to each channel. FIFO data can be read using CPU interface register reading, CPU interface DMA reading, or write transfer to the IDE.

To read FIFO data using CPU interface register reading, select one FIFO area joined to the corresponding channel by the AREAn{n=0-5}Join_0.JoinCPU_Rd bit. The selected FIFO area can be read in the order received using the FIFO_Rd_0,1 or FIFO_ByteRd register. The amount of FIFO data that can be read can be checked using the FIFO_RdRemain_H,L register. Data cannot be read from an empty FIFO, and so the data quantity must always be checked using the FIFO_RdRemain_H,L register to ensure that reading does not exceed that quantity.

To read FIFO data using CPU interface DMA reading, select one FIFO area for each DMA channel using the AREAn{n=0-5}Join_1.JoinDMAx{x=0,1} bit, and set the DMAx{x=0,1}_Control.Dir bit to 1. The selected FIFO area can be read in the order received by executing a DMA sequence for the CPU interface. The amount of FIFO data remaining can be checked by the DMAx{x=0,1}_Remain_H,L register. If the FIFO becomes empty, the CPU interface automatically pauses DMA to control the flow.

To read FIFO data using IDE interface write transfer, select one FIFO area using the AREAn{n=0-5}Join_1.IDE bit, and set the IDE_Control.Dir bit to 1. The FIFO for the selected channel can be read in the order received by executing IDE transfer using IDE_Control.IDE_Go. If the FIFO becomes empty, the IDE interface automatically pauses the write transfer to control the flow.

1.3.7 Zero Length Packet Automatic Issuing Function

Setting the H_CHx{x=a-e}Config_1.AutoZeroLen bit for the OUT transfer channel enables the function that automatically issues zero-length packets.

Table 1-26 shows the settings for the zero-length packet automatic issuing function.

Table 1-26 Control settings for the zero-length packet automatic issuing function

Item	Register/bit	Description
Zero-length packet automatic issue	H_CHx{x=a-e}Config_1.AutoZeroLen	Enables the zero-length packet automatic issuing function. This bit is valid only for OUT transfers.

1.3.7.1 Bulk/Interrupt OUT Transfer Zero Length Packet Automatic Issuing Function

Even when the data size transfer set by the H_CHx{x=a-e}TotalSize_HH,HL,LH,LL register ends exactly as the max packet size for a channel executing bulk/interrupt OUT transfer, H_CHx{x=a-e}Config_0.TranGo is not automatically cleared, and the transfer continues. When this channel is rescheduled, the OUT transaction is executed using a zero-length packet. When this transaction ends normally, H_CHx{x=a-e}Config_0.TranGo is automatically cleared to end the transfer, and TotalSizeCmp status (H_CHx{x=a-e}IntStat.TotalSizeCmp) is issued to the firmware.

1.3.8 Bulk-only Support Function

This LSI provides a function for automatically managing USB Mass Storage Class (BulkOnly Transport Protocol) command transport (CBW), data transport, and status transport (CSW). This function eliminates the need for individual control of each transport by the firmware. Figure 1-36 illustrates the typical configuration when using the bulk-only support function. Figure 1-37 illustrates the configuration when not using the function and controlling transactions as individual transport.

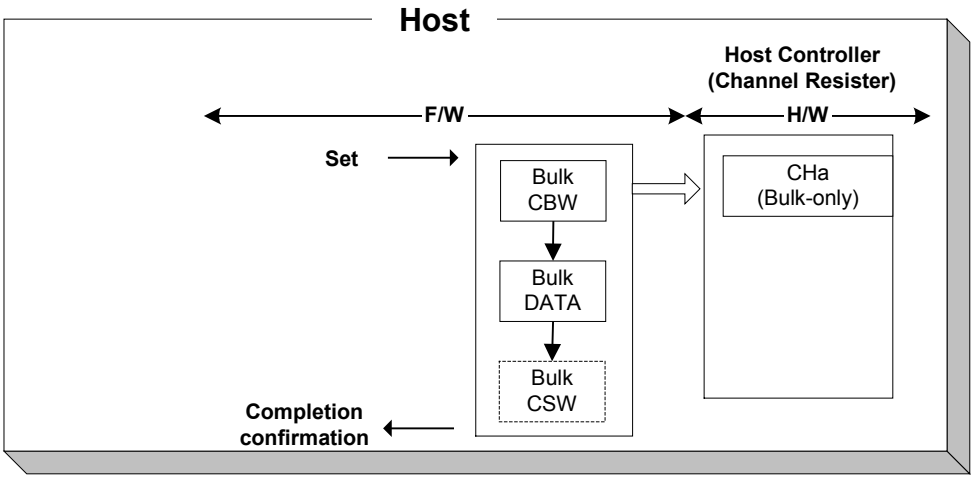


Figure 1-36 Bulk-only support function control

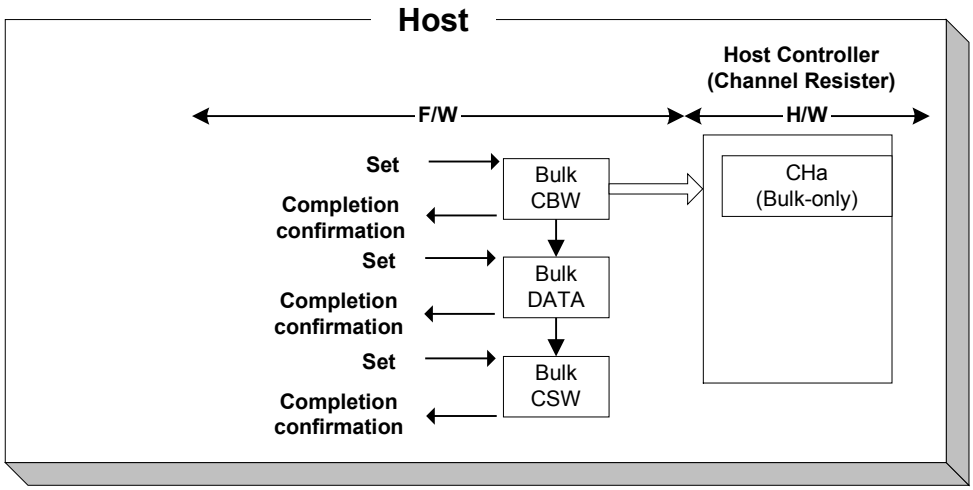


Figure 1-37 Control when not using the bulk-only support function (reference)

The bulk-only support function is valid for channel CHa. Select AREA1 as the FIFO area joined to channel CHa when using this function. The transport processing is as shown below when using this function. The firmware performs steps (1) to (5) and (8).

(1) The following basic setting registers are set appropriately in channel CHa.

Transfer speed (H_CHaConfig_0.SpeedMode), max packet size (H_CHaMaxPktSize), USB address (H_CHaFuncAdrs.FuncAdrs), FIFO area (AREA1StartAdrs_H,L,AREA1EndAdrs_H,L), FIFO area join (AREA1Join_1.JoinEPaCHa)

(2) The following control registers are set appropriately for the bulk-only support function.

OUT endpoint toggle sequence (H_CHaBO_OUT_EP_Control.OUT_Toggle), OUT endpoint number (H_CHaBO_OUT_EP_Control.OUT_EP_Number), IN endpoint toggle sequence (H_CHaBO_IN_EP_Control.OUT_Toggle), IN endpoint number (H_CHaBO_IN_EP_Control.IN_EP_Number)

(3) Writes the CBW data (31 bytes) to the FIFO CBW area.

(4) Sets the DMA or IDE settings to the FIFO area joined to CHa.

(5) Sets bulk-only support execution (H_CHaBO_SupportControl.BO_SupportGo).

The transport status (H_CHaBO_SupportControl.BO_TransportState) value is written to the H_CHaBO_SupportControl register as “Idle(00b).”

(6) Sends the CBW area data (31 bytes) to the OUT-direction endpoint indicated by the OUT endpoint number (H_CHaBO_OUT_EP_Control.OUT_EP_Number) using a bulk OUT transaction (command transport).

(7) Executes the data transport based on the CBW data.

- If the CBW data bmCBWFlags Bit7 is 0, the data in the FIFO area joined to CHa is sent to the OUT-direction endpoint indicated by the OUT endpoint number (H_CHaBO_OUT_EP_Control.OUT_EP_Number) by a bulk OUT transaction (OUT-direction data transport). The OUT-direction data transport ends once the data quantity indicated by the CBW data dCBWDataTransferLength has been sent by the OUT transaction.
- If the CBW data bmCBWFlags Bit7 is 1, a bulk IN transaction is issued to the IN-direction endpoint indicated by the IN endpoint number (H_CHaBO_IN_EP_Control.IN_EP_Number), and the data received is written to the FIFO area joined to CHa (IN-direction data transport). The IN-direction data transport ends once the data quantity indicated by the CBW data dCBWDataTransferLength has been received by the IN transaction. The IN-direction data transport also ends when a short packet is received by the IN transaction.
- If the CBW data dCBWDataTransferLength value is 0x00000000, no data transport is performed.

(8) If the FIFO area joined to CHa is smaller than the value indicated by the CBW data wCBWDataTransferLength, the firmware must divide the data transport data for processing.

- If the data transport is in the OUT direction, transactions are no longer issued once all data has been sent to the FIFO. The firmware must therefore write the remaining data to be sent to the FIFO in sequence while checking the FIFO free space.
- If the data transport is in the IN direction, transactions are no longer issued when no free FIFO space remains. The firmware must therefore read the data from the FIFO in the sequence received to create free space in the FIFO while checking the valid FIFO data quantity.

(9) A bulk IN transaction is issued to the IN-direction endpoint indicated by the IN endpoint number (H_CHaBO_IN_EP_Control.IN_EP_Number), and the data received is written to the FIFO CSW area (status transport). The data quantity received in the status transport corresponds to the status transport received data quantity (H_CHaBO_CSW_RcvDataSize.CSW_RcvDataSize).

- Status transport can be executed once the OUT-direction data transport ends.
- Status transport can be executed once all data received in the FIFO has been read out and the FIFO emptied when the IN-direction data transport has ended.

(10) The CSW data received in the status transport is checked. The details checked are as shown below.

- The received CSW data length is 13 bytes.
- CSW dCSWSignature is 0x53425355.
- CSW dCSWTag matches CBW dCBWTag.
- The BCSWStatus value is 0x00.

If any of the above conditions are not fulfilled, the bulk-only support execute (H_CHaBO_SupportControl.BO_SupportGo) is automatically cleared, and the bulk-only support function is stopped. A bulk-only support stop status (H_CHaIntStat.BO_SupportStop) is issued. The RAM_Monitor function can be used to read out data received in the CSW area.

(11) If the status transport ends normally, the bulk-only support execute (H_CHaBO_SupportControl.BO_SupportGo) is cleared automatically and a bulk-only support end status (H_CHaIntStat.BO_SupportCmp) is issued.

(12) If a transaction error is detected in any of the various transports, the bulk-only support execute (H_CHaBO_SupportControl.BO_SupportGo) is cleared automatically, halting the bulk-only support function and issuing a bulk-only support stop status (H_CHaIntStat.BO_SupportStop). The transport for which an error occurred is indicated by the TRANSPORT state (H_CHaBO_SupportControl.BO_TransportState). The condition code (H_CHaConditionCode) is set appropriately and a ChangeCondition status (H_CHaIntStat.ChangeCondition bit) issued.

The bulk-only support execute (H_CHaBO_SupportControl.BO_SupportGo) is cleared if the bulk-only support function is aborted. Status is issued once abort processing is completed for the bulk-only support function.

If the transport is completed up to the point of status transport before the end of abort processing, a bulk-only support completion status (CHaIntStat.BO_SupportCmp) is issued.

If abort processing ends before transport is completed up to the point of status transport, a bulk-only support stop status (H_CHaIntStat.BO_SupportStop) is issued.

The aborted transport is indicated by the TRANSPORT state (H_CHaBO_SupportControl.BO_TransportState).

To resume the bulk-only support function using the aborted transport, the TRANSPORT state (H_CHaBO_SupportControl.BO_TransportState) is set to the transport to be resumed (i.e., the settings are retained for the aborted transport) and the bulk-only support execute (H_CHaBO_SupportControl.BO_SupportGo) is set.

To execute a new bulk-only support function, the TRANSPORT state (H_CHaBO_SupportControl.BO_TransportState) is set to Idle(00b), and the bulk-only support execute (H_CHaBO_SupportControl.BO_SupportGo) is set.

The transfer execute bit (H_CHaConfig_0.TranGo), toggle sequence bit (H_CHaConfig_0.Toggle), transaction type (H_CHaConfig_1.TID), total size free bit (H_CHaConfig_1.TotalSizeFree), endpoint number (H_CHaFuncAdrs.EP_Number), and IRP data quantity (H_CHaTotalSize_HH,HL,LH,LL) are set or updated by hardware while the bulk-only support function is being executed. Do not write data to these bits.

See “1.3.3 Transactions” for detailed information on transaction errors.

See “1.6 FIFO Management” for detailed information on the FIFO CBW and CSW areas.

See “1.8.3.2 DMA0/DMA1(DMA ch.0/ch.1)” for detailed information on DMA.

See “1.10 IDE I/F” for detailed information on IDE.

Table 1-27 gives the settings and status for the bulk-only support function.

Table 1-27 Bulk-only support function settings and status

Item	Register/bit	Description
Bulk-only support execute	H_CHaBO_SupportControl.BO_SupportGo	Executes the bulk-only support function. See “1.3.8 Bulk-only Support Function” for particulars.
OUT endpoint toggle sequence	H_CHaBO_OUT_EP_Control.OUT_Toggle	Sets the OUT endpoint toggle sequence bit initial settings. Indicates the OUT endpoint toggle sequence bit state during or after the transaction.
OUT endpoint number	H_CHaBO_OUT_EP_Control.OUT_EP_Number	Sets the OUT endpoint number to a value between 0x0 and 0xF.
IN endpoint toggle sequence	H_CHaBO_IN_EP_Control.IN_Toggle	Sets the IN endpoint toggle sequence bit initial settings. Indicates the IN endpoint toggle sequence bit state during or after the transaction.
IN endpoint number	H_CHaBO_IN_EP_Control.IN_EP_Number	Sets the IN endpoint number to a value between 0x0 and 0xF.
Bulk-only support execute results	H_CHaIntStat.BO_SupportCmp H_CHaIntStat.BO_SupportStop	Indicates bulk-only support execute results.
Transaction status	H_CHaIntStat.TotalSizeCmp, H_CHaIntStat.TranACK, H_CHaIntStat.TranErr, H_CHaIntStat.ChangeCondition	Indicates transaction results.
Transaction condition code	H_CHaConditionCode	Indicates transaction result details.
TRANSPORT state	H_CHaBO_SupportControl.BO_TransportState	Indicates the transport during execution of the bulk-only support function. Indicates the transport for which an error occurred if stopped due to an error.
Status transport received data quantity	H_CSW_RcvDataSize	Indicates the data quantity received for the status transport.

1.3.9 Audio Class Assist Function

The audio class assist function is used when 16-bit 2-channel PCM data with a sampling frequency of 44.1 kHz is sent via isochronous transfer with a cycle of 1 ms. When this function is enabled, nine transactions are performed consecutively with a data packet size of 176 bytes, followed by one transaction of 180 bytes. The procedure is then repeated with nine transactions of 176 bytes followed by one transaction of 180 bytes, with the data packet size varying automatically.

This data packet size change sequence is maintained while the H_CHx{x=b-e}Config_1.Audio441 bit is "1." Clear the H_CHx{x=b-e}Config_1.Audio441 bit when initializing this sequence.

Table 1-28 Audio class assist function control settings

Item	Register/bit	Description
Audio class assist function	H_CHx{x=b-e}Config_1.Audio441	Enables the audio class assist function. This function can be used only for OUT-direction isochronous transfers. Do not set this bit to "1" when using other transfers.

1.3.10 Host State Management Support Function

1.3.10.1 Host State

The host state must be changed based on upstream requests or bus status. The state is managed by firmware. The hardware supports different settings and negotiations for each state.

Figure 1-38 illustrates host state transitions.

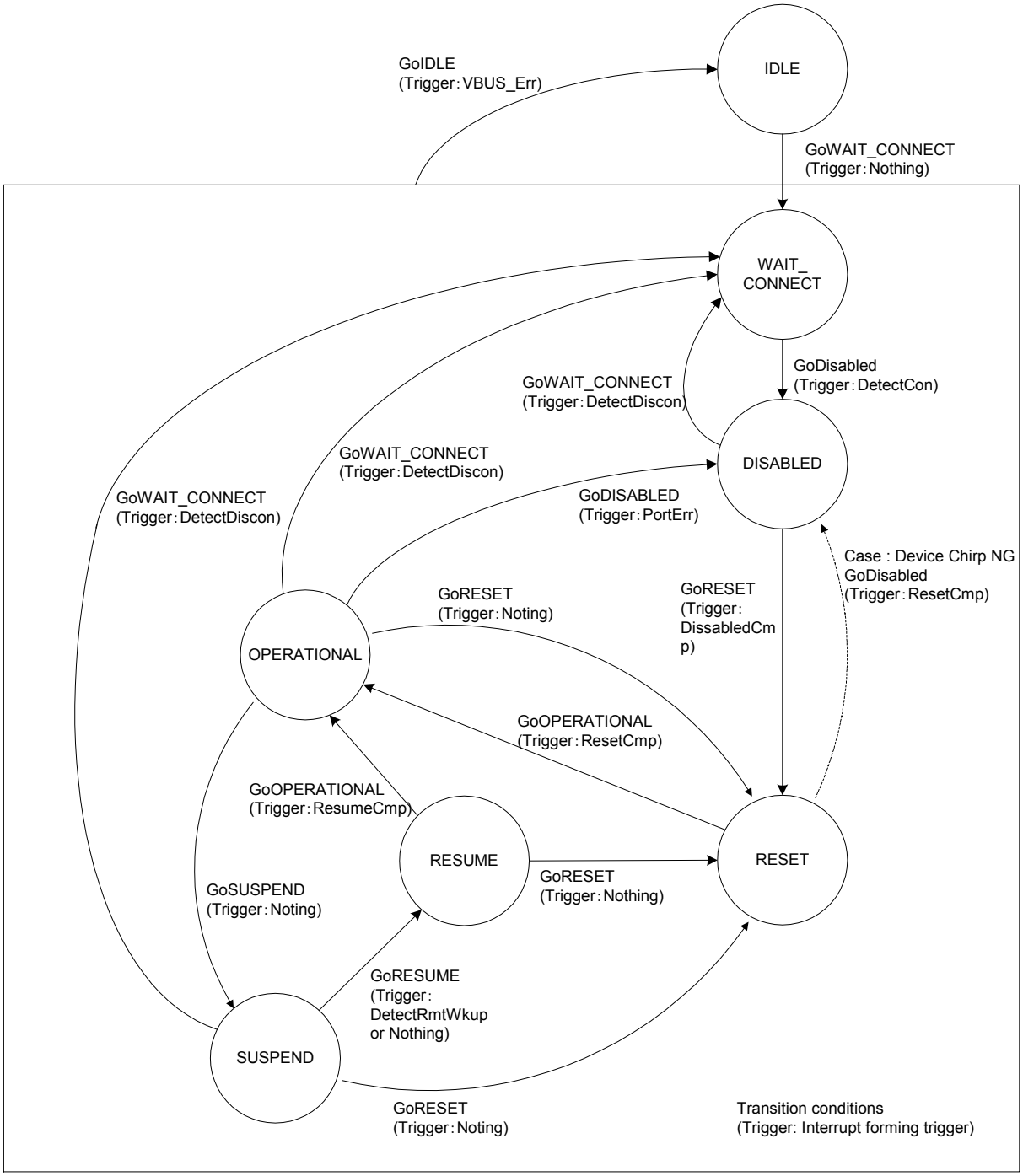


Figure 1-38 Host state transition diagram

Table 1-29 and Table 1-30 show host state management support function settings and status.

Table 1-29 Host state management support function settings and status

Item	Register/bit	Explanation
Host state transition execute	H_NegoControl_0.AutoMode	Sets the host state to be changed. Any of the following settings can be set. GoIDLE GoWAIT_CONNECT GoDISABLED GoRESET GoOPERATIONAL GoSUSPEND GoRESUME GoWAIT_CONNECTtoDIS GoWAIT_CONNECTtoOP GoRESETtoOP GoRESUMEtoOP GoSUSPENDtoOP
Host state transition execute cancel	H_NegoControl_0.AutoModeCancel	Aborts the current host state processing and remains in that state.
Host state monitor	H_NegoControl_0.HostState	Indicates the current host state (see below). IDLE WAIT_CONNECT DISABLED RESET OPERATIONAL SUSPEND RESUME
VBUS state monitor	H_USB_Status.VBUS_State	Indicates the VBUS state (normal/error).
Remote wakeup receipt permission	H_NegoControl_1.RmtWkupDetEnb	Permits the receipt of a remote wakeup.
Chirp completion disable	H_NegoControl_1.DisChirpFinish	Sets the operating mode for when the device chirp does not complete within the specified time.
VBUS error detection status	H_SIE_IntStat_0.VBUS_Err	Indicates an error on VBUS.
Connect detection status	H_SIE_IntStat_0.DetectCon	Indicates that a device is connected to the downstream port.
Disconnect detection status	H_SIE_IntStat_0.DetectDisCon	Indicates that the device has been disconnected from the downstream port.
Remote wakeup detection status	H_SIE_IntStat_0.DetectRmtWkup	Indicates that a remote wakeup signal from the device has been detected.
Device chirp normal detection status	H_SIE_IntStat_0.DetectDevChirpOK	Indicates a normal chirp signal from the device.
Device chirp error detection status	H_SIE_IntStat_0.DetectDevChirpNG	Indicates an error in the chirp signal from the device.
Reset completion status	H_SIE_IntStat_1.ResetCmp	Indicates successful completion of USB reset.
Suspend change completion status	H_SIE_IntStat_1.SuspendCmp	Indicates successful transition to suspend.
Resume completion status	H_SIE_IntStat_1.ResumeCmp	Indicates successful completion of resume.

(continued)

Table 1-30 Host state management support function settings and status (continued)

Item	Register/bit	Explanation
Port speed	H_NegoControl_1.PortSpeed	Indicates the downstream port operating speed (HS/FS/LS). Operation of device connected to port.
Line state	H_USB_Status.LineState	Indicates the signal state on the USB cable.
Transceiver selection	H_XcvrControl.XcvrSelect	Selects and enables the HS, FS, or LS transceiver.
Terminal selection	H_XcvrControl.TermSelect	Selects and enables either the HS or FS terminal.
Operating mode	H_XcvrControl.OpMode	Sets the HTM operating mode.

1.3.10.1.1 IDLE

This state initializes the USB host function. It is the default state when the host function is enabled.

In all other states, the state must be changed to this state when VBUS_Err is detected.

To change, write “0x80” to the H_NegoControl_0 register (“1” for H_NegoControl_0.AutoModeCancel and “0x0” for H_NegoControl_0.AutoMode) and halt the state operation during execution. Since the H_NegoControl_0.AutoModeCancel bit changes to “0” when the stop processing is completed (requires approximately 6 cycles with a 60 MHz clock), this must be confirmed to be “0” before writing “0x01” to the register (i.e., setting host state change execute (H_NegoControl0.AutoMode) to “GoIDLE”). Then the LSI switches to this state.

This state automatically sets the following items:

- Immediately halts the USB host transaction execute function.
- Sets the port to FS mode and “NonDriving.”
- Switches off VBUSEN_A.
- Switches off all detection functions, including connect detection, disconnect detection, remote wakeup detection, and device chirp detection.

1.3.10.1.2 WAIT_CONNECT

This state waits for a device to be connected to the downstream port.

This state must be switched to wait for device connection following an upstream request in “IDLE” state or device disconnect detection in the “DISABLED,” “OPERATIONAL” or “SUSPEND” states.

Setting the host state transition execute (H_NegoControl_0.AutoMode) to “GoWAIT_CONNECT” switches to this state.

This state first sets the following items automatically.

- Immediately halts the USB host transaction execute function.
- Sets the port to FS mode and "PowerDown."
- Switches on VBUSEN_A.
- Switches off all the detection functions, including connect detection, disconnect detection, remote wakeup detection, and device chirp detection.

The state then waits for the bus power device internal power supply stabilization time, switches on the connect detection function automatically, and waits for device connection. The time taken from switching VBUSEN_A on until device connect detection is not managed by hardware. This time must be managed by the firmware as needed.

The line state can be referenced as "J" when an FS or HS device is connected. If an LS device is connected, the line state can be referenced as "K." Device connection is detected if either of these states continues for more than 2.5 µs, and the port is switched to LS mode if an LS device connection is detected.

The disconnect detection function is automatically switched on once connection is detected.

Connect detection status (H_SIE_IntStat_0.DetectCon) is issued to the firmware if disconnection is not detected during the debounce interval, and the connection and disconnect detection functions are automatically switched off. If disconnection is detected, however, the disconnect detection function is automatically switched off, and connect detection is automatically repeated.

1.3.10.1.3 DISABLED

This state prevents the sending or receiving of bus signals while a device is connected to the downstream port.

The LSI switches to this state when connection is detected in "WAIT_CONNECT" state, when a chirp is detected from a device with an error in "RESET" state, or when a port error is detected in "OPERATIONAL" state.

Setting the host state transition execute (H_NegoControl_0.AutoMode) to "GoDISABLED" switches to this state.

This state sets the following items automatically.

- Waits for the current transaction to end and stops the USB host transaction execute function.
- The port is switched to FS mode if in HS mode when switching to this state, and the port connects in the same mode if in FS or LS mode.
- The port is switched to "PowerDown."

The following processing is then performed automatically after the end of the disconnect detection disable period.

- Turns on the disconnect detection function.
- Issues disabled transition completion status (H_SIE_IntStat_1.DisabledCmp).

1.3.10.1.4 RESET

This state issues a USB reset to the downstream port.

The LSI switches to this state and issues a USB reset when the disabled change completion status is issued in “DISABLED” state.

This state can be switched to from any USB state (“OPERATIONAL,” “SUSPEND,” or “RESUME”) if requested from upstream.

This system switches to this state if the host state change execute (H_NegoControl_0.AutoMode) is set to “GoRESET.”

The following items are set automatically in this state.

- The USB host transaction execute function is aborted after awaiting completion of the current transaction.
- The port is set to HS mode and “NormalOperation” (reset signal SE0 is driven for the USB cable signal).
- Connect detection, disconnect detection, and remote wakeup detection functions are switched off.
- The device chirp detection function is switched on.

Chirps from the device are detected using “HS K.” A normal chirp is detected if the line state is detected continuously as “K” for at least 2.5 µs and ends within the specified time after the USB reset is issued. If it does not end within the specified time, the chirp is detected as an error.

The following processing is performed automatically based on detection results:

(1) If a normal device chirp is detected

“HS K” (Chirp K) and “HS J” (Chirp J) are sent alternatively and continuously by the host on completion of the chirp from the device. Reset completion status (H_SIE_IntStat_1.ResetCmp) is issued to the firmware once the host has finished sending the chirp.

The port remains in HS mode.

(2) If a device error chirp is detected

Device chirp error detection status (H_SIE_IntStat_0.DetectDevChirpNG) is issued to the firmware once the specified time has elapsed. One of two subsequent operating modes can be selected using the chirp completion disable (H_NegoControl_1.DisChirpFinish) setting. See "1.3.10.2.4.2 If a Device Error Chirp Is Detected" for detailed information.

(3) If the connected device is FS and no chirp is detected from the device

The port is set to FS mode after issuing the USB reset for the specified time.

A reset completion status (H_SIE_IntStat_1.ResetCmp) is issued to the firmware.

(4) If the connected device is LS

The port is set to LS mode after issuing the USB reset for the specified time.

Reset completion status (H_SIE_IntStat_1.ResetCmp) is issued to the firmware.

1.3.10.1.5 OPERATIONAL

This state executes the USB transaction.

This system switches to this state, and the transaction is executed after "RESET" or "RESUME" ends.

This system switches to this state if the host state change execute (H_NegoControl_0.AutoMode) is set to "GoOPERATIONAL."

The following items are set automatically in this state.

- The port is set to "NormalOperation."
- The USB host transaction execute function is enabled.
- The disconnect detection function is switched on.

1.3.10.1.6 SUSPEND

This state suspends the USB.

This system switches to this state from "OPERATIONAL" when use of the USB is aborted.

This system switches to this state if the host state change execute (H_NegoControl_0.AutoMode) is set to "GoSUSPEND."

The following items are set automatically in this state:

- The disconnect detection and remote wakeup detection functions are switched off.
- The USB host transaction execute function is aborted after awaiting completion of the current transaction.

- The port is set to FS mode if it was in HS mode, and the port remains in the same mode if in FS or LS mode.
- The port is switched to "PowerDown."

The following processing is then performed automatically after the disconnect or remote wakeup detection disable time period has elapsed.

- The disconnect detection function is switched on.
- The remote wakeup detection function is switched on if the remote wakeup receipt permission (H_NegoControl_1.RmtWkupDetEnb) is enabled.
- A suspend change completion status (H_SIE_IntStat_1.SuspendCmp) is issued.

A remote wakeup detection status (H_SIE_IntStat_0.DetectRmtWkup) is then issued to the firmware on detection of a remote wakeup signal ("K" continuously for at least 2.5 µs) if the remote wakeup receipt permission (H_NegoControl_1.RmtWkupDetEnb) is enabled.

1.3.10.1.7 RESUME

This state issues a USB resume signal to the downstream port.

This system switches to this state from "SUSPEND" to recover the USB device from the SUSPEND state.

This state is switched when the host state change execute (H_NegoControl_0.AutoMode) is set to "GoRESUME."

The disconnect and remote wakeup detection functions are switched off automatically in this state.

A resume signal ("K") is issued for the specified duration.

When the resume signal has been issued, the port is returned to the mode setting before entering "SUSPEND" for "NormalOperation."

A resume completion status (H_SIE_IntStat_1.ResumeCmp) is issued to the firmware.

1.3.10.2 Detection Functions

1.3.10.2.1 VBUS Error Detection

A VBUS error is detected in the form of a level change (H to L) at the VBUSFLG_A input terminal. The execution procedure is as follows for detection of a VBUS error. Step (2) below is performed automatically by the LSI hardware.

(1) The VBUSFLG_A (external USB power switch error occurrence flag) input terminal changes to L (error occurrence) (T0).

(2) VBUS error detection status (HostIntStat.VBUS_Err) is issued to the firmware (T0).

VBUS must be switched off immediately when the host detects the VBUS error. The firmware therefore writes 0x80 to the H_NegoControl_0 register ("1" to H_NegoControl_0.AutoModeCancel and "0x0" to H_NegoControl_0.AutoMode) while identifying the VBUS error detection status to stop the state operation currently underway. Since the H_NegoControl_0.AutoModeCancel bit changes to "0" when the stop processing is completed (requires approximately 6 cycles with a 60 MHz clock), this must be confirmed to be "0" before writing "0x01" to the register (i.e., setting host state change execute (H_NegoControl0.AutoMode) to "GoIDLE"). This switches to IDLE state, disables the VBUSEN_A terminal logic, and allows the VBUS to be switched off.

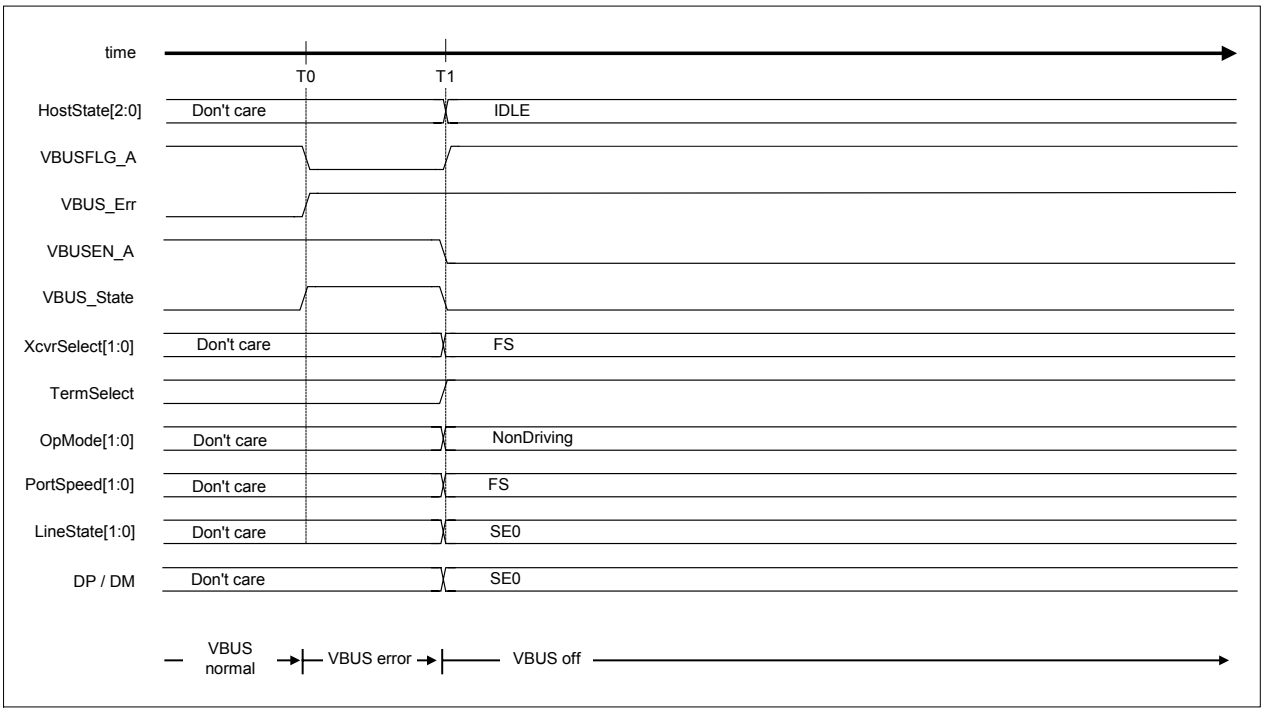


Figure 1-39 VBUS error detection timing

Table 1-31 VBUS error detection timing values

Timing Parameter	Description	Value
T0	VBUSFLG_A (external USB power switch error occurrence flag) input terminal changes to L (error). VBUS error detection status (HostIntStat.VBUS_Err) is issued. (H/W)	0 (reference)
T1 (reference)	Write 0x01 to change to “IDLE” state after writing 0x80 to H_NegoControl_0. (F/W)	T1

1.3.10.2.2 Disconnect Detection

This detects device disconnection while in “DISABLED,” “OPERATIONAL,” or “SUSPEND” states.

When a disconnection is detected, change the host state to “WAIT_CONNECT” to repeat from connect detection without switching off VBUS. Change the host state to “IDLE” to switch off VBUS.

1.3.10.2.2.1 If HS Device is Disconnected

HS device disconnection is detected in “OPERATIONAL” state.

The procedures are as shown below if an HS device is disconnected. Steps (2) to (3) below are performed automatically by the LSI hardware.

(1) The device is disconnected (T0).

(2) Disconnect detection is performed for the uSOF(HS_SOF) EOP time period, and the device is determined to be disconnected if identified as disconnected three times in succession (T1).

(3) A disconnect detection status (H_SIE_IntStat_0.DetectDiscon) is issued to the firmware (T1).

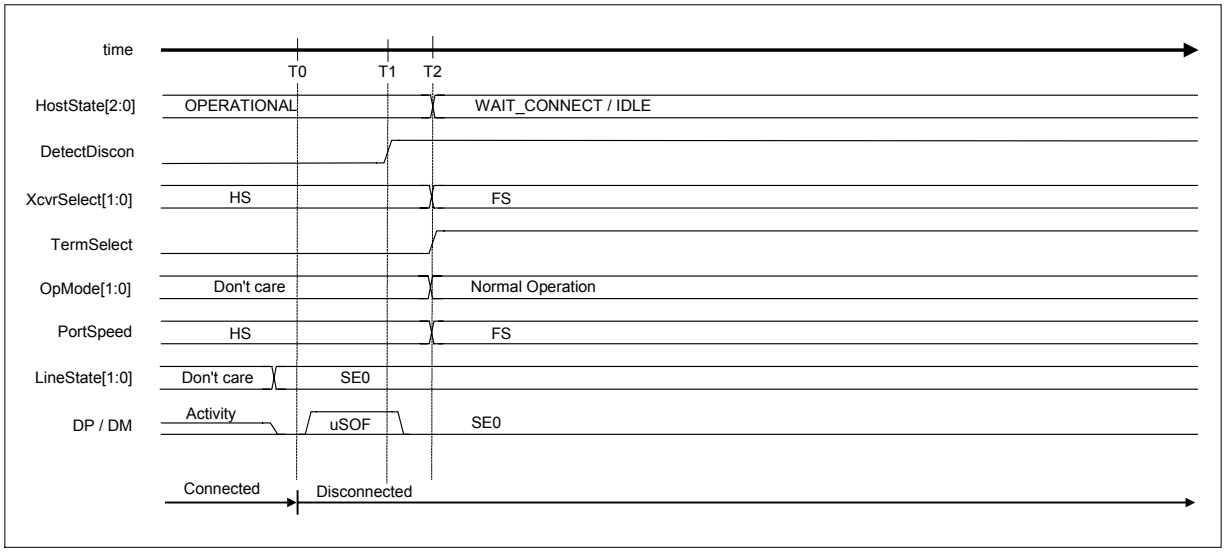


Figure 1-40 Disconnect detection timing (HS mode)

Table 1-32 Disconnect detection timing values (HS mode)

Timing Parameter	Description	Value
T0	Device is disconnected.	0 (reference)
T1	Disconnect detection status (H_SIE_IntStat_0.DetectDiscon) is issued. (H/W)	T1
T2 (reference)	Set the host state change execute (H_NegoControl_0.AutoMode) to “GoWAIT_CONNECT.” (F/W)	T2

1.3.10.2.2.2 If FS or LS Device is Disconnected

FS or LS device disconnection is detected in “DISABLED,” “OPERATIONAL,” and “SUSPEND” states.

The procedures are as shown below for disconnection of an FS or LS device. Steps (2) to (3) below are performed automatically by the LSI hardware.

(1) The device is disconnected (T0).

(2) Disconnection is detected from the signal line state (T1).

(3) Disconnect detection status (H_SIE_IntStat_0.DetectDiscon) is issued to the firmware (T1).

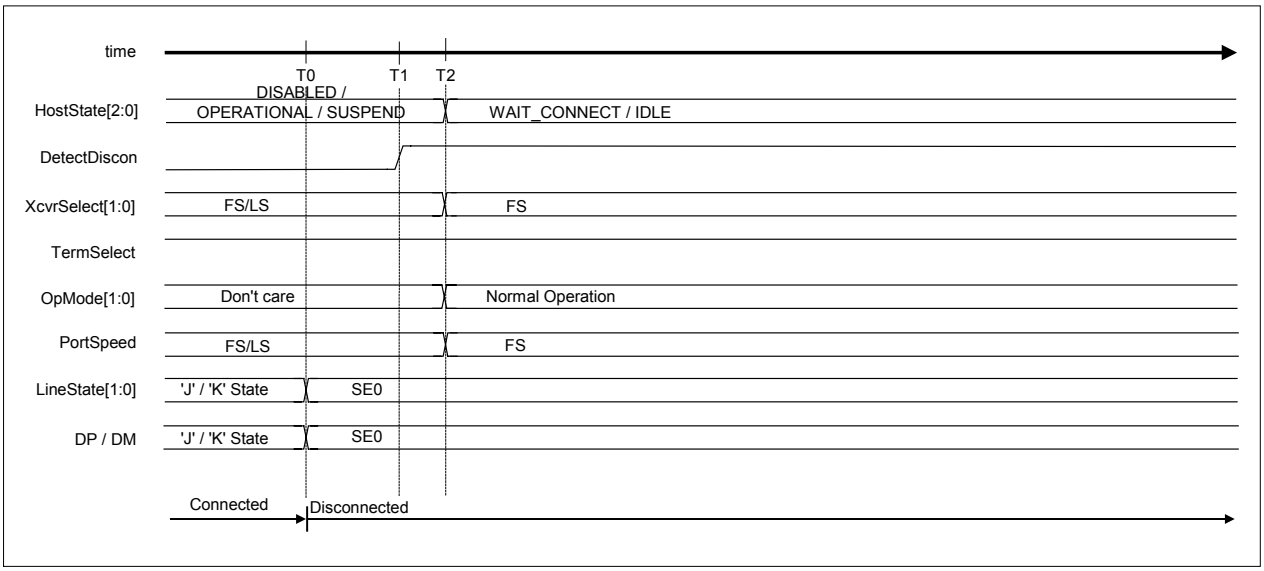


Figure 1-41 Disconnect detection timing (FS or LS mode)

Table 1-33 Disconnect detection timing values (FS or LS mode)

Timing Parameter	Description	Value
T0	Device is disconnected.	0 (reference)
T1	Disconnect detection status (H_SIE_IntStat_0.DetectDiscon) is issued. (H/W)	T0 + 2.5us< T1 {T_{DDIS}}
T2 (reference)	Set the host state change execute (H_NegoControl_0.AutoMode) to “GoWAIT_CONNECT.” (F/W)	No specifications

1.3.10.2.3 Remote Wakeup Detection

This detects remote wakeup in “SUSPEND” state if remote wakeup receipt permission (H_NegoControl_1.RmtWkupDetEnb) is enabled.

1.3.10.2.3.1 If HS Device Is Connected

The procedures are as shown below if an HS device is connected. Steps (2) to (3) below are performed automatically by the LSI hardware.

(1) The device begins sending a remote wakeup signal (K) (T0).

(2) The host detects the remote wakeup signal (K) (T1).

(3) Remote wakeup detection status (H_SIE_IntStat_0.DetectRmtWkup) is issued to the firmware (T1).

Note that the host must issue the resume signal (“K”) within 1 ms of detection of the device remote wakeup. The firmware must immediately recognize the remote wakeup detection status and set the host state change execute (H_NegoControl_0.AutoMode) to “GoRESUME” within 900 µs.

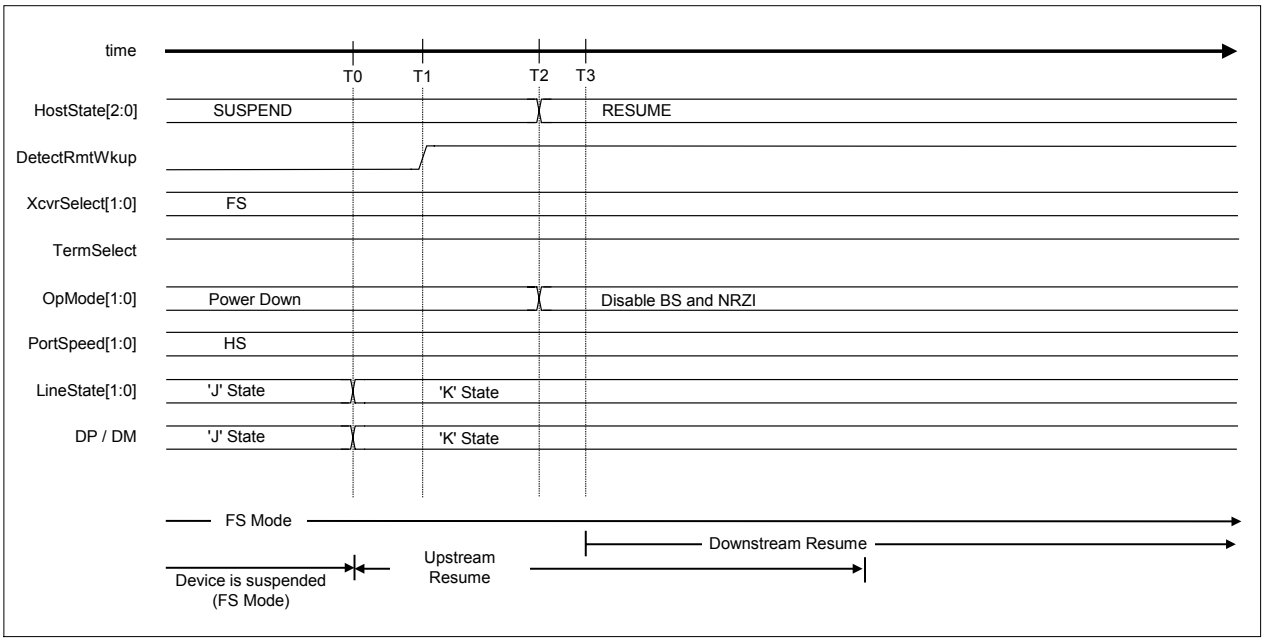


Figure 1-42 Remote wakeup timing (HS mode)

Table 1-34 Remote wakeup timing values (HS mode)

Timing Parameter	Description	Value
T0	The device begins sending the remote wakeup signal (K).	0 (reference)
T1	The remote wakeup signal (K) is detected. Remote wakeup detection status is issued. (H/W)	T0 + 2.5us{T_{URLK}} < T1
T2 (reference)	The host state change execute (H_NegoControl_0.AutoMode) is set to “GoRESUME.” (F//W)	T2< T1 + 900us
T3 (reference)	The host begins sending the resume signal (“K”). (H/W)	T3< T0 + 1ms{T_{URSM}}

Note: Brackets {} indicate terms defined in the USB 2.0 standards.

1.3.10.2.3.2 If FS Device Is Connected

The procedures when an FS device is connected are the same as when an HS device is connected.

For detailed information on the procedures performed, refer to "1.3.10.2.3.1 If HS Device Is Connected."

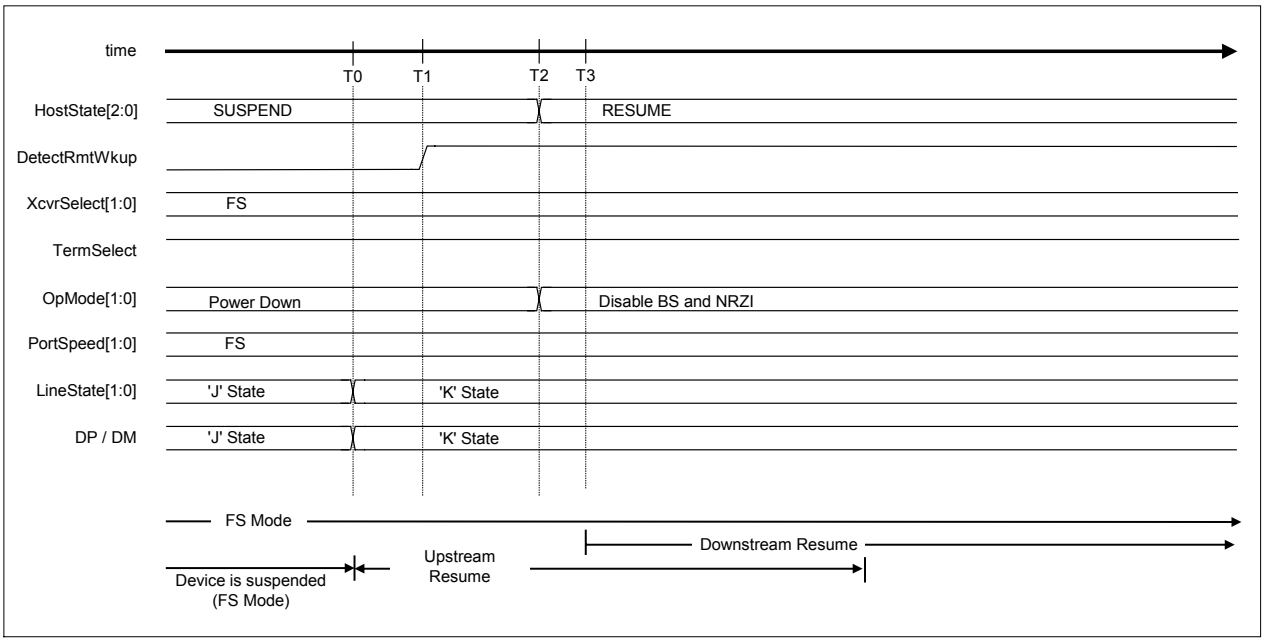


Figure 1-43 Remote wakeup timing (FS mode)

Table 1-35 Remote wakeup timing values (FS mode)

Timing Parameter	Description	Value
T0	The device begins sending the remote wakeup signal (K).	0 (reference)
T1	The remote wakeup signal (K) is detected. Remote wakeup detection status is issued. (H/W)	T0 + 2.5us< T1 {T_{URLK}}
T2 (reference)	The host state change execute (H_NegoControl_0.AutoMode) is set to "GoRESUME." (F//W)	T2< T1 + 900us
T3 (reference)	The host begins issuing the resume signal ("K"). (H/W)	T3< T0 + 1ms{T_{URSM}}

Note: Brackets {} indicate terms defined in the USB 2.0 standards.

1.3.10.2.3.3 If LS Device Is Connected

The procedures when an LS device is connected are the same as when an HS device is connected.

For detailed information on the procedures performed, refer to "1.3.10.2.3.1 If HS Device Is Connected."

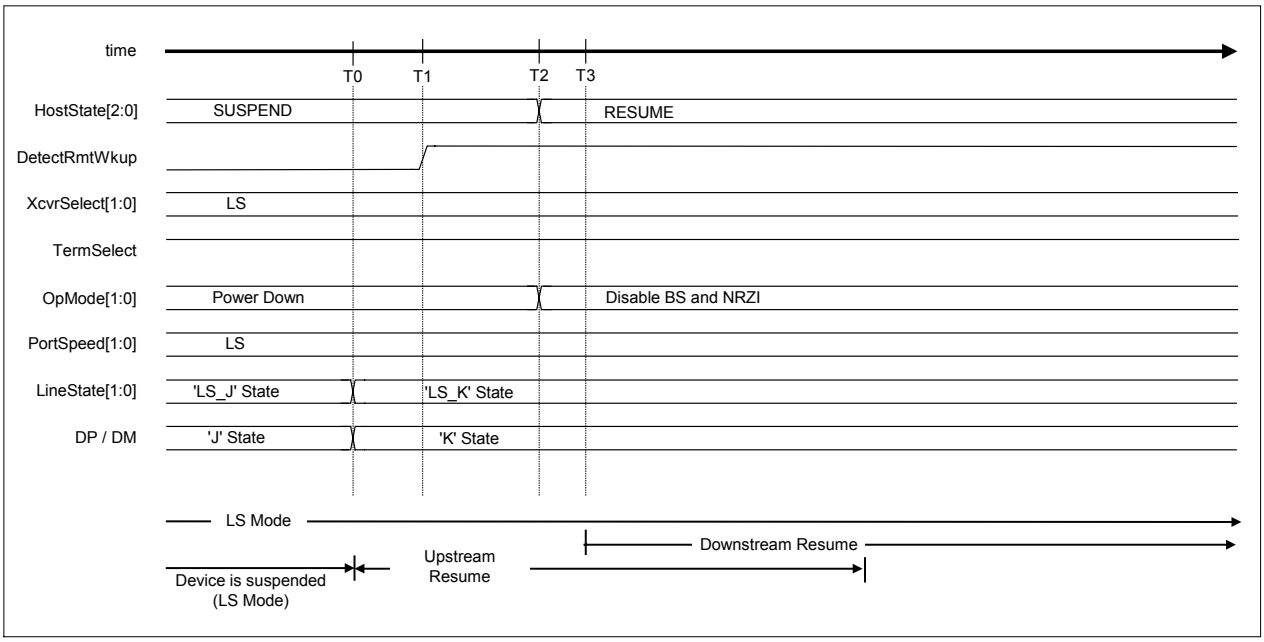


Figure 1-44 Remote wakeup timing (LS mode)

Table 1-36 Remote wakeup timing values (LS mode)

Timing Parameter	Description	Value
T0	The device begins sending the remote wakeup signal ("K").	0 (reference)
T1	The remote wakeup signal ("K") is detected. Remote wakeup detection status is issued. (H/W)	T0 + 2.5us< T1 {T_{URLK}}
T2 (reference)	The host state change execute (H_NegoControl_0.AutoMode) is set to "GoRESUME." (F//W)	T2< T1 + 900us
T3 (reference)	The host begins sending the resume signal ("K"). (H/W)	T3< T0 + 1ms{T_{URSM}}

Note: Brackets {} indicate terms defined in the USB 2.0 standards.

1.3.10.2.4 Device Chirp Detection Function

This detects device chirps.

The device chirp detection function is switched on in the “RESET” state.

1.3.10.2.4.1 If a Correct Device Chirp Is Detected

Given below is the device chirp detection procedure.

(1) The firmware sets the host state change execute (H_NegoControl_0.AutoMode) to “GoRESET” (T0).

(2) The device chirp detection function is switched on (T0).

(3) The device sends a chirp (T1).

(4) The device chirp is recognized if “K” persists beyond the specified duration in the line state (H_USB_Status.LineState[1:0]) (T2).

(5) Device chirp normal detection status (H_SIE_IntStat0.DetectDevChirpOK) is issued if the device chirp ends within the specified time after reset starts (line state (H_USB_Status.LineState[1:0]) becomes “SE0”) (T3).

(6) The device chirp detection function is switched off if a device chirp is detected (T3).

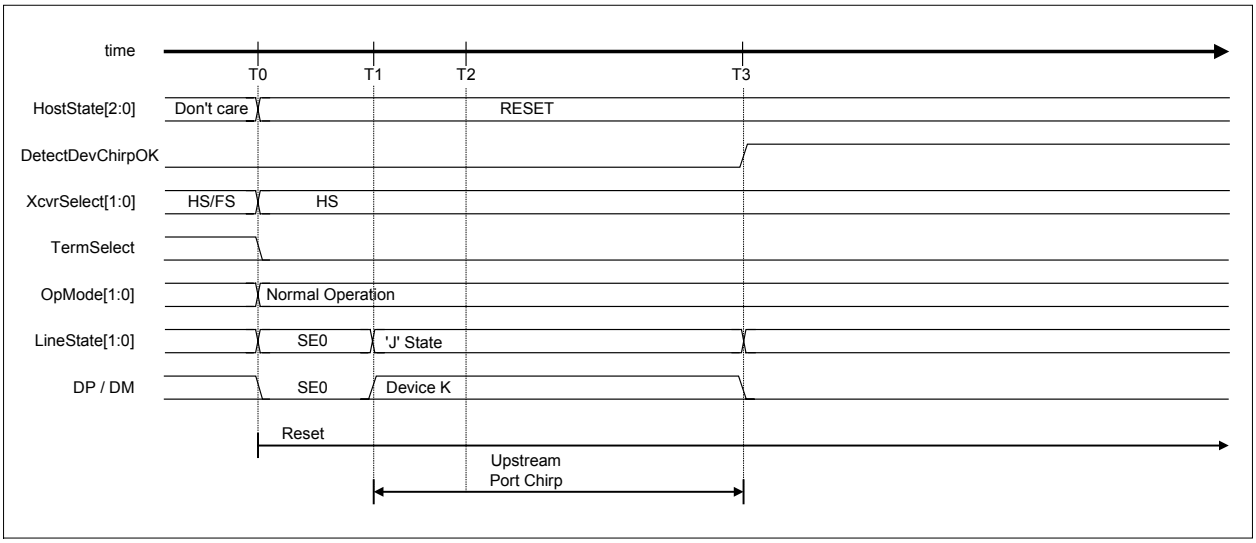


Figure 1-45 Device chirp timing

Table 1-37 Device chirp timing values

Timing Parameter	Description	Value
T0	H_NegoControl_0.AutoMode is set to “GoRESET.” (F/W)	0 (reference)
T1	The device starts the chirp.	T0 < T1 < T0 + 6.0ms
T2	The device recognizes the chirp. (H/W)	T1 + 2.5us {T_{FILT}} < T2
T3	The device ends the chirp. The device chirp detection function is switched off. Device chirp normal detection status (DetectDevChirpOK) is issued. (H/W)	T1 + 1.0ms {T_{UCH}} < T3 < T0 + 7.0ms {T_{UCHEND}}

Note: Brackets {} indicate terms defined in the USB 2.0 standards.

1.3.10.2.4.2 If a Device Error Chirp Is Detected

If the device chirp does not end within the specified time period, the device chirp detection function treats this as an error and issues the status.

(1) The firmware sets the host state change execute (H_NegoControl_0.AutoMode) to “GoRESET” (T0).

(2) The device chirp detection function is switched on (T0).

(3) The device sends a chirp (T1).

(4) The device chirp is recognized if “K” persists beyond the specified duration in the line state (H_USB_Status.LineState[1:0]) (T2).

(5) Device chirp error detection status (H_SIE_IntStat_0.DetectDevChirpNG) is issued if the device chirp does not end within the specified time after reset starts and treated as an error (T3).

(6) The device chirp detection function is switched off (T3).

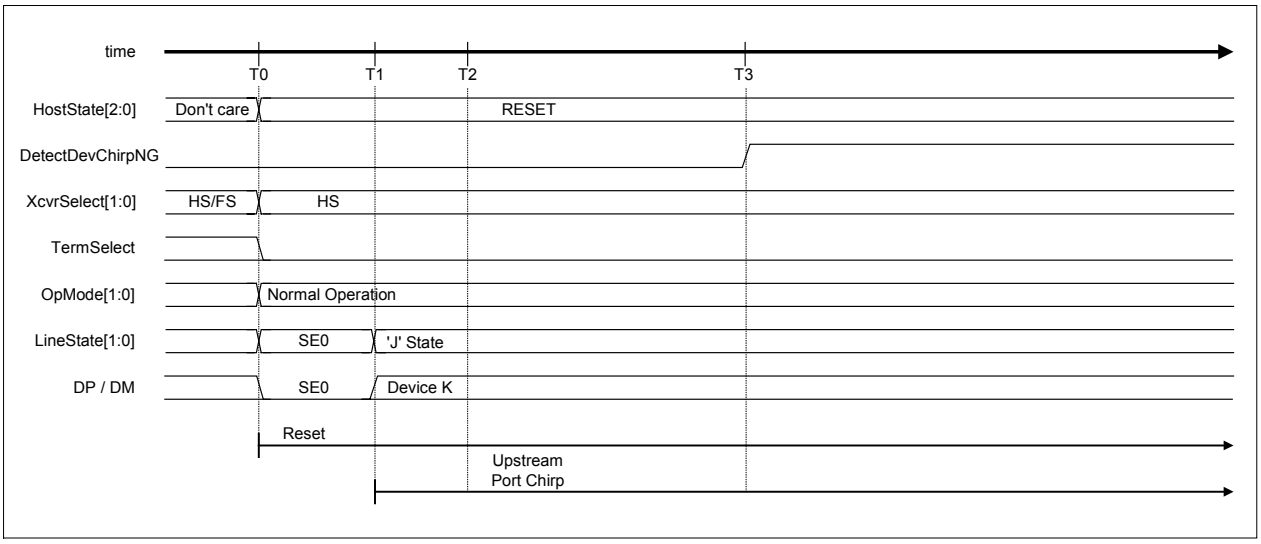


Figure 1-46 Device chirp timing (NG)

Table 1-38 Device chirp timing (NG) values

Timing Parameter	Description	Value
T0	H_NegoControl_0.AutoMode is set to “GoRESET.” (F/W)	0 (reference)
T1	The device starts the chirp.	T0 < T1 < T0 + 6.0ms
T2	The device recognizes the chirp. (H/W)	T1 + 2.5us {T_{FILT}} < T2
T3	Device chirp error detection status (DetectDevChirpNG) is issued. (H/W)	T0 + 7ms{T_{UCHEND}} < T3

Note: Brackets {} indicate terms defined in the USB 2.0 standards.

1.3.10.2.5 Port Error Detection

This detects port errors in “OPERATIONAL” state.

A port error is assumed if EOP cannot be detected for the packet being received, even after reaching the last (micro) frame.

The host issues the port error detection status (H_FrameIntStat.PortErr) to the firmware on detecting a port error, immediately halting the transaction. No subsequent transactions are issued, including SOF.

The firmware must perform the following processing if a port error occurs.

(1) Set H_NegoControl_0.AutoMode to “GoDISABLED.”

(2) Set H_NegoControl_0.ResetHTM to “1” and reset the host transceiver macro.

(3) Set H_NegoControl_0.ResetHTM to “0” after at least three cycles have elapsed with a 60 MHz clock and cancel the host transceiver macro reset.

1.3.10.3 Individual Host State Management Support Function Explanations

1.3.10.3.1 GoIDLE

The operation of the state being executed is stopped by writing 0x80 to the H_NegoControl_0 register ("1" to H_NegoControl_0.AutoModeCancel and "0x0" to H_NegoControl_0.AutoMode). The H_NegoControl_0.AutoModeCancel bit changes to "0" when stop processing is completed (requires approximately 6 cycles with a 60 MHz clock). Thus, it must be confirmed that the bit has changed to "0" when writing "0x01" to the register (i.e., setting host state change execute (H_NegoControl_0.AutoMode) to "GoIDLE"). The LSI hardware then automatically performs the processing necessary to change to "IDLE."

Steps (3) to (8) below are performed automatically by the LSI hardware.

(1) Writes 0x80 to H_NegoControl_0.AutoModeCancel ("1" to H_NegoControl_0.AutoModeCancel and "0x0" to H_NegoControl_0.AutoMode) (T0).

(2) Confirms that the H_NegoControl_0.AutoModeCancel bit has changed to "0," and writes "0x01" to H_NegoControl_0 ("0x1" to H_NegoControl_0.AutoMode) (T1).

(3) Sets the host state monitor (H_NegoControl_0.HostState) to "IDLE" (T1).

(4) Switches off VBUSEN_A (T1).

(5) Sets transceiver selection (H_XcvrControl.XcvrSelect) and terminal selection (H_XcvrControl.TermSelect) to FS mode (T1).

(6) Sets the operating mode (H_XcvrControl.OpMode[1:0]) to "NonDriving" (T1).

(7) Immediately halts the USB host transaction execute function (T1).

(8) Switches off all detection functions, including connect detection, disconnect detection, remote wakeup detection, and device chirp detection functions.

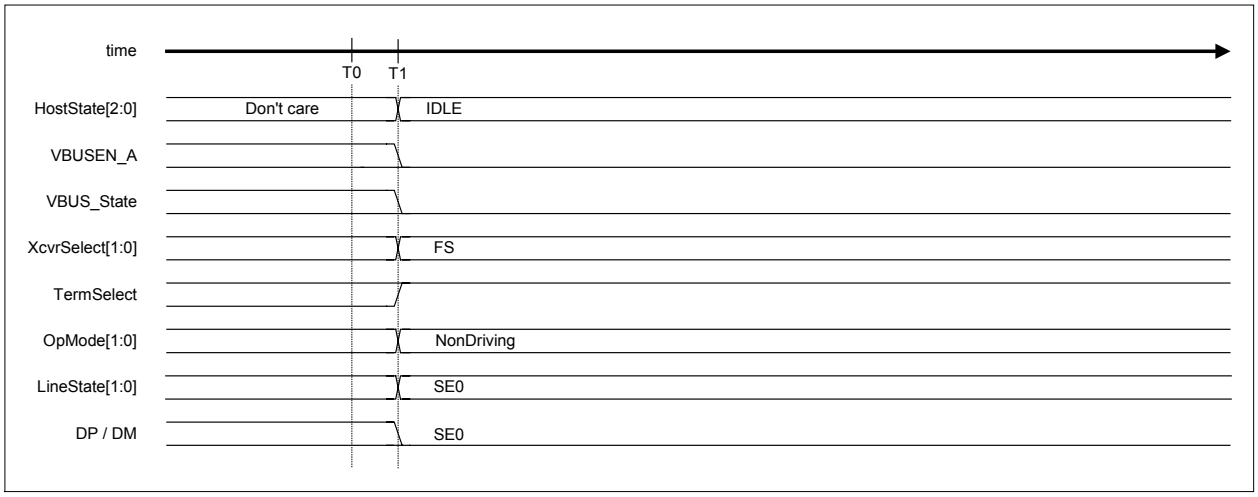


Figure 1-47 GoIDLE timing

Table 1-39 GoIDLE timing values

Timing Parameter	Description	Value
T0	Aborts operation for the state being executed. (F/W)	0 (reference)
T1	Confirms that the host state change execute cancel bit is “0” and sets H_NegoControl_0.AutoMode to “GoIDLE.” (F/W) Switches off VBUSEN_A. Sets the transceiver selection to FS mode. Sets the terminal selection to FS mode. Sets the operating mode to “NonDriving.” Immediately halts the transaction execute function. Switches off connect detection, disconnect detection, remote wakeup detection, and device chirp detection. (H/W)	T0 + 5cycle(60MHz) < T1

1.3.10.3.2 GoWAIT_CONNECT

The LSI hardware automatically performs the processing necessary to change to "WAIT_CONNECT" when "GoWAIT_CONNECT" is set to the host state change execute (H_NegoControl0.AutoMode).

Note that the HS device is connected as an FS device here. It is operated as an HS device following the HS Detection Handshake reset operation performed subsequently.

1.3.10.3.2.1 If an FS Device Is Connected

Given below is the procedures when an FS device is connected. Steps (2) to (12) below are performed automatically by the LSI hardware.

(1) The firmware sets the host state change execute (H_NegoControl_0.AutoMode) to "GoWAIT_CONNECT" (T0).

(2) Sets the host state monitor (H_NegoControl_0.HostState) to "WAIT_CONNECT" (T0).

(3) Switches on VBUSEN_A (T0).

(4) Sets transceiver selection (H_XcvrControl.XcvrSelect) and terminal selection (H_XcvrControl.TermSelect) to FS mode (T0).

(5) Sets the operating mode (H_XcvrControl.OpMode[1:0]) to "PowerDown" (T0).

(6) Sets the port speed (H_NegoControl_1.PortSpeed[1:0]) to "FS" (T0).

(7) Waits for the preset time for the device internal power supply to stabilize, then switches on the connect detection function (T1).

(8) The line state (H_USB_Status.LineState[1:0]) indicates "J" when an FS device is connected (T2).

(9) FS device connection is assumed if the line state (H_USB_Status.LineState[1:0]) remains at "J" continuously for at least 2.5 μs (T3).

(10) Switches on the disconnect detection function (T3).

(11) Issues connect detection status (H_SIE_IntStat_0.DetectCon) if no disconnection is detected during the debounce interval (T4). If a disconnection is detected during this period, the disconnect detection function is switched off, and connect detection is repeated from step (8). A disconnect detection status (H_SIE_IntStat_0.DetectDiscon) is not issued.

(12) Switches off the disconnect and connect detection functions (T4).

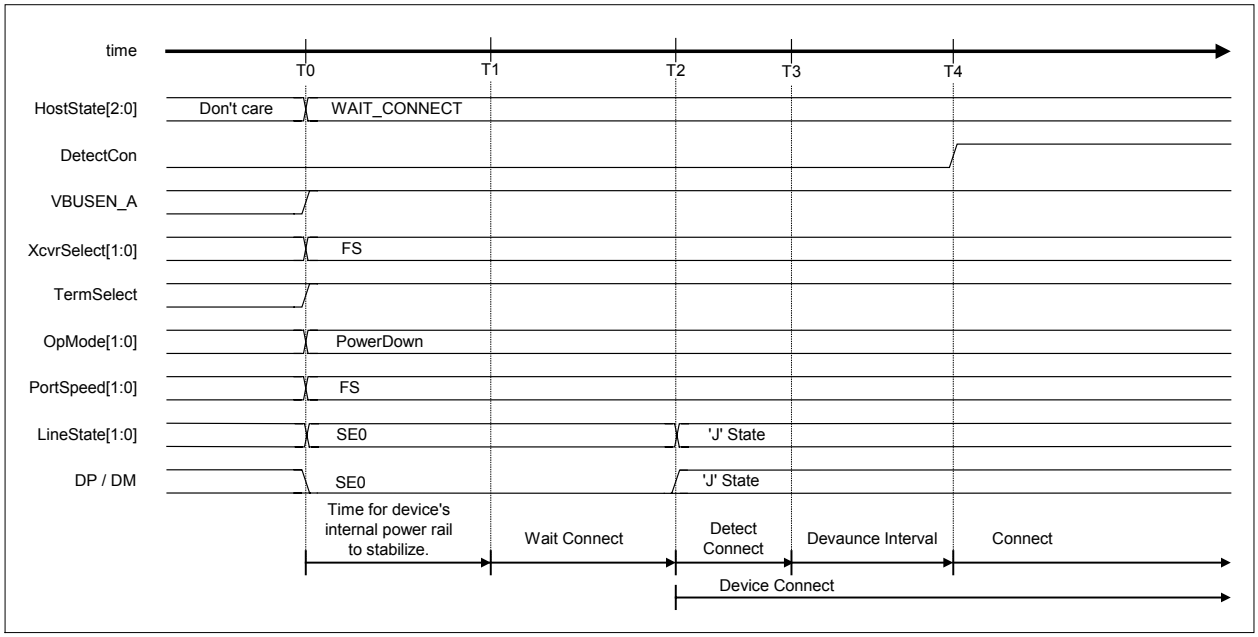


Figure 1-48 Device attach timing (FS mode)

Table 1-40 Device attach timing values (FS mode)

Timing Parameter	Description	Value
T0	Sets H_NegoControl_0.AutoMode to “GoWAIT_CONNECT.” (F/W)	0 (reference)
T1	Switches on the connect detection function. (H/W)	T0+100ms{T_{SIGATT}} < T1
T2	The device is connected.	T2
T3	Switches on the disconnect detection function. (H/W)	T2+2.5us{T_{DCNN}} < T3
T4	Issues connect detection status (DetectCon). Switches off the disconnect and connect detection functions. (H/W)	T3+100ms{T_{ATTDB}} <T4

Note: Brackets {} indicate terms defined in the USB 2.0 standards.

1.3.10.3.2.2 If an LS Device Is Connected

Shown below is the procedure with an LS device connected. Steps (2) to (14) below are performed automatically by the LSI hardware.

(1) The firmware sets the host state change execute (H_NegoControl_0.AutoMode) to "GoWAIT_CONNECT" (T0).

(2) Sets the host state monitor (H_NegoControl_0.HostState) to "WAIT_CONNECT" (T0).

(3) Switches on VBUSEN_A (T0).

(4) Sets transceiver selection (H_XcvrControl.XcvrSelect) and terminal selection (H_XcvrControl.TermSelect) to FS mode (T0).

(5) Sets the operating mode (H_XcvrControl.OpMode[1:0]) to "PowerDown" (T0).

(6) Sets port speed (H_NegoControl_1.PortSpeed[1:0]) to "FS" (T0).

(7) Waits 100 ms for the device internal power supply to stabilize, then switches on the connect detection function (T1).

(8) The line state (H_USB_Status.LineState[1:0]) indication is "K" when an LS device is connected (T2).

(9) An LS device connection is assumed if the line state (H_USB_Status.LineState[1:0]) remains "K" for at least 2.5 μs (T3).

(10) Sets transceiver selection (H_XcvrControl.XcvrSelect[1:0]) to "LS" (T3). The line state (H_USB_Status.LineState[1:0]) polarity changes to LS, and the line state (H_USB_Status.LineState[1:0]) indication becomes "J."

(11) Sets the port speed (H_NegoControl_1.PortSpeed[1:0]) to "LS" (T3).

(12) Switches on the disconnect detection function (T3).

(13) Issues connect detection status (H_SIE_IntStat_0.DetectCon) if no disconnection is detected during the debounce interval (T4). If disconnection is detected during this period, the disconnect detection function is switched off, transceiver selection (H_XcvrControl.XcvrSelect) and port speed (H_NegoControl_1.PortSpeed[1:0]) are both set to "FS," and connect detection is repeated from step (8). Disconnect detection status (H_SIE_IntStat_0.DetectDiscon) is not issued.

(14) Switches off the disconnect and connect detection functions (T4).

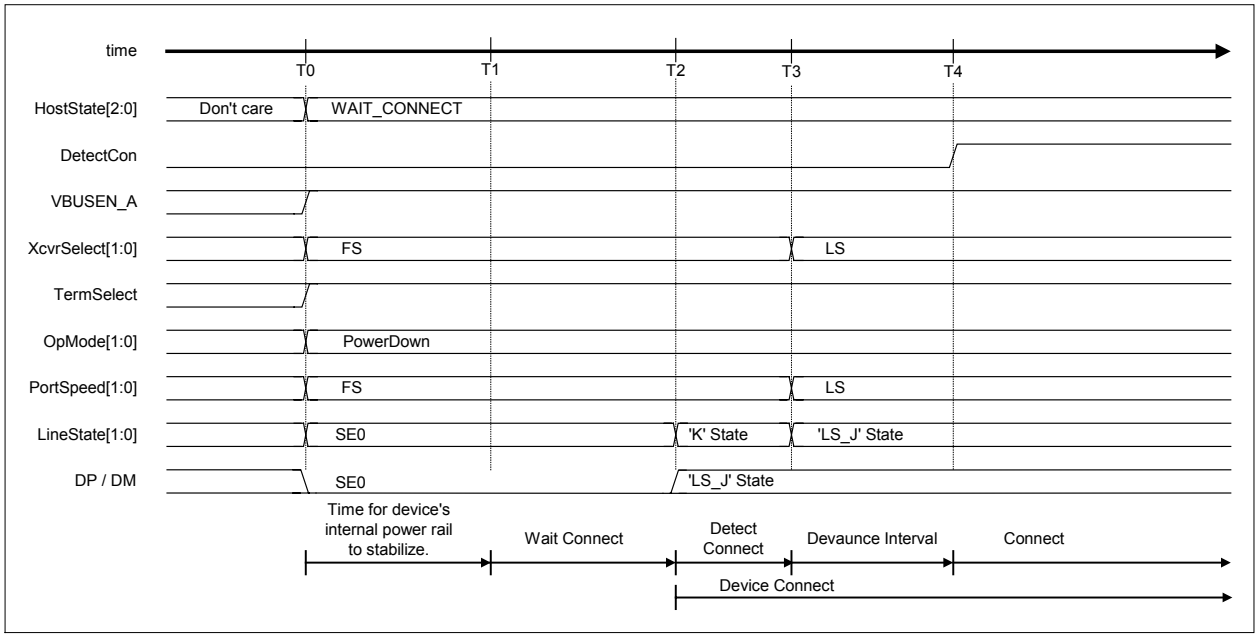


Figure 1-49 Device attach timing (LS mode)

Table 1-41 Device attach timing values (LS mode)

Timing Parameter	Description	Value
T0	Sets H_NegoControl_0.AutoMode to “GoWAIT_CONNECT.” (F/W)	0 (reference)
T1	Switches on the connect detection function. (H/W)	T0+100ms{T_{SIGATT}} < T1
T2	The device is connected.	T2
T3	Switches on the disconnect detection function. (H/W)	T2+2.5us{T_{DCNN}} < T3
T4	Issues connect detection status (DetectCon). (H/W) Switches off the disconnect and connect detection functions. (H/W)	T3+100ms{T_{ATTDB}} <T4

Note: Brackets {} indicate terms defined in the USB 2.0 standards.

1.3.10.3.3 GoDISABLED

The LSI hardware automatically performs the processing necessary to change to "DISABLED" when "GoDISABLED" is set to the host state change execute (H_NegoControl0.AutoMode).

The system enters this state when a connection is detected in a "WAIT_CONNECT" state, when an error device chirp is detected in a "RESET" state, or when a port error is detected in an "OPERATIONAL" state.

1.3.10.3.3.1 If an HS Device Is Connected

Shown below is the procedure with an HS device is connected. Steps (2) to (6) below are performed automatically by the LSI hardware.

(1) The firmware sets the host state change execute (H_NegoControl_0.AutoMode) to "GoDISABLED" (T0).

(2) Sets the host state monitor (H_NegoControl_0.HostState) to "DISABLED" (T0).

(3) Switches off the disconnect detection function (T0).

(4) Waits for the transaction currently being executed to end, sets the transceiver selection (H_XcvrControl.XcvrSelect), terminal selection (H_XcvrControl.TermSelect), and port speed (H_NegoControl_1.PortSpeed) to "FS" mode, then sets the operating mode (H_XcvrControl.OpMode[1:0]) to "PowerDown" (T1).

(5) Switches on the disconnect detection function (T3).

(6) Issues disabled change completion status (H_SIE_IntStat_1.DisabledCmp) to the firmware (T3).

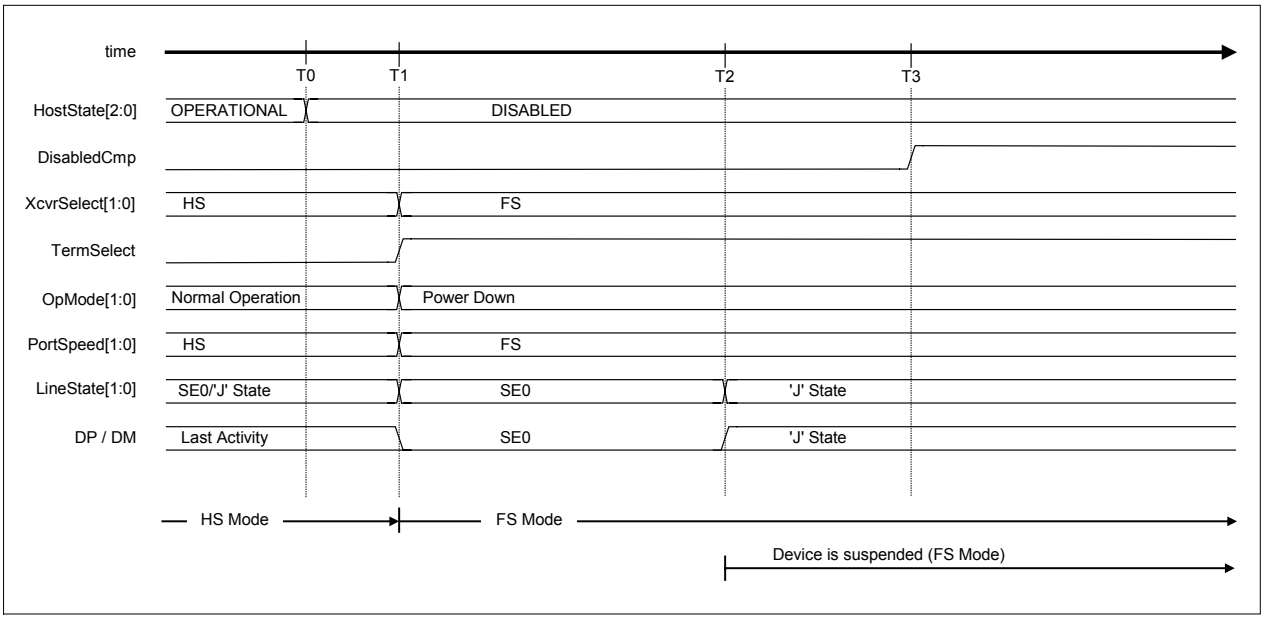


Figure 1-50 Disabled timing (HS mode)

Table 1-42 Disabled timing values (HS mode)

Timing Parameter	Description	Value
T0	Sets H_NegoControl_0.AutoMode to “GoDISABLED.” (F/W) Switches off the disconnect detection function. (H/W)	0 (reference)
T1	Sets the transceiver selection and terminal selection port speed to “FS” mode after the completion of the last bus activity and sets the operating mode (H_XcvrControl.OpMode[1:0]) to “PowerDown.” (H/W)	T1
T2	The device detects Suspend and switches to FS mode.	T1 + 3.0ms < T2 {T_{WTREV}} < T1 + 3.125ms
T3	Switches on the disconnect detection function. (H/W) Issues disabled change completion status. (H/W)	T1 + 4ms < T3

Note: Brackets {} indicate terms defined in the USB 2.0 standards.

1.3.10.3.3.2 If an FS Device Is Connected

Shown below is the procedure with an FS device connected. Steps (2) to (6) below are performed automatically by the LSI hardware.

(1) The firmware sets the host state change execute (H_NegoControl_0.AutoMode) to “GoDISABLED” (T0).

(2) Sets the host state monitor (H_NegoControl_0.HostState) to “DISABLED” (T0).

(3) Switches off the disconnect detection function (T0).

(4) Waits for the transaction currently being executed to end, then sets the operating mode (H_XcvrControl.OpMode[1:0]) to “PowerDown” (T1).

(5) Switches on the disconnect detection function (T3).

(6) Issues disabled change completion status (H_SIE_IntStat_1.DisabledCmp) to the firmware (T3).

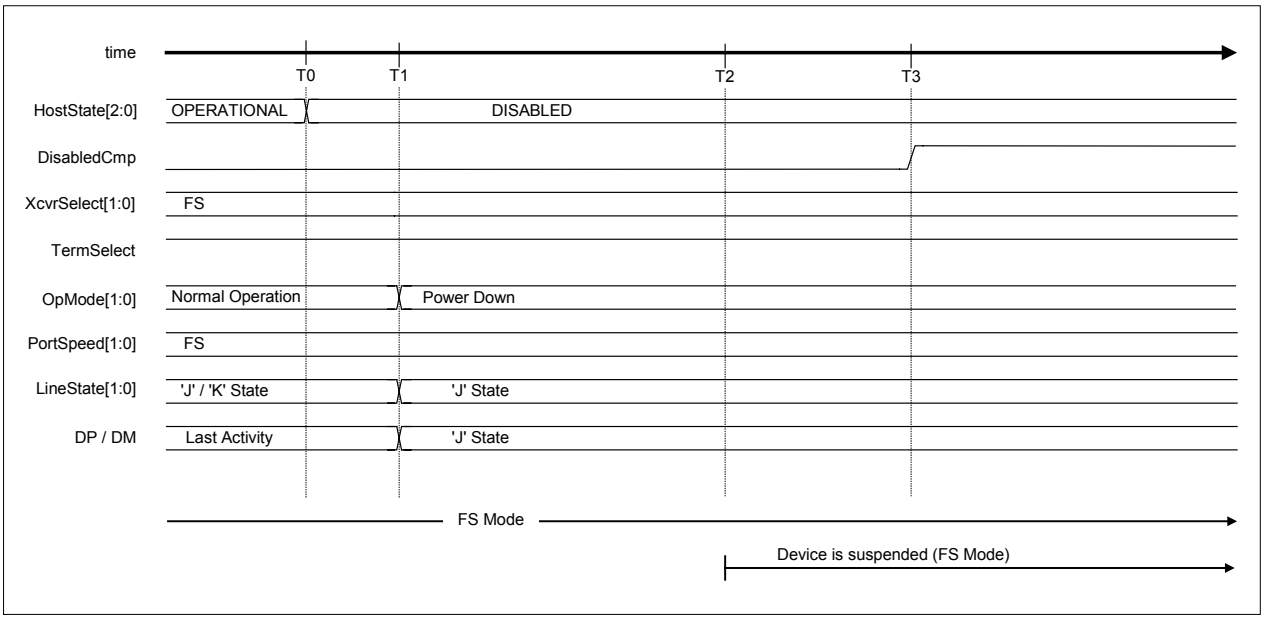


Figure 1-51 Disabled timing (FS mode)

Table 1-43 Disabled timing values (FS mode)

Timing Parameter	Description	Value
T0	Sets H_NegoControl_0.AutoMode to “GoDISABLED.” (F/W) Switches off the disconnect detection function. (H/W)	0 (reference)
T1	Sets the operating mode (H_XcvrControl.OpMode[1:0]) to “PowerDown” after the completion of the last bus activity. (H/W)	T1
T2	The device detects Suspend.	T1 + 3.0ms < T2 {T_{WTREV}} < T1 + 3.125ms
T3	Switches on the disconnect detection function. Issues disabled change completion status. (H/W)	T1 + 4ms < T3

Note: Brackets {} indicate terms defined in the USB 2.0 standards.

1.3.10.3.3.3 If an LS Device Is Connected

Shown below is the procedure with an LS device connected. Steps (2) to (6) below are performed automatically by the LSI hardware.

(1) The firmware sets the host state change execute (H_NegoControl_0.AutoMode) to “GoDISABLED” (T0).

(2) Sets the host state monitor (H_NegoControl_0.HostState) to “DISABLED” (T0).

(3) Switches off the disconnect detection function (T0).

(4) Waits for the transaction currently being executed to end, then sets the operating mode (H_XcvrControl.OpMode[1:0]) to “PowerDown” (T1).

(5) Switches on the disconnect detection function (T3).

(6) Issues disabled change completion status (H_SIE_IntStat_1.DisabledCmp) to the firmware (T3).

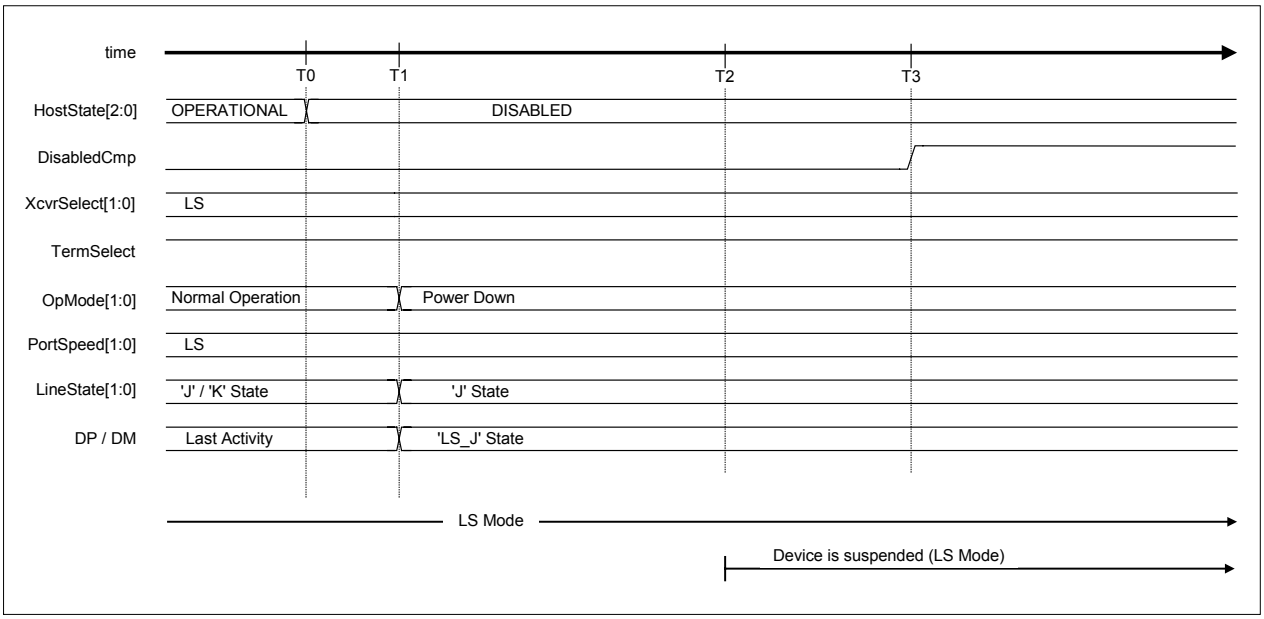


Figure 1-52 Disabled timing (LS mode)

Table 1-44 Disabled timing values (LS mode)

Timing Parameter	Description	Value
T0	Sets H_NegoControl_0.AutoMode to “GoDISABLED.” (F/W) Switches off the disconnect detection function. (H/W)	0 (reference)
T1	Sets the operating mode (H_XcvrControl.OpMode[1:0]) to “PowerDown” after the completion of the last bus activity. (H/W)	T1
T2	The device detects Suspend.	T1 + 3.0ms < T2 {T_{WTREV}} < T1 + 3.125ms
T3	Switches on the disconnect detection function. Issues disabled change completion status. (H/W)	T1 + 4ms < T3

Note: Brackets {} indicate terms defined in the USB 2.0 standards.

1.3.10.3.4 GoRESET

The LSI hardware automatically performs the processing necessary to change to "RESET" when "GoRESET" is set to the host state change execute (H_NegoControl_0.AutoMode). When changing to this state from "OPERATIONAL," the hardware waits for the transaction currently being executed to end before starting "RESET" processing.

1.3.10.3.4.1 HResetting an HS Device

Shown below is the procedure for resetting an HS device. Steps (2) to (14) below are performed automatically by the LSI hardware.

(1) The firmware sets the host state change execute (H_NegoControl_0.AutoMode) to "GoRESET" (T0).

(2) Sets the host state monitor (H_NegoControl_0.HostState) to "RESET" (T0).

(3) Sets the transceiver selection (H_XcvrControl.XcvrSelect) and terminal selection (H_XcvrControl.TermSelect) to "HS" mode (T0).

(4) Sets the operating mode (H_XcvrControl.OpMode[1:0]) to "Normal" (T0).

(5) Switches on the device chirp detection function (T0).

(6) The device chirp is recognized if continuous activity ("J" state) is detected in the line state (H_USB_Status.LineState[1:0]) for at least 2.5 µs. A device chirp normal detection status (H_SIE_IntStat_0.DetectDevChirpOK) is then issued when the device chirp ends within the specified time after the reset starts (line state (H_USB_Status.LineState[1:0]) changes to no activity (treated as "SE0")) (T2).

The hardware determines a normal chirp signal has been received from the device if the device chirp normal detection status (H_SIE_IntStat_0.DetectDevChirpOK) is set to "1," then performs the following processing. Thus, the device chirp normal detection status (H_SIE_IntStat_0.DetectDevChirpOK) must always be cleared to "0" if the HS device is disconnected. If an FS device is connected while the device chirp normal detection status (H_SIE_IntStat_0.DetectDevChirpOK) is set to "1" without being cleared, subsequent processing will be performed with the device incorrectly recognized as an HS device.

(7) Switches off the device chirp detection function (T2).

(8) The host starts to output "Chirp K" after the device chirp ends (T3).

(9) The host outputs switching from "Chirp K" to "Chirp J" (T4).

(10) The host outputs switching from "Chirp J" to "Chirp K" (T5). The host then outputs the "Chirp K" and "Chirp J" sequence in alternation.

(11) The device switches to HS mode on detecting the host chirp (T6). The change in chirp height after point T7 indicates that device HS termination is enabled. The chirp is approximately 800 mV when the device is in FS mode and approximately 400 mV when in HS mode.

(12) The host ends the chirp (T8).

(13) Reset is ended (T9).

(14) Issues reset completion status (H_SIE_IntStat_1.ResetCmp) (T9).

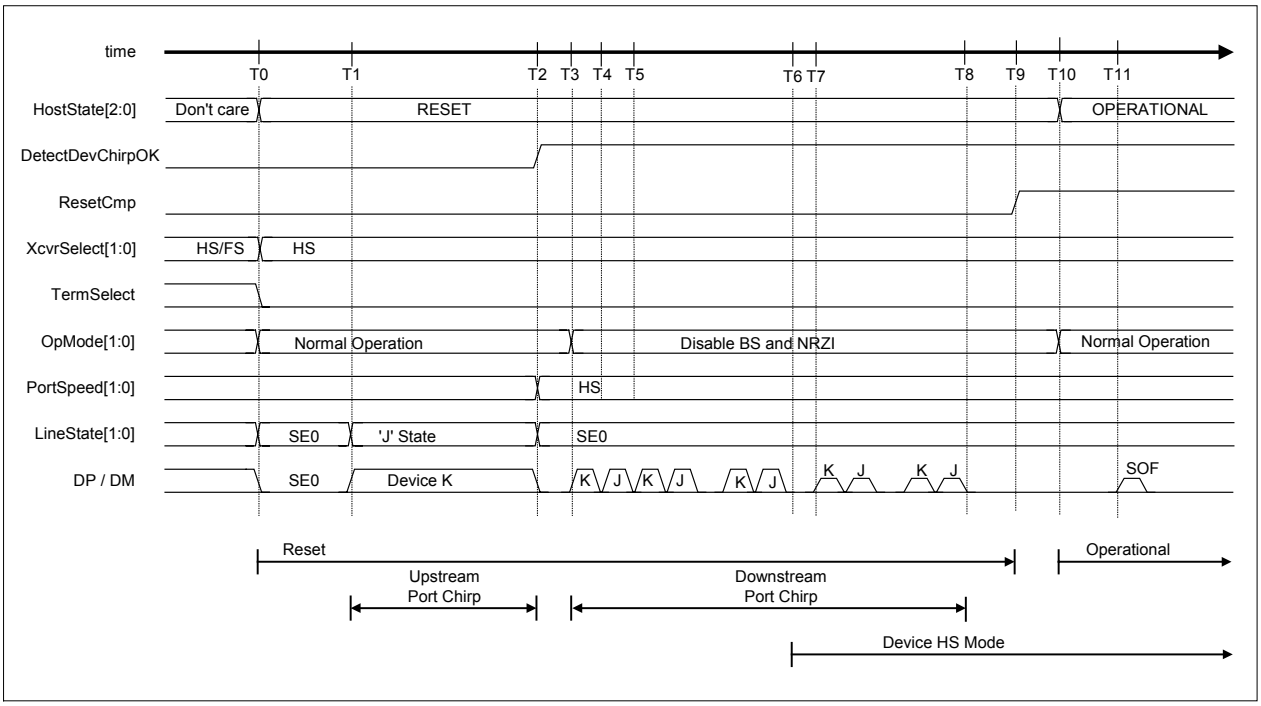


Figure 1-53 Reset timing (HS mode)

Table 1-45 Reset timing values (HS mode)

Timing Parameter	Description	Value
T0	Sets H_NegoControl_0.AutoMode to “GoRESET.” (F/W) Switches on the device chirp detection function. (H/W)	0 (reference)
T1	The device begins chirping.	T0 < T1 < T0 + 6.0ms
T2	The device stops chirping. Sets the port speed to HS. Switches off the device chirp detection function. Issues device chirp normal detection status (DetectDevChirpOK). (H/W)	T1 + 1.0ms {T_{UCH}} < T2 < T0 + 7.0ms {T_{UCHEND}}
T3	The host outputs the first chirp (“Chirp K”). (H/W)	T2 < T3 < T2 + 100us {T_{WTDCH}}
T4	The host outputs switching from “Chirp K” to “Chirp J.” (H/W)	T3 + 40us {T_{DCHBIT}} < T4 < T3 + 60us {T_{DCHBIT}}
T5	The host outputs switching from “Chirp J” to “Chirp K.” (H/W)	T4 + 40us {T_{DCHBIT}} < T5 < T4 + 60us {T_{DCHBIT}}
T6	The device detects the host chirp.	T6
T7	The device switches to HS mode.	T6 < T7 < T6 + 500us
T8	The host ends the chirp. (H/W)	T3+ 50ms{T_{DRSTR}} < T8
T9	The reset is completed. Issues reset completion status (ResetCmp). (H/W)	T8 < T9 <T8 + 150us
T10 (reference)	Sets H_NegoControl_0.AutoMode to “GoOPERATIONAL.” (F/W)	T10 < T9 + 200us
T11 (reference)	Issues the first SOF. (H/W)	T10 + 120us < T11 < T10 + 130us T8 + 100us{T_{DCHSE0}} < T11 < T8 + 500us {T_{DCHSE0}}

Note: Brackets {} indicate terms defined in the USB 2.0 standards.

1.3.10.3.4.2 Device Error Chirp Detection

The operation is indicated if a device error chirp is detected in the HS Detection Handshake. One of two operating modes can be selected, depending on the chirp completion disable (H_NegoControl_1.DisChirpFinish) setting.

1.3.10.3.4.2.1 If Chirp Completion Disable (H_NegoControl_1.DisChirpFinish) Setting Is 0

Host chirping is not performed after error detection. If a device chirp error detection status is issued, the firmware waits for the reset completion status (H_SIE_IntStat_1.ResetCmp) to be issued, then sets the host state change execute (H_NegoControl_0.AutoMode) to "GoDISABLED" before changing the host state to "DISABLED." Steps (2) to (9) below are performed automatically by the LSI hardware.

(1) The firmware sets the host state change execute (H_NegoControl_0.AutoMode) to "GoRESET" (T0).

(2) Sets the host state monitor (H_NegoControl_0.HostState) to "RESET" (T0).

(3) Sets the transceiver selection (H_XcvrControl.XcvrSelect) and terminal selection (H_XcvrControl.TermSelect) to HS mode (T0).

(4) Sets the operating mode (H_XcvrControl.OpMode[1:0]) to "Normal" (T0).

(5) Switches on the device chirp detection function (T0).

(6) The device chirp is recognized if continuous activity ("J" state) is detected in the line state (H_USB_Status.LineState[1:0]) for at least 2.5 µs. However, an error is assumed if the device chirp does not end within the specified time after a reset starts, and a device chirp error detection status (H_SIE_IntStat_0.DetectDevChirpNG) is issued (T2).

(7) Switches off the device chirp detection function (T2).

(8) Reset is ended (T3).

(9) Issues reset completion status (H_SIE_IntStat_1.ResetCmp) (T3).

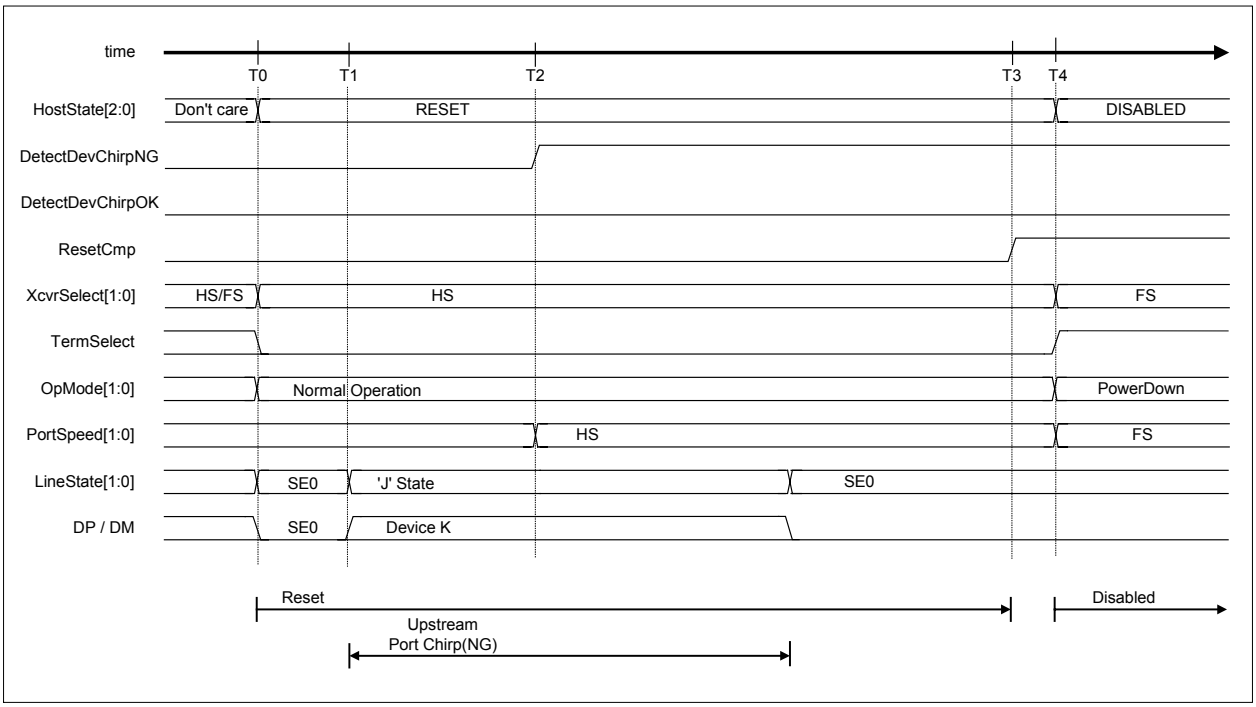


Figure 1-54 Detect device chirp NG timing (with chirp completion disable set to 0)

Table 1-46 Detect device chirp timing values (with chirp completion disable set to 0)

Timing Parameter	Description	Value
T0	Sets H_NegoControl_0.AutoMode to “GoRESET.” (F/W) Switches on the device chirp detection function. (H/W)	0 (reference)
T1	The device begins chirping.	T0 < T1 < T0 + 6.0ms
T2	Issues device chirp error detection status (DetectDevChirpNG). Switches off the device chirp detection function. (H/W)	T0 + 7ms{T_{UCHEND}} < T2
T3	Reset ends. Issues reset completion status (ResetCmp). (H/W)	T2 + 50ms{T_{DRSTR}}< T3
T4 (reference)	Sets H_NegoControl_0.AutoMode to “GoDISABLED.” (F/W)	T4

Note: Brackets {} indicate terms defined in the USB 2.0 standards.

1.3.10.3.4.2.2 If Chirp Completion Disable (H_NegoControl_1.DisChirpFinish) Setting is 1

Host chirping is performed once device chirping is complete after error detection.

To change the host state to "DISABLED" when using this mode without waiting for the reset completion status (H_SIE_IntStat_1.ResetCmp) to be issued, halt the operation for the state currently being executed by writing 0x80 to the H_NegoControl_0 register ("1" to H_NegoControl_0.AutoModeCancel and "0x0" to H_NegoControl_0.AutoMode). Since the value of the H_NegoControl_0.AutoModeCancel bit becomes "0" when stop processing is complete (requires approximately 6 cycles with a 60 MHz clock), confirm that the H_NegoControl_0.AutoModeCancel bit has changed to "0" before writing 0x03 to the register (i.e., set the host state change execute (H_NegoControl_0.AutoMode) to "GoDISABLED").

Steps (2) to (15) below are performed automatically by the LSI hardware.

(1) The firmware sets the host state change execute (H_NegoControl_0.AutoMode) to "GoRESET" (T0).

(2) Sets the host state monitor (H_NegoControl_0.HostState) to "RESET" (T0).

(3) Sets the transceiver selection (H_XcvrControl.XcvrSelect) and terminal selection (H_XcvrControl.TermSelect) to HS mode (T0).

(4) Sets the operating mode (H_XcvrControl.OpMode[1:0]) to "Normal" (T0).

(5) Switches on the device chirp detection function (T0).

(6) The device chirp is recognized if continuous activity ("J" state) is detected in the line state (H_USB_Status.LineState[1:0]) for at least 2.5 µs. However, an error is assumed if the device chirp does not end within the specified time after a reset starts, and a device chirp error detection status (H_SIE_IntStat_0.DetectDevChirpNG) is issued (T2).

(7) Switches off the device chirp detection function (T2).

(8) A device chirp normal detection status (H_SIE_IntStat_0.DetectDevChirpOK) is issued when the device chirp ends (line state (H_USB_Status.LineState[1:0]) changes to no activity (treated as "SE0")) (T3).

(9) The host starts to output "Chirp K" after the device chirp ends (T4).

(10) The host outputs switching from "Chirp K" to "Chirp J" (T5).

(11) The host outputs switching from “Chirp J” to “Chirp K” (T6). The host then outputs the “Chirp K” and “Chirp J” sequence in alternation.

(12) The device switches to HS mode on detecting the host chirp (T7).

The change in chirp height after point T8 indicates that device HS termination is enabled. The chirp is approximately 800 mV when the device is in FS mode and approximately 400 mV when in HS mode.

(13) The host ends the chirp (T9).

(14) Reset is ended (T10).

(15) Issues reset completion status (H_SIE_IntStat_1.ResetCmp) (T10).

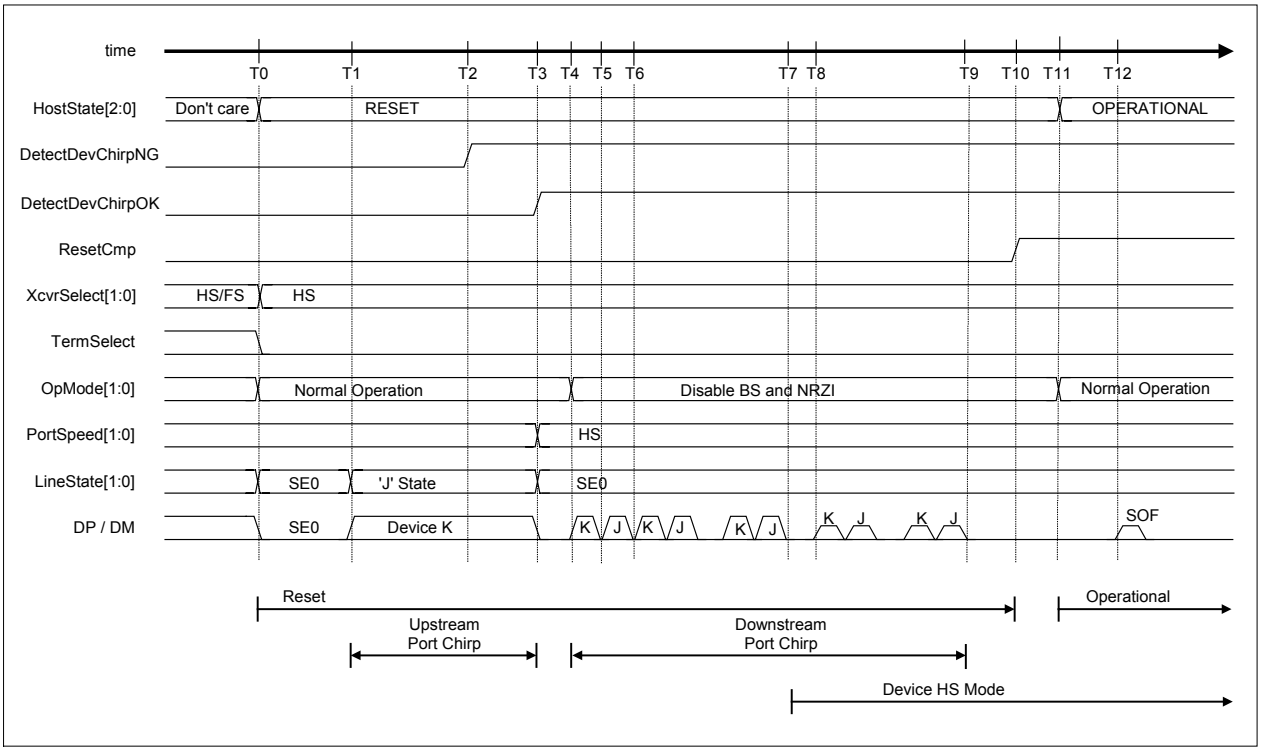


Figure 1-55 Detect device chirp NG timing (with chirp completion disable set to 1)

Table 1-47 Detect device chirp timing values (with chirp completion disable set to 1)

Timing Parameter	Description	Value
T0	Sets H_NegoControl_0.AutoMode to “GoRESET.” (F/W) Switches on the device chirp detection function. (H/W)	0 (reference)
T1	The device begins chirping.	T0 < T1 < T0 + 6.0ms
T2	Issues device chirp error detection status (DetectDevChirpNG). Switches off the device chirp detection function. (H/W)	T0 + 7ms{T_{UCHEND}} < T2
T3	The device stops chirping. Sets the port speed to HS. Issues device chirp normal detection status (DetectDevChirpOK). (H/W)	T3
T4	The host outputs the first chirp (“Chirp K”). (H/W)	T3< T4 < T3 + 100us {T_{WTDCH}}
T5	The host outputs switching from “Chirp K” to “Chirp J.” (H/W)	T4 + 40us {T_{DCHBIT}} < T45 T4 + 60us {T_{DCHBIT}}
T6	The host outputs switching from “Chirp J” to “Chirp K.” (H/W)	T5 + 40us {T_{DCHBIT}} < T6 < T5 + 60us {T_{DCHBIT}}
T7	The device detects the host chirp.	T7
T8	The device switches to HS mode.	T7 < T8 < T6 + 500us
T9	The host ends the chirp. (H/W)	T4 + 50ms{T_{DRSTR}} < T9
T10	Reset is ended. Issues reset completion status (ResetCmp). (H/W)	T9 < T10 <T9 + 150us
T11 (reference)	Sets H_NegoControl0.AutoMode to “GoOPERATIONAL.” (F/W)	T11< T10 + 200us
T12 (reference)	Issues the first SOF. (H/W)	T11 + 120us < T12 < T11 + 130us T9+ 100us{T_{DCHSE0}} < T12 < T9 + 500us {T_{DCHSE0}}

Note: Brackets {} indicate terms defined in the USB 2.0 standards.

1.3.10.3.4.3 Resetting an FS Device

Shown below is the procedure for resetting an FS device. Steps (2) to (9) below are performed automatically by the LSI hardware.

(1) The firmware sets the host state change execute (H_NegoControl_0.AutoMode) to “GoRESET” (T0).

(2) Sets the host state monitor (H_NegoControl_0.HostState) to “RESET” (T0).

(3) Sets the transceiver selection (H_XcvrControl.XcvrSelect) and terminal selection (H_XcvrControl.TermSelect) to HS mode (T0).

(4) Sets the operating mode (H_XcvrControl.OpMode[1:0]) to “Normal” (T0).

(5) Switches on the device chirp detection function (T0).

(6) Since the port speed (H_NegoControl_1.PortSpeed[1:0]) is HS/FS, the connected device is identified as an FS device without detecting a device chirp. Transceiver selection (H_XcvrControl.XcvrSelect) and port speed (H_NegoControl_1.PortSpeed[1:0]) are set to “FS” (T1).

(7) Switches off the device chirp detection function (T1).

(8) Sets terminal selection (H_XcvrControl.TermSelect) to “FS” (T2).

(9) Issues reset completion status (H_SIE_IntStat_1.ResetCmp) (T3).

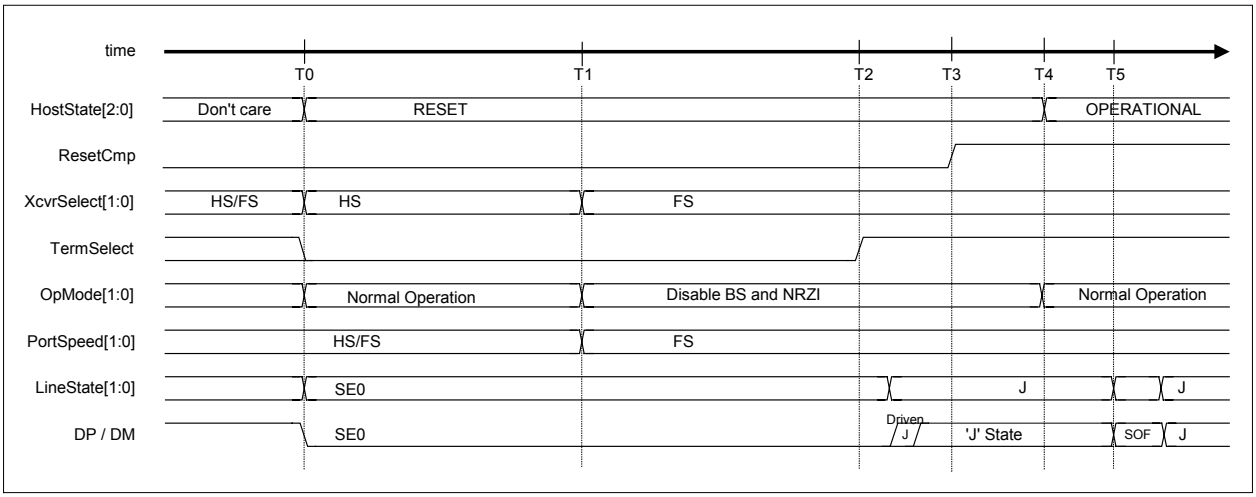


Figure 1-56 Reset timing (FS mode)

Table 1-48 Reset timing values (FS mode)

Timing Parameter	Description	Value
T0	Sets H_NegoControl_0.AutoMode to “GoRESET.” (F/W) Switches on the device chirp detection function. (H/W)	0 (reference)
T1	Sets the transceiver selection to “FS.” Sets the host speed to “FS.” Switches off the device chirp detection function. (H/W)	T0 + 7.0ms{T_{UCHEND}} < T1
T2	Sets the terminal selection to “FS. ” (H/W)	T0 + 50ms{T_{DRSTR}} < T2
T3	Issues reset completion status. (H/W)	T2 + 150us < T3
T4 (reference)	Sets H_NegoControl_0.AutoMode to “GoOPERATIONAL.” (F/W)	T4
T5 (reference)	Issues the first SOF. (H/W)	T4 + 0.9ms < T5 < T4 + 1.1ms (T5 < T2 + 3ms)

Note: Brackets {} indicate terms defined in the USB 2.0 standards.

1.3.10.3.4.4 Resetting an LS Device

Shown below is the procedure for resetting an LS device. Steps (2) to (7) below are performed automatically by the LSI hardware.

(1) The firmware sets the host state change execute (H_NegoControl_0.AutoMode) to “GoRESET” (T0).

(2) Sets the host state monitor (H_NegoControl_0.HostState) to “RESET” (T0).

(3) Sets the transceiver selection (H_XcvrControl.XcvrSelect) and terminal selection (H_XcvrControl.TermSelect) to HS mode (T0).

(4) Sets the operating mode (H_XcvrControl.OpMode[1:0]) to “Normal” (T0).

(5) Since the port speed (H_NegoControl_1.PortSpeed[1:0]) is LS, the transceiver selection (H_XcvrControl.XcvrSelect) is set to “LS” (T1).

(6) Sets the terminal selection (H_XcvrControl.TermSelect) to “FS” (T2).

(7) Issues reset completion status (H_SIE_IntStat_1.ResetCmp) (T3).

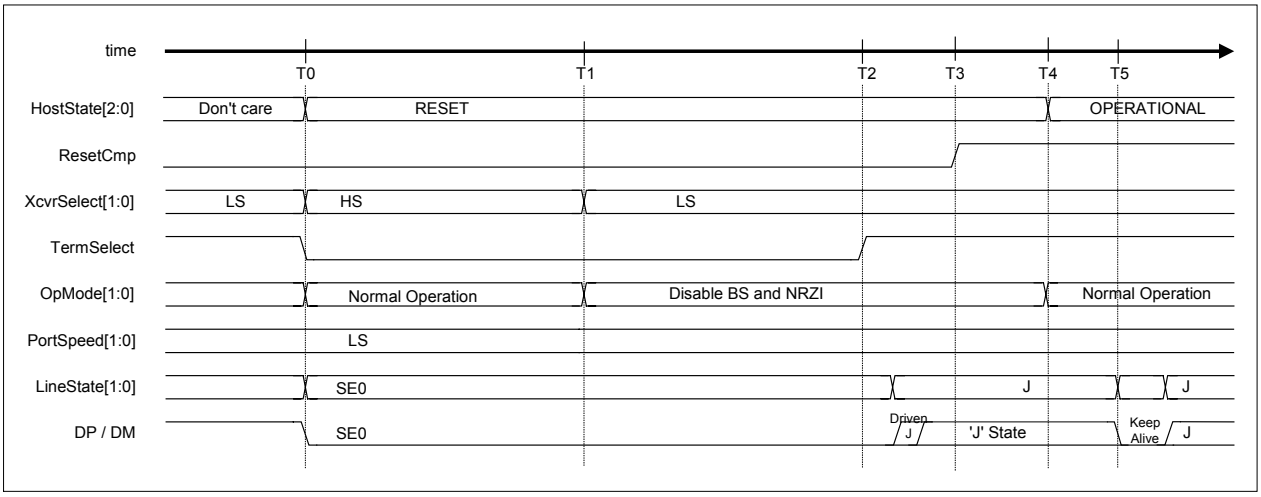


Figure 1-57 Reset timing (LS mode)

Table 1-49 Reset timing values (LS mode)

Timing Parameter	Description	Value
T0	Sets H_NegoControl_0.AutoMode to “GoRESET.” (F/W)	0 (reference)
T1	Sets the transceiver selection to “LS.” (H/W)	T0 + 7.0ms{T_{UCHEND}} < T1
T2	Sets the terminal selection to “FS.” (H/W)	T0+ 50ms{T_{DRSTR}} < T2
T3	Issues reset completion status. (H/W)	T2 + 150us < T3
T4 (reference)	Sets H_NegoControl_0.AutoMode to “GoOPERATIONAL.” (F/W)	T4
T5 (reference)	Issues the first KeepAlive. (H/W)	T4 + 0.9ms < T5 < T4 + 1.1ms (T5 < T2 + 3ms)

Note: Brackets {} indicate terms defined in the USB 2.0 standards.

1.3.10.3.5 GoOPERATIONAL

The LSI hardware automatically performs the processing necessary to change to "OPERATIONAL" when "GoOPERATIONAL" is set to host state change execute (H_NegoControl_0.AutoMode).

(1) The firmware sets the host state change execute (H_NegoControl_0.AutoMode) to "GoOPERATIONAL" (T0).

(2) Sets the host state monitor (H_NegoControl_0.HostState) to "OPERATION" (T0).

(3) Sets the operating mode (H_XcvrControl.OpMode[1:0]) to "Normal" to enable USB transactions (T0).

(4) Switches on the disconnect detection function (T0).

(5) Issues the first SOF if the port speed (H_NegoControl_1.PortSpeed[1:0]) is set to "HS" or "FS" (T1). Issues the first KeepAlive if set to "LS" (T1). Subsequent operations are performed in accordance with channel settings.

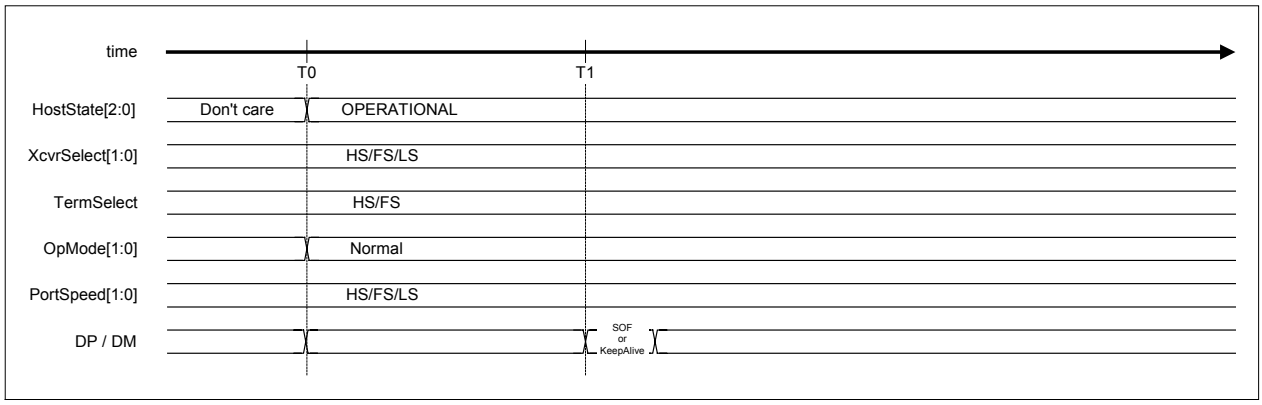


Figure 1-58 GoOPERATIONAL timing

Table 1-50 GoOPERATIONAL timing values

Timing Parameter	Description	Value
T0	Sets H_NegoControl_0.AutoMode to "GoOPERATIONAL." (F/W)	0 (reference)
T1	Issues the first SOF (HS/FS) or KeepAlive (LS).	T0+120us < T1(HS) < T0 + 130us T0+0.9ms <T1(FS,LS) < T0 + 1.1ms

1.3.10.3.6 GoSUSPEND

The LSI hardware automatically performs the processing necessary to change to "SUSPEND" when "GoSUSPEND" is set to host state change execute (H_NegoControl_0.AutoMode).

1.3.10.3.6.1 If HS Device Is Connected

Shown below is the procedure when an HS device is connected. Steps (2) to (7) below are performed automatically by the LSI hardware.

(1) The firmware sets the host state change execute (H_NegoControl_0.AutoMode) to "GoSUSPEND" (T0).

(2) Sets the host state monitor (H_NegoControl_0.HostState) to "SUSPEND" (T0).

(3) Switches off the disconnect detection and remote wakeup detection functions (T0).

(4) Waits for the transaction currently being executed to end, sets the transceiver selection (H_XcvrControl.XcvrSelect) and terminal selection (H_XcvrControl.TermSelect) to FS mode, then sets the operating mode (H_XcvrControl.OpMode[1:0]) to "PowerDown" (T1).

(5) Switches on the disconnect detection function (T3).

(6) Switches on the remote wakeup detection function if the remote wakeup receipt permission (H_NegoControl_1.RmtWkupDetEnb) is enabled (T3).

(7) Issues suspend change completion status (H_SIE_IntStat_1.SuspendCmp) to the firmware (T3).

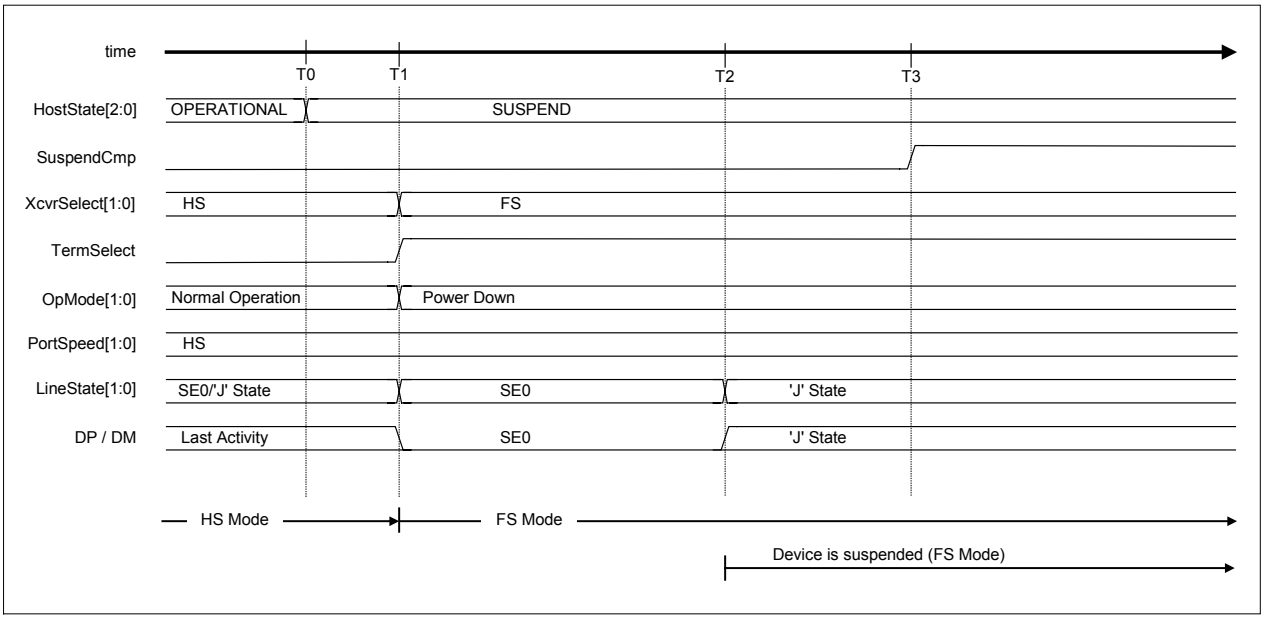


Figure 1-59 Suspend timing (HS mode)

Table 1-51 Suspend timing values (HS mode)

Timing Parameter	Description	Value
T0	Sets H_NegoControl_0.AutoMode to “GoSUSPEND.” (F/W) Switches off disconnect detection and remote wakeup detection functions. (H/W).	0 (reference)
T1	Sets the transceiver selection and terminal selection to “FS” mode after the completion of the last bus activity and sets the operating mode (H_XcvrControl.OpMode[1:0]) to “PowerDown.” (H/W)	T1
T2	The device detects Suspend and switches to FS mode.	T1 + 3.0ms < T2 {T_{WTREV}} < T1 + 3.125ms
T3	Switches on the disconnect detection function. Switches on the remote wakeup detection function if the remote wakeup receipt permission is enabled. Issues suspend change completion status. (H/W)	T1 + 5ms{T_{WTRSM}} < T3

Note: Brackets {} indicate terms defined in the USB 2.0 standards.

1.3.10.3.6.2 If FS Device Is Connected

Shown below is the procedure when an FS device is connected. Steps (2) to (7) below are performed automatically by the LSI hardware.

(1) The firmware sets the host state change execute (H_NegoControl_0.AutoMode) to “GoSUSPEND” (T0).

(2) Sets the host state monitor (H_NegoControl_0.HostState) to “SUSPEND” (T0).

(3) Switches off the disconnect detection and remote wakeup detection functions (T0).

(4) Waits for the transaction currently being executed to end, then sets the operating mode (H_XcvrControl.OpMode[1:0]) to “PowerDown” (T1).

(5) Switches on the disconnect detection function (T3).

(6) Switches on the remote wakeup detection function if remote wakeup receipt permission (H_NegoControl_1.RmtWkupDetEnb) is enabled (T3).

(7) Issues suspend change completion status (H_SIE_IntStat_1.SuspendCmp) to the firmware (T3).

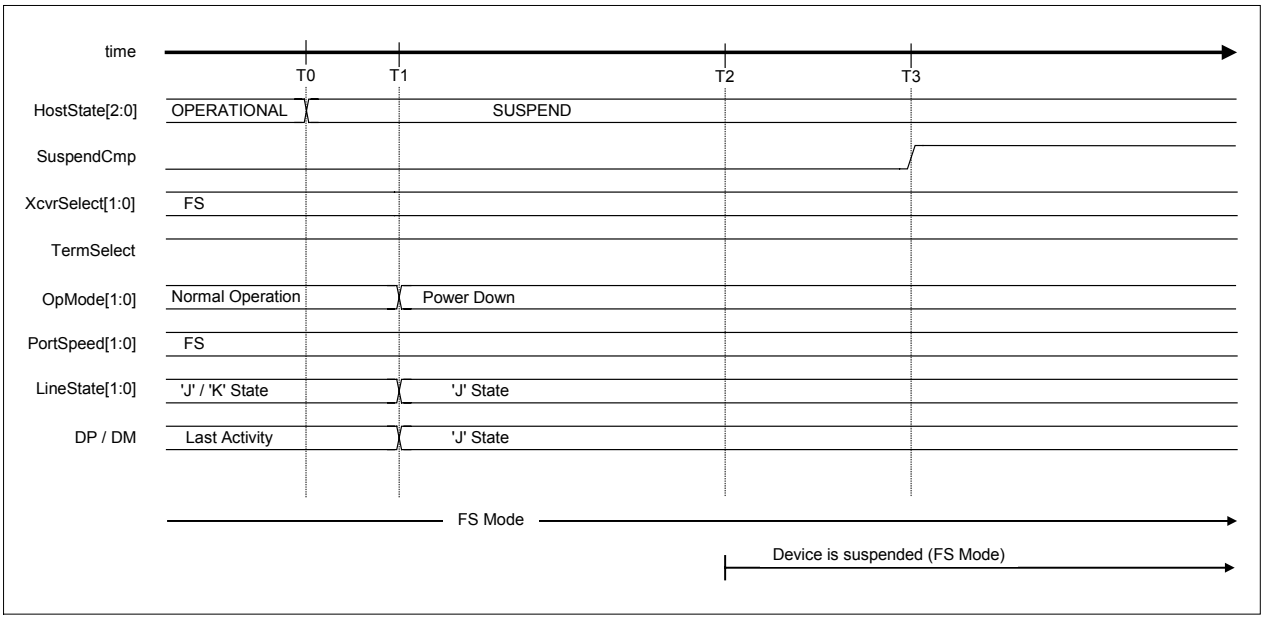


Figure 1-60 Suspend timing (FS mode)

Table 1-52 Suspend timing values (FS mode)

Timing Parameter	Description	Value
T0	Sets H_NegoControl_0.AutoMode to “GoSUSPEND.” (F/W) Switches off the disconnect detection and remote wakeup detection functions. (H/W).	0 (reference)
T1	Sets the operating mode (H_XcvrControl.OpMode[1:0]) to “PowerDown” after the completion of the last bus activity.	T1
T2	The device detects Suspend.	T1 + 3.0ms < T2 {T_{WTREV}} < T1 + 3.125ms
T3	Switches on the disconnect detection function. Switches on the remote wakeup detection function if the remote wakeup receipt permission is enabled. Issues suspend change completion status. (H/W)	T1 + 5ms{T_{WTRSM}} < T3

Note: Brackets {} indicate terms defined in the USB 2.0 standards.

1.3.10.3.6.3 If LS Device Is Connected

Shown below is the procedure when an LS device is connected. Steps (2) to (7) below are performed automatically by the LSI hardware.

(1) The firmware sets the host state change execute (H_NegoControl_0.AutoMode) to “GoSUSPEND” (T0).

(2) Sets the host state monitor (H_NegoControl0.HostState) to “SUSPEND” (T0).

(3) Switches off the disconnect detection and remote wakeup detection functions (T0).

(4) Waits for the transaction currently being executed to end, then sets the operating mode (H_XcvrControl.OpMode[1:0]) to “PowerDown” (T1).

(5) Switches on the disconnect detection function (T3).

(6) Switches on the remote wakeup detection function if the remote wakeup receipt permission (H_NegoControl_1.RmtWkupDetEnb) is enabled (T3).

(7) Issues suspend change completion status (H_SIE_IntStat_1.SuspendCmp) to the firmware (T3).

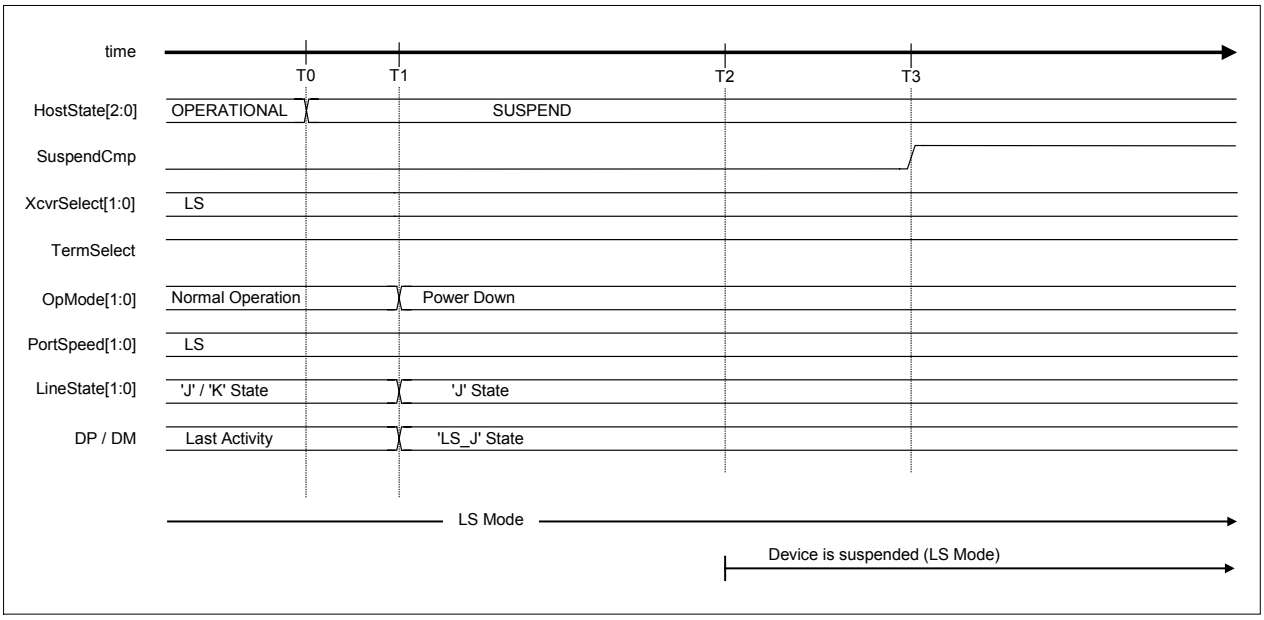


Figure 1-61 Suspend timing (LS mode)

Table 1-53 Suspend timing values (LS mode)

Timing Parameter	Description	Value
T0	Sets H_NegoControl_0.AutoMode to “GoSUSPEND.” (F/W) Switches off the disconnect detection and remote wakeup detection functions. (H/W).	0 (reference)
T1	Sets the operating mode (H_XcvrControl.OpMode[1:0]) to “PowerDown” after the completion of the last bus activity.	T1
T2	The device detects Suspend.	T1 + 3.0ms < T2 {T_{WTREV}} < T1 + 3.125ms
T3	Switches on the disconnect detection function. Switches on the remote wakeup detection function if the remote wakeup receipt permission is enabled. Issues suspend change completion status. (H/W)	T1 + 5ms{T_{WTRSM}} < T3

Note: Brackets {} indicate terms defined in the USB 2.0 standards.

1.3.10.3.7 GoRESUME

The LSI hardware automatically performs the processing necessary to change to "RESUME" when "GoRESUME" is set to the host state change execute (H_NegoControl0.AutoMode).

1.3.10.3.7.1 If an HS Device Is Connected

Shown below is the procedure when an HS device is connected. Steps (2) to (8) below are performed automatically by the LSI hardware.

(1) The firmware sets the host state change execute (H_NegoControl_0.AutoMode) to "GoRESUME" (T0).

(2) Sets the host state monitor (H_NegoControl_0.HostState) to "RESUME" (T0).

(3) Switches off the disconnect detection and remote wakeup detection functions (T0).

(4) Sets the operating mode (H_XcvrControl.OpMode) to "Disable BS and NRZI" and begins issuing the resume "K" signal (T0).

(5) Completes issuance of the resume "K" signal (T1).

(6) Sets the terminal selection (H_XcvrControl.TermSelect) to "HS" (T2).

(7) Sets the transceiver selection (H_XcvrControl.XcvrSelect) to "HS" (T3).

(8) Issues resume completion status (H_SIE_IntStat_1.ResumeCmp) to the firmware (T3).

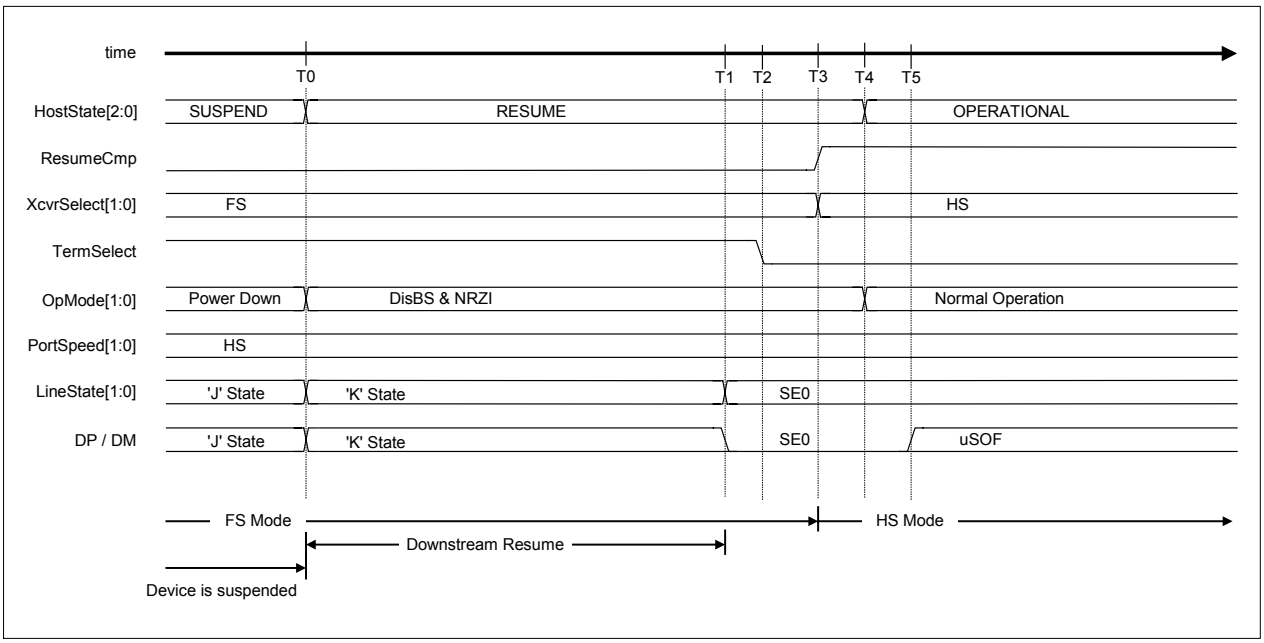


Figure 1-62 Resume timing (HS mode)

Table 1-54 Resume timing values (HS mode)

Timing Parameter	Description	Value
T0	Sets H_NegoControl_0.AutoMode to “GoRESUME.” (F/W) Switches off the disconnect detection and remote wakeup detection functions. Sets the operating mode to “Disable BS and NRZI” and begins issuing the resume “K” signal. (H/W)	0 (reference)
T1	Completes issuance of the resume “K” signal. Sets the terminal selection to “HS.” (H/W)	T0 + 20ms{T_{DRSMDN}}< T1
T2	Sets the transceiver selection to “HS.” (H/W)	T1 + 100ns < T2 < T1 + 2.0us
T3	Issues resume completion status (H_SIE_IntStat_1.ResumeCmp). (H/W)	T1 + 90us < T3 < T1 + 110us
T4 (reference)	Sets GoOPERATIONAL. (F/W) Sets the operating mode to “NormalOperation.” (H/W)	T4
T5 (reference)	Issues the first micro SOF. (H/W)	T5 < T1 + 3ms T4 + 120us < T5 < T4 + 130us

Note: Brackets {} indicate terms defined in the USB 2.0 standards.

1.3.10.3.7.2 If FS Device Is Connected

Shown below is the procedure when an FS device is connected. Steps (2) to (6) below are performed automatically by the LSI hardware.。

(1) The firmware sets the host state change execute (H_NegoControl_0.AutoMode) to “GoRESUME” (T0).

(2) Sets the host state monitor (H_NegoControl_0.HostState) to “RESUME” (T0).

(3) Switches off the disconnect detection and remote wakeup detection functions (T0).

(4) Sets the operating mode (H_XcvrControl.OpMode) to “Disable BS and NRZI” and begins issuing the resume “K” signal (T0).

(5) Completes issuance of the resume “K” signal (T1) and finally enables LS bit time EOP.

(6) Issues resume completion status (H_SIE_IntStat_1.ResumeCmp) to the firmware (T2).

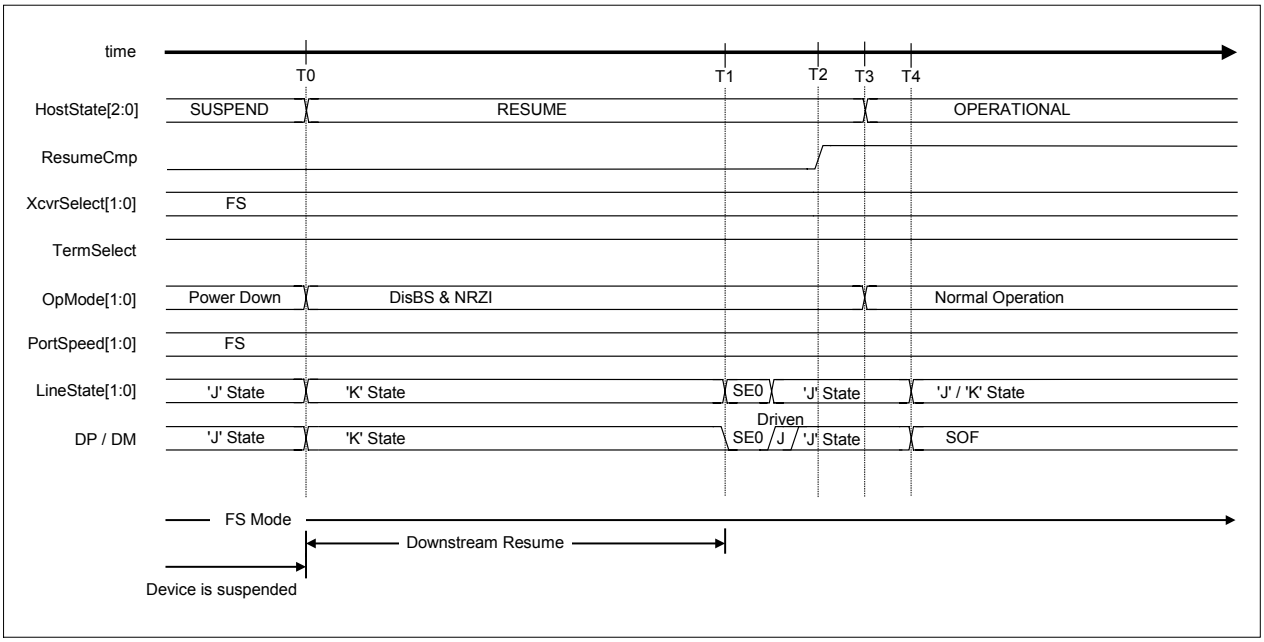


Figure 1-63 Resume timing (FS mode)

Table 1-55 Resume timing values (FS mode)

Timing Parameter	Description	Value
T0	Sets H_NegoControl_0.AutoMode to “GoRESUME.” (F/W) Switches off the disconnect detection and remote wakeup detection functions. Sets the operating mode to “Disable BS and NRZI” and begins issuing the resume “K” signal. (H/W)	0 (reference)
T1	Completes issuance of the resume “K” signal and finally enables the LS bit time EOP. (H/W)	T0 + 20ms{T_{DRSMDN}}< T1
T2	Issues resume completion status (H_SIE_IntStat_1.ResumeCmp). (H/W)	T1 + 90us < T2 < T1 + 110us
T3 (reference)	Sets GoOPERATIONAL. (F/W) Sets the operating mode to “NormalOperation.” (H/W)	T3
T4 (reference)	Issues the first SOF. (H/W)	T4 < T1 + 3ms T3 + 0.9ms < T4 < T3 + 1.1ms

Note: Brackets {} indicate terms defined in the USB 2.0 standards.

1.3.10.3.7.3 If an LS Device Is Connected

Shown below is the procedure when an LS device is connected. Steps (2) to (6) below are performed automatically by the LSI hardware.

(1) The firmware sets the host state change execute (H_NegoControl_0.AutoMode) to “GoRESUME” (T0).

(2) Sets the host state monitor (H_NegoControl_0.HostState) to “RESUME” (T0).

(3) Switches off the disconnect detection and remote wakeup detection functions (T0).

(4) Sets the operating mode (H_XcvrControl.OpMode) to “Disable BS and NRZI” and begins issuing the resume “K” signal (T0).

(5) Completes issuance of the resume “K” signal (T1), and finally enables LS bit time EOP.

(6) Issues resume completion status (H_SIE_IntStat_1.ResumeCmp) to the firmware (T2).

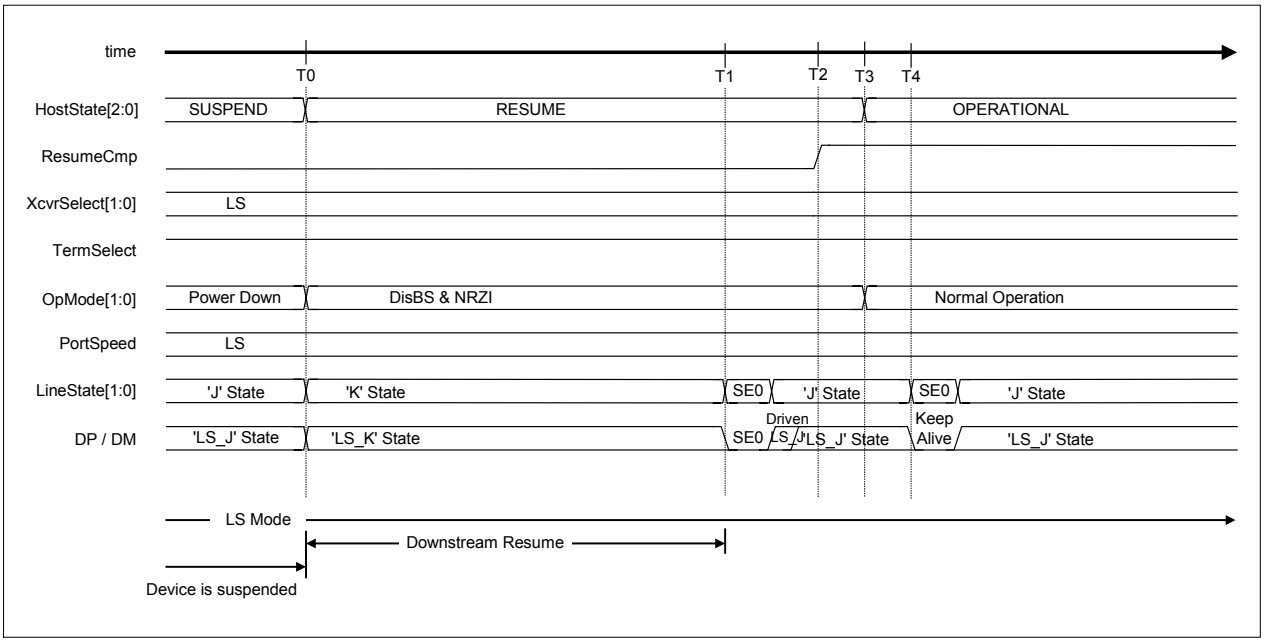


Figure 1-64 Resume timing (LS mode)

Table 1-56 Resume timing values (LS mode)

Timing Parameter	Description	Value
T0	Sets H_NegoControl_0.AutoMode to “GoRESUME.” (F/W) Switches off the disconnect detection and remote wakeup detection functions. Sets the operating mode to “Disable BS and NRZI” and begins issuing the resume “K” signal. (H/W)	0 (reference)
T1	Completes issuance of the resume “K” signal and finally enables the LS bit time EOP. (H/W)	T0 + 20ms{T_{DRSMDN}}< T1
T2	Issues resume completion status (H_SIE_IntStat_1.ResumeCmp). (H/W)	T1 + 90us < T2 < T1 + 110us
T3 (reference)	Sets GoOPERATIONAL. (F/W) Sets the operating mode to “NormalOperation.” (H/W)	T3
T4 (reference)	Issues the first KeepAlive. (H/W)	T4 < T1 + 3ms T3 + 0.9ms < T4 < T3 + 1.1ms

Note: Brackets {} indicate terms defined in the USB 2.0 standards.

1.3.10.3.8 GoWAIT_CONNECTtoDIS

The LSI hardware automatically performs the processing necessary from the "WAIT_CONNECT" state to "DISABLED" state when "GoWAIT_CONNECTtoDIS" is set to the host state change execute (H_NegoControl_0.AutoMode).

The procedure is shown below. Steps (2) to (5) below are performed automatically by the LSI hardware.

(1) The firmware sets the host state change execute (H_NegoControl_0.AutoMode) to "GoWAIT_CONNECTtoDIS" (T0).

(2) Performs the same processing as for "GoWAIT_CONNECT" (T0).

(3) Detects connection and issues connect detection status (H_SIE_IntStat_0.DetectCon) (T1).

(4) Performs the same processing as for "GoDISABLED" (T1).

(5) Issues disabled completion status (H_SIE_IntStat_1.DisabledCmp) (T2).

Note that the timing for each state is the same as for when executing "GoWAIT_CONNECT" and "GoDISABLED." See "1.3.10.3.2 GoWAIT_CONNECT" and "1.3.10.3.3 GoDISABLED," respectively, for timing details.

See "1.3.10.2.2 Disconnect Detection" and "1.3.10.2.1 VBUS Error Detection" for details of execution procedures and timing when an error (disconnection or VBUS error) is detected in midprocess.

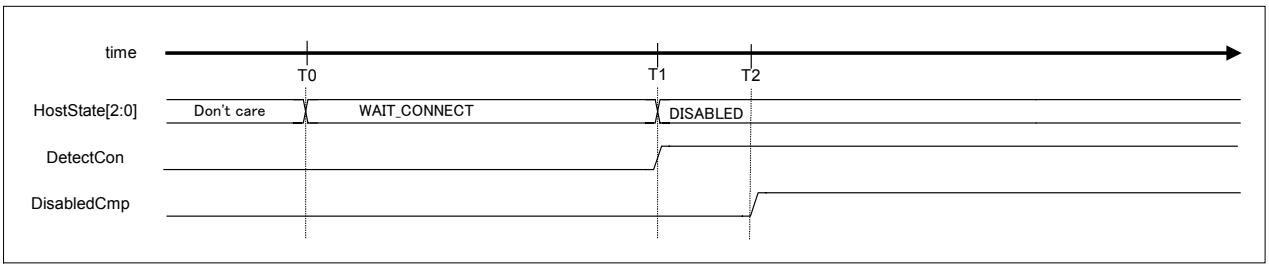


Figure 1-65 GoWAIT_CONNECTtoDIS timing (HS mode)

Table 1-57 GoWAIT_CONNECTtoDIS timing values (HS mode)

Timing Parameter	Description	Value
T0	Sets H_NegoControl_0.AutoMode to "GoWAIT_CONNECTtoDIS." (F/W) Performs the same processing as for "GoWAIT_CONNECT." (H/W)	0 (reference)
T1	Detects a connection and issues a connect detection status. Performs the same processing as for "GoDISABLED." (H/W)	T1
T2	Issues disabled completion status. (H/W)	T2

1.3.10.3.9 GoWAIT_CONNECTtoOP

The LSI hardware automatically performs the processing necessary to change from "WAIT_CONNECT" to "OPERATIONAL" states when "GoWAIT_CONNECTtoOP" is set to host state change execute (H_NegoControl_0.AutoMode).

1.3.10.3.9.1 If HS Device Is Connected

The following procedure is performed if an HS device is connected. Steps (2) to (9) below are performed automatically by the LSI hardware.

(1) The firmware sets the host state change execute (H_NegoControl_0.AutoMode) to "GoWAIT_CONNECTtoOP" (T0).

(2) Performs the same processing performed for "GoWAIT_CONNECT" (T0).

(3) Detects connection and issues connect detection status (H_SIE_IntStat_0.DetectCon) (T1).

(4) Performs the same processing performed for "GoDISABLED" (T1).

(5) Issues the disabled completion status (H_SIE_IntStat_1.DisabledCmp) (T2).

(6) Performs the same processing performed for "GoRESET" (T2).

(7) Detects the device chirp and issues the device chirp normal detection status (H_SIE_IntStat_0.DetectDevChirpOK) (T3).

(8) Issues the reset completion status (H_SIE_IntStat_1.ResetCmp) (T4).

(9) Performs the same processing performed for "GoOPERATIONAL" (T4).

Note that the timing for each state is the same as when executing "GoWAIT_CONNECT," "GoDISABLED," "GoRESET," and "GoOPERATIONAL." See "1.3.10.3.2 GoWAIT_CONNECT," "1.3.10.3.3 GoDISABLED," "1.3.10.3.4 GoRESET," and "1.3.10.3.5 GoOPERATIONAL," respectively, for timing particulars.

For detailed information on procedures and timing when errors (disconnection,VBUS error, or device chirp error) are detected in midprocess, see "1.3.10.2.2 Disconnect Detection" and "1.3.10.2.1 VBUS Error Detection."

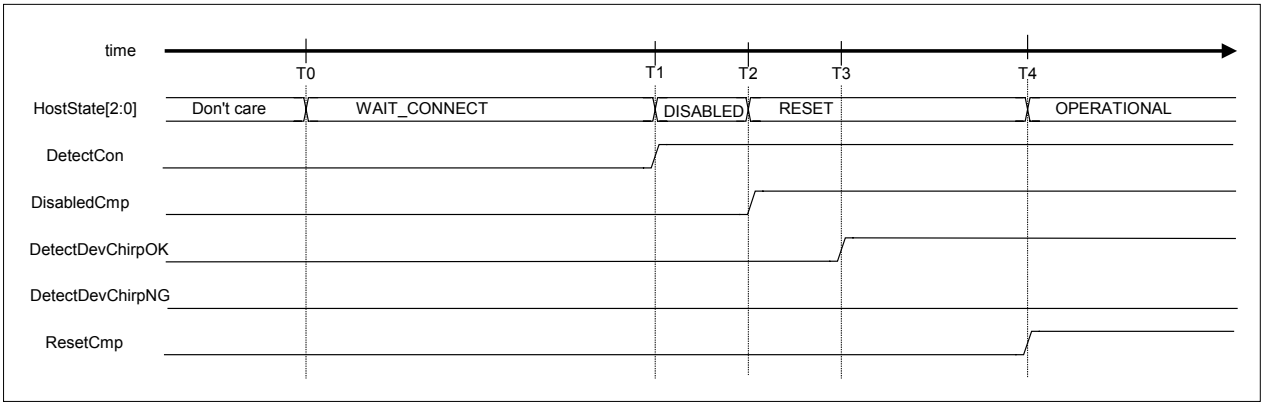


Figure 1-66 GoWAIT_CONNECTtoOP timing (HS mode)

Table 1-58 GoWAIT_CONNECTtoOP timing values (HS mode)

Timing Parameter	Description	Value
T0	Sets H_NegoControl_0.AutoMode to “GoWAIT_CONNECTtoOP.” (F/W) Performs the same processing performed for “GoWAIT_CONNECT.” (H/W)	0 (reference)
T1	Detects connection and issues the connect detection status. Performs the same processing performed for “GoDISABLED.” (H/W)	T1
T2	Issues the disabled completion status. Performs the same processing performed for “GoRESET.” (H/W)	T2
T3	Detects the device chirp and outputs the device chirp normal detection status. (H/W)	T3
T4	Issues the reset completion status. Performs the same processing performed for “GoOPERATIONAL.” (H/W)	T4

1.3.10.3.9.2 If FS or LS Device Is Connected

The following procedure is performed if an FS or LS device is connected. Steps (2) to (9) below are performed automatically by the LSI hardware.

(1) The firmware sets the host state change execute (H_NegoControl_0.AutoMode) to "GoWAIT_CONNECTtoOP" (T0).

(2) Performs the same processing performed for "GoWAIT_CONNECT" (T0).

(3) Detects connection and issues connect detection status (H_SIE_IntStat_0.DetectCon) (T1).

(4) Performs the same processing performed for "GoDISABLED" (T1).

(5) Issues the disabled completion status (H_SIE_IntStat_1.DisabledCmp) (T2).

(6) Performs the same processing performed for "GoRESET" (T2).

(7) Does not detect a device chirp and does not issue the device chirp normal/error detection status (H_SIE_IntStat_0.DetectDevChirpOK/NG) (T3).

(8) Issues the reset completion status (H_SIE_IntStat_1.ResetCmp) (T4).

(9) Performs the same processing performed for "GoOPERATIONAL" (T4).

Note that the timing for each state is the same as when executing "GoWAIT_CONNECT," "GoDISABLED," "GoRESET," and "GoOPERATIONAL." See "1.3.10.3.2 GoWAIT_CONNECT," "1.3.10.3.3 GoDISABLED," "1.3.10.3.4 GoRESET," and "1.3.10.3.5 GoOPERATIONAL," respectively, for timing particulars.

For detailed information on procedures and timing when errors (disconnection or VBUS error) are detected in midprocess, see "1.3.10.2.2 Disconnect Detection" and "1.3.10.2.1 VBUS Error Detection."

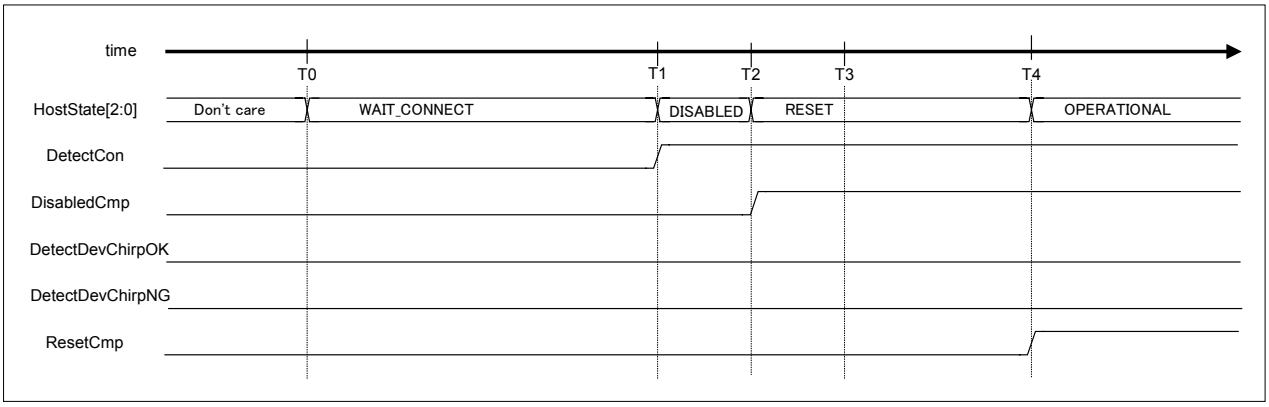


Figure 1-67 GoWAIT_CONNECTtoOP timing (FS or LS mode)

Table 1-59 GoWAIT_CONNECTtoOP timing values (FS or LS mode)

Timing Parameter	Description	Value
T0	Sets H_NegoControl_0.AutoMode to “GoWAIT_CONNECTtoOP.” (F/W) Performs the same processing performed for “GoWAIT_CONNECT.” (H/W)	0 (reference)
T1	Detects connection and issues the connect detection status. Performs the same processing performed for “GoDISABLED.” (H/W)	T1
T2	Issues the disabled completion status. Performs the same processing performed for “GoRESET.” (H/W)	T2
T3	Does not detect the device chirp and does not issue the device chirp normal/error detection status. (H/W)	T3
T4	Issues the reset completion status. Performs the same processing performed for “GoOPERATIONAL.” (H/W)	T4

1.3.10.3.10 GoRESETtoOP

The LSI hardware automatically performs the processing necessary to change from "RESET" to "OPERATIONAL" states when "GoRESETtoOP" is set to the host state change execute (H_NegoControl_0.AutoMode).

1.3.10.3.10.1 If HS Device Is Connected

The following procedure is performed if an HS device is connected. Steps (2) to (5) below are automatically performed by the LSI hardware.

(1) The firmware sets the host state change execute (H_NegoControl_0.AutoMode) to "GoRESETtoOP" (T0).

(2) Performs the same processing performed for "GoRESET" (T0).

(3) Detects the device chirp and issues the device chirp normal detection status (H_SIE_IntStat_0.DetectDevChirpOK) (T1).

(4) Issues the reset completion status (H_SIE_IntStat_1.ResetCmp) (T2).

(5) Performs the same processing performed for "GoOPERATIONAL" (T2).

Note that the timing for each state is the same as when executing "GoRESET" and "GoOPERATIONAL." See "1.3.10.3.4 GoRESET" and "1.3.10.3.5 GoOPERATIONAL," respectively, for timing particulars.

For detailed information on procedures and timing when errors (VBUS error, or device chirp error) are detected in midprocess, see "1.3.10.2.2 Disconnect Detection" and "1.3.10.2.1 VBUS Error Detection."

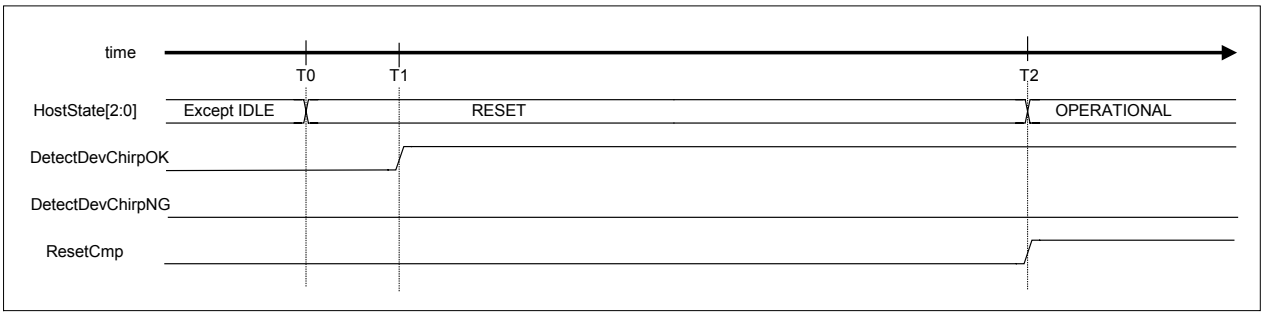


Figure 1-68 GoRESETtoOP timing (HS mode)

Table 1-60 GoRESETtoOP timing values (HS mode)

Timing Parameter	Description	Value
T0	Sets H_NegoControl_0.AutoMode to "GoRESETtoOP." (F/W) Performs the same processing performed for "GoRESET." (H/W)	0 (reference)
T1	Detects the device chirp and outputs the device chirp normal detection status. (H/W)	T1
T2	Issues the reset completion status. Performs the same processing performed for "GoOPERATIONAL." (H/W)	T2

1.3.10.3.10.2 If FS or LS Device Is Connected

The following procedure is performed if an FS or LS device is connected. Steps (2) to (5) below are performed automatically by the LSI hardware.

(1) The firmware sets the host state change execute (H_NegoControl_0.AutoMode) to "GoRESETtoOP" (T0).

(2) Performs the same processing performed for "GoRESET" (T0).

(3) Does not detect a device chirp and does not issue the device chirp normal/error detection status (H_SIE_IntStat_0.DetectDevChirpOK/NG) (T1).

(4) Issues the reset completion status (H_SIE_IntStat_1.ResetCmp) (T2).

(5) Performs the same processing performed for "GoOPERATIONAL" (T2).

Note that the timing for each state is the same as when executing "GoRESET" and "GoOPERATIONAL." See "1.3.10.3.4 GoRESET" and "1.3.10.3.5 GoOPERATIONAL," respectively, for timing particulars.

For detailed information on procedures and timing when errors (VBUS error) are detected in midprocess, see "1.3.10.2.1 VBUS Error Detection."

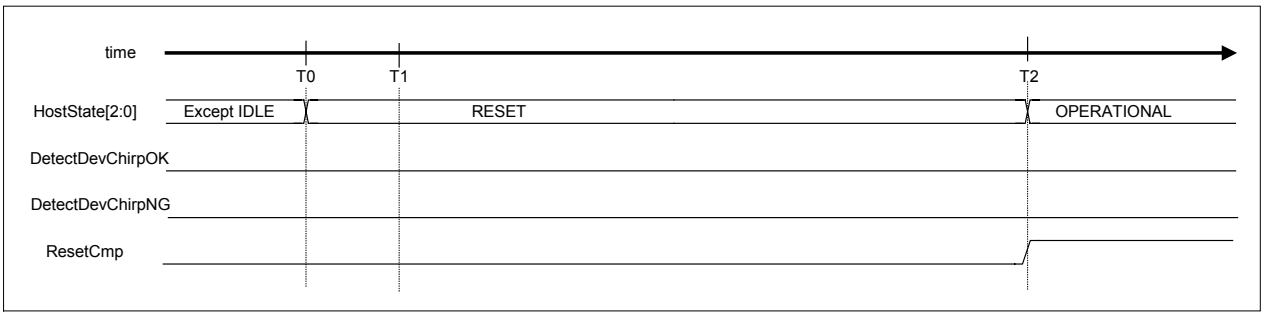


Figure 1-69 GoRESETtoOP timing (FS or LS mode)

Table 1-61 GoRESETtoOP timing values (FS or LS mode)

Timing Parameter	Description	Value
T0	Sets H_NegoControl_0.AutoMode to "GoRESETtoOP." (F/W) Performs the same processing performed for "GoRESET." (H/W)	0 (reference)
T1	Detects the device chirp and does not issue the device chirp normal/error detection status. (H/W)	T1
T2	Issues the reset completion status. Performs the same processing performed for "GoOPERATIONAL." (H/W)	T2

1.3.10.3.11 GoSUSPENDtoOP

The LSI hardware automatically performs the processing necessary to change from "SUSPEND" to "OPERATIONAL" states when "GoSUSPENDtoOP" is set to the host state change execute (H_NegoControl_0.AutoMode).

The remote wakeup detection function is automatically turned on and off by the hardware when "GoSUSPENDtoOP" is set. (Note that this is not shown by the remote wakeup receipt permission (H_NegoControl_1.RmtWkupDetEnb), and there is no need for firmware to manipulate the remote wakeup receipt permission (H_NegoControl_1.RmtWkupDetEnb).)

Avoid using the power management function with this setting.

The procedures for this setting are shown below. Steps (2) to (7) below are performed automatically by the LSI hardware.

(1) The firmware sets the host state change execute (H_NegoControl_0.AutoMode) to "GoSUSPENDtoOP" (T0).

(2) Performs the same processing performed for "GoSUSPEND" (T0).

(3) Issues suspend change completion status (H_SIE_IntStat_1.SuspendCmp) (T1).

(4) Detects the remote wakeup and issues remote wakeup detection status (H_SIE_IntStat_0.DetectRmtWkup) (T2).

(5) Performs the same processing performed for "GoRESUME" (T2).

(6) Issues the resume completion status (H_SIE_IntStat_1.ResumeCmp) (T3).

(7) Performs the same processing performed for "GoOPERATIONAL" (T3).

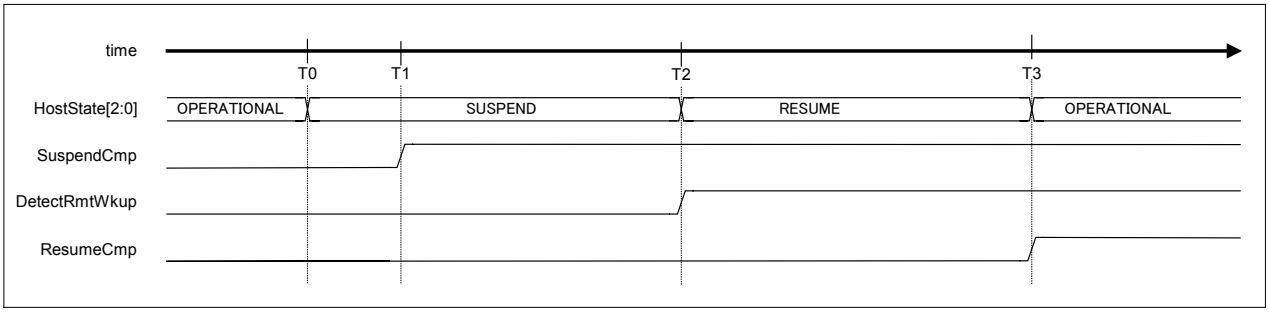


Figure 1-70 GoSUSPENDtoOP timing

Table 1-62 GoSUSPENDtoOP timing values

Timing Parameter	Description	Value
T0	Sets H_NegoControl_0.AutoMode to “GoSUSPENDtoOP.” (F/W) Performs the same processing performed for “GoSUSPEND.” (H/W)	0 (reference)
T1	Issues suspend change completion status. (H/W)	T1
T2	Detects the remote wakeup and issues remote wakeup detection status. Performs the same processing performed for “GoRESUME.” (H/W)	T2
T3	Issues the resume completion status. Performs the same processing performed for “GoOPERATIONAL.” (H/W)	T3

1.3.10.3.12 GoRESUMEtoOP

The LSI hardware automatically performs the processing necessary to change from "RESUME" to "OPERATIONAL" states when "GoRESUMEtoOP" is set to the host state change execute (H_NegoControl_0.AutoMode).

The procedures for this setting are shown below. Steps (2) to (4) below are performed automatically by the LSI hardware.

(1) The firmware sets the host state change execute (H_NegoControl_0.AutoMode) to "GoRESUMEtoOP" (T0).

(2) Performs the same processing performed for "GoRESUME" (T0).

(3) Issues the resume completion status (H_SIE_IntStat_1.ResumeCmp) (T1).

(4) Performs the same processing performed for "GoOPERATIONAL" (T1).

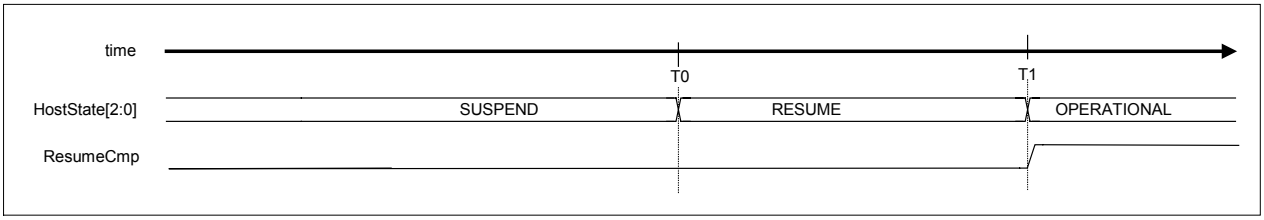


Figure 1-71 GoRESUMEtoOP

Table 1-63 GoRESUMEtoOP timing values

Timing Parameter	Description	Value
T0	Sets H_NegoControl_0.AutoMode to "GoRESUMEtoOP." (F/W) Performs the same processing performed for "GoRESUME." (H/W)	0 (reference)
T1	Issues the resume completion status. Performs the same processing performed for "GoOPERATIONAL." (H/W)	T1

1.4 Media Data Transfer Function

1.4.1 Media Data

The data type used by devices depends on the application and includes music and video data. In this manual, such data is generally referred to as media data.

1.4.2 Media Data Transfers

The LSI includes a 64-byte FIFO (media FIFO) used to transfer media data separately from the FIFO used for USB transfers. This FIFO allows media data to be transferred easily between the HDD and memory.

In this manual, the function in this LSI for performing transfers between the HDD and memory via media FIFO is called the media data transfer function.

A typical media data transfer is illustrated for playback (HDD to memory).

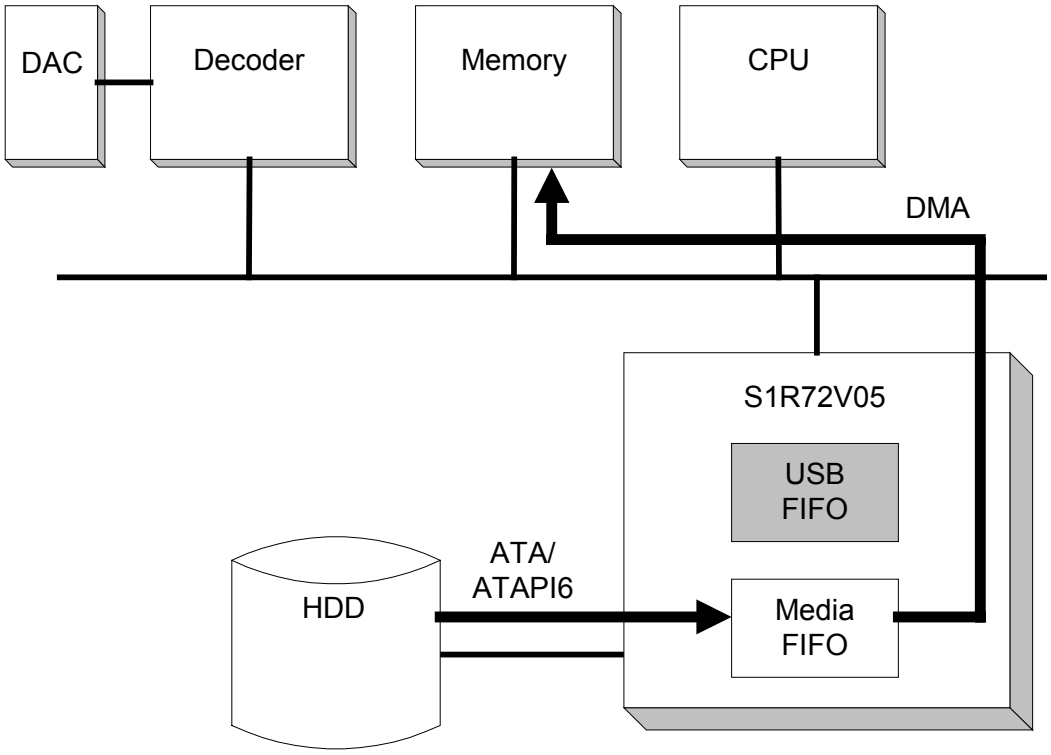


Figure 1-72 Media data transfer example (for playback)

The media FIFO is accessed by specifying the IDE port and one other port (DMA0/1, CPU_Rd, or CPU_Wr) via the MediaFIFO_Join register.

For detailed information on media FIFO access methods, see “1.7 Media FIFO Management.”

1.4.3 Reduced Power Consumption

The media data transfer function can be used in the ACTIVE60, ACT_DEVICE, ACT_HOST, or ACT_ALL power mode states of the LSI. In particular, the ACTIVE60 state halts the USB device and USB host functions, allowing significant power consumption reductions.

See "1.5 Power Management Functions" for detailed information on power management functions.

The respective data flows are shown below for combined USB and media data transfers and for media data transfers alone.

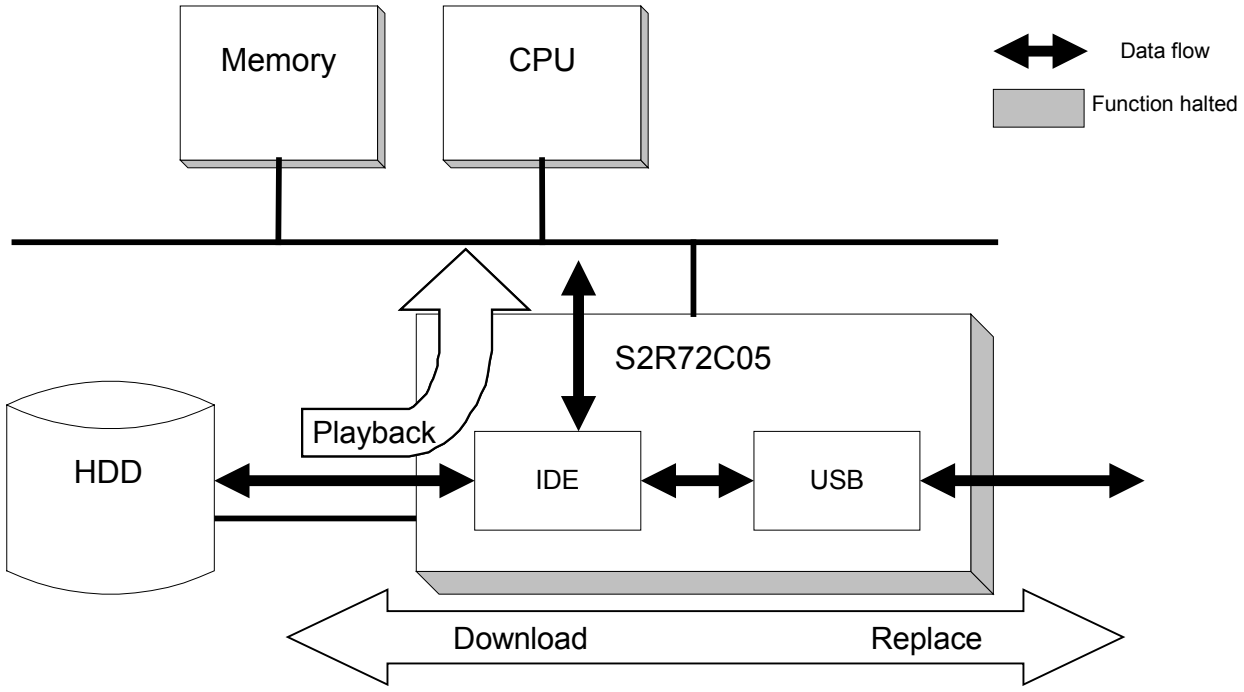


Figure 1-73 USB and media data transfers (for playback)

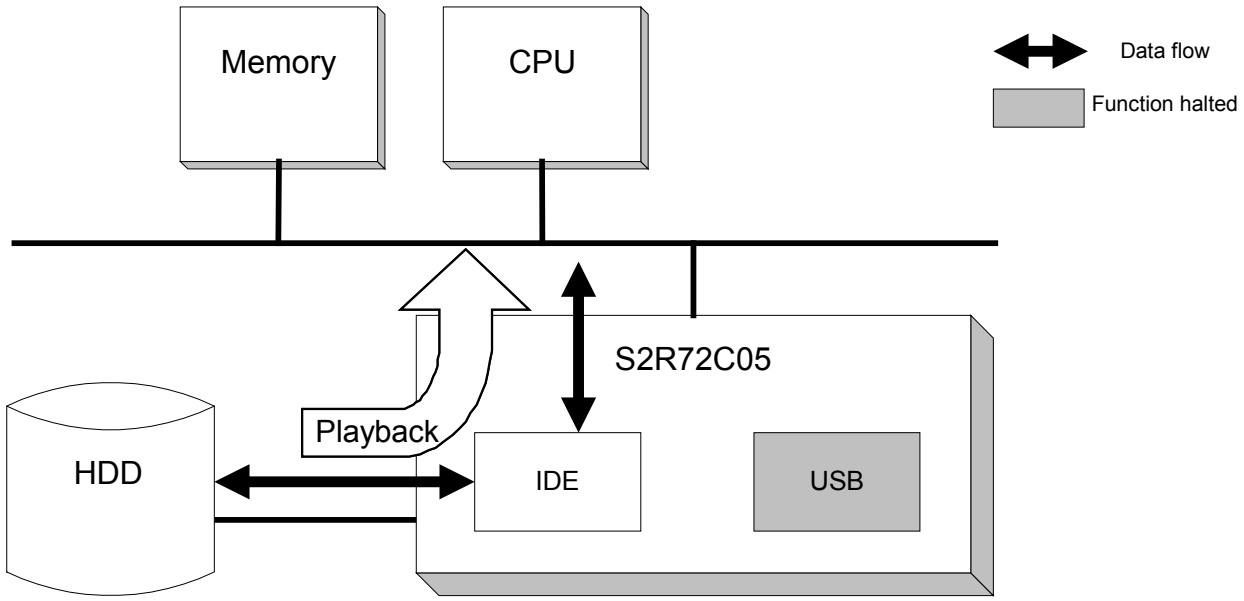


Figure 1-74 Media data transfers only (for playback)

1.5 Power Management Functions

When using this IC, the ClkSelect.Port1x2 bit must first be set appropriately to specify whether the IC is being used in 2-port or 1-port mode. 2-port mode uses USB port A for the host and USB port B for the device. 1-port mode uses USB port B for both the host and device. The default value is cleared to 0, specifying 2-port mode, but 1-port mode can be specified by setting the bit to 1. Note that the ClkSelect.Port1x2 bit should be changed in either SLEEP or SNOOZE state (described later).

1.5.1 2-port Mode (ClkSelect.Port1x2 Is Cleared to 0)

The power management function controls oscillator and PLL (DevicePLL480, HostPLL480, PLL60) operations and changes the SLEEP state, SNOOZE state, ACTIVE60 state, ACT_DEVICE state, ACT_HOST state, and ACT_ALL state. To change to other states, the function is initiated by setting the PM_Control_0.GoSLEEP, PM_Control_0.GoSNOOZE, PM_Control_0.GoActive60, PM_Control_0.GoActDevice, PM_Control_0.GoActHost, PM_Control_0.GoActAllDev, or PM_Control_0.GoActAllHost bit, and is terminated once processing is complete. PM_Control_1.PM_State[3:0] should be checked to determine the current state. The MainIntStat.FinishedPM event occurs when the change is complete. An interrupt XINT occurs here if the MainIntEnb.EnFinishedPM bit is set.

It is possible to change from any state to any other state, and a MainIntStat.FinishedPM event occurs once the change to SLEEP state is completed via the ACTIVE60 and SNOOZE states if the PM_Control_0.GoSLEEP bit is set in the ACT_DEVICE or ACT_HOST state. A MainIntStat.FinishedPM event occurs once the change to ACT_DEVICE or ACT_HOST state is completed via SNOOZE and ACTIVE60 states if the PM_Control_0.GoActDevice or PM_Control_0.GoActHost bit is set in the SLEEP state. Similarly, a MainIntStat.FinishedPM event occurs once the change to SLEEP state is completed via SNOOZE state if the PM_Control_0.GoSLEEP bit is set in the ACTIVE60 state. A MainIntStat.FinishedPM event occurs once the change to the ACTIVE60 state is completed via SNOOZE state if the PM_Control_0.GoActive60 bit is set in the SLEEP state. A MainIntStat.FinishedPM event occurs once the change to ACT_HOST state is completed via the ACTIVE60 state if the PM_Control_0.GoActHost bit is set in the ACT_DEVICE state. Similarly, a MainIntStat.FinishedPM event occurs once the change to ACT_DEVICE state is completed via the ACTIVE60 state if the PM_Control_0.GoActDevice bit is set in the ACT_HOST state.

Similarly, a MainIntStat.FinishedPM event occurs once the various midway states have been changed, and the change to ACT_ALL state is completed by setting the PM_Control_0.GoActAllDev or PM_Control_0.GoActAllHost bit when changing from SLEEP, SNOOZE, or ACTIVE60 state to the ACT_ALL state. Conversely, the various PM_Control_0.GoSLEEP, PM_Control_0.GoSNOOZE, and PM_Control_0.GoActive60 bits can be set to change from ACT_ALL state to SLEEP, SNOOZE, or ACTIVE60 state. In these cases, too, a MainIntStat.FinishedPM event occurs once the change is complete.

Except for the ACT_ALL state, note that it is not possible to change from the current state to the same state. The PM_Control_0.GoXXXX bit set will not be cleared in this case. For example, if PM_Control_0.GoActive60 is set while operating in ACTIVE60 state, the PM_Control_0.GoActive60 bit will remain set without changing states (even though it appears correct). This means the ensuing operations will be invalid.

Both DevicePLL480 and HostPLL480 operate in the ACT_ALL state, allowing use of both functions. However, only one transfer can be used. To perform device transfers while using host transfers, set the PM_Control_0.GoActAllDev bit and switch to device transfer mode. A MainIntStat.FinishedPM event occurs as soon as the change is completed in the same way for changes to other states, and the PM_Control_1.MonActFunc bit is cleared at the same time. Similarly, to perform host transfers while using device transfer, set the PM_Control_0.GoActAllHost bit and switch to host transfer mode. A MainIntStat.FinishedPM event occurs as soon as the change is complete in the same way as for changes to other states, and the PM_Control_1.MonActFunc bit is set concurrently. Thus, only the ACT_ALL state exists as a power management state, and ACT_ALL_DEVICE and ACT_ALL_HOST states do not exist, forming a substate of the PM_Control_1.MonActFunc.

The PM_Control_1.MonActFunc bit is also cleared in the ACT_DEVICE state and set in the ACT_HOST state, as well as being set and cleared in the ACT_ALL state, indicating whether either device or host transfer is possible in 2-port mode. Note that this bit has no significance in SLEEP, SNOOZE, and ACTIVE60 states.

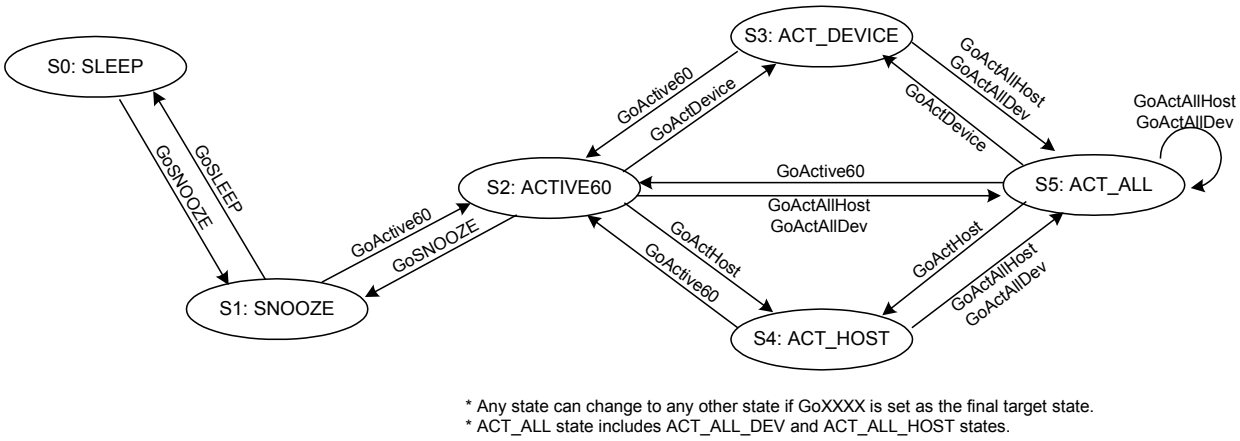


Figure 1-75 Power management for port mode

1.5.1.1 SLEEP

Since the oscillator does not oscillate in this state, PLL also does not oscillate.

Setting the PM_Control_0.GoSLEEP bit and changing to SLEEP while in SNOOZE, ACTIVE60, ACT_DEVICE, ACT_HOST, or ACT_ALL state stops the PLL currently operating (DevicePLL480, HostPLL480, or both), stops the PLL60, stops the OSCCLK output, and finally halts oscillation.

Conversely, setting the PM_Control_0.GoSNOOZE, PM_Control_0.GoActve60, PM_Control_0.GoActDevice, PM_Control_0.GoActHost, PM_Control_0.GoActAllDev, or PM_Control_0.GoActAllHost bit while in the SLEEP state to change to SNOOZE uses an oscillation stabilization time gate to avoid subjecting internal circuits to OSCCLK until oscillator oscillations have stabilized. This time till oscillation stabilization depends on the oscillation cell, oscillator, peripheral circuits, and circuit board and should be set using the WakeUpTim_H,L register. The WakeUpTim_H,L register can be accessed asynchronously, allowing reading and writing even in the SLEEP state.

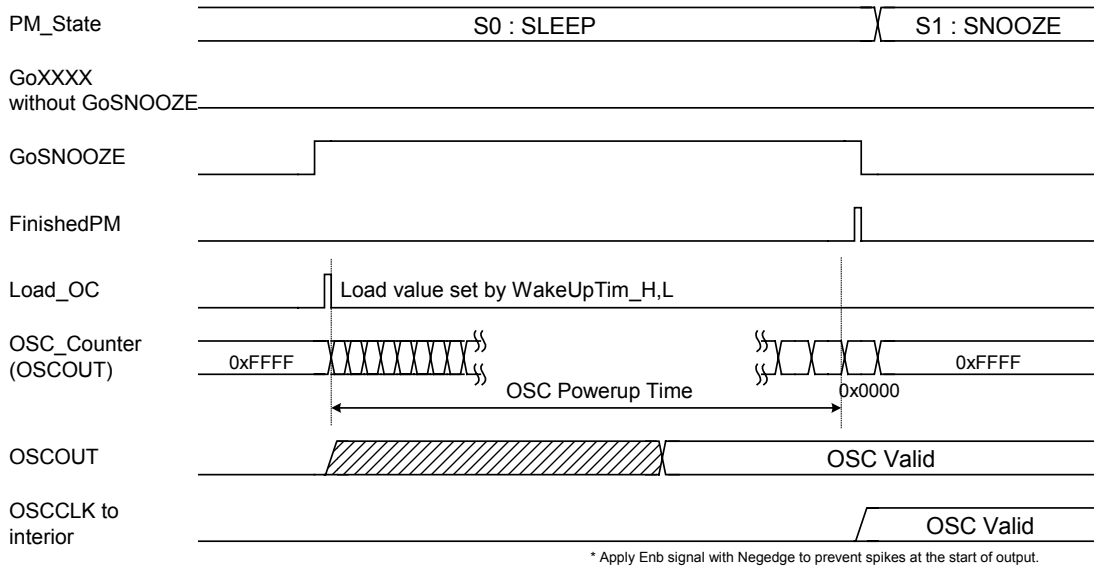


Figure 1-76 Disengaging from SLEEP state (for GoSNOOZE)

1.5.1.2 SNOOZE

The oscillator oscillates in this state; the PLL does not oscillate in this state.

Setting the PM_Control_0.GoSNOOZE bit and changing to SNOOZE while in ACTIVE60, ACT_DEVICE, ACT_HOST, or ACT_ALL state stops the clock output, stops DevicePLL480, HostPLL480, or both, and finally stops PLL60.

Conversely, setting the PM_Control_0.GoActive60, PM_Control_0.GoActDevice, PM_Control_0.GoActHost, PM_Control_0.GoActAllHost, or PM_Control_0.GoActAllDev bit while in SNOOZE state to change to ACTIVE uses a PLL stabilization time gate (approximately 250 µs) to avoid subjecting internal circuits to SCLK until PLL oscillations have stabilized.

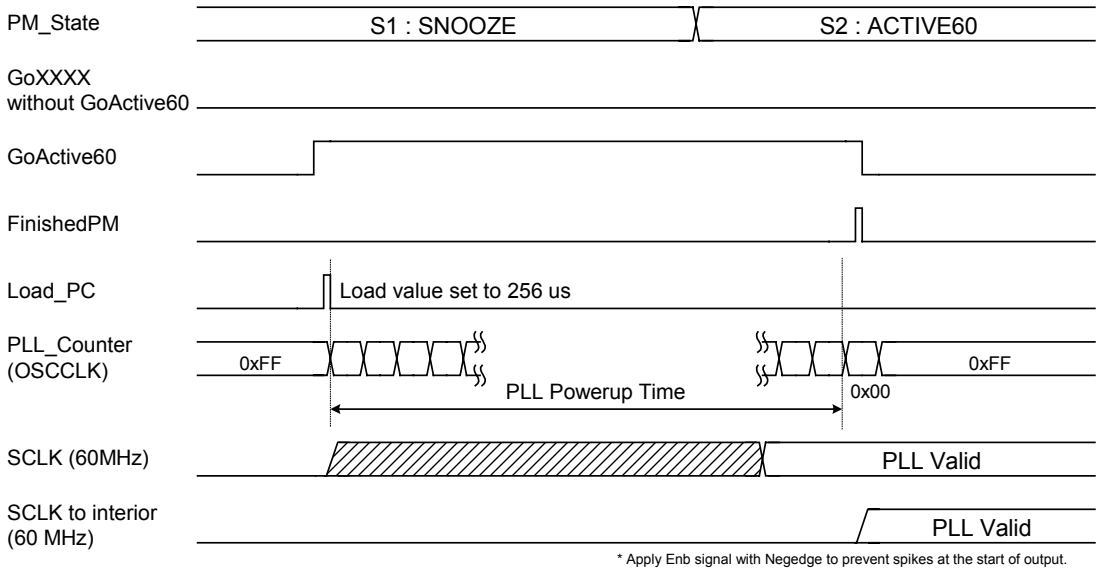


Figure 1-77 Disengaging from SNOOZE state (for GoActive60)

1.5.1.3 ACTIVE60

In this state, the oscillator and PLL60 operate, and the DevicePLL480 and HostPLL480 are stopped. The registers and bits indicated in ***bold italics*** in the register map allow reading and writing even in SNOOZE and SLEEP states. Otherwise, the registers described in "2.2 Device/Host Common Register Map" allow reading and writing, and the registers in "2.3 Device Register Map" and "2.4 Host Register Map" can be read in the ACTIVE60 state.

The USB circuit does not operate for either the host or the device in this state, since an SCLK480 with a frequency of 480 MHz is required.

The IDE and CPU circuits do not require a 480 MHz SCLK480. The PLL current consumption for 480 MHz is relatively high compared to that for 60 MHz. Power requirements can be reduced by changing to the ACTIVE60 state if the USB is not used.

1.5.1.4 ACT_DEVICE

In this state, the oscillator, PLL60, and DevicePLL480 operate. The registers and bits indicated in ***bold italics*** in the register map allow reading and writing even in SNOOZE and SLEEP states. Otherwise, the registers described in "2.2 Device/Host Common Register Map" and "2.3 Device Register Map" allow reading and writing, and the registers in "2.4 Host Register Map" can be read in the ACT_DEVICE state. The PM_Control_1.MonActFunc bit is cleared when the change to the ACT_DEVICE state is completed, enabling data transfers as a USB device.

The USB device circuit operates in this state, since it requires a 480 MHz SCLK480.

The IDE and CPU circuits do not require a 480 MHz SCLK480. The PLL current consumption for 480 MHz is relatively high compared to that for 60 MHz. Power requirements can be reduced by changing to the ACTIVE60 state if the USB is not used.

1.5.1.5 ACT_HOST

In this state, the oscillator, PLL60, and HostPLL480 operate. The registers and bits indicated in ***bold italics*** in the register map allow reading and writing even in SNOOZE and SLEEP states. Otherwise, the registers described in "2.2 Device/Host Common Register Map" and "2.4 Host Register Map" allow reading and writing, and the registers in "2.3 Device Register Map" can be read in the ACT_HOST state. The PM_Control_1.MonActFunc bit is set when the change to the ACT_HOST state is completed, enabling data transfers as a USB host.

The USB host circuit operates in this state, since it requires a 480 MHz SCLK480.

The IDE and CPU circuits do not require a 480 MHz SCLK480. The PLL current consumption for 480 MHz is relatively high compared to that for 60 MHz. Power requirements can be reduced by changing to the ACTIVE60 state if the USB is not used.

1.5.1.6 ACT_ALL

In this state, the oscillator, PLL60, DevicePLL480, HostPLL480, and all clock sources operate. The registers and bits indicated in ***bold italics*** in the register map allow reading and writing even in SNOOZE and SLEEP states. Otherwise, all the registers allow reading and writing in 2-port mode (when the ClkSelect.Port1x2 bit is cleared) in the ACT_ALL state. In other words, the registers described in "2.2 Device/Host Common Register Map," "2.3 Device Register Map," and "2.4 Host Register Map" allow reading and writing. Note that the HostDeviceSel.HOSTxDEVICE bit must be set appropriately when reading and writing to the registers, since host register and device register maps exist in the same address space.

The USB device and host circuits operate in this state, since they require a 480 MHz SCLK480. The PM_Control_1.MonActFunc bit is cleared when the change to the ACT_ALL state is completed with the PM_Control_0.GoActAllDev bit set as the ACT_ALL state change start bit, enabling data transfers as a USB device. This state is defined here as the ACT_ALL_DEV state, forming a substate of the ACT_ALL state. The USB host cannot transfer data here, but allows SOF transmission to be continued and suspended without disconnecting peripheral devices.

Conversely, the PM_Control_1.MonActFunc bit is set when the change to the ACT_ALL state is completed with the PM_Control_0.GoActAllHost bit set as the ACT_ALL state change start bit, enabling data transfers as a USB host. This state is defined here as the ACT_ALL_HOST state, forming a substate of the ACT_ALL state. The USB device cannot transfer data here, but it returns an NAK in response to the token from the PC to maintain the connection without disconnecting the PC. An ACK is normally returned to SETUP transactions for endpoint EP0.

The ACT_ALL state allows host operations without fully stopping operation as a device. For mass storage devices, for example, reading and writing to a USB memory as a host is possible while keeping the disk mounted without disconnecting the PC.

1.5.2 1-port Mode (ClkSelect.Port1x2 Set to 1)

The USB port B alone can be used to allow use of either the USB device or host function. The HostDeviceSel.HOSTxDEVICE bit is used to switch functions, and if this bit is cleared to 0, the device function is selected together with the register map selection, the clock is supplied to the device function block and not supplied to the host function block. Conversely, if the HostDeviceSel.HOSTxDEVICE bit is set to 1, the host function is selected together with the register map selection, the clock is supplied to the host function block, and not supplied to the device function block. The actual clock feed to the function block is started and stopped by the SLEEP, SNOOZE, ACTIVE60, or ACT_DEVICE states described later. The HostDeviceSel.HOSTxDEVICE bit should be set and cleared in SLEEP or SNOOZE modes with no clock feed to each module.

Note that the ACT_HOST and ACT_ALL states used in 2-port mode cannot be used in 1-port mode. Additionally, the USB port A HTM (Host Transceiver Macro) does not operate, as it is not used in 1-port mode. The terminals should be connected as shown in the examples.

The power management function controls the oscillator and PLL (DevicePLL480, PLL60) operations, and changes the four SLEEP, SNOOZE, ACTIVE60, and ACT_DEVICE states. To change to other states, setting the PM_Control_0.GoSLEEP, PM_Control_0.GoSNOOZE, PM_Control_0.GoActive60, PM_Control_0.GoActDevice bits starts and ends the change after performing the required processing. Check PM_Control_1.PM_State[3:0] to determine the current state. A MainIntStat.FinishedPM event occurs after the change. An interrupt XINT occurs here if the MainIntEnb.EnFinishedPM bit is set.

It is possible to change from any state to any other state, and a MainIntStat.FinishedPM event occurs once the change to SLEEP state is completed via the ACTIVE60 and SNOOZE states if the PM_Control_0.GoSLEEP bit is set in the ACT_DEVICE state. A MainIntStat.FinishedPM event occurs once the change to ACT_DEVICE state is completed via SNOOZE and ACTIVE60 states if the PM_Control_0.GoActHost bit is set in the SLEEP state. Similarly, a MainIntStat.FinishedPM event occurs once the change to SLEEP state is completed via SNOOZE state if the PM_Control_0.GoSLEEP bit is set in the ACTIVE60 state. A MainIntStat.FinishedPM event occurs once the change to the ACTIVE60 state is completed via SNOOZE state if the PM_Control_0.GoActive60 bit is set in the SLEEP state.

Note that it is not possible to change from the current state to the same state. The PM_Control_0.GoXXXX bit set will not be cleared in this case. If, for example, PM_Control_0.GoActive60 is set while operating in ACTIVE60 state, the PM_Control_0.GoActive60 bit will remain set without changing states (even though it appears correct).

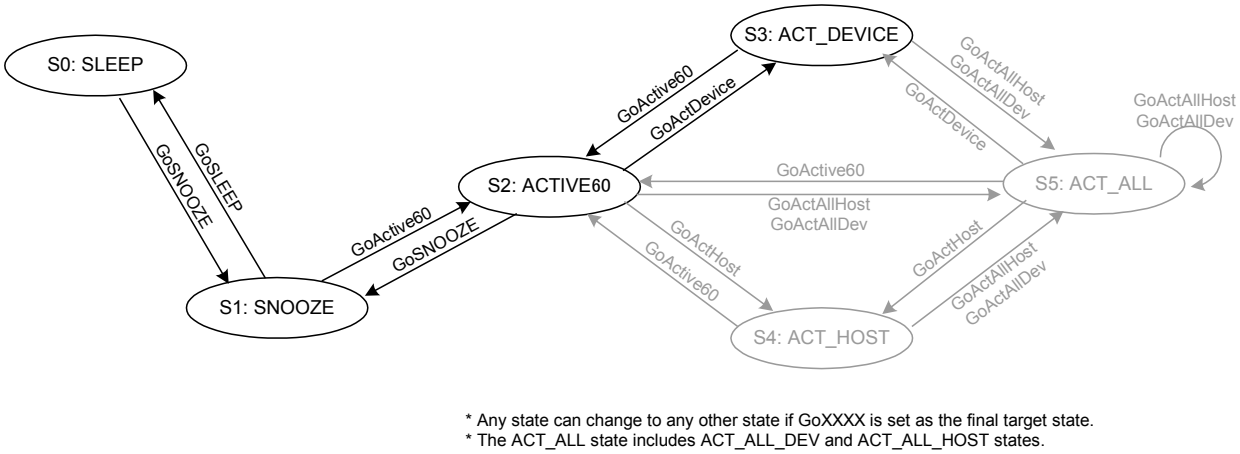


Figure 1-78 Power management for 1-port mode

1.5.2.1 SLEEP

Neither the oscillator nor PLL oscillates in this state.

Setting the PM_Control_0.GoSLEEP bit and changing to SLEEP while in SNOOZE, ACTIVE60, or ACT_DEVICE states stops DevicePLL480, stops the PLL60, stops the OSCCLK output, and finally halts oscillation.

Conversely, setting the PM_Control_0.GoSNOOZE, PM_Control_0.GoActve60, or PM_Control_0.GoActDevice bit while in the SLEEP state to change to SNOOZE uses an oscillation stabilization time gate to avoid subjecting internal circuits to OSCCLK until oscillator oscillations have stabilized. This oscillation stabilization time depends on the oscillation cell, oscillator, peripheral circuits and circuit board and should be set using the WakeUpTim_H,L register. The WakeUpTim_H,L register can be accessed asynchronously, allowing reading and writing even in the SLEEP state.

1.5.2.2 SNOOZE

The oscillator oscillates in this state, but the PLL does not oscillate.

Setting the PM_Control_0.GoSNOOZE bit and changing to SNOOZE while in ACTIVE60 or ACT_DEVICE states stops the clock output, stops DevicePLL480, and finally stops PLL60.

Conversely, setting the PM_Control_0.GoActive60 or PM_Control_0.GoActDevice bit while in SNOOZE state to change to ACTIVE uses a PLL stabilization time gate (approximately 250 µs) to avoid subjecting internal circuits to SCLK until PLL oscillations have stabilized.

1.5.2.3 ACTIVE60

The oscillator and PLL60 operate in this state, and the DevicePLL480 is stopped. The registers and bits indicated in ***bold italics*** in the register map allow reading and writing even in SNOOZE and SLEEP states. Otherwise, the registers described in "2.2 Device/Host Common Register Map" allow reading and writing, and the registers in "2.3 Device Register Map" and "2.4 Host Register Map" can be read in the ACTIVE60 state.

The USB circuit does not operate for either the host or the device in this state, since an SCLK480 is required with a frequency of 480 MHz.

The IDE and CPU circuits do not require a 480 MHz SCLK480. The PLL current consumption for 480 MHz is relatively high compared to that for 60 MHz. Power requirements can be reduced by changing to the ACTIVE60 state if the USB is not used.

1.5.2.4 ACT_DEVICE

In this state, the oscillator, PLL60, and DevicePLL480 (USB port B 480 MHz PLL) operate. The registers and bits indicated in ***bold italics*** in the register map allow reading and writing even in SNOOZE and SLEEP states. Otherwise, the registers described in "2.2 Device/Host Common Register Map" allow reading and writing in the ACT_DEVICE state. The "2.3 Device Register Map," "2.4 Host Register Map," and the HostDeviceSelect.HOSTxDEVICE bit setting determines whether reading or writing is possible. The registers described in "2.3 Device Register Map" can be read or written to if the HostDeviceSelect.HOSTxDEVICE bit is cleared to 0, and the registers described in "2.4 Host Register Map" can be read or written to if the HostDeviceSelect.HOSTxDEVICE bit is set to 1.

The USB device and host circuits operate in this state, since it requires a 480 MHz SCLK480.

The IDE and CPU circuits do not require a 480 MHz SCLK480. The PLL current consumption for 480 MHz is relatively high compared to that for 60 MHz. Power requirements can be reduced by changing to the ACTIVE60 state if the USB is not used.

1.5.2.5 ACT_HOST

The ACT_HOST state and the PM_Control_0.GoActHost bit allowing changes to this state cannot be used.

1.5.2.6 ACT_ALL

The ACT_ALL state and the PM_Control_0.GoActALL bit allowing changes to this state cannot be used.

1.6 FIFO Management

This section describes the FIFO management.

1.6.1 FIFO Memory Map

The FIFO memory map is shown below.

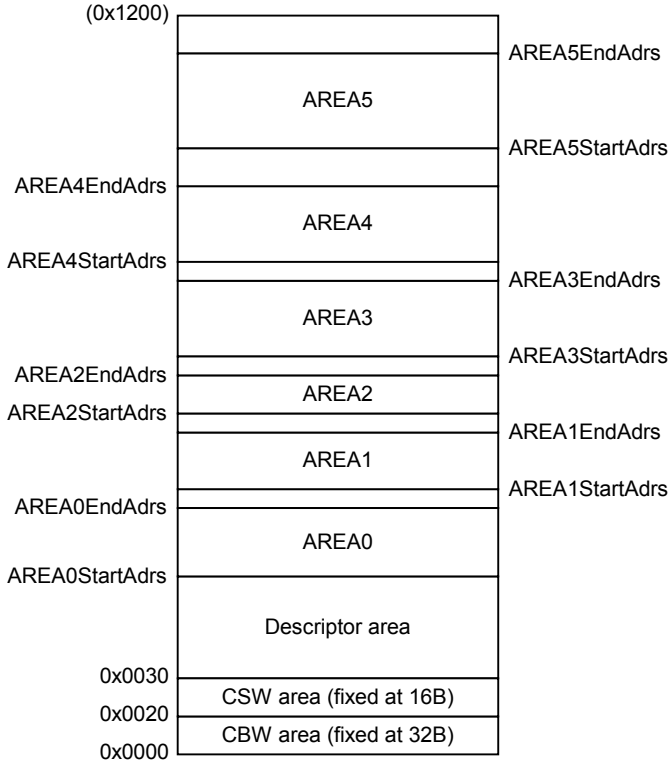


Figure 1-79 Typical FIFO memory map

FIFO memory can be used divided up into CBW, CSW and descriptor areas, and AREA0, AREA1, AREA2, AREA3, AREA4, and AREA5. The CBW and CSW areas are assigned fixed areas as shown in Figure 1-79. The other areas (AREAx{x=0-5}) can be set flexibly using the FIFO area setting register (AREAx{x=0-5}StartAdrs_H,L, AREAx{x=0-5}EndAdrs_H,L). The descriptor area can use unused areas as required.

The descriptor area is an area used by the descriptor response function for USB devices. Any unused FIFO area can be used. For detailed information on actual usage procedures, see "1.6.2 Descriptor Area." All FIFO areas can be set for use with the descriptor response function, but it is recommended that the areas shown here be used to prevent interference.

The CBW area is used for bulk only support function CBW support for USB devices. 32 bytes are reserved, and a 31-byte area is used starting from address 0x0000. For detailed information on actual usage procedures, see "1.6.3.1 CBW Area (for USB device)." This CBW area is also used for the CHa bulk only support function with USB hosts. For detailed information on actual usage procedures, see "1.6.3.2 CBW Area (for USB host)."

The CSW area is used for bulk only support function CSW support for USB devices. 16 bytes are reserved, and a 13-byte area is used starting from address 0x0020. For detailed information on actual usage procedures, see "1.6.4.1 CSW Area (for USB device)." This CSW area is also used for the CHa bulk only support function with USB hosts. For detailed information on actual usage procedures, see "1.6.4.2 CSW Area (for USB Host)."

AREA0, AREA1, AREA2, AREA3, AREA4, and AREA5 are the endpoint areas which can be used by joining to endpoints (EPx{x=0, a-e}) for USB devices. For USB hosts, they can be used by joining to channels (CHx{x=0, a-e}) in the same way. The JoinEPxCHx{x=0,a-e} bit in the AREA join setting register (AREAx{x=0-5}Join_1) is set to join areas. Note that the same endpoint or channel must not be joined to multiple areas.

The AREA0, AREA1, AREA2, AREA3, AREA4, and AREA5 areas are each controlled as FIFO and have data storage quantities. To clear this stored state, set the AREAnFIFO_Clr.ClearAREAx{x=0-5} bits.

Note that clearing these states only initializes the data storage information, and does not write or clear data. Data is thus not actually cleared on the RAM by these bits, and the data stored within the descriptor areas is not removed. There is no need to rewrite data after clearing.

1.6.2 Descriptor Area

The descriptor area is the area used by the descriptor response function for USB devices. The descriptor response function can be used for endpoint 0 when the data stage is in the IN transfer.

The IN-direction data stage is performed automatically by setting the initial address of the data to be written to the area and the data size to be returned, and then executing the descriptor response function.

Commands can be specified to return the area data on receipt of a request by writing the details such as the device descriptor specified uniquely by the device to the area at initialization after turning on the power. This allows a fast response to be given without the need to write data to the EP0 FIFO area for each request.

1.6.2.1 Writing Data to Descriptor Area

The RAM_WrDoor function is used to write data to the descriptor area. The initial write address is set to the RAM_WrAdrs_H,L register, allowing the data to be written to the RAM_WrDoor_0,1 register. The RAM_WrAdrs_H,L register value is updated by the quantity of data written for each write cycle. It is therefore possible to write continuously to the RAM_WrDoor_0,1 register when writing data to continuous addresses.

Note that the RAM_WrDoor_0,1 register allows writing only.

1.6.2.2 Executing Data Stage (IN) with Descriptor Area

When using the written data with the descriptor response function, the initial address of the data to be sent to the data stage is set in the D_DescAdrs_H,L register, the data size to be returned is set in the D_DescSize_H,L register, and then the D_EP0Control.ReplyDescriptor bit is set to 1. The D_EP0Control.INxOUT bit is also set to "1" to allow IN transactions. Take care to clear the D_EP0Control_IN.ForceNAK bit after clearing D_SETUP_Control.ProtectEP0 to allow data packets to be returned to data stage IN transactions.

After setting, data packets up to the data quantity set by the D_DescSize_H,L register are returned to the host in response to the IN transaction from the host while dividing automatically into the max packet size (set by D_EP0MaxSize). Data is automatically sent as short packets if the D_DescSize_H,L register value is smaller than the max packet size or if the data quantity remaining after division is smaller than the max packet size.

D_EP0Control.ReplyDescriptor is cleared and D_EP0IntStat.DescriptorCmp is set when an OUT transaction is issued by the host. The firmware should switch to status stage processing.

1.6.3 CBW Area

1.6.3.1 CBW Area (for USB device)

The CBW area is the area used by the bulk only support function CBW support for USB devices. This area can be used for receiving data when performing Bulk Only Transport Protocol command transport for Bulk OUT endpoints (endpoints EPa, EPb, EPc, EPd, EPe). This enables only data received by data transport to be processed by the endpoint FIFO.

When using CBW support, data is received in the CBW area if the OUT transaction is performed for the endpoint involved and the data size is 31 bytes. Error status is issued and the data is discarded if the data is not 31 bytes long.

The RAM_Rd function is used to read out the data received in the CBW area. Setting the RAM_RdControl.RAM_GoRdCBW_CSW bit reads the CBW area data and copies it to registers RAM_Rd_00 to RAM_Rd_1E before issuing completion status (CPU_IntStat.RAM_RdCmp bit).

1.6.3.2 CBW Area (for USB host)

The CBW area is the area used for the bulk only support function for USB hosts. CBW data is sent as data packets from these areas when using Bulk Only Transport Protocol command transport with channel CHa. This enables only data sent by data transport to be processed by the channel FIFO.

CBW data (31 bytes) should be provided in the CBW area from 0x0000 before sending data packets.

The RAM_WrDoor function is used to write data to the CBW area. The CBW area initial address (0x0000) is written to the RAM_WrAdrs_H,L register, and then 31 bytes of valid data are written via the RAM_WrDoor_0,1 register. The CBW area holds 32 bytes. There is no problem of leakage into other areas even if 32 bytes are written in word access.

1.6.4 CSW Area

1.6.4.1 CSW Area (for USB device)

The CSW area is the area used by the bulk only support function CSW support for USB devices. This area can be used for sending data when performing Bulk Only Transport Protocol status transport for Bulk IN endpoints (endpoints EPa, EPb, EPc, EPd, EPe). This enables only data sent by data transport to be processed by the endpoint FIFO.

When using CSW support, 13 bytes of data are sent as a data packet from the CSW area if the IN transaction is performed for the endpoint involved and the data size is 13 bytes.

The RAM_WrDoor function is used to write data to the CSW area. The CSW area initial address (0x0020) is written to the RAM_WrAdrs_H,L register, and then 13 bytes of valid data are written via the RAM_WrDoor_0,1 register. The CSW area holds 16 bytes. There is no problem of leakage into other areas even if 14 bytes are written in word access.

1.6.4.2 CSW Area (for USB Host)

The CSW area is the area used for the bulk only support function for USB hosts. CSW data is received in this area when using Bulk Only Transport Protocol status transport with channel CHa. This enables only data received by data transport to be processed by the channel FIFO.

The RAM_Rd function is used to read out the data received in the CSW area. Setting the RAM_RdControl.RAM_GoRdCBW_CSW bit reads the CSW area data and copies it to registers RAM_Rd_00 to RAM_Rd_0C before issuing completion status (CPU_IntStat.RAM_RdCmp bit).

1.6.5 FIFO Access Methods

FIFO data can be accessed using RAM access or FIFO access. FIFO access means include CPU (register), CPU (DMA), USB, and IDE. Data can be read from or written to any FIFO area using RAM access. Number of FIFO data is not updated depending on RAM access. This means that FIFO area is not freed when FIFO data is read out using RAM_Rd function. In the same way, FIFO area is not occupied even if data is written to FIFO using RAM_WrDoor function.

1.6.5.1 RAM Access Methods (RAM_Rd)

To read from the FIFO using the RAM_Rd register, before setting the RAM_RdControl.RAM_GoRd bit, set the FIFO area initial address and data size to be read in the RAM_RdAdrs_H,L and RAM_RdCount registers. The CPU_IntStat.RAM_RdCmp bit is set to “1” once the specified FIFO area data can be read from the RAM_Rd register. Read out the data from registers RAM_Rd_00 to RAM_Rd_1F after checking the RAM_RdCmp bit. The data read out is stored in sequence from RAM_Rd_00. The RAM_Rd register values are invalid beyond the preset size if the size set in the RAM_RdCount register is smaller than 32.

FIFO data can be read out from the RAM_Rd register as required, regardless of FIFO area settings.

The RAM_RdAdrs_H,L and RAM_RdCount register values are updated in sequence while the RAM_Rd function is operating. Avoid accessing these registers after the RAM_Rd function starts

until the CPU_IntStat.RAM_RdCmp bit has been set. Values read out from these registers are not guaranteed while the RAM_Rd function is operating. Writing to these registers may result in errors.

1.6.5.2 RAM Access Methods (RAM_WrDoor)

To write to the FIFO using the RAM_WrDoor_0,1 register, set the initial write address in the RAM_WrAdrs_H,L register and write the data using the RAM_WrDoor_0,1 register. The RAM_WrAdrs_H,L register is incremented automatically by the data quantity written for each access, allowing continuous writing to the RAM_WrDoor_0,1 register when writing data to continuous addresses.

FIFO data can be written to the FIFO using the RAM_WrDoor_0,1 register as required, regardless of the FIFO area settings.

1.6.5.3 FIFO Access Methods (Register Access)

To read data from the FIFO by register access, set AREAx{x=0-5}Join_0.JoinCPU_Rd to 1 for one area, and then read using the FIFO_Rd_0,1 or FIFO_ByteRd registers.

To write data to the FIFO by register access, set AREAx{x=0-5}Join_0.JoinCPU_Wr to 1 for one area, and then write using the FIFO_Wr_0,1 or FIFO_ByteWr registers.

The FIFO_RdRemain_H,L register indicates the remaining data size that can be read from the FIFO for a single area set using JoinCPU_Rd. Similarly, the FIFO_WrRemain_H,L register indicates the remaining space that can be written to in the FIFO for a single area set using JoinCPU_Wr.

Note that data will be read from the FIFO during register dumping if either JoinCPU_Rd bit is set for register dumping when debugging the firmware using ICE.

Accessing the CPU register requires that the FIFO and media FIFO be joined exclusively. In other words, JoinCPU_Wr cannot be simultaneously set for both media FIFO and a FIFO area. Likewise, JoinCPU_Rd cannot be simultaneously set for both media FIFO and a FIFO area.

1.6.5.4 FIFO Access Methods (DMA)

To read from the FIFO using CPU DMA access, a single area is selected by the AREAx{x=0-5}Join_0.JoinDMAx{x=0,1} bit, the DMAx{x=0,1}_Control.Dir bit is set to 1, and the data are read out using the DMA procedures.

To write to the FIFO using CPU DMA access, a single area is selected by the AREAx{x=0-5}Join_0.JoinDMAx{x=0,1} bit for each DMA channel, the DMAx{x=0,1}_Control.Dir bit is set to 0, and the data are written using the DMA procedures.

The DMAx{x=0,1}_Remain_H,L register indicates the remaining data size that can be read from the FIFO for the single area selected by the AREAx{x=0-5}Join_0.JoinDMAx{x=0,1} bit for each DMA channel. It also indicates the remaining space that can be written to in the FIFO for a single area selected by the AREAx{x=0-5}Join_0.JoinDMAx{x=0,1} bit for each DMA channel.

Accessing the CPU DMA requires that the FIFO and media FIFO be joined exclusively. In other words, JoinDMA0 or JoinDMA1 cannot be set simultaneously for both media FIFO and a FIFO area.

1.6.5.5 FIFO Access Methods (IDE)

When the IDE accesses the FIFO, a single area is selected by the AREAx{x=0-5}Join_0.JoinIDE bit for one of the areas, and data is transferred by IDE procedures. The IDE transfer direction depends on the IDE_Control.Dir bit.

IDE access requires that the FIFO and media FIFO be joined exclusively. In other words, JoinIDE cannot be simultaneously set to both media FIFO and a FIFO area.

1.6.5.6 FIFO Access Restrictions

The FIFO in this LSI allows simultaneous transfers to and from the USB, register reading and writing from the CPU bus, reading and writing using DMA, and reading and writing using IDE. Reading from the CPU bus also involves pre-reading.

The following exclusive rules apply when accessing (joining) the FIFO in the respective areas.

- Multiple write means cannot be simultaneously set for the same FIFO area.
- Multiple read means cannot be simultaneously set for the same FIFO area.
- Only one of JoinCPU_Wr, JoinCPU_Rd, or JoinDMA can be set to a single area.
- JoinIDE, JoinCPU_Wr, JoinCPU_Rd, and JoinDMAx{x=0,1} respectively can be set to only one area at any given time.
- JoinIDE, JoinCPU_Wr, JoinCPU_Rd, JoinDMAx{x=0,1} cannot be set to FIFO and media FIFO simultaneously.

The following exceptions are permitted for access means from USB (for devices) with duplicated read/write means for the same area. For example, JoinCPU_Wr and JoinDMAx{x=0,1}(DMAx{x=0,1}Control.Dir==0) can be set for writing to the OUT endpoint FIFO area. In this case, JoinCPU_Wr or JoinDMAx{x=0,1} must be set for writing from the CPU after ensuring that No OUT transactions are being performed. Similarly, JoinCPU_Rd and JoinDMAx{x=0,1}(DMAx{x=0,1}Control.Dir==1) can be set for reading from the IN endpoint FIFO area. In this case, JoinCPU_Rd or JoinDMAx{x=0,1} must be set for reading from the CPU after ensuring that no IN transactions are being performed.

Cases in which no transactions are performed include cases in which the ActiveUSB bit is cleared, when each endpoint is not joined to FIFO areas, and when ForceNAK is set.

1.7 Media FIFO Management

This section describes FIFO management for the media data transfer functions.

1.7.1 Media FIFO

The media FIFO is used to transfer media data between the IDE and memory.

The media FIFO is independent of the USB FIFO and is automatically assigned 64 bytes as a fixed usage area. Area settings by firmware are not required.

The usage area is controlled as the FIFO, and the data storage capacity is maintained. Set MediaFIFO_Control.MediaFIFO_Clr to clear this maintained state.

1.7.2 Media FIFO Access Methods

The media FIFO can be accessed via the CPU (register), CPU (DMA), and IDE.

1.7.2.1 Media FIFO Access Methods (Register Access)

To read from the media FIFO using CPU register access, MediaFIFO_Join.JoinCPU_Rd is set to 1, and data is read out using the FIFO_Rd_0,1 or FIFO_ByteRd register.

To write to the media FIFO using CPU register access, MediaFIFO_Join.JoinCPU_Wr is set to 1, and data is written to the FIFO_Wr_0,1 register.

The FIFO_RdRemain_H,L register indicates the amount of data that can be read out from the FIFO when set via MediaFIFO_Join.JoinCPU_Rd. Similarly, the FIFO_WrRemain_H,L register indicates the amount of remaining free space in the FIFO when set via MediaFIFO_Join.JoinCPU_Wr.

Note that data will be read from the FIFO during register dumping if MediaFIFO_Join.JoinCPU_Rd register is set for register dumping while debugging the firmware using ICE.

Accessing the CPU register requires exclusive joining of FIFO and media FIFO. That is, JoinCPU_Wr cannot be set concurrently for both media FIFO and a FIFO area. Likewise, JoinCPU_Rd cannot be set concurrently for both media FIFO and a FIFO area.

1.7.2.2 Media FIFO Access Methods (DMA)

To read from the media FIFO using CPU DMA access, a single channel is selected by the MediaFIFO_Join.JoinDMAx{x=0,1} bit for each DMA channel, the DMAx{x=0,1}_Control.Dir bit is set to 1, and data is read out using DMA procedures.

To write to the media FIFO using CPU DMA access, a single channel is selected by the MediaFIFO_Join.JoinDMAx{x=0,1} bit for each DMA channel, the DMAx{x=0,1}_Control.Dir bit is set to 0, and data is written using DMA procedures.

The DMAx{x=0,1}_Remain_H,L register indicates the amount of remaining data that can be read from the FIFO for a single channel selected by the MediaFIFO_Join.JoinDMAx{x=0,1} bit for

each DMA channel. It also indicates the remaining space that can be written to in the FIFO for a single channel selected by the MediaFIFO_Join.JoinDMAx{x=0,1} bit for each DMA channel.

Accessing the CPU DMA requires exclusive joining of FIFO and media FIFO. That is, JoinDMA0 or JoinDMA1 cannot be set concurrently for both media FIFO and a FIFO area.

1.7.2.3 Media FIFO Access Methods (IDE)

When the IDE accesses the media FIFO, the MediaFIFO_Join.JoinIDE bit is set to 1, and data is transferred using IDE procedures. The IDE transfer direction is determined by the IDE_Control.Dir bit.

IDE access requires exclusive joining of FIFO and media FIFO. That is, JoinIDE cannot be set concurrently to both media FIFO and FIFO areas.

1.7.2.4 Media FIFO Access Restrictions

The media FIFO in this LSI allows simultaneous register reading and writing from the CPU bus, reading and writing using DMA, and reading and writing using IDE. Pre-reading is also used when reading from the CPU bus.

The following exclusive rules apply for accessing (joining) the media FIFO.

- Multiple write means must not be set concurrently.
- Multiple read means must not be set concurrently.
- Only one of JoinCPU_Wr, JoinCPU_Rd, or JoinDMAx{x=0,1} can be set to the media FIFO.
- JoinIDE, JoinCPU_Wr, and JoinCPU_Rd,JoinDMAx{x=0,1} cannot be set to the FIFO and media FIFO simultaneously.

1.8 CPUIF

1.8.1 CPUIF Mode

The S2R72C05 CPUIF supports the two BUS modes, BE mode and strobe mode. The access method for writing depends on the particular BUS mode. BE mode allows writing using the write strobe signal indicating the write timing and the upper (CD[15:8]) and lower (CD[7:0]) byte enable signal. Strobe mode allows writing using the individual write strobe signals for the upper and lower bytes (refer to Figure 1-80). The read access methods are the same for either BUS mode, and both use the read strobe signal.

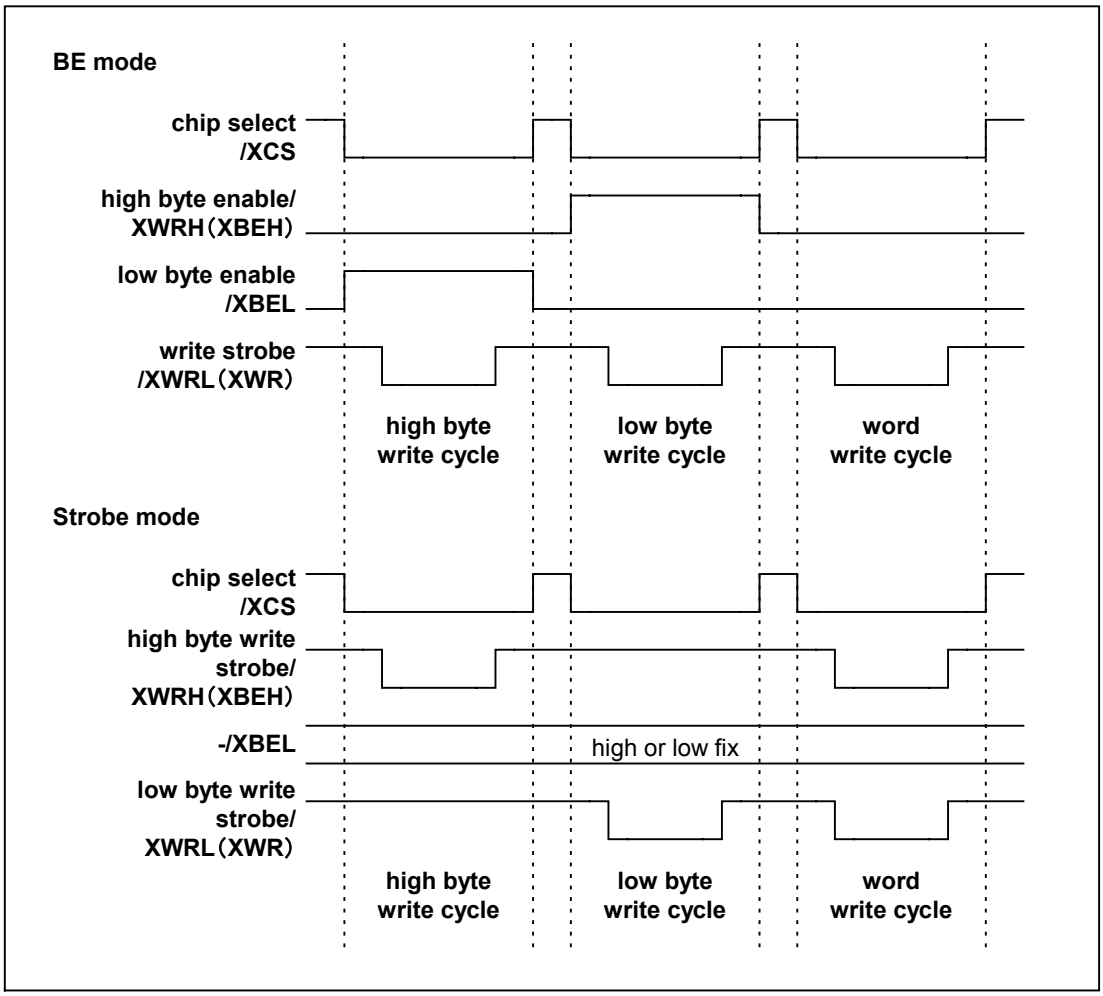


Figure 1-80 BUS modes

In addition to the BUS modes described above, the S2R72C05 CPUIF also supports the two BigEndian and LittleEndian ENDIAN modes to provide CPU endian support. In BigEndian mode, the even address register takes the upper position, the odd address register takes the lower position, the first byte data takes the upper position, and the second byte data takes the lower position. Conversely, in LittleEndian mode, the even address register takes the lower position, the odd address register takes

the upper position, the first byte data takes the lower position, and the second byte data takes the upper position. See “Appendix B Connection to Little-endian CPU” for detailed information on ENDIAN modes.

The S2R72C05 should be set to the correct BUS and ENDIAN modes for the CPU before use.

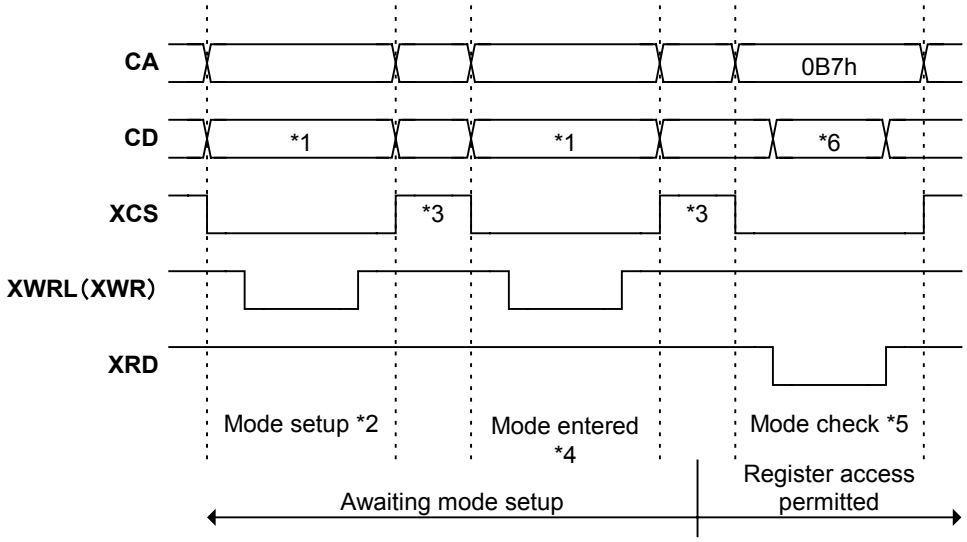
The following sections describe the mode setup procedures.

1.8.2 CPUIF Mode Setup

The S2R72C05 awaits CPUIF mode setup (while not initialized) after hardware resetting. No access is permitted in this state except for mode setup. Register access is allowed once mode setup is complete (initialized). The mode setup procedures are shown below.

1) Asserts the XCS and XWRL signals and writes the mode setup data in the specified data pattern (mode setup). For example, data is written using word to the memory space mapping C05.
2) The XCS signal is temporarily negated. For example, NOP is inserted or memory space other than that in which C05 is mapped is accessed.
3) The XCS signal is asserted again (mode confirmation). For example, the same operation is performed as in step 1).
4) The XCS signal is temporarily negated. For example, the same operation is performed as in step 2).

Normal register access is possible once this mode setup is complete. We recommend reading the ChipConfig register after setting to check the mode setup.



*1:

ENDIAN mode	BUS mode	CD[15:0]
BigEndian	Strobe	****_*00*_****_*00*
BigEndian	BE	****_*01*_****_*01*
LittleEndian	Strobe	****_*10*_****_*10*
LittleEndian	BE	****_*11*_****_*11*

*2: This must be set.
*3: Negate the chip select signal temporarily ("1").
*4: Assert the chip select signal. (e.g., same writing as for mode setup)
*5: Checking is recommended (register access is possible even if not checked.).
*6: Mode set can be checked.

Figure 1-81 CPUIF mode setup

Signals other than those included in Figure 1-81 in mode setup can be either High or Low, provided the AC ratings are satisfied.

CPUIF mode setting must be performed after hard resetting, and can be performed only during the uninitialized period after hard resetting.

The following CPUIF descriptions primarily explain Strobe and BigEndian modes.

1.8.3 Block Configuration

The S2R72C05 CPUIF (referred to as CPUIF hereafter) block configuration is illustrated in Figure 1-82.

The CPUIF consists of three blocks: REG, DMA0, and DMA1.

- REG: Controls access to the S2R72C05 register area
- DMA0: DMA channel 0
- DMA1: DMA channel 1

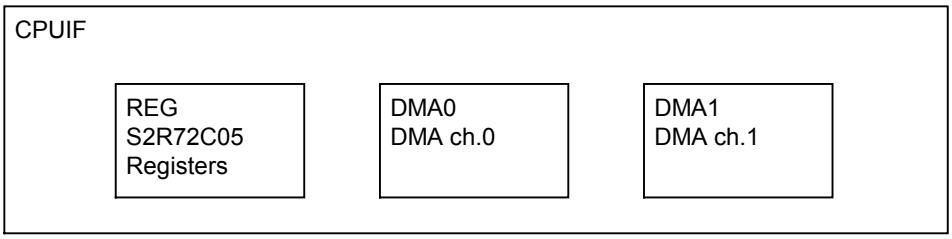


Figure 1-82 Block configuration

1.8.3.1 REG (S2R72C05 Registers)

These control access to the S2R72C05 register areas. They include the following functions.

- Synchronous register access
- FIFO access
- RAM_Rd access
- Asynchronous register access

1.8.3.1.1 Synchronous Register Access (Write)

Writes data from an external bus to the register synchronized with the internal clock.

1.8.3.1.2 Synchronous Register Access (Read)

Outputs register data to an external bus with the read (asserted for both XCS and XRD) time as the output enable period.

Significant registers with at least three bytes such as count values in the register read operation include those that maintain the lower byte register value for the uppermost byte read timing and output the value to an external bus when reading the lower byte to prevent an incorrect count from being read – for example, due to carry-over of count values during the access cycle.

1.8.3.1.3 FIFO Access (Write)

FIFO write access refers to writing to the FIFO_Wr_0,1, FIFO_ByteWr, and RAM_WrDoor_0,1 registers.

FIFO Access (Write) includes the following restrictions.

- Check the amount of data that can be written using the FIFO_WrRemain_H,L register before accessing after setting the AREAx{x=0-5}Join_0.JoinCPU_Wr bit or MediaFIFO_Join.JoinCPU_Wr. The RAM_WrDoor_0,1 register is not subject to this restriction.

- The FIFO should normally be accessed in word (2-byte) units. Control the strobe signal, accounting for the FIFO byte boundary when writing fractional (odd) bytes. Otherwise, use the FIFO_ByteWr register. See "1.8.3.1.5 FIFO Access Fractional Number Processing" for detailed information.
- The FIFO free space cannot be checked accurately if the FIFO_WrRemain_H,L register is checked immediately after writing to the FIFO_Wr_0,1 and FIFO_ByteWr registers. Always allow an interval of at least one CPU cycle before checking.
- The address cannot be checked accurately if the RAM_WrDoorAdrs_H,L register is checked immediately after writing to the RAM_WrDoor_0,1 register. Always allow an interval of at least one CPU cycle before checking.

1.8.3.1.4 FIFO Access (Read)

FIFO read access refers to reading from the FIFO_Rd_0,1 and FIFO_ByteRd registers.

FIFO Access (Read) includes the following restrictions.

- Check the amount of data that can be read using the FIFO_RdRemain_H,L register and the RdRemainvalid bit before accessing after setting the AREAx{x=0-5}Join_0.JoinCPU_Rd bit or MediaFIFO_Join.JoinCPU_Rd.
- Use the FIFO_Rd_0,1 register for word reading. For byte reading, use the FIFO_ByteRd register. Use byte reading if a byte boundary exists. Here, if the FIFO_Rd_0,1 register is used for word reading, valid data is output only on one side. See "1.8.3.1.5 FIFO Access Fractional Number Processing" for detailed information.

1.8.3.1.5 FIFO Access Fractional Number Processing

This section describes the relationship between FIFO access and data storage status in the FIFO when handling fractional (odd) numbers. The actual FIFO has a 4-byte width, but a 2-byte width is used here to simplify the discussion. The operations are the same for either 2-byte or 4-byte.

[Writing]

Under normal conditions, we recommend writing from a state without byte boundaries.

When setting the AREAnFIFO_Clr.ClrAREAx{x=0-5} or MediaFIFO_Control.MediaFIFO_Clr bit (for media data transfer) and word writing from a state without byte boundaries and with odd data quantities, write the last byte (data Z) of the continuous data to the High side only. This situation is illustrated in Figure 1-83 (1). Data is output from the USB in the sequence A, B, C, D ... X, Y, Z.

When writing from a state in which byte boundaries exist in the FIFO, first write the data to the Low side (write data K) to clear the byte boundaries before word writing (data L and M). This situation is illustrated in Figure 1-83 (2).

These are the normal writing operations.

A: Data A exists in FIFO
Free: No data exists in FIFO
start: Writing start point

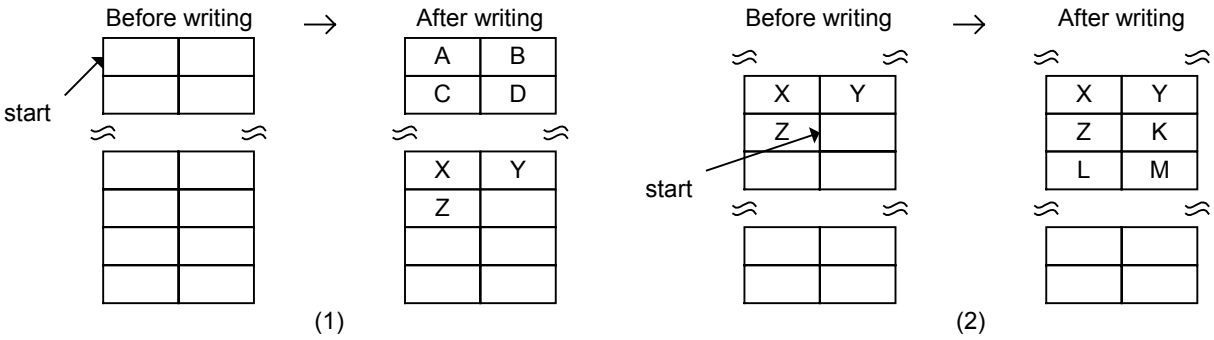


Figure 1-83 FIFO writing processing (normal operation)

The writing operations below require caution.

If byte writing is performed from a state in which byte boundaries exist in the FIFO, writing to High will be ignored, and only writing to Low will be performed (Figure 1-84 (3)). That is, the operation is identical to that when writing to Low. If writing to High only from a state in which byte boundaries exist in the FIFO, writing will be ignored (Figure 1-84 (4)).

If writing only to Low from a state in which byte boundaries do not exist in the FIFO, this writing will be ignored (Figure 1-84 (5)). If word writing from a state in which byte boundaries do not exist in the FIFO and the quantity that can be written is "1," writing to Low will be ignored, and only writing to High will be performed (Figure 1-84 (6)). That is, the operation is identical to that when writing to High.

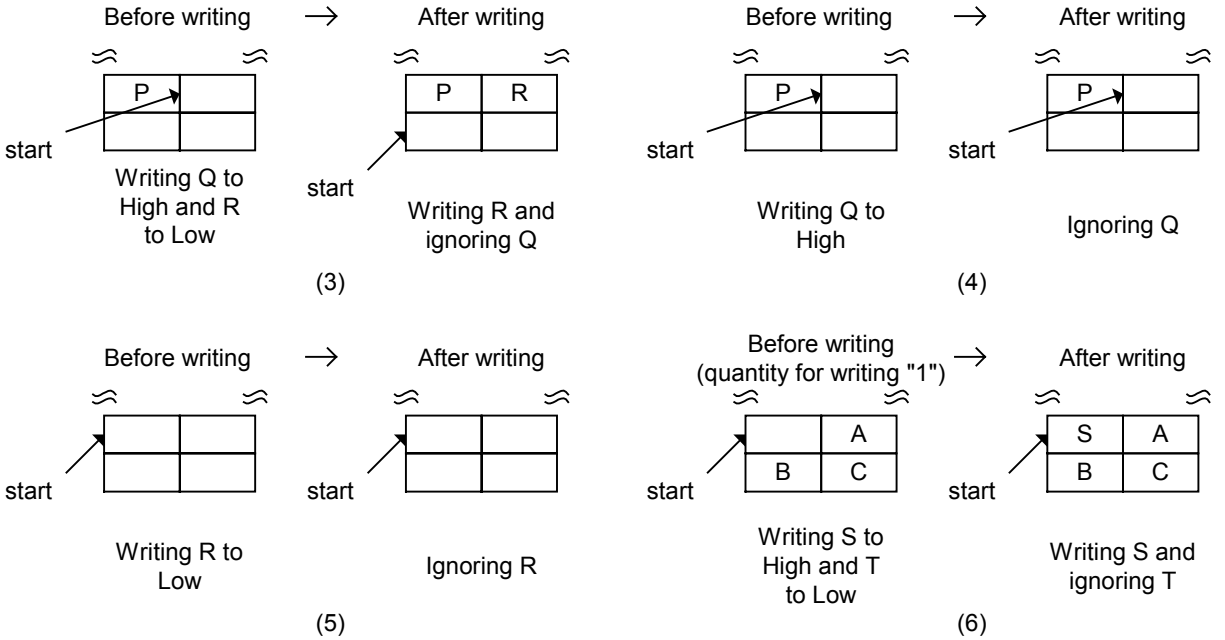


Figure 1-84 FIFO writing processing (operation requiring caution)

[Reading]

When there are no byte boundaries, word reading using the FIFO_Rd_0,1 register or byte reading using the FIFO_ByteRd register presents no problems. If there are byte boundaries, read bytes using the FIFO_ByteRd register. Word reading or byte reading after the byte boundaries have been cleared will lead to no problems.

Figure 1-85 (1) illustrates a case of word reading from a state without byte boundaries. Data A and B followed by data C and D are read out for each access. Figure 1-85 (2) illustrates a case of byte reading. Data is read out in the sequence A, B, C, and D for each access. These are the normal reading operations.

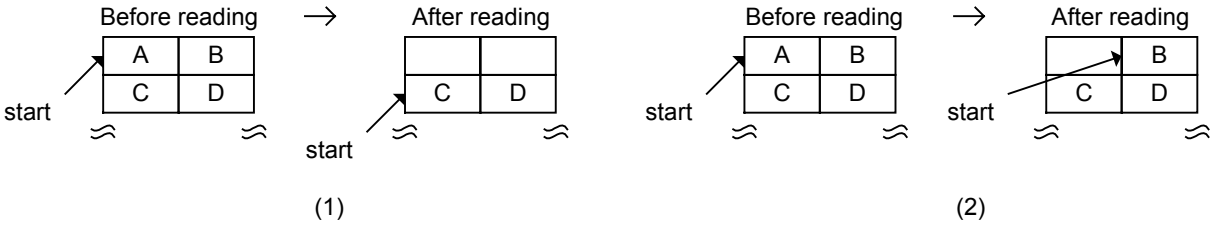


Figure 1-85 FIFO reading processing (normal operation)

The reading operations below require caution.

Figure 1-86 (3) illustrates a case of word reading using the FIFO_Rd_0,1 register from a state in which byte boundaries exist. Indeterminate data is output to High and data J is output to Low. The read pointer increments one byte at a time. Figure 1-86 (4) shows the operation performed when word reading using the FIFO_Rd_0,1 register from a state in which byte boundaries do not exist and one data byte remains. Data X is output to High and indeterminate data is output to Low. The read pointer increments one byte at a time.

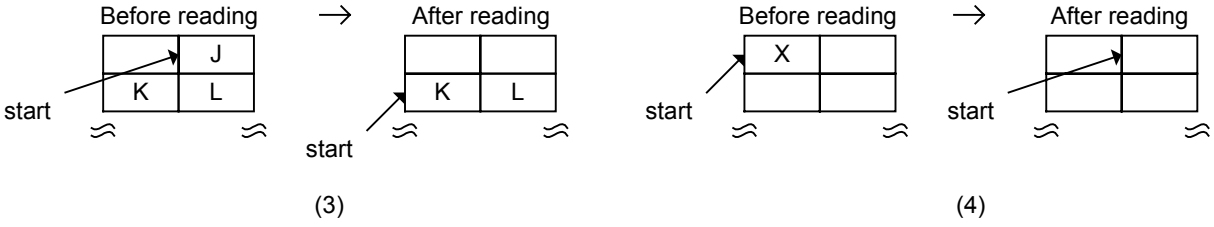


Figure 1-86 FIFO reading processing (operation requiring caution)

Described below is an example of reading processing for fractional number processing based on the details given above.

1) When data sent from the USB as 64 bytes is read out as 31 and 32 bytes

   (1) The CPUIF latches the 64-byte ready and starts the continuous read sequence.

   (2) 30 bytes of data are word-read by the FIFO_Rd_0,1 register or byte-read by the FIFO_ByteRd register.

(3) The 31st data byte is read by the FIFO_ByteRd register. Byte boundaries are formed.

(4) The 32nd data byte is byte-read. Byte reading using the FIFO_ByteRd register is recommended here. Data is output to Low if it was word-read using the FIFO_Rd_0,1 register. Byte boundaries are then cleared.

(5) The remaining 32 bytes of data are word-read by the FIFO_Rd_0,1 register or byte-read by the FIFO_ByteRd register.

2) When all 64 bytes of the data sent from the USB as 31 and 33 bytes with JoinCPU_Rd set are to be word-read by the FIFO_Rd_0,1 register

(1) The CPUIF latches the 31-byte ready and starts the continuous read sequence once 31 bytes of data are received from the USB.

(2) 30 bytes of data are word-read.

(3) The join is temporarily severed to clear the cached 31st data byte (byte boundary).

(4) The join is restored after the 33 bytes of data have been sent from the USB. (1 + 33 bytes)

(5) The CPUIF latches the 34-byte ready and starts the continuous operation sequence.

(6) 34 bytes of data are word-read.

1.8.3.1.6 RAM_Rd Access

As with synchronous register reading, data is output to the external bus with the read (asserted for both XCS and XRD) period as the output enable period. See “1.6.5.1 RAM Access Methods (RAM_Rd)” for detailed information.

1.8.3.1.7 Asynchronous Register Access (Writing)

A write pulse is created from the external write signal (XCS, XWRL, H), and the external bus data is written to the register.

1.8.3.1.8 Asynchronous Register Access (Reading)

As with synchronous register reading, register data is output to the external bus with the read (asserted for both XCS and XRD) period as the output enable period.

1.8.3.2 DMA0/DMA1(DMA ch.0/ch.1)

1.8.3.2.1 Basic Functions

The basic DMA operations are as shown below.

[Writing]

XDREQ is asserted to permit DMA transfers if free space is found in the FIFO for writing.

[Reading]

If data that can be read is found in the FIFO, XDREQ is asserted and DMA transfer enabled when reading is enabled.

There are three different DMA access methods, as described below.

- Access using XDACK

  This access method is used when ChipConfig.CS_Mode="0" and DMAx{x=0,1}_Config.DMA_Mode="0." Transfers are performed when the XDACK signal is asserted at the logic level set in ChipConfig.DACK_Level, and DMA access is identified.

- Access using XDACK&XCS

  This access method is used when ChipConfig.CS_Mode="1" and DMAx{x=0,1}_Config.DMA_Mode="0." Transfers are performed when the XDACK signal is asserted at the logic level set in ChipConfig.DACK_Level, XCS is asserted ("0"), and DMA access is identified.

- Access using DMA_Mode

  This access method is used when ChipConfig.CS_Mode="0" and DMAx{x=0,1}_Config.DMA_Mode="1." Transfers are performed with access to the DMAx{x=0,1}_Rd/WrData_H/L register identified as DMA access.

In all cases, DMAx{x=0,1}_Config.ActiveDMA should be set to "1."

DMA has two operating modes and one operating option.

- Count mode

  Performs DMA transfer for the preset number of counts.

  If free space for writing or data for reading is found in the internal FIFO and counts remain in the DMAx{x=0,1}_Count_HH,HL,LH,LL register, XDREQ is asserted and DMA transfer enabled.

- Free-run mode

  If free space for writing or data for reading is found in the internal FIFO, XDREQ is asserted and DMA transfer enabled.

- REQ assert count option

  This option supports CPU burst reads and writes. It can be used in either count or free-run modes. XDREQ is asserted, enabling DMA transfer, if the free space for writing or data quantity for reading is at least equal to the assert count set in the DMAx{x=0,1}_Config.ReqAssertCount[1:0] bit. Thus, transfer of the assert count quantity set is essentially guaranteed once XDREQ is asserted. However, XDREQ is asserted if the FIFO free space or data quantity is less than the set assert count or if the FIFO free space or data quantity is at least equal to the remaining count when set to count mode. In this case, the guaranteed transfer quantity is the remaining count.

DMA data processing is generally performed in word units. Data processing in byte units is possible only in count mode when the remaining count is "1." Shown below are the XDREQ assert conditions and correlations for permissible XDREQ assert transfer quantities for the various operating modes and options.

Table 1-64 Operating modes, options, and transfer start conditions

Count mode: Using ReqAssertCount option

Condition	Count mode (Count > 0)			
	Count ≥ Req		Count < Req	
	Ready ≥ Req	Ready < Req	Ready ≥ Count	Ready < Count
XDREQ	Assert	Negate	Assert	Negate
Transfer quantity	Req	-	Count	-

Free-run mode: Using ReqAssertCount option

Condition	Free-run mode	
	-	
	Ready ≥ Req	Ready < Req
XDREQ	Assert	Negate
Transfer quantity	Req	-

Count mode: Not using ReqAssertCount option

Condition	Count mode (Count > 0)		
	Count ≥ Ready		Count < Ready
	Ready ≥ 2	Ready < 2	Ready ≥ Count
XDREQ	Assert	Negate	Assert
Transfer quantity	Ready (Ready-1 if ready is odd number)	-	Count

Free-run mode: Not using ReqAssertCount option

Condition	Free-run mode	
	-	
	Ready ≥ 2	Ready < 2
XDREQ	Assert	Negate
Transfer quantity	Ready (Ready-1 if ready is odd number)	-

* Req here is the DMAx{x=0,1}Config.ReqAssertCount setting, Ready is the FIFO free space and data quantity, Count is the DMAx{x=0,1}Count_HH,HL,LH,LL value.

1.8.3.2.2 Terminal Setup

The XDREQx{x=1,0} and XDACKx{x=1,0} logic levels can be set using the ChipConfig register setting. Unless stated otherwise, the description below applies to negative logic for both XDREQ and XDACK.

1.8.3.2.3 Count mode (Write)

[Operation start]

Set the DMAx{x=0,1}_Count_HH,HL,LH,LL register to the count value, then set the DMAx{x=0,1}_Control.DMA_Go bit to "1." XDREQ can be asserted and DMA transfer enabled if at least 2 bytes of free space (DMA_Ready) exist for writing in the internal FIFO and counts remain. If only one byte of free space remains in the FIFO, XDREQ is asserted only when set to count mode and the remaining count number is "1."

If byte boundaries are formed in the FIFO when writing an odd number of bytes, use FIFO clear to clear the byte boundaries after the data has been transferred from the USB, then initiate the next write operation. For example, to write data every 31 bytes from the DMA and transfer data every 31 bytes from the USB, (1) set the DMA count to 31 and write 31 bytes of data; (2) wait for the 31 bytes of data to be transferred to the USB; (3) clear the FIFO after checking that the 31 bytes of data have been transferred from the USB, then repeat these steps.

"1" can be read in DMAx{x=0,1}_Control.DMA_Running bit until operations stop.

[Stopping operations]

The following two conditions must be met to stop operations.

- DMA transfer must be complete for the count number set in the DMAx{x=0,1}_Count_HH,HL,LH,LL register.
- "1" is written to the DMAx{x=0,1}_Control.DMA_Stop bit.

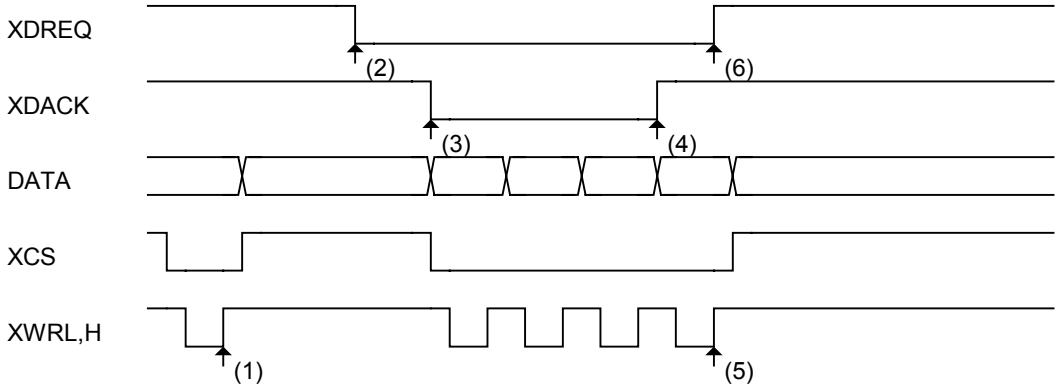
The CPU_IntStat..DMAx{x=0,1}_Cmp bit is set when DMA operations stop.

XDREQ is negated while the last access strobe is being asserted when the transfer is stopped by the DMAx{x=0,1}_Count_HH,HL,LH,LL register.

When transfers are stopped by the DMA_Stop bit, chip internal operations are stopped using the synchronous register access write timing, and XDREQ is negated. To stop the DMA using the DMA_Stop bit, the CPU DMAC (master) must be stopped first.

Figure 1-87 shows the operational timing for starting transfers in count mode and stopping transfers using the DMAx{x=0, 1}_Control.DMA_Stop bit before the preset number of counts have been transferred.

Ex 1: Transfer start conditions: Count (8 bytes) < FIFO free space (16 bytes); Transfer stoppage conditions: DMA_Stop

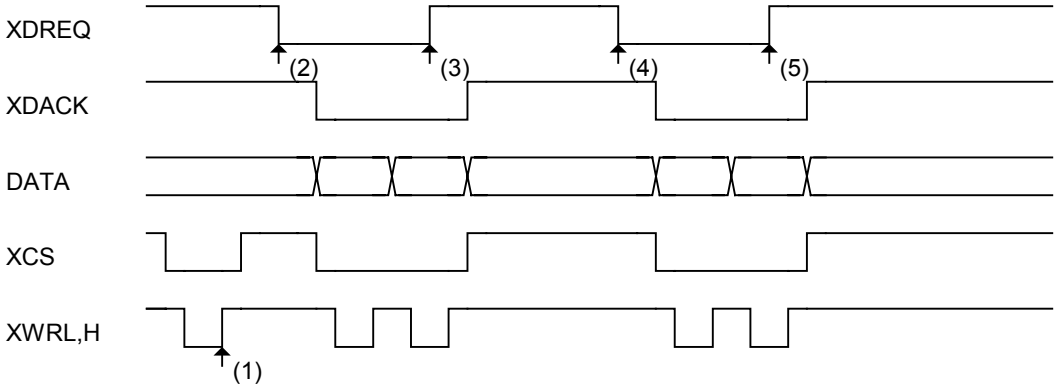


(1) DMA circuit operation is initiated by writing "1" to the DMA_Control.DMA_Go bit.
(2) Transferring data from the USB creates free space in the FIFO (DMA_Ready). XDREQ is asserted on receipt of DMA_Ready.
(3) XDACK is asserted and DMA transfer starts.
(4) The master stops and XDACK is negated before the count mode transfer quantity is complete.
(5) DMA circuit operation is stopped by writing "1" to the DMA_Control.DMA_Stop bit.
(6) XDREQ is negated on receipt of DMA circuit stop.

Figure 1-87 Count mode write timing 1

Figure 1-88 shows the operational timing for starting transfers in count mode and for stopping DMA transfers when the preset number of counts have been transferred.

Ex 2: Transfer start conditions: Count (8 bytes) > FIFO free space (4 bytes); Transfer stoppage conditions: Count 0



(1) DMA circuit operation is initiated by writing "1" to the DMA_Control.DMA_Go bit.
(2) Transferring data from the USB creates free space in the FIFO (DMA_Ready). XDREQ is asserted on receipt of DMA_Ready.
(3) XDREQ is negated when DMA_Ready disappears.
(4) Transferring data from the USB creates free space in the FIFO (DMA_Ready). XDREQ is asserted on receipt of DMA_Ready.
(5) XDREQ is negated when the DMA_Count last data is transferred. The DMA circuit is stopped once the DMA_Count quantity has been transferred.

Figure 1-88 Count mode write timing 2

1.8.3.2.4 Count mode (Read)

[Operation start]

Set the DMAx{x=0,1}_Count_HH,HL,LH,LL register to the count value, then set the DMAx{x=0,1}_Control.DMA_Go bit to "1." XDREQ is asserted if at least 2 bytes of data for reading are found in the internal FIFO and counts remain and reading from externally is enabled. If only one data byte remains in the FIFO, count mode is set and XDREQ is asserted only when the remaining count number is "1."

To describe a typical device operation, the ForceNAK bit is automatically set to "1" to return an NAK in the count mode reading operation when data exceeding the count number set in the DMAx{x=0,1}_Count_HH,HL,LH,LL register remains in the FIFO for the endpoint to which that DMA is connected. Similarly, if a short packet is received from the USB, the ForceNAK bit of the corresponding endpoint is automatically set to "1" and an NAK response is returned unless the DisAF_NAK_Short bit is set.

If byte boundaries are formed when reading an odd number of bytes, use FIFO clear to clear the byte boundaries before the next transfer. For example, to transfer data every 31 bytes from the USB and read data every 31 bytes from the DMA, (1) receive 31 bytes of data from the USB (ForceNAK is set here and the corresponding endpoint returns an NAK response); (2) read 31 bytes of data from the DMA; (3) clear the FIFO, clear the ForceNAK, permit receipt from the USB, then repeat these steps.

"1" can be read in the DMAx{x=0,1}_Control.DMA_Running bit until operations stop.

[Stopping operations]

The following two conditions must be met to stop operations.

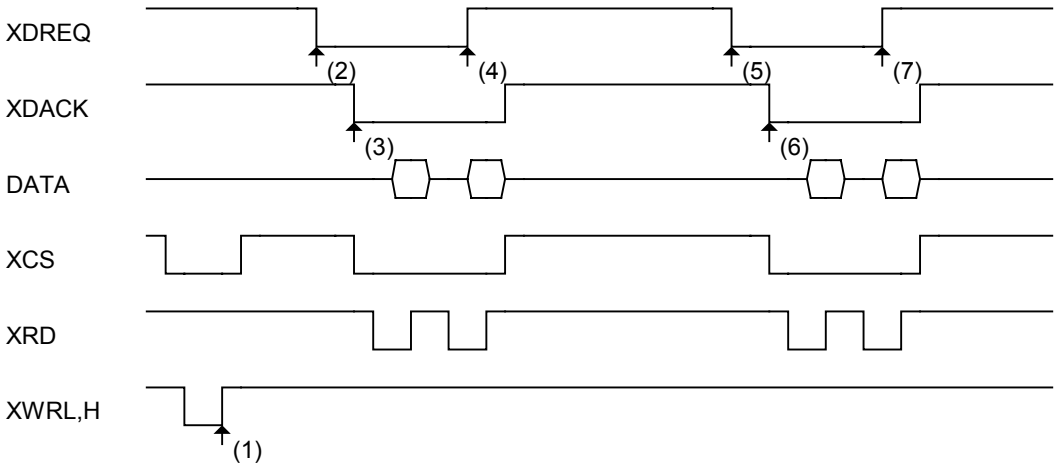
- DMA transfer must be complete for the count number set in the DMAx{x=0,1}_Count_HH,HL,LH,LL register.
- "1" is written to the DMAx{x=0,1}_Control.DMA_Stop bit.

XDREQ is negated while the last access strobe is being asserted when the operation is stopped by the DMAx{x=0,1}_Count_HH,HL,LH,LL register.

When transfers are stopped by the DMA_Stop bit, chip internal operations are stopped using the synchronous register access write timing, and XDREQ is negated. To stop the DMA using the DMA_Stop bit, the CPU DMAC (master) must be stopped first.

Figure 1-89 shows the operational timing for starting transfers in count mode, and the DMA transfer ending once the preset number of counts have been transferred.

Ex: Transfer start: Count (8 bytes) > FIFO data (4 bytes); Transfer stoppage: Count 0



(1) DMA circuit operation is initiated by writing "1" to the DMA_Control.DMA_Go bit.
(2) XDREQ is asserted when data is written to the FIFO from the USB, permitting data to be read from externally.
(3) XDACK is asserted and DMA transfer starts.
(4) XDREQ is negated when the FIFO data become empty.
(5) XDREQ is asserted when data is written to the FIFO from the USB, permitting data to be read from externally.
(6) XDACK is asserted and DMA transfer starts.
(7) XDREQ is negated when the DMA_Count last data is transferred.

Figure 1-89 Count mode read timing

1.8.3.2.5 Free-run Mode (Write)

[Operation start]

Set the DMAx{x=0,1}_Config.FreeRun bit, then write "1" to the DMAx{x=0,1}_Control.DMA_Go bit to "1." XDREQ is asserted and DMA transfer enabled if at least 2 bytes of free space are found for writing in the internal FIFO. If only one byte of free space remains in the FIFO, XDREQ is not asserted in free-run mode. To transfer data, see "1.8.3.2.3 Count mode (Write)."

"1" can be read in the DMAx{x=0,1}_Control.DMA_Running bit until operations stop.

[Stopping operations]

The following condition must be met to stop operations.

- "1" is written to the DMAx{x=0,1}_Control.DMA_Stop bit.

When transfers are stopped by the DMA_Stop bit, chip internal operations are stopped using the synchronous register access write timing, and XDREQ is negated. The CPU DMAC (master) must be stopped before stopping the DMA using the DMA_Stop bit.

The CPU_IntStat.DMAx{x=0,1}_Countup bit is set if the DMAx{x=0,1}_Count_HH,HL,LH,LL register value overflows during DMA transfers in

free-run mode. DMA transfer continues even if this occurs.
DMAx{x=0,1}_Count_HH,HL,LH,LL also continues and is counted.

The operational timing is the same as for count mode except when no restrictions apply by DMAx{x=0,1}_Count_HH,HL,LH,LL.

1.8.3.2.6 Free-run Mode (Read)

[Operation start]

Set the DMAx{x=0,1}_Config.FreeRun bit, then set the DMAx{x=0,1}_Control.DMA_Go bit to "1." XDREQ is asserted if at least 2 bytes of data for reading are found in the internal FIFO and reading from externally is enabled. If only one byte of valid data remains in the FIFO, DMA operation is not started. To transfer data, see "1.8.3.2.4 Count mode (Read)."

"1" can be read in the DMAx{x=0,1}_Control.DMA_Running bit until operations stop.

[Stopping operations]

The following condition must be met to stop operations.

- "1" is written to the DMAx{x=0,1}_Control.DMA_Stop bit.

When transfers are stopped by the DMA_Stop bit, chip internal operations are stopped using the synchronous register access write timing, and XDREQ is negated. To stop the DMA using the DMA_Stop bit, the CPU DMAC (master) must be stopped first.

The CPU_IntStat.DMAx{x=0,1}_Countup bit is set if the DMAx{x=0,1}_Count_HH,HL,LH,LL register value overflows during DMA transfers in free-run mode. DMA transfer continues even if this occurs, and DMAx{x=0,1}_Count_HH,HL,LH,LL also continues and is counted.

The operational timing is the same as for count mode, except when no restrictions apply by DMAx{x=0,1}_Count_HH,HL,LH,LL.

1.8.3.2.7 REQ Assert Count Option (Write)

[Operation start]

Set the assert count using the DMAx{x=0,1}_Config.ReqAssertCount[1:0] bit, then set the DMAx{x=0,1}_Control.DMA_Go bit to "1." XDREQ is asserted and DMA transfer enabled if the free space for writing in the internal FIFO is at least equal to the assert count number. Thus, transfer of the assert count quantity set is guaranteed once XDREQ is asserted. However, XDREQ is asserted even if the free space is smaller than the assert count when set to count mode and the free space exceeds the remaining count number. In this case, the guaranteed transfer quantity will be the remaining count.

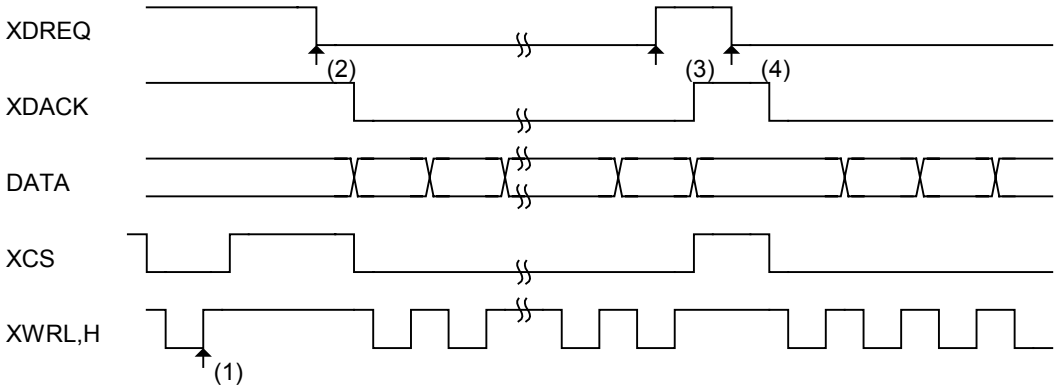
XDREQ is temporarily negated for each transfer quantity set in the ReqAssertCount[1:0] bit in this mode.

"1" can be read in the DMAx{x=0,1}_Control.DMA_Running bit until operations stop.

[Stopping operations]

See “1.8.3.2.3 Count mode (Write)” and “1.8.3.2.5 Free-run Mode (Write)”for detailed information on operation stoppage conditions.

Ex: Transfer start: REQ assert count (8-beat: 16-byte)



(1) DMA circuit operation is initiated by writing “1” to the DMA_Control.DMA_Go bit. XDREQ is not asserted since the value of DMA_Ready is less than the continuous transfer quantity.
(2) Transferring data from the USB creates free space in the FIFO (DMA_Ready) exceeding the continuous transfer quantity. XDREQ is asserted on receipt of DMA_Ready.
(3) XDREQ is negated when transfer of the continuous transfer quantity (REQ assert count) is completed.
(4) Free space for the next continuous transfer (DMA_Ready) is available once the first continuous transfer is complete. DREQ is asserted on receipt of DMA_Ready.

Figure 1-90 REQ assert count option write timing

1.8.3.2.8 REQ Assert Count Option (Read)

[Operation start]

Set the assert count using the DMAx{x=0,1}_Config.ReqAssertCount[1:0] bit, then set the DMAx{x=0,1}_Control.DMA_Go bit to “1.” XDREQ is asserted and DMA transfer enabled if the amount of data for reading in the internal FIFO is at least equal to the assert count number and reading from externally is enabled. Thus, transfer of the assert count quantity set is guaranteed once XDREQ is asserted. However, XDREQ is asserted even if the FIFO data quantity is less than the assert count when set to count mode and the data remaining exceeds the remaining count number. In this case, the guaranteed transfer quantity will be the remaining count.

XDREQ is temporarily negated for each transfer quantity set in the ReqAssertCount[1:0] bit in this mode.

“1” can be read in the DMAx{x=0,1}_Control.DMA_Running bit until operations stop.

[Stopping operations]

See “1.8.3.2.4 Count mode (Read)” and “1.8.3.2.6 Free-run Mode (Read)” for detailed information on operation stoppage conditions.

See Figure 1-89 and Figure 1-90 for detailed information on operational timing.

1.8.3.2.9 DMA FIFO Access Fractional Number Processing

See “1.8.3.1.5 FIFO Access Fractional Number Processing.” Note that the DMA lacks an opening for byte reading or writing.

1.9 Timer

The S2R72C05 includes a 16-bit timer. The timer resolution (clock) is approximately 1.1 ms (16-bit division at 60 MHz).

Down counting is performed by loading the count value set in the TimerSet_H,L register to the down counter. Interrupt status is issued when the count reaches 0x0.

1.9.1 Operating Mode

Two operating modes are supported: Single mode and free-run mode.

Single mode: The timer stops when the down counter value reaches 0x0.

Free-run mode: The TimerSet_H,L register value is loaded to the down counter when the count reaches 0x0, and the down count continues.

1.9.2 Operation Start/Stop

To start the timer operation, write the down count value to the TimerSet_H,L register. Once written, the value is loaded to the down counter, and the timer begins its countdown.

To stop the timer, set the TimerConfig.TimerStop bit to “1.”

1.10 IDE I/F

This section describes the IDE I/F function.

Use of the IDE I/F function requires writing “1” to the IDE_Config_1.ActiveIDE bit beforehand.

1.10.1 IDE Task File Register Access

This section describes methods for accessing the IDE task file register.

The firmware can access the IDE task file register via the registers from the IDE_RegAdrs register to IDE_RegConfig register. The read-write sequence on the IDE bus is performed by the LSI hardware until the end of the operation, freeing the firmware from operations involving access to the IDE task file register from the time the command is set until the sequence completion notification is issued via polling or interrupt.

The IDE_Rmod/IDE_Tmod register must be set appropriately beforehand in accordance with the IDE transfer mode when accessing the IDE task file register. This is because the IDE_Tmod register setting is used for transfers when the IDE bus XHCS0=0/HDA[2:0]=0 and the IDE_Rmod register setting is used for all other addresses.

1.10.1.1 Reading from IDE Task File Register

Reading from the IDE task file register is performed by setting the address to be accessed in the IDE_RegAdrs.IDE_RegAddress[3:0] bit beforehand or at the same time, then writing “1” to the IDE_RegAdrs.IDE_RdReg bit.

Reading ends when the IDE_RegAdrs.IDE_RdReg bit becomes “0” and the IDE_IntStat.IDE_RegCmp bit becomes “1.”

The data read from the IDE task file register is stored in the IDE_RdRegValue register.

1.10.1.2 Writing to IDE Task File Register

Writing to the IDE task file register is performed by setting the data to be written in the IDE_WrRegValue register beforehand, setting the address to be accessed in the IDE_RegAdrs.IDE_RegAddress[3:0] bit beforehand or at the same time, then writing “1” to the IDE_RegAdrs.IDE_WrReg bit.

Writing ends when the IDE_RegAdrs.IDE_WrReg bit becomes “0” and the IDE_IntStat.IDE_RegCmp bit becomes “1.”

1.10.1.3 Sequential Writing to IDE Task File Register

The IDE_SeqWrRegControl register can be used for sequentially writing to the IDE task file register with up to 16 address/data sets set beforehand.

Write up to 16 sets of addresses to the IDE_SeqWrRegAdrs.IDE_SeqWrRegAddress[3:0] in advance and up to 16 corresponding sets of byte data to the IDE_SeqWrRegValue register. Since word access is used for XHCS0=0/HDA[2:0]=0 addresses, data must be written twice in the sequence of lower, then upper. Two address/data pairs are used. Sequential writing is then performed when "1" is subsequently written to the IDE_SeqWrRegControl.IDE_SeqWrReg bit. The sequential writing operation ends when the IDE_SeqWrRegControl.IDE_SeqWrReg bit becomes "0" and the IDE_IntStat.IDE_SeqWrRegCmp bit becomes "1."

"1" can be written to the IDE_SeqWrRegControl.IDE_SeqWrRegClr bit and the address/data can be discarded if the address/data set in advance is not required before the start of sequential writing.

If the firmware performs normal IDE task file register reading or writing during sequential writing, the IDE_IntStat.IDE_RegErr bit becomes "1" and the reading and writing is ignored.

1.10.1.4 Auto Status Register Reading from IDE Task File Register

If the IDE bus HINTRQ is asserted when "1" has been set to the IDE_RegConfig.EnAutoStsRd bit, the LSI automatically reads the IDE status register (XHCS0=0,HDA[2:0]=7), and the IDE_IntStat.IDE_CompleteINTRQ bit becomes "1" once the data read has been stored in the IDE_RdRegValue register.

If the firmware performs normal IDE task file register reading or writing from auto status register reading until the time the firmware reads the IDE_RdRegValue_1 register, the IDE_IntStat.IDE_RegErr bit becomes "1," and reading and writing is ignored.

1.10.2 PIO Access

This section describes the DMA function using PIO mode.

DMA using PIO mode involves DMA transfers with IDE_Tmod register settings. The IDE_Tmod register must be appropriately set in advance for the specific IDE transfer mode.

1.10.2.1 PIO Read DMA

The PIO read DMA operates as follows:

"0" is written to both the IDE_Config_0.DMA and IDE_Config_0.Ultra bits.

The number of transfer bytes is set in the IDE_CountH/M/L register.

"1" is written to the IDE_Control.IDE_Clr bit, initializing the IDE circuit. (This is not essential.)

Writing "1" to the IDE_Control.IDE_Go bit will start the DMA operation. The IDE_CountH/M/L register contents are reduced as the transfer proceeds. When the count reaches 0 and the IDE DMA operation and writing to the internal FIFO of data read from the IDE are complete, the IDE_Control.IDE_Go bit assumes the value of "0," and the IDE_IntStat.IDE_Cmp bit assumes the value of "1," ending the DMA operation.

Writing "0" to the IDE_Control.IDE_Go bit during the DMA transfer aborts the DMA operation and ends the DMA. The data transferred to the internal FIFO is valid, but the IDE transfer byte quantity cannot be managed precisely, since the data remaining in the intermediate circuit buffer is discarded.

1.10.2.2 PIO Write DMA

The PIO write DMA operates as follows:

"0" is written to both the IDE_Config_0.DMA and IDE_Config_0.Ultra bits.

The number of transfer bytes is set in the IDE_CountH/M/L register.

"1" is written to the IDE_Control.IDE_Clr bit, initializing the IDE circuit. (This is not essential.)

Writing "1" to the IDE_Control.IDE_Go bit will start the DMA operation. The IDE_CountH/M/L register contents are reduced as the transfer proceeds. When the count reaches 0 and DMA writing to the IDE is complete, the IDE_Control.IDE_Go bit assumes the value of "0," and the IDE_IntStat.IDE_Cmp bit assumes the value of "1," ending the DMA operation.

Writing "0" to the IDE_Control.IDE_Go bit during the DMA transfer aborts the DMA operation and ends the DMA. The data written to the IDE up to this point is valid, but the IDE transfer byte quantity cannot be managed precisely, since the data remaining in the intermediate circuit buffer is discarded.

1.10.3 Multi-Word DMA

This section describes the DMA function using multi-word DMA mode.

DMA using multi-word DMA mode involves DMA transfers with IDE_Tmod register settings. The IDE_Tmod register must be appropriately set in advance for the specific IDE transfer mode.

1.10.3.1 Multi-Word DMA Read

The multi-word DMA read operates as follows:

"1" is written to the IDE_Config_0.DMA bit, and "0" is written to the IDE_Config_0.Ultra bit.

The appropriate values are written to the IDE_Config_1.DelayStrobe and IDE_Config_1.InterLock bits.

The number of transfer bytes is set in the IDE_CountH/M/L register.

"1" is written to the IDE_Control.IDE_Clr bit, initializing the IDE circuit. (This is not essential.)

Writing "1" to the IDE_Control.IDE_Go bit will start the DMA operation. The IDE_CountH/M/L register contents are reduced as the transfer proceeds. When the count reaches 0 and the IDE DMA operation and writing to the internal FIFO of data read from the IDE are complete, the IDE_Control.IDE_Go bit assumes the value of "0," and the IDE_IntStat.IDE_Cmp bit assumes the value of "1," ending the DMA operation.

Writing "0" to the IDE_Control.IDE_Go bit during the DMA transfer aborts the DMA operation and ends the DMA. The data transferred to the internal FIFO is valid, but the IDE transfer byte quantity cannot be managed precisely, since the data remaining in the intermediate circuit buffer is discarded.

1.10.3.2 Multi-Word DMA Write

The multi-word DMA write operates as follows:

"1" is written to the IDE_Config_0.DMA bit, and "0" is written to the IDE_Config_0.Ultra bit.

The appropriate values are written to the IDE_Config_1.DelayStrobe and IDE_Config_1.InterLock bits.

The number of transfer bytes is set in the IDE_CountH/M/L register.

"1" is written to the IDE_Control.IDE_Clr bit, initializing the IDE circuit. (This is not essential.)

Writing "1" to the IDE_Control.IDE_Go bit will start the DMA operation. The IDE_CountH/M/L register contents are reduced as the transfer proceeds. When the count reaches 0 and DMA writing to the IDE is complete, the IDE_Control.IDE_Go bit assumes the value of "0," and the IDE_IntStat.IDE_Cmp bit assumes the value of "1," ending the DMA operation.

Writing "0" to the IDE_Control.IDE_Go bit during the DMA transfer aborts the DMA operation and ends the DMA. The data written to the IDE up to this point is valid, but the IDE transfer byte quantity cannot be managed precisely, since the data remaining in the intermediate circuit buffer is discarded.

1.10.4 Ultra DMA

This section describes the DMA function using Ultra DMA mode.

DMA using Ultra DMA mode involves DMA transfers with IDE_Umod register settings. The IDE_Umod register must be appropriately set in advance for the specific IDE transfer mode.

1.10.4.1 Ultra DMA Read

The Ultra DMA read operates as follows:

"1" is written to both the IDE_Config_0.DMA and IDE_Config_0.Ultra bits.

The number of transfer bytes is set in the IDE_CountH/M/L register.

"1" is written to the IDE_Control.IDE_Clr bit, initializing the IDE circuit. (This is not essential.)

Writing "1" to the IDE_Control.IDE_Go bit will start the DMA operation. The IDE_CountH/M/L register contents are reduced as the transfer proceeds. When the count reaches 0 and the IDE DMA operation and writing to the internal FIFO of data read from the IDE are complete, the IDE_Control.IDE_Go bit assumes the value of "0," and the IDE_IntStat.IDE_Cmp bit assumes the value of "1," ending the DMA operation.

Writing "0" to the IDE_Control.IDE_Go bit during the DMA transfer aborts the DMA operation and ends the DMA. The data transferred to the internal FIFO is valid, but the IDE transfer byte quantity cannot be managed precisely, since the data remaining in the intermediate circuit buffer is discarded. A host terminate notification is also issued to the connected IDE device.

If a device terminate notification is issued during the DMA transfer, the IDE_IntStat.DetectTerm bit assumes the value of "1," and the connected IDE device stops the transfer. However, a value of "0" must be written to the IDE_Control.IDE_Go bit to end the DMA, since the DMA started will not otherwise end.

1.10.4.2 Ultra DMA Write

The Ultra DMA write operates as follows:

"1" is written to both the IDE_Config_0.DMA and IDE_Config_0.Ultra bits.

The number of transfer bytes is set in the IDE_CountH/M/L register.

"1" is written to the IDE_Control.IDE_Clr bit, initializing the IDE circuit. (This is not essential.)

Writing "1" to the IDE_Control.IDE_Go bit will start the DMA operation. The IDE_CountH/M/L register contents are reduced as the transfer proceeds. When the count reaches 0 and DMA writing to the IDE is complete, the IDE_Control.IDE_Go bit assumes the value of "0," and the IDE_IntStat.IDE_Cmp bit assumes the value of "1," ending the DMA operation.

Writing "0" to the IDE_Control.IDE_Go bit during the DMA transfer aborts the DMA operation and ends the DMA. The data written to the IDE up to this point is valid, but the IDE transfer byte quantity cannot be managed precisely, since the data remaining in the intermediate circuit buffer is discarded. A host terminate notification is also issued to the connected IDE device.

If a device terminate notification is issued during the DMA transfer, the IDE_IntStat.DetectTerm bit assumes the value of "1," and the connected IDE device stops the transfer. However, a value of "0" must be written to the IDE_Control.IDE_Go bit to end the DMA, since the DMA started will not otherwise end.

1.10.5 IDE Transfer Mode Settings

The following tables show the settings used for the various IDE transfer modes.

- Register Mode (when the IDE task file register is accessed by the firmware in cases other than when XHCS0=0 and HDA=0)

Mode	Register settings (mandatory)		Register settings (recommended)
	IDE_Config_0.Ultra	IDE_Config_0.DMA	IDE_Rmod
0	No effect	No effect	FFh
1	No effect	No effect	F1h
2	No effect	No effect	F0h
3	No effect	No effect	22h
4	No effect	No effect	10h

- PIO Mode (when the IDE task file register is accessed by the firmware in cases when XHCS0=0 and HDA=0)

Mode	Register settings (mandatory)		Register settings (recommended)
	IDE_Config_0.Ultra	IDE_Config_0.DMA	IDE_Tmod
0	No effect	No effect	FFh
1	No effect	No effect	88h
2	No effect	No effect	44h
3	No effect	No effect	22h
4	No effect	No effect	10h

- PIO Mode (For DMA transfer)

Mode	Register settings (mandatory)		Register settings (recommended)
	IDE_Config_0.Ultra	IDE_Config_0.DMA	IDE_Tmod
0	0	0	FFh
1	0	0	88h
2	0	0	44h
3	0	0	22h
4	0	0	10h

- Multi-Word DMA Mode (For DMA transfer)

Mode	Register settings (mandatory)		Register settings (recommended)
	IDE_Config_0.Ultra	IDE_Config_0.DMA	IDE_Tmod
0	0	1	BBh
1	0	1	20h
2	0	1	10h

- Ultra Mode (For DMA transfer, DATA-OUT)

Mode	Register settings (mandatory)		Register settings (recommended)
	IDE_Config_0.Ultra	IDE_Config_0.DMA	IDE_Umod
0	1	1	06h
1	1	1	04h
2	1	1	03h
3	1	1	02h
4	1	1	01h
5	1	1	00h

Note: Data can be received in all modes regardless of the IDE_Umod register setting for DATA-IN inputting data from the IDE bus in Ultra Mode.

1.11 Boundary Scan (JTAG)

Boundary Scan (JTAG) can be used when the TEST terminal is set to L (the default setting). A Boundary Scan consists of a BSR (Boundary Scan Register) conforming to JTAG (IEEE 1149.1) specifications, the scan path to connect it, and a TAP controller. Boundary scan connection information can be provided in BSDL format.

1.11.1 Supported Instructions

The LSI JTAG instruction bit width is 4 bits, and the following JTAG instructions are supported:

Table 1-65 JTAG instruction codes

Instruction	Description	Code
SAMPLE/PRELOAD	Scans LSI internal status into BSR and sets data.	0010
BYPASS	Bypasses scan path using BSR.	1111
EXTEST	Checks device physical connection.	0000
CLAMP	Bypasses scan path while retaining output values.	0011
HIGHZ	Sets all output to Hi-Z.	0100
IDCODE	Outputs specified DEVICE_CODE.	0001

1.11.2 DEVICE_CODE

The DEVICE_CODE corresponding to the IDCODE instruction consists of the following components.

Table 1-66 DEVICE_CODE

Version	1
Part Number	0x0015
Manufacturer	0x0BE

The DEVICE_CODE response to the IDCODE instruction will therefore be

0001_0000000000010101_00010111110_1.

1.11.3 Terminals Excluded from Boundary Scan

The following terminals on this LSI are excluded from scanning, since they do not include boundary scan cells.

DP_A, DM_A, DP_B, DM_B, R1_A, R1_B, XI, XO, VBUS_B, and TEST

2 REGISTERS

Registers are either initial, device/host common, device, or host registers. The device register and host register switch mapping using the HostDeviceSel.HOSTxDEVICE bit setting. The device register is selected if the HostDeviceSel.HOSTxDEVICE bit is set to "0"; the host register is selected if it is set to "1." Changing the bit setting will not clear the register setting.

Do not write "1" to the reserve register bit.

The register map for this LSI is defined as big endian. When connecting to a little endian CPU, refer to "Appendix B: Connection to Little-endian CPU".

2.1 Initial Register Map

Only this "Initial register" can be accessed with the S2R72C05 from the time it is hard-reset until the initial register is set and the setting is enabled (uninitialized period). See "1.8.2 CPUIF Mode Setup" for detailed information.

Word Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0xXXX*	CPUIF_MODE	W	0xXXXX						*CPU_Endian*	*BusMode*	
									CPU_Endian	*BusMode*	

* The address is "Don't care." All write access to the LSI is treated as write access to this register for processing during the uninitialized period.

2.2 Device/Host Common Register Map

Registers that can be read from or written to even in SLEEP or SNOOZE states are indicated in ***bold italic***.

All other registers can be read from or written to in ACTIVE60, ACT_DEVICE, ACT_HOST, and ACT_ALL states. (See "1.5 Power Management Functions" for detailed information on individual states.)

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x000	***MainIntStat***	R/(W)	0x00	***DeviceIntStat***	***HostIntStat***	CPU_IntStat	IDE_IntStat	MediaFIFO_Intstat	AREAnIntStat		***FinishedPM***
0x001	***DeviceIntStat***	R/(W)	0x00	***VBUS_Changed***		***D_SIE_IntStat***	D_BulkIntStat	RcvEP0SETUP	D_FIFO_IntStat	D_EP0IntStat	D_EPrIntStat
0x002	***HostIntStat***	R/(W)	0x00	***VBUS_Err***	***LineStateChanged***	H_SIE_IntStat_1	H_SIE_IntStat_0	H_FrameIntStat	H_FIFO_IntStat	H_CH0IntStat	H_CHrIntStat
0x003	CPU_IntStat	R/(W)	0x00	RAM_RdCmp	C_TimerCmp			DMA1_Countup	DMA1_Cmp	DMA0_Countup	DMA0_Cmp
0x004	IDE_IntStat	R/(W)	0x00	IDE_RegCmp	IDE_RegErr	IDE_SeqWrRegCmp	CompleteINTRQ		IDE_Cmp	DetectINTRQ	DetectTerm
0x005	MediaFIFO_IntStat	R/(W)	0x01		MediaIDE_Cmp				FIFO_NotEmpty	FIFO_Full	FIFO_Empty
0x006			0xXX								
0x007			0xXX								
0x008			0xXX								
0x009			0xXX								
0x00A			0xXX								
0x00B			0xXX								
0x00C			0xXX								
0x00D			0xXX								
0x00E			0xXX								
0x00F			0xXX								
Byte Addr.	**Register Name**	**R/W**	**Reset**	**bit15**	**bit14**	**bit13**	**bit12**	**bit11**	**bit10**	**bit9**	**bit8**
				bit7	**bit6**	**bit5**	**bit4**	**bit3**	**bit2**	**bit1**	**bit0**
0x010	***MainIntEnb***	R/W	0x00	***EnDeviceIntStat***	***EnHostIntStat***	EnCPU_IntStat	EnIDE_IntStat	EnMediaFIFO_IntStat	EnAREAnIntStat		***EnFinishedPM***
0x011	***DeviceIntEnb***	R/W	0x00	***EnVBUS_Changed***		***EnD_SIE_IntStat***	EnD_BulkIntStat	EnRcvEP0SETUP	EnD_FIFO_IntStat	EnD_EP0IntStat	EnD_EPrIntStat
0x012	***HostIntEnb***	R/W	0x00	***EnVBUS_Err***	***EnLineStateChanged***	EnH_SIE_IntStat_1	EnH_SIE_IntStat_0	EnH_FrameIntStat	EnH_FIFO_IntStat	EnH_CH0IntStat	EnH_CHrIntStat
0x013	CPU_IntEnb	R/W	0x00	EnRAM_RdCmp	EnC_TimerCmp			EnDMA1_Countup	EnDMA1_Cmp	EnDMA0_Countup	EnDMA0_Cmp
0x014	IDE_IntEnb	R/W	0x00	EnIDE_RegCmp	EnIDE_RegErr	EnIDE_SeqWrRegCmp	EnCompleteINTRQ		EnIDE_Cmp	EnDetectINTRQ	EnDetectTerm
0x015	MediaFIFO_IntEnb	R/W	0x00		EnMediaIDE_Cmp				EnFIFO_NotEmpty	EnFIFO_Full	EnFIFO_Empty
0x016			0xXX								
0x017			0xXX								
0x018			0xXX								
0x019			0xXX								
0x01A			0xXX								
0x01B			0xXX								
0x01C			0xXX								
0x01D			0xXX								
0x01E			0xXX								
0x01F			0xXX								

Registers that can be read from or written to even in SLEEP or SNOOZE states are indicated in ***bold italic***.

All other registers can be read from or written to in ACTIVE60, ACT_DEVICE, ACT_HOST, and ACT_ALL states. (See "1.5 Power Management Functions" for detailed information on individual states.)

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x020	***RevisionNum***	R	0x60	***RevisionNumer***							
0x021	***ChipReset***	W	0xXX								***AllReset***
0x022	***PM_Control_0***	R/W	0x00	***GoSLEEP***	***GoSNOOZE***	***GoActive60***	***GoActDevice***	***GoActHost***	***GoActAllDev***	***GoActAllHost***	
0x023	***PM_Control_1***	R	0x00	***MonActFunc***				***PM_State[3:0]***			
0x024	***WakeupTim_H***	R/W	0x00	***WakeupTim[15:8]***							
0x025	***WakeupTim_L***	R/W	0x00	***WakeupTim[7:0]***							
0x026	***H_USB_Control***	R/W	0x00	***VBUS_Enb***							
0x027	***H_XcvrControl***	R/W	0x91	***TermSelect***	***RemoveRPD***	***XcvrSelect[1:0]***				***OpMode[1:0]***	
0x028	***D_USB_Status***	R/W	0xXX	***VBUS***	FSxHS					***LineState[1:0]***	
0x029	***H_USB_Status***	R	0xXX	***VBUS_State***						***LineState[1:0]***	
0x02A			0xXX								
0x02B			0xXX								
0x02C			0xXX								
0x02D			0xXX								
0x02E			0xXX								
0x02F			0xXX								
Byte Addr.	**Register Name**	**R/W**	**Reset**	**bit15**	**bit14**	**bit13**	**bit12**	**bit11**	**bit10**	**bit9**	**bit8**
				bit7	**bit6**	**bit5**	**bit4**	**bit3**	**bit2**	**bit1**	**bit0**
0x030	FIFO_Rd_0	R	0xXX	FIFO_Rd_0[7:0]							
0x031	FIFO_Rd_1	R	0xXX	FIFO_Rd_1[7:0]							
0x032	FIFO_Wr_0	W	0xXX	FIFO_Wr_0[7:0]							
0x033	FIFO_Wr_1	W	0xXX	FIFO_Wr_1[7:0]							
0x034	FIFO_RdRemain_H	R	0x00	RdRemainValid			RdRemain[12:8]				
0x035	FIFO_RdRemain_L	R	0x00	RdRemain[7:0]							
0x036	FIFO_WrRemain_H	R	0x00				WrRemain[12:8]				
0x037	FIFO_WrRemain_L	R	0x00	WrRemain[7:0]							
0x038	FIFO_ByteRd	R	0xXX	FIFO_ByteRd[7:0]							
0x039			0xXX								
0x03A	FIFO_ByteWr	W	0xXX	FIFO_ByteWr[7:0]							
0x03B			0xXX								
0x03C			0xXX								
0x03D			0xXX								
0x03E			0xXX								
0x03F			0xXX								

Registers that can be read from or written to even in SLEEP or SNOOZE states are indicated in ***bold italic***.

All other registers can be read from or written to in ACTIVE60, ACT_DEVICE, ACT_HOST, and ACT_ALL states. (See "1.5 Power Management Functions" for detailed information on individual states.)

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x040	RAM_RdAdrs_H	R/W	0x00					RAM_RdAdrs[12:8]			
0x041	RAM_RdAdrs_L	R/W	0x00	RAM_RdAdrs[7:2]							
0x042	RAM_RdControl	R/W	0x00	RAM_GoRdCBW_CSW	RAM_GoRd						
0x043	RAM_RdCount	R/W	0x00			RAM_RdCount[5:2]					
0x044	RAM_WrAdrs_H	R/W	0x00					RAM_WrAdrs[12:8]			
0x045	RAM_WrAdrs_L	R/W	0x00	RAM_WrAdrs[7:0]							
0x046	RAM_WrDoor_0	W	0xXX	RAM_WrDoor_0[7:0]							
0x047	RAM_WrDoor_1	W	0xXX	RAM_WrDoor_1[7:0]							
0x048	MediaFIFO_Control	W	0xXX								MediaFIFO_Clr
0x049	ClrAllMediaFIFO_Join	W	0xXX	ClrJoinIDE				ClrJoinDMA1	ClrJoinDMA0	ClrJoinCPU_Rd	ClrJoinCPU_Wr
0x04A	MediaFIFO_Join	R/W	0x00	JoinIDE				JoinDMA1	JoinDMA0	JoinCPU_Rd	JoinCPU_Wr
0x04B			0xXX								
0x04C			0xXX								
0x04D			0xXX								
0x04E			0xXX								
0x04F			0xXX								
Byte Addr.	**Register Name**	**R/W**	**Reset**	**bit15**	**bit14**	**bit13**	**bit12**	**bit11**	**bit10**	**bit9**	**bit8**
				bit7	**bit6**	**bit5**	**bit4**	**bit3**	**bit2**	**bit1**	**bit0**
0x050	RAM_Rd_00	R	0x00	RAM_Rd_00[7:0]							
0x051	RAM_Rd_01	R	0x00	RAM_Rd_01[7:0]							
0x052	RAM_Rd_02	R	0x00	RAM_Rd_02[7:0]							
0x053	RAM_Rd_03	R	0x00	RAM_Rd_03[7:0]							
0x054	RAM_Rd_04	R	0x00	RAM_Rd_04[7:0]							
0x055	RAM_Rd_05	R	0x00	RAM_Rd_05[7:0]							
0x056	RAM_Rd_06	R	0x00	RAM_Rd_06[7:0]							
0x057	RAM_Rd_07	R	0x00	RAM_Rd_07[7:0]							
0x058	RAM_Rd_08	R	0x00	RAM_Rd_08[7:0]							
0x059	RAM_Rd_09	R	0x00	RAM_Rd_09[7:0]							
0x05A	RAM_Rd_0A	R	0x00	RAM_Rd_0A[7:0]							
0x05B	RAM_Rd_0B	R	0x00	RAM_Rd_0B[7:0]							
0x05C	RAM_Rd_0C	R	0x00	RAM_Rd_0C[7:0]							
0x05D	RAM_Rd_0D	R	0x00	RAM_Rd_0D[7:0]							
0x05E	RAM_Rd_0E	R	0x00	RAM_Rd_0E[7:0]							
0x05F	RAM_Rd_0F	R	0x00	RAM_Rd_0F[7:0]							

Registers that can be read from or written to even in SLEEP or SNOOZE states are indicated in ***bold italic***.

All other registers can be read from or written to in ACTIVE60, ACT_DEVICE, ACT_HOST, and ACT_ALL states. (See "1.5 Power Management Functions" for detailed information on individual states.)

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x060	RAM_Rd_10	R	0x00	RAM_Rd_10[7:0]							
0x061	RAM_Rd_11	R	0x00	RAM_Rd_11[7:0]							
0x062	RAM_Rd_12	R	0x00	RAM_Rd_12[7:0]							
0x063	RAM_Rd_13	R	0x00	RAM_Rd_13[7:0]							
0x064	RAM_Rd_14	R	0x00	RAM_Rd_14[7:0]							
0x065	RAM_Rd_15	R	0x00	RAM_Rd_15[7:0]							
0x066	RAM_Rd_16	R	0x00	RAM_Rd_16[7:0]							
0x067	RAM_Rd_17	R	0x00	RAM_Rd_17[7:0]							
0x068	RAM_Rd_18	R	0x00	RAM_Rd_18[7:0]							
0x069	RAM_Rd_19	R	0x00	RAM_Rd_19[7:0]							
0x06A	RAM_Rd_1A	R	0x00	RAM_Rd_1A[7:0]							
0x06B	RAM_Rd_1B	R	0x00	RAM_Rd_1B[7:0]							
0x06C	RAM_Rd_1C	R	0x00	RAM_Rd_1C[7:0]							
0x06D	RAM_Rd_1D	R	0x00	RAM_Rd_1D[7:0]							
0x06E	RAM_Rd_1E	R	0x00	RAM_Rd_1E[7:0]							
0x06F	RAM_Rd_1F	R	0x00	RAM_Rd_1F[7:0]							
Byte Addr.	**Register Name**	**R/W**	**Reset**	**bit15**	**bit14**	**bit13**	**bit12**	**bit11**	**bit10**	**bit9**	**bit8**
				bit7	**bit6**	**bit5**	**bit4**	**bit3**	**bit2**	**bit1**	**bit0**
0x070			0xXX								
0x071	DMA0_Config	R/W	0x00	FreeRun	DMA_Mode			ActiveDMA		ReqAssertCount[1:0]	
0x072	DMA0_Control	R/W	0x00	DMA_Running			CounterClr	Dir		DMA_Stop	DMA_Go
0x073			0xXX								
0x074	DMA0_Remain_H	R	0x00				DMA_Remain[12:8]				
0x075	DMA0_Remain_L	R	0x00	DMA_Remain[7:0]							
0x076			0xXX								
0x077			0xXX								
0x078	DMA0_Count_HH	R/W	0x00	DMA_Count[31:24]							
0x079	DMA0_Count_HL	R/W	0x00	DMA_Count[23:16]							
0x07A	DMA0_Count_LH	R/W	0x00	DMA_Count[15:8]							
0x07B	DMA0_Count_LL	R/W	0x00	DMA_Count[7:0]							
0x07C	DMA0_RdData_0	R	0xXX	DMA_RdData_0[7:0]							
0x07D	DMA0_RdData_1	R	0xXX	DMA_RdData_1[7:0]							
0x07E	DMA0_WrData_0	W	0xXX	DMA_WrData_0[7:0]							
0x07F	DMA0_WrData_1	W	0xXX	DMA_WrData_1[7:0]							

Registers that can be read from or written to even in SLEEP or SNOOZE states are indicated in ***bold italic***.

All other registers can be read from or written to in ACTIVE60, ACT_DEVICE, ACT_HOST, and ACT_ALL states. (See "1.5 Power Management Functions" for detailed information on individual states.)

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x080			0xXX								
0x081	DMA1_Config	R/W	0x00	FreeRun	DMA_Mode			ActiveDMA		ReqAssertCount[1:0]	
0x082	DMA1_Control	R/W	0x00	DMA_Running			CounterClr	Dir		DMA_Stop	DMA_Go
0x083			0xXX								
0x084	DMA1_Remain_H	R	0x00				DMA_Remain[12:8]				
0x085	DMA1_Remain_L	R	0x00	DMA_Remain[7:0]							
0x086			0xXX								
0x087			0xXX								
0x088	DMA1_Count_HH	R/W	0x00	DMA_Count[31:24]							
0x089	DMA1_Count_HL	R/W	0x00	DMA_Count[23:16]							
0x08A	DMA1_Count_LH	R/W	0x00	DMA_Count[15:8]							
0x08B	DMA1_Count_LL	R/W	0x00	DMA_Count[7:0]							
0x08C	DMA1_RdData_0	R	0xXX	DMA_RdData_0[7:0]							
0x08D	DMA1_RdData_1	R	0xXX	DMA_RdData_1[7:0]							
0x08E	DMA1_WrData_0	W	0xXX	DMA_WrData_0[7:0]							
0x08F	DMA1_WrData_1	W	0xXX	DMA_WrData_1[7:0]							
Byte Addr.	**Register Name**	**R/W**	**Reset**	**bit15**	**bit14**	**bit13**	**bit12**	**bit11**	**bit10**	**bit9**	**bit8**
				bit7	**bit6**	**bit5**	**bit4**	**bit3**	**bit2**	**bit1**	**bit0**
0x090	IDE_Status	R/W	0x00	DMARQ	DMACK	INTRQ	IORDY			PDIAG	DASP
0x091	IDE_Control	R/W	0x00		IDE_Clr			Dir			IDE_Go
0x092	IDE_Config_0	R/W	0x00	IDE_BusReset	IDE_LongBusReset					Ultra	DMA
0x093	IDE_Config_1	R/W	0x04	ActiveIDE	DelayStrobe		InterLock		Swap		
0x094	IDE_Rmod	R/W	0x00	RegisterAssertPulseWidth[3:0]				RegisterNegatePulseWidth[3:0]			
0x095	IDE_Tmod	R/W	0x00	TransferAssertPulseWidth[3:0]				TransferNegatePulseWidth[3:0]			
0x096	IDE_Umod	R/W	0x00					UltraDMA_Cycle[3:0]			
0x097			0xXX								
0x098			0xXX								
0x099			0xXX								
0x09A	IDE_CRC_H	R	0x00	IDE_CRC[15:8]							
0x09B	IDE_CRC_L	R	0x00	IDE_CRC[7:0]							
0x09C			0xXX								
0x09D	IDE_Count_H	R/W	0x00	IDE_Count[23:16]							
0x09E	IDE_Count_M	R/W	0x00	IDE_Count[15:8]							
0x09F	IDE_Count_L	R/W	0x00	IDE_Count[7:1]							

Registers that can be read from or written to even in SLEEP or SNOOZE states are indicated in ***bold italic***.

All other registers can be read from or written to in ACTIVE60, ACT_DEVICE, ACT_HOST, and ACT_ALL states. (See "1.5 Power Management Functions" for detailed information on individual states.)

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x0A0	IDE_RegAdrs	R/W	0x00	IDE_WrReg	IDE_RdReg			IDE_RegAddress[3:0]			
0x0A1			0xXX								
0x0A2	IDE_RdRegValue_0	R	0x00	IDE_RdRegValue_0[7:0]							
0x0A3	IDE_RdRegValue_1	R	0x00	IDE_RdRegValue_1[7:0]							
0x0A4	IDE_WrRegValue_0	R/W	0x00	IDE_WrRegValue_0[7:0]							
0x0A5	IDE_WrRegValue_1	R/W	0x00	IDE_WrRegValue_1[7:0]							
0x0A6	IDE_SeqWrRegControl	R/W	0x00	IDE_SeqWrReg	IDE_SeqWrRegClr						
0x0A7	IDE_SeqWrRegCnt	R	0x00				IDE_SeqWrRegCnt[4:0]				
0x0A8	IDE_SeqWrRegAdrs	W	0xXX					IDE_SeqRegAddress[3:0]			
0x0A9	IDE_SeqWrRegValue	W	0xXX	IDE_SeqWrRegValue[7:0]							
0x0AA			0xXX								
0x0AB			0xXX								
0x0AC	IDE_RegConfig	R/W	0x00	EnAutoStsRd							
0x0AD			0xXX								
0x0AE			0xXX								
0x0AF			0xXX								
Byte Addr.	**Register Name**	**R/W**	**Reset**	**bit15**	**bit14**	**bit13**	**bit12**	**bit11**	**bit10**	**bit9**	**bit8**
				bit7	**bit6**	**bit5**	**bit4**	**bit3**	**bit2**	**bit1**	**bit0**
0x0B0			0xXX								
0x0B1	***HostDeviceSel***	R/W	0x00								***HOSTxDEVICE***
0x0B2			0xXX								
0x0B3	***ModeProtect***	R/W	0x56	***ModeProtect[7:0](56 以外を書くとプロテクト、0x56 で解除)***							
0x0B4			0xXX								
0x0B5	***ClkSelect***	R/W	0x41	***xActIDE_Term***	***xActIDE_DD_Term***					***PORT1x2***	***ClkSelect***
0x0B6			0xXX								
0x0B7	***ChipConfig***	R/W	0x00	***IntLevel***	***IntMode***	***DREQ_Level***	***DACK_Level***	***CS_Mode***	***CPU_Endian***	***BusMode***	***Initialized***
0x0B8			0xXX								
0x0B9			0xXX								
0x0BA			0xXX								
0x0BB			0xXX								
0x0BC			0xXX								
0x0BD	TimerConfig	R/W	0x00	TimerStop	TimerRunning		FreeRun				
0x0BE	TimerSet_H	R/W	0xFF	TimerSet[15:8]							
0x0BF	TimerSet_L	R/W	0xFF	TimerSet[7:0]							

Registers that can be read from or written to even in SLEEP or SNOOZE states are indicated in ***bold italic***.

All other registers can be read from or written to in ACTIVE60, ACT_DEVICE, ACT_HOST, and ACT_ALL states. (See "1.5 Power Management Functions" for detailed information on individual states.)

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x0C0	AREAnIntStat	R	0x00			AREA5IntStat	AREA4IntStat	AREA3IntStat	AREA2IntStat	AREA1IntStat	AREA0IntStat
0x0C1	AREA0IntStat	R/(W)	0x01						FIFO_NotEmpty	FIFO_Full	FIFO_Empty
0x0C2	AREA1IntStat	R/(W)	0x01						FIFO_NotEmpty	FIFO_Full	FIFO_Empty
0x0C3	AREA2IntStat	R/(W)	0x01						FIFO_NotEmpty	FIFO_Full	FIFO_Empty
0x0C4	AREA3IntStat	R/(W)	0x01						FIFO_NotEmpty	FIFO_Full	FIFO_Empty
0x0C5	AREA4IntStat	R/(W)	0x01						FIFO_NotEmpty	FIFO_Full	FIFO_Empty
0x0C6	AREA5IntStat	R/(W)	0x01						FIFO_NotEmpty	FIFO_Full	FIFO_Empty
0x0C7											
0x0C8	AREAnIntEnb	R/W	0x00			EnAREA5IntStat	EnAREA4IntStat	EnAREA3IntStat	EnAREA2IntStat	EnAREA1IntStat	EnAREA0IntStat
0x0C9	AREA0IntEnb	R/W	0x00						EnFIFO_NotEmpty	EnFIFO_Full	EnFIFO_Empty
0x0CA	AREA1IntEnb	R/W	0x00						EnFIFO_NotEmpty	EnFIFO_Full	EnFIFO_Empty
0x0CB	AREA2IntEnb	R/W	0x00						EnFIFO_NotEmpty	EnFIFO_Full	EnFIFO_Empty
0x0CC	AREA3IntEnb	R/W	0x00						EnFIFO_NotEmpty	EnFIFO_Full	EnFIFO_Empty
0x0CD	AREA4IntEnb	R/W	0x00						EnFIFO_NotEmpty	EnFIFO_Full	EnFIFO_Empty
0x0CE	AREA5IntEnb	R/W	0x00						EnFIFO_NotEmpty	EnFIFO_Full	EnFIFO_Empty
0x0CF											
Byte Addr.	**Register Name**	**R/W**	**Reset**	**bit15**	**bit14**	**bit13**	**bit12**	**bit11**	**bit10**	**bit9**	**bit8**
				bit7	**bit6**	**bit5**	**bit4**	**bit3**	**bit2**	**bit1**	**bit0**
0x0D0	AREA0Join_0	R/W	0x00	JoinIDE	JoinFIFO_Stat			JoinDMA1	JoinDMA0	JoinCPU_Rd	JoinCPU_Wr
0x0D1	AREA0Join_1	R/W	0x00			JoinEPeCHe	JoinEPdCHd	JoinEPcCHc	JoinEPbCHb	JoinEPaCHa	JoinEP0CH0
0x0D2	AREA1Join_0	R/W	0x00	JoinIDE	JoinFIFO_Stat			JoinDMA1	JoinDMA0	JoinCPU_Rd	JoinCPU_Wr
0x0D3	AREA1Join_1	R/W	0x00			JoinEPeCHe	JoinEPdCHd	JoinEPcCHc	JoinEPbCHb	JoinEPaCHa	JoinEP0CH0
0x0D4	AREA2Join_0	R/W	0x00	JoinIDE	JoinFIFO_Stat			JoinDMA1	JoinDMA0	JoinCPU_Rd	JoinCPU_Wr
0x0D5	AREA2Join_1	R/W	0x00			JoinEPeCHe	JoinEPdCHd	JoinEPcCHc	JoinEPbCHb	JoinEPaCHa	JoinEP0CH0
0x0D6	AREA3Join_0	R/W	0x00	JoinIDE	JoinFIFO_Stat			JoinDMA1	JoinDMA0	JoinCPU_Rd	JoinCPU_Wr
0x0D7	AREA3Join_1	R/W	0x00			JoinEPeCHe	JoinEPdCHd	JoinEPcCHc	JoinEPbCHb	JoinEPaCHa	JoinEP0CH0
0x0D8	AREA4Join_0	R/W	0x00	JoinIDE	JoinFIFO_Stat			JoinDMA1	JoinDMA0	JoinCPU_Rd	JoinCPU_Wr
0x0D9	AREA4Join_1	R/W	0x00			JoinEPeCHe	JoinEPdCHd	JoinEPcCHc	JoinEPbCHb	JoinEPaCHa	JoinEP0CH0
0x0DA	AREA5Join_0	R/W	0x00	JoinIDE	JoinFIFO_Stat			JoinDMA1	JoinDMA0	JoinCPU_Rd	JoinCPU_Wr
0x0DB	AREA5Join_1	R/W	0x00			JoinEPeCHe	JoinEPdCHd	JoinEPcCHc	JoinEPbCHb	JoinEPaCHa	JoinEP0CH0
0x0DC			0xXX								
0x0DD			0xXX								
0x0DE	ClrAREAnJoin_0	W	0xXX	ClrJoinIDE	ClrJoinFIFO_Stat			ClrJoinDMA1	ClrJoinDMA0	ClrJoinCPU_Rd	ClrJoinCPU_Wr
0x0DF	ClrAREAnJoin_1	W	0xXX			ClrJoinEPeCHe	ClrJoinEPdCHd	ClrJoinEPcCHc	ClrJoinEPbCHb	ClrJoinEPaCHa	ClrJoinEP0CH0

Registers that can be read from or written to even in SLEEP or SNOOZE states are indicated in ***bold italic***.

All other registers can be read from or written to in ACTIVE60, ACT_DEVICE, ACT_HOST, and ACT_ALL states. (See "1.5 Power Management Functions" for detailed information on individual states.)

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x180	AREA0StartAdrs_H	R/W	0x00				StartAdrs[12:8]				
0x181	AREA0StartAdrs_L	R/W	0x00	StartAdrs[7:2]							
0x182	AREA0EndAdrs_H	R/W	0x00				EndAdrs[12:8]				
0x183	AREA0EndAdrs_L	R/W	0x00	EndAdrs[7:2]							
0x184	AREA1StartAdrs_H	R/W	0x00				StartAdrs[12:8]				
0x185	AREA1StartAdrs_L	R/W	0x00	StartAdrs[7:2]							
0x186	AREA1EndAdrs_H	R/W	0x00				EndAdrs[12:8]				
0x187	AREA1EndAdrs_L	R/W	0x00	EndAdrs[7:2]							
0x188	AREA2StartAdrs_H	R/W	0x00				StartAdrs[12:8]				
0x189	AREA2StartAdrs_L	R/W	0x00	StartAdrs[7:2]							
0x18A	AREA2EndAdrs_H	R/W	0x00				EndAdrs[12:8]				
0x18B	AREA2EndAdrs_L	R/W	0x00	EndAdrs[7:2]							
0x18C	AREA3StartAdrs_H	R/W	0x00				StartAdrs[12:8]				
0x18D	AREA3StartAdrs_L	R/W	0x00	StartAdrs[7:2]							
0x18E	AREA3EndAdrs_H	R/W	0x00				EndAdrs[12:8]				
0x18F	AREA3EndAdrs_L	R/W	0x00	EndAdrs[7:2]							
Byte Addr.	**Register Name**	**R/W**	**Reset**	**bit15**	**bit14**	**bit13**	**bit12**	**bit11**	**bit10**	**bit9**	**bit8**
				bit7	**bit6**	**bit5**	**bit4**	**bit3**	**bit2**	**bit1**	**bit0**
0x190	AREA4StartAdrs_H	R/W	0x00				StartAdrs[12:8]				
0x191	AREA4StartAdrs_L	R/W	0x00	StartAdrs[7:2]							
0x192	AREA4EndAdrs_H	R/W	0x00				EndAdrs[12:8]				
0x193	AREA4EndAdrs_L	R/W	0x00	EndAdrs[7:2]							
0x194	AREA5StartAdrs_H	R/W	0x00				StartAdrs[12:8]				
0x195	AREA5StartAdrs_L	R/W	0x00	StartAdrs[7:2]							
0x196	AREA5EndAdrs_H	R/W	0x00				EndAdrs[12:8]				
0x197	AREA5EndAdrs_L	R/W	0x00	EndAdrs[7:2]							
			0xXX								
			0xXX								
			0xXX								
			0xXX								
			0xXX								
			0xXX								
			0xXX								
0x19F	AREAnFIFO_Clr	W	0xXX			ClrAREA5	ClrAREA4	ClrAREA3	ClrAREA2	ClrAREA1	ClrAREA0

2.3 Device Register Map

Registers that can be read from or written to even in SLEEP or SNOOZE states are indicated in ***bold italic***.

All other registers can be read from in ACTIVE60, ACT_DEVICE, ACT_HOST, and ACT_ALL states, and written to in ACT_DEVICE and ACT_ALL states. (See "1.5 Power Management Functions" for detailed information on individual states.)

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x0E0	***D_SIE_IntStat***	R/(W)	0x00		***NonJ***	RcvSOF	DetectRESET	DetectSUSPEND	ChirpCmp	RestoreCmp	SetAddressCmp
0x0E1			0xXX								
0x0E2	D_FIFO_IntSat	R/(W)	0x00	DescriptorCmp	FIFO_IDE_Cmp	FIFO1_Cmp	FIFO0_Cmp		FIFO_NotEmpty	FIFO_Full	FIFO_Empty
0x0E3	D_BulkIntStat	R/(W)	0x00	CBW_Cmp	CBW_LengthErr	CBW_Err		CSW_Cmp	CSW_Err		
0x0E4	D_EPrIntStat	R	0x00				D_EPeIntStat	D_EPdIntStat	D_EPcIntStat	D_EPbIntStat	D_EPaIntStat
0x0E5	D_EP0IntStat	R/(W)	0x00		OUT_ShortACK	IN_TranACK	OUT_TranACK	IN_TranNAK	OUT_TranNAK	IN_TranErr	OUT_TranErr
0x0E6	D_EPaIntStat	R/(W)	0x00		OUT_ShortACK	IN_TranACK	OUT_TranACK	IN_TranNAK	OUT_TranNAK	IN_TranErr	OUT_TranErr
0x0E7	D_EPbIntStat	R/(W)	0x00		OUT_ShortACK	IN_TranACK	OUT_TranACK	IN_TranNAK	OUT_TranNAK	IN_TranErr	OUT_TranErr
0x0E8	D_EPcIntStat	R/(W)	0x00		OUT_ShortACK	IN_TranACK	OUT_TranACK	IN_TranNAK	OUT_TranNAK	IN_TranErr	OUT_TranErr
0x0E9	D_EPdIntStat	R/(W)	0x00		OUT_ShortACK	IN_TranACK	OUT_TranACK	IN_TranNAK	OUT_TranNAK	IN_TranErr	OUT_TranErr
0x0EA	D_EPeIntStat	R/(W)	0x00		OUT_ShortACK	IN_TranACK	OUT_TranACK	IN_TranNAK	OUT_TranNAK	IN_TranErr	OUT_TranErr
0x0EB			0xXX								
0x0EC			0xXX								
0x0ED			0xXX								
0x0EE			0xXX								
0x0EF			0xXX								
Byte Addr.	**Register Name**	**R/W**	**Reset**	**bit15**	**bit14**	**bit13**	**bit12**	**bit11**	**bit10**	**bit9**	**bit8**
				bit7	**bit6**	**bit5**	**bit4**	**bit3**	**bit2**	**bit1**	**bit0**
0x0F0	***D_SIE_IntEnb***	R/W	0x00		***EnNonJ***	EnRcvSOF	EnDetectRESET	EnDetectSUSPEND	EnChirpCmp	EnRestoreCmp	EnSetAddressCmp
0x0F1			0xXX								
0x0F2	D_FIFO_IntEnb	R/W	0x00	EnDescriptorCmp	EnFIFO_IDE_Cmp	EnFIFO1_Cmp	EnFIFO0_Cmp		EnFIFO_NotEmpty	EnFIFO_Full	EnFIFO_Empty
0x0F3	D_BulkIntEnb	R/W	0x00	EnCBW_Cmp	EnCBW_LengthErr	EnCBW_Err		EnCSW_Cmp	EnCSW_Err		
0x0F4	D_EPrIntEnb	R/W	0x00				EnD_EPeIntStat	EnD_EPdIntStat	EnD_EPcIntStat	EnD_EPbIntStat	EnD_EPaIntStat
0x0F5	D_EP0IntEnb	R/W	0x00		EnOUT_ShortACK	EnIN_TranACK	EnOUT_TranACK	EnIN_TranNAK	EnOUT_TranNAK	EnIN_TranErr	EnOUT_TranErr
0x0F6	D_EPaIntEnb	R/W	0x00		EnOUT_ShortACK	EnIN_TranACK	EnOUT_TranACK	EnIN_TranNAK	EnOUT_TranNAK	EnIN_TranErr	EnOUT_TranErr
0x0F7	D_EPbIntEnb	R/W	0x00		EnOUT_ShortACK	EnIN_TranACK	EnOUT_TranACK	EnIN_TranNAK	EnOUT_TranNAK	EnIN_TranErr	EnOUT_TranErr
0x0F8	D_EPcIntEnb	R/W	0x00		EnOUT_ShortACK	EnIN_TranACK	EnOUT_TranACK	EnIN_TranNAK	EnOUT_TranNAK	EnIN_TranErr	EnOUT_TranErr
0x0F9	D_EPdIntEnb	R/W	0x00		EnOUT_ShortACK	EnIN_TranACK	EnOUT_TranACK	EnIN_TranNAK	EnOUT_TranNAK	EnIN_TranErr	EnOUT_TranErr
0x0FA	D_EPeIntEnb	R/W	0x00		EnOUT_ShortACK	EnIN_TranACK	EnOUT_TranACK	EnIN_TranNAK	EnOUT_TranNAK	EnIN_TranErr	EnOUT_TranErr
0x0FB			0xXX								
0x0FC			0xXX								
0x0FD			0xXX								
0x0FE			0xXX								
0x0FF			0xXX								

Registers that can be read from or written to even in SLEEP or SNOOZE states are indicated in ***bold italic***.

All other registers can be read from in ACTIVE60, ACT_DEVICE, ACT_HOST, and ACT_ALL states, and written to in ACT_DEVICE and ACT_ALL states. (See "1.5 Power Management Functions" for detailed information on individual states.)

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x100	***D_Reset***	R/W	0x01								***ResetDTM***
0x101			0xXX								
0x102	D_NegoControl	R/W	0x00	DisBusDetect	EnAutoNego	InSUSPEND	DisableHS	SendWakeup	RestoreUSB	GoChirp	ActiveUSB
0x103			0xXX								
0x104			0xXX								
0x105	D_XcvrControl	R/W	0x41	TermSelect	XcvrSelect					OpMode[1:0]	
0x106	D_USB_Test	R/W	0x00	EnHS_Test				Test_SE0_NAK	Test_J	Test_K	Test_Packet
0x107			0xXX								
0x108	D_EPnControl	W	0xXX	AllForceNAK	EPrForceSTALL						
0x109			0xXX								
0x10A	D_BulkOnlyControl	R/W	0x00	AutoForceNAK_CBW					GoCBW_Mode	GoCSW_Mode	
0x10B	D_BulkOnlyConfig	R/W	0x00				EPeBulkOnly	EPdBulkOnly	EPcBulkOnly	EPbBulkOnly	EPaBulkOnly
0x10C			0xXX								
0x10D			0xXX								
0x10E			0xXX								
0x10F			0xXX								
Byte Addr.	**Register Name**	**R/W**	**Reset**	**bit15**	**bit14**	**bit13**	**bit12**	**bit11**	**bit10**	**bit9**	**bit8**
				bit7	**bit6**	**bit5**	**bit4**	**bit3**	**bit2**	**bit1**	**bit0**
0x110	D_EP0SETUP_0	R	0x00	SETUP0[7:0]							
0x111	D_EP0SETUP_1	R	0x00	SETUP1[7:0]							
0x112	D_EP0SETUP_2	R	0x00	SETUP2[7:0]							
0x113	D_EP0SETUP_3	R	0x00	SETUP3[7:0]							
0x114	D_EP0SETUP_4	R	0x00	SETUP4[7:0]							
0x115	D_EP0SETUP_5	R	0x00	SETUP5[7:0]							
0x116	D_EP0SETUP_6	R	0x00	SETUP6[7:0]							
0x117	D_EP0SETUP_7	R	0x00	SETUP7[7:0]							
0x118	D_USB_Address	R/W	0x00	SetAddress	USB_Address[6:0]						
0x119			0xXX								
0x11A	D_SETUP_Control	R/W	0x00								ProtectEP0
0x11B			0xXX								
0x11C			0xXX								
0x11D			0xXX								
0x11E	D_FrameNumber_H	R	0x80	Fn_Invalid					FrameNumber[10:8]		
0x11F	D_FrameNumber_L	R	0x00	FrameNumber[7:0]							

Registers that can be read from or written to even in SLEEP or SNOOZE states are indicated in ***bold italic***.

All other registers can be read from in ACTIVE60, ACT_DEVICE, ACT_HOST, and ACT_ALL states, and written to in ACT_DEVICE and ACT_ALL states. (See "1.5 Power Management Functions" for detailed information on individual states.)

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x120	D_EP0MaxSize	R/W	0x40	EP0MaxSize[6:3]							
0x121	D_EP0Control	R/W	0x00	INxOUT							ReplyDescriptor
0x122	D_EP0ControlIN	R/W	0x00		EnShortPkt		ToggleStat	ToggleSet	ToggleClr	ForceNAK	ForceSTALL
0x123	D_EP0ControlOUT	R/W	0x00	AutoForceNAK			ToggleStat	ToggleSet	ToggleClr	ForceNAK	ForceSTALL
0x124			0xXX								
0x125			0xXX								
0x126			0xXX								
0x127			0xXX								
0x128			0xXX								
0x129			0xXX								
0x12A			0xXX								
0x12B			0xXX								
0x12C			0xXX								
0x12D			0xXX								
0x12E			0xXX								
0x12F			0xXX								
Byte Addr.	**Register Name**	**R/W**	**Reset**	**bit15**	**bit14**	**bit13**	**bit12**	**bit11**	**bit10**	**bit9**	**bit8**
				bit7	**bit6**	**bit5**	**bit4**	**bit3**	**bit2**	**bit1**	**bit0**
0x130	D_EPaMaxSize_H	R/W	0x00						EPaMaxSize[10:8]		
0x131	D_EPaMaxSize_L	R/W	0x00	EPaMaxSize[7:0]							
0x132	D_EPaConfig_0	R/W	0x00	INxOUT	IntEP_Mode		ISO	EndpointNumber[3:0]			
0x133			0xXX								
0x134	D_EPaControl	R/W	0x00	AutoForceNAK	EnShortPkt	DisAF_NAK_Short	ToggleStat	ToggleSet	ToggleClr	ForceNAK	ForceSTALL
0x135			0xXX								
0x136			0xXX								
0x137			0xXX								
0x138			0xXX								
0x139			0xXX								
0x13A			0xXX								
0x13B			0xXX								
0x13C			0xXX								
0x13D			0xXX								
0x13E			0xXX								
0x13F			0xXX								

Registers that can be read from or written to even in SLEEP or SNOOZE states are indicated in ***bold italic***.

All other registers can be read from in ACTIVE60, ACT_DEVICE, ACT_HOST, and ACT_ALL states, and written to in ACT_DEVICE and ACT_ALL states. (See "1.5 Power Management Functions" for detailed information on individual states.)

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x140	D_EPbMaxSize_H	R/W	0x00						EPbMaxSize[10:8]		
0x141	D_EPbMaxSize_L	R/W	0x00	EPbMaxSize[7:0]							
0x142	D_EPbConfig_0	R/W	0x00	INxOUT	IntEP_Mode		ISO	EndpointNumber[3:0]			
0x143			0xXX								
0x144	D_EPbControl	R/W	0x00	AutoForceNAK	EnShortPkt	DisAF_NAK_Short	ToggleStat	ToggleSet	ToggleClr	ForceNAK	ForceSTALL
0x145			0xXX								
0x146			0xXX								
0x147			0xXX								
0x148			0xXX								
0x149			0xXX								
0x14A			0xXX								
0x14B			0xXX								
0x14C			0xXX								
0x14D			0xXX								
0x14E			0xXX								
0x14F			0xXX								
Byte Addr.	**Register Name**	**R/W**	**Reset**	**bit15**	**bit14**	**bit13**	**bit12**	**bit11**	**bit10**	**bit9**	**bit8**
				bit7	**bit6**	**bit5**	**bit4**	**bit3**	**bit2**	**bit1**	**bit0**
0x150	D_EPcMaxSize_H	R/W	0x00						EPcMaxSize[10:8]		
0x151	D_EPcMaxSize_L	R/W	0x00	EPcMaxSize[7:0]							
0x152	D_EPcConfig_0	R/W	0x00	INxOUT	IntEP_Mode		ISO	EndpointNumber[3:0]			
0x153			0xXX								
0x154	D_EPcControl	R/W	0x00	AutoForceNAK	EnShortPkt	DisAF_NAK_Short	ToggleStat	ToggleSet	ToggleClr	ForceNAK	ForceSTALL
0x155			0xXX								
0x156			0xXX								
0x157			0xXX								
0x158			0xXX								
0x159			0xXX								
0x15A			0xXX								
0x15B			0xXX								
0x15C			0xXX								
0x15D			0xXX								
0x15E			0xXX								
0x15F			0xXX								

Registers that can be read from or written to even in SLEEP or SNOOZE states are indicated in ***bold italic***.

All other registers can be read from in ACTIVE60, ACT_DEVICE, ACT_HOST, and ACT_ALL states, and written to in ACT_DEVICE and ACT_ALL states. (See "1.5 Power Management Functions" for detailed information on individual states.)

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x160	D_DescAdrs_H	R/W	0x00				DescAdrs[12:8]				
0x161	D_DescAdrs_L	R/W	0x00	DescAdrs[7:0]							
0x162	D_DescSize_H	R/W	0x00							DescSize[9:8]	
0x163	D_DescSize_L	R/W	0x00	DescSize[7:0]							
0x164			0xXX								
0x165			0xXX								
0x166			0xXX								
0x167			0xXX								
0x168			0xXX								
0x169			0xXX								
0x16A			0xXX								
0x16B			0xXX								
0x16C			0xXX								
0x16D			0xXX								
0x16E			0xXX								
0x16F			0xXX								

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x170	D_DMA0_FIFO_Control	R/W	0x00	FIFO_Running	AutoEnShort						
0x171			0xXX								
0x172	D_DMA1_FIFO_Control	R/W	0x00	FIFO_Running	AutoEnShort						
0x173			0xXX								
0x174			0xXX								
0x175			0xXX								
0x176			0xXX								
0x177			0xXX								
0x178			0xXX								
0x179			0xXX								
0x17A			0xXX								
0x17B			0xXX								
0x17C			0xXX								
0x17D			0xXX								
0x17E			0xXX								
0x17F			0xXX								

Registers that can be read from or written to even in SLEEP or SNOOZE states are indicated in ***bold italic***.

All other registers can be read from in ACTIVE60, ACT_DEVICE, ACT_HOST, and ACT_ALL states, and written to in ACT_DEVICE and ACT_ALL states. (See "1.5 Power Management Functions" for detailed information on individual states.)

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x1A0	D_EPdMaxSize_H	R/W	0x00						EPdMaxSize[10:8]		
0x1A1	D_EPdMaxSize_L	R/W	0x00	EPdMaxSize[7:0]							
0x1A2	D_EPdConfig_0	R/W	0x00	INxOUT	IntEP_Mode		ISO	EndpointNumber[3:0]			
0x1A3			0xXX								
0x1A4	D_EPdControl	R/W	0x00	AutoForceNAK	EnShortPkt	DisAF_NAK_Short	ToggleStat	ToggleSet	ToggleClr	ForceNAK	ForceSTALL
0x1A5			0xXX								
0x1A6			0xXX								
0x1A7			0xXX								
0x1A8			0xXX								
0x1A9			0xXX								
0x1AA			0xXX								
0x1AB			0xXX								
0x1AC			0xXX								
0x1AD			0xXX								
0x1AE			0xXX								
0x1AF			0xXX								

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x1B0	D_EPeMaxSize_H	R/W	0x00						EPeMaxSize[10:8]		
0x1B1	D_EPeMaxSize_L	R/W	0x00	EPeMaxSize[7:0]							
0x1B2	D_EPeConfig_0	R/W	0x00	INxOUT	IntEP_Mode		ISO	EndpointNumber[3:0]			
0x1B3			0xXX								
0x1B4	D_EPeControl	R/W	0x00	AutoForceNAK	EnShortPkt	DisAF_NAK_Short	ToggleStat	ToggleSet	ToggleClr	ForceNAK	ForceSTALL
0x1B5			0xXX								
0x1B6			0xXX								
0x1B7			0xXX								
0x1B8			0xXX								
0x1B9			0xXX								
0x1BA			0xXX								
0x1BB			0xXX								
0x1BC			0xXX								
0x1BD			0xXX								
0x1BE	***DTM_Config***	R/W	0x00			***DTM_SlopeValue[1:0]***				***DTM_TermValue[1:0]***	
0x1BF			0xXX								

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x1E0	(Reserved)		0xXX								
0x1E1	D_ModeControl	W	0xXX	(Reserved)	(Reserved)	(Reserved)	SetAddressMode	(Reserved)	(Reserved)	(Reserved)	(Reserved)

2.4 Host Register Map

Registers that can be read from or written to even in SLEEP or SNOOZE states are indicated in ***bold italic***.

All other registers can be read from in ACTIVE60, ACT_DEVICE, ACT_HOST, and ACT_ALL states, and written to in ACT_DEVICE and ACT_ALL states. (See "1.5 Power Management Functions" for detailed information on individual states.)

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x0E0	H_SIE_IntStat_0	R(W)	0x00				DetectCon	DetectDiscon	DetectRmtWkup	DetectDevChirpOK	DetectDevChirpNG
0x0E1	H_SIE_IntStat_1	R(W)	0x00					DisabledCmp	ResumeCmp	SuspendCmp	ResetCmp
0x0E2	H_FIFO_IntStat	R(W)	0x00		FIFO_IDE_Cmp	FIFO1_Cmp	FIFO0_Cmp		FIFO_NotEmpty	FIFO_Full	FIFO_Empty
0x0E3	H_FrameIntStat	R(W)	0x00	TriggerFrame					PortErr	FrameNumOver	SOF
0x0E4	H_CHrIntStat	R	0x00				H_CHeIntStat	H_CHdIntStat	H_CHcIntStat	H_CHbIntStat	H_CHaIntStat
0x0E5	H_CH0IntStat	R(W)	0x00	TotalSizeCmp	TranACK	TranErr	ChangeCondition			CTL_SupportCmp	CTL_SupportStop
0x0E6	H_CHaIntStat	R(W)	0x00	TotalSizeCmp	TranACK	TranErr	ChangeCondition			BO_SupportCmp	BO_SupportStop
0x0E7	H_CHbIntStat	R(W)	0x00	TotalSizeCmp	TranACK	TranErr	ChangeCondition				
0x0E8	H_CHcIntStat	R(W)	0x00	TotalSizeCmp	TranACK	TranErr	ChangeCondition				
0x0E9	H_CHdIntStat	R(W)	0x00	TotalSizeCmp	TranACK	TranErr	ChangeCondition				
0x0EA	H_CHeIntStat	R(W)	0x00	TotalSizeCmp	TranACK	TranErr	ChangeCondition				
0x0EB			0xXX								
0x0EC			0xXX								
0x0ED			0xXX								
0x0EE			0xXX								
0x0EF			0xXX								
Byte Addr.	**Register Name**	**R/W**	**Reset**	**bit15**	**bit14**	**bit13**	**bit12**	**bit11**	**bit10**	**bit9**	**bit8**
				bit7	**bit6**	**bit5**	**bit4**	**bit3**	**bit2**	**bit1**	**bit0**
0x0F0	H_SIE_IntEnb_0	R/W	0x00				EnDetectCon	EnDetectDiscon	EnDetectRmtWkup	EnDetectDevChirpOK	EnDetectDevChirpNG
0x0F1	H_SIE_IntEnb_1	R/W	0x00					EnDisabledCmp	EnResumeCmp	EnSuspendCmp	EnResetCmp
0x0F2	H_FIFO_IntEnb	R/W	0x00		EnFIFO_IDE_Cmp	EnFIFO1_Cmp	EnFIFO0_Cmp		EnFIFO_NotEmpty	EnFIFO_Full	EnFIFO_Empty
0x0F3	H_FrameIntEnb	R/W	0x00	EnTriggerFrame					EnPortErr	EnFrameNumOver	EnSOF
0x0F4	H_CHrIntEnb	R/W	0x00				EnH_CHeIntStat	EnH_CHdIntStat	EnH_CHcIntStat	EnH_CHbIntStat	EnH_CHaIntStat
0x0F5	H_CH0IntEnb	R/W	0x00	EnTotalSizeCmp	EnTranACK	EnTranErr	EnChangeCondition			EnCTL_SupportCmp	EnCTL_SupportStop
0x0F6	H_CHaIntEnb	R/W	0x00	EnTotalSizeCmp	EnTranACK	EnTranErr	EnChangeCondition			EnBO_SupportCmp	EnBO_SupportStop
0x0F7	H_CHbIntEnb	R/W	0x00	EnTotalSizeCmp	EnTranACK	EnTranErr	EnChangeCondition				
0x0F8	H_CHcIntEnb	R/W	0x00	EnTotalSizeCmp	EnTranACK	EnTranErr	EnChangeCondition				
0x0F9	H_CHdIntEnb	R/W	0x00	EnTotalSizeCmp	EnTranACK	EnTranErr	EnChangeCondition				
0x0FA	H_CHeIntEnb	R/W	0x00	EnTotalSizeCmp	EnTranACK	EnTranErr	EnChangeCondition				
0x0FB			0xXX								
0x0FC			0xXX								
0x0FD			0xXX								
0x0FE			0xXX								
0x0FF			0xXX								

Registers that can be read from or written to even in SLEEP or SNOOZE states are indicated in ***bold italic***.

All other registers can be read from in ACTIVE60, ACT_DEVICE, ACT_HOST, and ACT_ALL states, and written to in ACT_DEVICE and ACT_ALL states. (See "1.5 Power Management Functions" for detailed information on individual states.)

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x100	***H_Reset***	R/W	0x01								***ResetHTM***
0x101			0xXX								
0x102	H_NegoControl_0	R/W	0x1X	AutoModeCancel	HostState[2:0]			AutoMode[3:0]			
0x103											
0x104	H_NegoControl_1	R/W	0x10			PortSpeed[1:0]				DisChirpFinish	RmtWkupDetEnb
0x105			0xXX								
0x106	H_USB_Test	R/W	0x00	EnHS_Test			Test_Force_Enable	Test_SE0_NAK	Test_J	Test_K	Test_Packet
0x107			0xXX								
0x108			0xXX								
0x109			0xXX								
0x10A			0xXX								
0x10B			0xXX								
0x10C			0xXX								
0x10D			0xXX								
0x10E			0xXX								
0x10F			0xXX								
Byte Addr.	**Register Name**	**R/W**	**Reset**	**bit15**	**bit14**	**bit13**	**bit12**	**bit11**	**bit10**	**bit9**	**bit8**
				bit7	**bit6**	**bit5**	**bit4**	**bit3**	**bit2**	**bit1**	**bit0**
0x110	H_CH0SETUP_0	R/W	0x00	SETUP_0[7:0]							
0x111	H_CH0SETUP_1	R/W	0x00	SETUP_1[7:0]							
0x112	H_CH0SETUP_2	R/W	0x00	SETUP_2[7:0]							
0x113	H_CH0SETUP_3	R/W	0x00	SETUP_3[7:0]							
0x114	H_CH0SETUP_4	R/W	0x00	SETUP_4[7:0]							
0x115	H_CH0SETUP_5	R/W	0x00	SETUP_5[7:0]							
0x116	H_CH0SETUP_6	R/W	0x00	SETUP_6[7:0]							
0x117	H_CH0SETUP_7	R/W	0x00	SETUP_7[7:0]							
0x118			0xXX								
0x119			0xXX								
0x11A			0xXX								
0x11B			0xXX								
0x11C	(Reserved)	R/W	0xXX								(Reserved)
0x11D			0xXX								
0x11E	H_FrameNumber_H	R	0x07						FrameNumber[10:8]		
0x11F	H_FrameNumber_L	R	0xFF	FrameNumber[7:0]							

Registers that can be read from or written to even in SLEEP or SNOOZE states are indicated in ***bold italic***.

All other registers can be read from in ACTIVE60, ACT_DEVICE, ACT_HOST, and ACT_ALL states, and written to in ACT_DEVICE and ACT_ALL states. (See "1.5 Power Management Functions" for detailed information on individual states.)

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x120	H_CH0Config_0	R/W	0x00	ACK_Cnt[3:0]				SpeedMode[1:0]		Toggle	TranGo
0x121	H_CH0Config_1	R/W	0x00	TID[1:0]							
0x122			0xXX								
0x123	H_CH0MaxPktSize	R/W	0x00	MaxPktSize[6:0]							
0x124			0xXX								
0x125			0xXX								
0x126	H_CH0TotalSize_H	R/W	0x00	TotalSize[15:8]							
0x127	H_CH0TotalSize_L	R/W	0x00	TotalSize[7:0]							
0x128	H_CH0HubAdrs	R/W	0x00	HubAdrs[3:0]					Port[2:0]		
0x129	H_CH0FuncAdrs	R/W	0x00	FuncAdrs[3:0]				EP_Number[3:0]			
0x12A			0xXX								
0x12B	H_CTL_SupportControl	R/W	0xXX			CTL_SupportState[1:0]					CTL_SupportGo
0x12C			0xXX								
0x12D			0xXX								
0x12E	H_CH0ConditionCode	R	0x00	ConditonCode[2:0]							
0x12F			0xXX								

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x130	H_CHaConfig_0	R/W	0x00	ACK_Cnt[3:0]				SpeedMode[1:0]		Toggle	TranGo
0x131	H_CHaConfig_1	R/W	0x00	TID[1:0]				AutoZerolen			TotalSizeFree
0x132	H_CHaMaxPktSize_H	R/W	0x00						MaxPktSize[10:8]		
0x133	H_CHaMaxPktSize_L	R/W	0x00	MaxPktSize[7:0]							
0x134	H_CHaTotalSize_HH	R/W	0x00	TotalSize[31:24]							
0x135	H_CHaTotalSize_HL	R/W	0x00	TotalSize[23:16]							
0x136	H_CHaTotalSize_LH	R/W	0x00	TotalSize[15:8]							
0x137	H_CHaTotalSize_LL	R/W	0x00	TotalSize[7:0]							
0x138	H_CHaHubAdrs	R/W	0x00	HubAdrs[3:0]					Port[2:0]		
0x139	H_CHaFuncAdrs	R/W	0x00	FuncAdrs[3:0]				EP_Number[3:0]			
0x13A	H_CHaBO_SupportControl	R/W	0x00			BO_TransportState[1:0]					BO_SupportGo
0x13B	H_CHaCSW_RcvDataSize	R/W	0x00					CSW_RcvDataSize[3:0]			
0x13C	H_CHaBO_OUT_EP_Control	R/W	0x00				OUT_Toggle	OUT_EP_Number[3:0]			
0x13D	H_CHaBO_IN_EP_Control	R/W	0x00				IN_Toggle	IN_EP_Number[3:0]			
0x13E	H_CHaConditionCode	R	0x00	ConditonCode[2:0]							
0x13F			0xXX								

Registers that can be read from or written to even in SLEEP or SNOOZE states are indicated in ***bold italic***.

All other registers can be read from in ACTIVE60, ACT_DEVICE, ACT_HOST, and ACT_ALL states, and written to in ACT_DEVICE and ACT_ALL states. (See "1.5 Power Management Functions" for detailed information on individual states.)

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x140	H_CHbConfig_0	R/W	0x00	ACK_Cnt[3:0]				SpeedMode[1:0]		Toggle	TranGo
0x141	H_CHbConfig_1	R/W	0x00	TID[1:0]		TranType[1:0]		AutoZerolen	Audio441	TotalSizeFree[1:0]	
0x142	H_CHbMaxPktSize_H	R/W	0x00						MaxPktSize[10:8]		
0x143	H_CHbMaxPktSize_L	R/W	0x00	MaxPktSize[7:0]							
0x144	H_CHbTotalSize_HH	R/W	0x00	TotalSize[31:24]							
0x145	H_CHbTotalSize_HL	R/W	0x00	TotalSize[23:16]							
0x146	H_CHbTotalSize_LH	R/W	0x00	TotalSize[15:8]							
0x147	H_CHbTotalSize_LL	R/W	0x00	TotalSize[7:0]							
0x148	H_CHbHubAdrs	R/W	0x00	HubAdrs[3:0]					Port[2:0]		
0x149	H_CHbFuncAdrs	R/W	0x00	FuncAdrs[3:0]				EP_Number[3:0]			
0x14A	H_CHbInterval_H	R/W	0x00						Interval[10:8]		
0x14B	H_CHbInterval_L	R/W	0x00	Interval[7:0]							
0x14C	H_CHbTranPause	R/W	0x00							EnTranPause	TranPause
0x14D			0xXX								
0x14E	H_CHbConditionCode	R	0x00		ConditonCode[2:0]						
0x14F			0xXX								
Byte Addr.	**Register Name**	**R/W**	**Reset**	**bit15**	**bit14**	**bit13**	**bit12**	**bit11**	**bit10**	**bit9**	**bit8**
				bit7	**bit6**	**bit5**	**bit4**	**bit3**	**bit2**	**bit1**	**bit0**
0x150	H_CHcConfig_0	R/W	0x00	ACK_Cnt[3:0]				SpeedMode[1:0]		Toggle	TranGo
0x151	H_CHcConfig_1	R/W	0x00	TID[1:0]		TranType[1:0]		AutoZerolen	Audio441	TotalSizeFree[1:0]	
0x152	H_CHcMaxPktSize_H	R/W	0x00						MaxPktSize[10:8]		
0x153	H_CHcMaxPktSize_L	R/W	0x00	MaxPktSize[7:0]							
0x154	H_CHcTotalSize_HH	R/W	0x00	TotalSize[31:24]							
0x155	H_CHcTotalSize_HL	R/W	0x00	TotalSize[23:16]							
0x156	H_CHcTotalSize_LH	R/W	0x00	TotalSize[15:8]							
0x157	H_CHcTotalSize_LL	R/W	0x00	TotalSize[7:0]							
0x158	H_CHcHubAdrs	R/W	0x00	HubAdrs[3:0]						Port[2:0]	
0x159	H_CHcFuncAdrs	R/W	0x00	FuncAdrs[3:0]				EP_Number[3:0]			
0x15A	H_CHcInterval_H	R/W	0x00						Interval[10:8]		
0x15B	H_CHcInterval_L	R/W	0x00	Interval[7:0]							
0x15C	H_CHcTranPause	R/W	0x00							EnTranPause	TranPause
0x15D			0xXX								
0x15E	H_CHcConditionCode	R	0x00		ConditonCode[2:0]						
0x15F			0xXX								

Registers that can be read from or written to even in SLEEP or SNOOZE states are indicated in ***bold italic***.

All other registers can be read from in ACTIVE60, ACT_DEVICE, ACT_HOST, and ACT_ALL states, and written to in ACT_DEVICE and ACT_ALL states. (See "1.5 Power Management Functions" for detailed information on individual states.)

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x160	H_CHdConfig_0	R/W	0x00	ACK_Cnt[3:0]				SpeedMode[1:0]		Toggle	TranGo
0x161	H_CHdConfig_1	R/W	0x00	TID[1:0]		TranType[1:0]		AutoZerolen	Audio441	TotalSizeFree[1:0]	
0x162	H_CHdMaxPktSize_H	R/W	0x00						MaxPktSize[10:8]		
0x163	H_CHdMaxPktSize_L	R/W	0x00	MaxPktSize[7:0]							
0x164	H_CHdTotalSize_HH	R/W	0x00	TotalSize[31:24]							
0x165	H_CHdTotalSize_HL	R/W	0x00	TotalSize[23:16]							
0x166	H_CHdTotalSize_LH	R/W	0x00	TotalSize[15:8]							
0x167	H_CHdTotalSize_LL	R/W	0x00	TotalSize[7:0]							
0x168	H_CHdHubAdrs	R/W	0x00	HubAdrs[3:0]					Port[2:0]		
0x169	H_CHdFuncAdrs	R/W	0x00	FuncAdrs[3:0]				EP_Number[3:0]			
0x16A	H_CHdInterval_H	R/W	0x00						Interval[10:8]		
0x16B	H_CHdbnterval_L	R/W	0x00	Interval[7:0]							
0x16C	H_CHdTranPause	R/W	0x00							EnTranPause	TranPause
0x16D			0xXX								
0x16E	H_CHdConditionCode	R	0x00		ConditonCode[2:0]						
0x16F			0xXX								
Byte Addr.	**Register Name**	**R/W**	**Reset**	**bit15**	**bit14**	**bit13**	**bit12**	**bit11**	**bit10**	**bit9**	**bit8**
				bit7	**bit6**	**bit5**	**bit4**	**bit3**	**bit2**	**bit1**	**bit0**
0x170	H_CHeConfig_0	R/W	0x00	ACK_Cnt[3:0]				SpeedMode[1:0]		Toggle	TranGo
0x171	H_CHeConfig_1	R/W	0x00	TID[1:0]		TranType[1:0]		AutoZerolen	Audio441	TotalSizeFree[1:0]	
0x172	H_CHeMaxPktSize_H	R/W	0x00						MaxPktSize[10:8]		
0x173	H_CHeMaxPktSize_L	R/W	0x00	MaxPktSize[7:0]							
0x174	H_CHeTotalSize_HH	R/W	0x00	TotalSize[31:24]							
0x175	H_CHeTotalSize_HL	R/W	0x00	TotalSize[23:16]							
0x176	H_CHeTotalSize_LH	R/W	0x00	TotalSize[15:8]							
0x177	H_CHeTotalSize_LL	R/W	0x00	TotalSize[7:0]							
0x178	H_CHeHubAdrs	R/W	0x00	HubAdrs[3:0]						Port[2:0]	
0x179	H_CHeFuncAdrs	R/W	0x00	FuncAdrs[3:0]				EP_Number[3:0]			
0x17A	H_CHeInterval_H	R/W	0x00						Interval[10:8]		
0x17B	H_CHeInterval_L	R/W	0x00	Interval[7:0]							
0x17C	H_CHeTranPause	R/W	0x00							EnTranPause	TranPause
0x17D			0xXX								
0x17E	H_CHeConditionCode	R	0x00		ConditonCode[2:0]						
0x17F			0xXX								

Registers that can be read from or written to even in SLEEP or SNOOZE states are indicated in ***bold italic***.

All other registers can be read from in ACTIVE60, ACT_DEVICE, ACT_HOST, and ACT_ALL states, and written to in ACT_DEVICE and ACT_ALL states. (See "1.5 Power Management Functions" for detailed information on individual states.)

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x1B0	H_TriggerFrameNum_H	R/W	0x00	UseTriggerFrame					TriggerFrameNum[10:8]		
0x1B1	H_TriggerFrameNum_L	R/W	0x00	TriggerFrameNum[7:0]							
0x1B2			0xXX								
0x1B3			0xXX								
0x1B4			0xXX								
0x1B5			0xXX								
0x1B6			0xXX								
0x1B7			0xXX								
0x1B8			0xXX								
0x1B9			0xXX								
0x1BA			0xXX								
0x1BB			0xXX								
0x1BC			0xXX								
0x1BD			0xXX								
0x1BE	***HTM_Config***	R/W	0x00			***HTM_SlopeValue[1:0]***				***HTM_TermValue[1:0]***	
0x1BF			0xXX								

Byte Addr.	Register Name	R/W	Reset	bit15	bit14	bit13	bit12	bit11	bit10	bit9	bit8
				bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
0x1F4	(Reserved)		0xXX	(Reserved)	(Reserved)	(Reserved)	(Reserved)	(Reserved)	(Reserved)	(Reserved)	(Reserved)
0x1F5	H_Protect	R/W	0x00	(Reserved)	(Reserved)	(Reserved)	(Reserved)	PortSpeedWrEnb	(Reserved)	(Reserved)	(Reserved)

2.5 Initial Register Details

2.5.1 xxxh *CPUIF_MODE (CPU I/F Mode)*

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
uninitialized	xxxh	***CPUIF_MODE***		15,7:	0:	1:	xxh
				14,6:	0:	1:	
				13,5:	0:	1:	
				12,4:	0:	1:	
				11,3:	0:	1:	
			W	10,2: ***CPU_Endain***	0: BigEndian Mode	1: LittleEndian Mode	
			W	9,1: ***BusMode***	0: Strobe Mode	1: BE Mode	
				8,0:	0:	1:	

This register is enabled during the uninitialized period. It should be set to suit the particular CPU used. See "1.8.1 CPUIF Mode" and "1.8.2 CPUIF Mode Setup" for detailed information.

It should be accessed so that the identical contents are found (mirrored) in the upper and lower bytes.

Bit15-11, Bit7-3 Reserved

Bit10, Bit2 *CPU_Endian*

Sets the CPUIF endian.

0: BigEndian mode

1: LittleEndian mode

The settings for this bit can be checked using the ChipConfig.CPU_Endian bit during the initialized period.

Bit9, Bit1 *BusMode*

Sets the CPUIF Bus mode.

0: Strobe mode

1: BE mode

The settings for this bit can be checked using the ChipConfig.BusMode bit during the initialized period.

Bit8, Bit0 Reserved

2.6 Device/Host Register Details

2.6.1 000h *MainIntStat (Main Interrupt Status)*

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device / Host	000h	***MainIntStat***	R	7: ***DeviceIntStat***	0: None	1: Device Interrupts	00h
			R	6: ***HostIntStat***	0: None	1: Host Interrupts	
			R	5: CPU_IntStat	0: None	1: CPU Interrupts	
			R	4: IDE_IntStat	0: None	1: IDE Interrupts	
			R	3: MediaFIFO_IntStat	0: None	1: MediaFIFO Interrupts	
			R	2: AREAnIntStat	0: None	1: AREA Interrupts	
				1:	0:	1:	
			R (W)	0: ***FinishedPM***	0: None	1:Detect FinishedPM	

Indicates the interrupt factors for the LSI.

This register includes bits that indirectly and directly instruct the interrupt factors. The bit for indirectly instructing interrupt factors can read the corresponding interrupt status registers to follow the bit for directly instructing interrupt factors. The bit for indirectly instructing interrupt factors is read-only and is automatically cleared by clearing the bit for directly instructing major interrupt factors. The bit for directly instructing interrupt factors can be written to; writing "1" to this bit enables the interrupt factor to be cleared. Setting the interrupt factor to "1" when the corresponding bit interrupt is enabled by the MainIntEnb register asserts the XINT terminal and issues an interrupt to the CPU. Clearing all corresponding interrupt factors negates the XINT terminal.

Bit7 ***DeviceIntStat***

Indirectly instructs interrupt factors.

This is set to "1" when the DeviceIntStat register includes interrupt factors and the DeviceIntEnb register bit corresponding to the interrupt factors is enabled. This bit allows reading even in SLEEP and SNOOZE states.

Bit6 ***HostIntStat***

Indirectly instructs interrupt factors.

This is set to "1" when the HostIntStat register includes interrupt factors and the HostIntEnb register bit corresponding to the interrupt factors is enabled. This bit allows reading even in SLEEP and SNOOZE states.

Bit5 **CPU_IntStat**

Indirectly instructs interrupt factors.

This is set to "1" when the CPU_IntStat register includes interrupt factors and the CPU_IntEnb register bit corresponding to the interrupt factors is enabled.

Bit4 **IDE_IntStat**

Indirectly instructs interrupt factors.

This is set to "1" when the IDE_IntStat register includes interrupt factors and the IDE_IntEnb register bit corresponding to the interrupt factors is enabled.

Bit3 **MediaFIFO_IntStat**

Indirectly instructs interrupt factors.

This is set to "1" when the MediaFIFO_IntStat register includes interrupt factors and the MediaFIFO_IntEnb register bit corresponding to the interrupt factors is enabled.

Bit2 **AREAnIntStat**

Indirectly instructs interrupt factors.

This is set to "1" when the AREAnIntStat register includes interrupt factors and the AREAnIntEnb register bit corresponding to the interrupt factors is enabled.

Bit1 **Reserved**

Bit0 ***FinishedPM***

Directly instructs interrupt factors.

This bit is set to "1" on reaching the particular instructed state when GoSLEEP, GoSNOOZE, GoActive60, GoActDevice, GoActHost, GoActAllDev, or GoActAllHost is set by the PM_Control_0 register.

This bit is enabled even in SLEEP and SNOOZE states.

2.6.2 001h *DeviceIntStat (Device Interrupt Status)*

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device / Host	001h	***DeviceIntStat***	R (W)	7: ***VBUS_Changed***	0: None	1: VBUS is Changed	00h
				6:	0:	1:	
			R	5: ***D_SIE_IntStat***	0: None	1: SIE Interrupts	
			R	4: D_BulkIntStat	0: None	1: Bulk Interrupts	
			R (W)	3: RcvEP0SETUP	0: None	1: Receive EP0 SETUP	
			R	2: D_FIFO_IntStat	0: None	1: FIFO Interrupts	
			R	1: D_EP0IntStat	0: None	1: EP0 Interrupts	
			R	0: D_EPrIntStat	0: None	1: EPr Interrupts	

Indicates device-related interrupts.

This register includes bits that indirectly and directly instruct the interrupt factors. The bit for indirectly instructing interrupt factors can read the corresponding interrupt status registers to follow the bit for directly instructing interrupt factors. The bit for indirectly instructing interrupt factors is read-only and is automatically cleared by clearing the bit for directly instructing major interrupt factors. The bit for directly instructing interrupt factors can be written to; writing "1" to this bit enables the interrupt factor to be cleared.

Bit7 ***VBUS_Changed***

Directly instructs interrupt factors.

This is set to "1" when the VBUS_B terminal status has changed.

Check the VBUS_B terminal status using the D_USB_Status register VBUS bit. If VBUS is "0," the cable is disconnected. This bit is enabled even in SLEEP and SNOOZE states.

Bit6 **Reserved**

Bit5 ***D_SIE_IntStat***

Indirectly instructs interrupt factors.

This is set to "1" when the D_SIE_IntStat register includes interrupt factors and the D_SIE_IntEnb register bit corresponding to the interrupt factors is enabled. This bit allows reading even in SLEEP and SNOOZE states.

Bit4 **D_BulkIntStat**

Indirectly instructs interrupt factors.

This is set to "1" when D_BulkIntStat register includes interrupt factors and the D_BulkIntEnb register bit corresponding to the interrupt factors is enabled.

Bit3 RcvEP0SETUP

Indirectly instructs interrupt factors.

This is set to "1" when the control transfer setup stage is complete and the received data is stored in registers D_EP0SETUP_0 to D_EP0SETUP_7. The D_EP0ControlIN and D_EP0ControlOUT register ForceSTALL bit is automatically set to "0" at the same time, and the D_EP0ControlIN, D_EP0ControlOUT register ForceNAK and ToggleStat bits and D_SETUP_Control register ProtectEP0 bit are automatically set to "1." The AutoSetAddress function automatically responds to the SetAddress() request, and this status is not set.

Bit2 D_FIFO_IntStat

Indirectly instructs interrupt factors.

This is set to "1" when the D_FIFO_IntStat register includes interrupt factors and the D_FIFO_IntEnb register bit corresponding to the interrupt factors is enabled.

Bit1 D_EP0IntStat

Indirectly instructs interrupt factors.

This is set to "1" when the D_EP0IntStat register includes interrupt factors and the D_EP0IntEnb register bit corresponding to the interrupt factors is enabled.

Bit0 D_EPrIntStat

Indirectly instructs interrupt factors.

This is set to "1" when the D_EPrIntStat register includes interrupt factors and the D_EPaIntEnb register bit corresponding to the interrupt factors is enabled.

2.6.3 002h *HostIntStat (Host Interrupt Status)*

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device / Host	002h	***HostIntStat***	R/(W)	7: ***VBUS_Err***	0: None	1: VBUS Error	00h
			R/(W)	6: ***LineStatusChanged***	0: None	1: Line Status Changed	
			R	5: H_SIE_IntStat1	0: None	1: SIE Interrupts1	
			R	4: H_SIE_IntStat0	0: None	1: SIE Interrupts0	
			R	3: H_FrameIntStat	0: None	1: Frame Interrupts	
			R	2: H_FIFOIntStat	0: None	1: FIFO Interrupts	
			R	1: H_CH0IntStat	0: None	1: CH0 Interrupts	
			R	0: H_CHrIntStat	0: None	1: CHr Interrupts	

Indicates host-related interrupts.

This register includes bits that indirectly and directly instruct the interrupt factors. The bit for indirectly instructing interrupt factors can read the corresponding interrupt status registers to follow the bit for directly instructing interrupt factors. The bit for indirectly instructing interrupt factors is read-only and is automatically cleared by clearing the bit for directly instructing major interrupt factors. The bit for directly instructing interrupt factors can be written to; writing "1" to this bit enables the interrupt factor to be cleared.

Bit7 ***VBUS_Err***

Directly instructs interrupt factors. This bit is enabled even in SLEEP and SNOOZE states.

This is set to "1" when a signal is input to the VBUSFLG_A terminal from the external connection VBUS power switch notifying of a VBUS error (change Edge from High to Low).

Check the VBUSFLG_A terminal status using the H_USB_Status register VBUS_State bit.

This error signal will vary, depending on the particular external connection VBUS power switch specifications. Check the specifications beforehand.

Bit6 ***LineStateChanged***

Directly instructs interrupt factors. This bit is enabled even in SLEEP and SNOOZE states.

It indicates that the host port DP and DM terminal states have changed.

This interrupt is used to detect host port signal line changes when the USB host function is not being used or when the USB host function is being used and SLEEP or SNOOZE are in use.

Bit5 **H_SIE_IntStat1**

Indirectly instructs interrupt factors.

This is set to "1" when the H_SIE_IntStat1 register includes interrupt factors and the H_SIE_IntEnb1 register bit corresponding to the interrupt factors is enabled.

Bit4 H_SIE_IntStat0

Indirectly instructs interrupt factors.

This is set to "1" when the H_SIE_IntStat0 register includes interrupt factors and the H_SIE_IntEnb0 register bit corresponding to the interrupt factors is enabled.

Bit3 H_FrameIntStat

Indirectly instructs interrupt factors.

This is set to "1" when the H_FrameIntStat register includes interrupt factors and the H_FrameIntEnb register bit corresponding to the interrupt factors is enabled.

Bit2 H_FIFO_IntStat

Indirectly instructs interrupt factors.

This is set to "1" when the H_FIFO_IntStat register includes interrupt factors and the H_FIFO_IntEnb register bit corresponding to the interrupt factors is enabled.

Bit1 H_CH0_IntStat

Indirectly instructs interrupt factors.

This is set to "1" when the H_CH0_IntStat register includes interrupt factors and the H_CH0_IntEnb register bit corresponding to the interrupt factors is enabled.

Bit0 H_CHr_IntStat

Indirectly instructs interrupt factors.

This is set to "1" when the H_CHr_IntStat register includes interrupt factors and the H_CHr_IntEnb register bit corresponding to the interrupt factors is enabled.

2.6.4 003h CPU_IntStat (CPU Interrupt Status)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device / Host	003h	CPU_IntStat	R (W)	7: RAM_RdCmp	0: None	1: RAM Read Complete	00h
			R (W)	6: C_TimerCmp	0: None	1: Timer Interrupts	
				5:	0:	1:	
				4:	0:	1:	
			R (W)	3: DMA1_Countup	0: None	1: DMA1 Counter Overflow	
			R (W)	2: DMA1_Cmp	0: None	1: DMA1 Complete	
			R (W)	1: DMA0_CountUp	0: None	1: DMA0 Counter Overflow	
			R (W)	0: DMA0_Cmp	0: None	1: DMA0 Complete	

Indicates CPU interface-related interrupts.

All bits can be set to “1” to clear the interrupt factors.

Bit7 RAM_RdCmp

Directly instructs interrupt factors.

This is set to “1” when data is read from the RAM by the RAM_Rd function and the RAM_Rd_XX data is enabled.

Bit6 C_TimerCmp

Directly instructs interrupt factors.

This is set to “1” when the timer value overflows from the value set to TimerSet_H,L.

Bit5-4 Reserved

Bit3 DMA1_CountUp

Directly instructs interrupt factors.

This is set to “1” when the DMA1_Count_HH,HL,LH,LL value overflows while the transfer mode is operating in free-run mode. The DMA1_Count_HH,HL,LH,LL value returns to 0, and DMA operation continues.

Bit2 DMA1_Cmp

Directly instructs interrupt factors.

This is set to “1” when the DMA transfer is stopped or if the specified transfer quantity has been sent and end processing is complete.

Bit1 DMA0_CountUp

Directly instructs interrupt factors.

This is set to “1” when the DMA0_Count_HH,HL,LH,LL value overflows while the transfer mode is operating in free-run mode. The DMA0_Count_HH,HL,LH,LL value returns to 0, and DMA operation continues.

Bit0 DMA0_Cmp

Directly instructs interrupt factors.

This is set to “1” when the DMA transfer is stopped or if the specified transfer quantity has been sent and end processing is complete.

2.6.5 004h IDE_IntStat (IDE Interrupt Status)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device / Host	004h	IDE_IntStat	R (W)	7: IDE_RegCmp	0: None	1: Register Access Cmplete	00h
			R (W)	6: IDE_RegErr	0: None	1: Register Access Error	
			R (W)	5: IDE_SeqWrRegCmp	0: None	1: Sequence Write Complete	
			R (W)	4: CompleteINTRQ	0: None	1: Auto Status Read Compl.	
				3:	0:	1:	
			R (W)	2: IDE_Cmp	0: None	1: DMA Complete	
			R (W)	1: DetectINTRQ	0: None	1: Detected Interrupt	
			R (W)	0: DetectTerm	0: None	1: Detected Device terminate	

Indicates IDE-related interrupt status.

All bits can be set to "1" to clear the interrupt factors.

Bit7 IDE_RegCmp

Directly instructs interrupt factors.

This is set to "1" when the read/write access to the IDE register by the IDE_RegAdrs register is complete.

Bit6 IDE_RegErr

Directly instructs interrupt factors.

This is set to "1" in the following situations.

1) The IDE_RegAdrs register read/write accesses the IDE register during the automatic status reading operation sequence by the IDE_RegConfig register.
2) The IDE_RegAdrs register read/write accesses the IDE register during the sequence writing operation sequence by the IDE_SeqWrRegControl register.

Bit5 IDE_SeqWrRegCmp

Directly instructs interrupt factors.

This is set to "1" when the sequence writing operation by the IDE_SeqWrRegControl register is complete.

Bit4 CompleteINTRQ

Directly instructs interrupt factors.

This is set to "1" when the automatic status reading operation by the IDE_RegConfig register is complete.

Bit3 Reserved

Bit2 IDE_Cmp

Directly instructs interrupt factors.

This is set to “1” when the DMA operation by the IDE_Control register is complete.

Bit1 DetectINTRQ

Directly instructs interrupt factors.

This is set to “1” when IDE HINTRQ signal startup is detected while the IDE_RegConfig register EnAutoStsRd bit is not set.

Bit0 DetectTerm

Directly instructs interrupt factors.

This is set to “1” when a device terminate is detected in ultra-DMA transfer mode during IDE DMA operation by the IDE_Control register.

2.6.6 005h MediaFIFO_IntStat (Media FIFO Interrupt Status)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device / Host	005h	MediaFIFO_IntStat		7:	0:	1:	00h
			R (W)	6: MediaIDE_Cmp	0: None	1: Media FIFO IDE Complete	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
			R (W)	2: FIFO_NotEmpty	0: None	1: FIFO NotEmpty	
			R (W)	1: FIFO_Full	0: None	1: FIFO Full	
			R (W)	0: FIFO_Empty	0: None	1: FIFO Empty	

Indicates Media FIFO-related interrupt status.

All bits can be set to “1” to clear the interrupt factors.

Bit7 Reserved

Bit6 MediaIDE_Cmp

Directly instructs interrupt factors.

This is set to “1” if MediaFIFO is emptied after the IDE transfer ends when the IDE is “Join” and Dir is “0.” This is set to “1” if the IDE transfer ends when Dir is “1.”

Bit5-3 Reserved

Bit2 FIFO_NotEmpty

Directly instructs interrupt factors.

This is set to “1” if the Media FIFO contains data (NotEmpty).

Bit1 FIFO_Full

Directly instructs interrupt factors.

This is set to “1” when the Media FIFO is full.

Bit0 FIFO_Empty

Directly instructs interrupt factors.

This is set to “1” when the Media FIFO is empty.

2.6.7 010h *MainIntEnb (Main Interrupt Enable)*

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device / Host	010h	***MainIntEnb***	R/W	7: ***EnDeviceIntStat***	0: Disable	1: Enable	00h
			R/W	6: ***EnHostIntStat***	0: Disable	1: Enable	
			R/W	5: EnCPU_IntStat	0: Disable	1: Enable	
			R/W	4: En IDE_IntStat	0: Disable	1: Enable	
			R/W	3: EnMediaFIFO_IntStat	0: Disable	1: Enable	
			R/W	2: EnAREAnIntStat	0: Disable	1: Enable	
				1:	0:	1	
			R/W	0: ***EnFinishedPM***	0: Disable	1: Enable	

This register permits or prohibits interrupt signal (XINT) asserting by MainIntStat interrupt factors.

An interrupt is permitted if the corresponding bit is set to “1.”

The EnDeviceIntStat, EnHostIntStat, and EnFinishedPM bits are enabled even in SLEEP and SNOOZE states.

2.6.8 011h *DeviceIntEnb (Device Interrupt Enable)*

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device / Host	011h	***DeviceIntEnb***	R/W	7: ***EnVBUS_Changed***	0: Disable	1: Enable	00h
				6:	0:	1:	
			R/W	5: ***EnD_SIE_IntStat***	0: Disable	1: Enable	
			R/W	4: EnD_BulkIntStat	0: Disable	1: Enable	
			R/W	3: EnRcvEP0SETUP	0: Disable	1: Enable	
			R/W	2: EnD_FIFO_IntStat	0: Disable	1: Enable	
			R/W	1: EnD_EP0IntStat	0: Disable	1: Enable	
			R/W	0: EnD_EPrIntStat	0: Disable	1: Enable	

This permits or prohibits MainIntStat register DeviceIntStat bit asserting by DeviceIntStat register interrupt factors.

The EnVBUS_Changed and EnD_SIE_IntStat bits are enabled even in SLEEP and SNOOZE states.

2.6.9 012h *HostIntEnb (Host Interrupt Enable)*

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device / Host	012h	***HostIntEnb***	R/W	7: ***EnVBUS_Err***	0: Disable	1: Enable	00h
			R/W	6: ***EnLineStatusChanged***	0: Disable	1: Enable	
			R/W	5: EnH_SIE_IntStat1	0: Disable	1: Enable	
			R/W	4: EnH_SIE_IntStat0	0: Disable	1: Enable	
			R/W	3: EnH_FrameIntStat	0: Disable	1: Enable	
			R/W	2: EnH_FIFOIntStat	0: Disable	1: Enable	
			R/W	1: EnH_CH0IntStat	0: Disable	1: Enable	
			R/W	0: EnH_CHrIntStat	0: Disable	1: Enable	

This permits or prohibits MainIntStat register HostIntStat bit asserting by HostIntStat register interrupt factors.

The EnVBUS_Err and EnLineStatusChanged bits are enabled even in SLEEP and SNOOZE states.

2.6.10 013h CPU_IntEnb (CPU Interrupt Enable)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device / Host	013h	CPU_IntEnb	R/W	7: EnRAM_RdCmp	0: Disable	1: Enable	00h
			R/W	6: EnC_TimerCmp	0: Disable	1: Enable	
				5:	0:	1:	
				4:	0:	1:	
			R/W	3: EnDMA1_Countup	0: Disable	1: Enable	
			R/W	2: EnDMA1_Cmp	0: Disable	1: Enable	
			R/W	1: EnDMA0_CountUp	0: Disable	1: Enable	
			R/W	0: EnDMA0_Cmp	0: Disable	1: Enable	

This permits or prohibits MainIntStat register CPU_IntStat bit asserting by CPU_IntStat register interrupt factors.

2.6.11 014h IDE_IntEnb (IDE Interrupt Enable)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device / Host	014h	IDE_IntEnb	R/W	7: EnIDE_RegCmp	0: Disable	1: Enable	00h
			R/W	6: EnIDE_RegErr	0: Disable	1: Enable	
			R/W	5: En_SeqWrRegCmp	0: Disable	1: Enable	
			R/W	4: EnCompleteINTRQ	0: Disable	1: Enable	
				3:	0:	1:	
			R/W	2: EnIDE_Cmp	0: Disable	1: Enable	
			R/W	1: EnDetectINTRQ	0: Disable	1: Enable	
			R/W	0: EnDetectTerm	0: Disable	1: Enable	

This permits or prohibits MainIntStat register IDE_IntStat bit asserting by IDE_IntStat register interrupt factors.

2.6.12 015h MediaFIFO_IntEnb (Media FIFO Interrupt Enable)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device / Host	015h	MediaFIFO_IntEnb		7:	0:	1:	00h
			R/W	6: EnMediaIDE_Cmp	0: Disable	1: Enable	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
			R/W	2: EnFIFO_NotEmpty	0: Disable	1: Enable	
			R/W	1: EnFIFO_Full	0: Disable	1: Enable	
			R/W	0: EnFIFO_Empty	0: Disable	1: Enable	

This permits or prohibits MainIntStat register MediaFIFO_IntStat bit asserting by MediaFIFO_IntStat register interrupt factors.

2.6.13 020h *RevisionNum (Revision Number)*

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device / Host	020h	***RevisionNum***	R	7: ***RevisionNum[7]***	Revision Number	60h
				6: ***RevisionNum[6]***		
				5: ***RevisionNum[5]***		
				4: ***RevisionNum[4]***		
				3: ***RevisionNum[3]***		
				2: ***RevisionNum[2]***		
				1: ***RevisionNum[1]***		
				0: ***RevisionNum[0]***		

This indicates the LSI revision number. This register can be accessed even in SLEEP and SNOOZE states.

The revision number is 0x60 for the current specifications.

2.6.14 021h *ChipReset (Chip Reset)*

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device / Host	021h	***ChipReset***		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
			W	0: ***AllReset***	0: None	1: Reset	

This resets the LSI.

It can be accessed even in SLEEP and SNOOZE states.

Bit7-1 Reserved

Bit0 *AllReset*

This initializes all circuits except for the CPUIF_MODE register. Soft resetting using this register does not switch to the uninitialized period.

Note that this register should not be written to except for resetting purposes.

Writing to this register in contravention of the AC spec except for resetting purposes will cause malfunctions.

2.6.15 022h *PM_Control_0 (Power Management Control 0)*

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device / Host	022h	***PM_Control_0***	R/W	7: ***GoSLEEP***	0: Do nothing	1: Go to SLEEP	00h
			R/W	6: ***GoSNOOZE***	0: Do nothing	1: Go to SNOOZE	
			R/W	5: ***GoActive60***	0: Do nothing	1: Go to ACTIVE60	
			R/W	4: ***GoActDevice***	0: Do nothing	1: Go to ACT_DEVICE	
			R/W	3: ***GoActHost***	0: Do nothing	1: Go to ACT_HOST	
			R/W	2: ***GoActAllDev***	0: Do nothing	1: Go to ACT_ALL_DEV	
			R/W	1: ***GoActAllHost***	0: Do nothing	1: Go to ACT_ALL_HOST	
				0:	0:	1:	

This sets the LSI power management related operations.

The register is enabled even in SLEEP and SNOOZE states.

Bit7 ***GoSLEEP***

This starts the switch to SLEEP state from states other than SLEEP.

Setting this bit to "1" while in SNOOZE state stops the oscillator oscillation and switches to SLEEP state. Setting the bit to "1" while in ACTIVE60 state stops PLL60 oscillation before stopping oscillator oscillation and switching to SLEEP state. Setting the bit to "1" while in ACT_DEVICE state stops DevicePLL480 oscillation, then stops PLL60 oscillation before stopping oscillator oscillation and switching to SLEEP state. Setting the bit to "1" while in ACT_HOST state stops HostPLL480 oscillation, then stops PLL60 oscillation before stopping oscillator oscillation and switching to SLEEP state. Setting the bit to "1" while in ACT_ALL state stops DevicePLL480 oscillation, then stops HostPLL480 oscillation before stopping PLL60 oscillation, stopping oscillator oscillation and switching to SLEEP state.

This bit is automatically cleared as soon as the switch is complete, regardless of the state switched from, and the MainIntStat.FinishedPM bit is set concurrently.

Bit6 ***GoSNOOZE***

This starts the switch to SNOOZE state from states other than SNOOZE.

Setting this bit to "1" while in SLEEP state starts the oscillator oscillation and switches to SNOOZE state once the oscillator oscillation stabilization time has elapsed (time set in WakeupTim_H, L). Setting the bit to "1" while in ACTIVE60 state stops PLL60 oscillation and switches to SNOOZE state. Setting the bit to "1" while in ACT_DEVICE state stops DevicePLL480 oscillation, then stops PLL60 oscillation before switching to SNOOZE state. Setting the bit to "1" while in ACT_HOST state stops HostPLL480 oscillation, then stops PLL60 oscillation before switching to SNOOZE state. Setting the bit to "1" while in ACT_ALL state stops DevicePLL480 oscillation, then stops HostPLL480 oscillation before stopping PLL60 oscillation, switching to SNOOZE state.

This bit is automatically cleared as soon as the switch is complete, regardless of the state switched from, and the MainIntStat.FinishedPM bit is set concurrently.

Bit5 ***GoActive60***

This starts the switch to ACTIVE60 state from states other than ACTIVE60.

Setting this bit to "1" while in SLEEP state starts the oscillator oscillation, starts the PLL60 oscillation after the oscillator oscillation stabilization time has elapsed (time set in WakeupTim_H, L), then switches to ACTIVE60 state after the PLL60 oscillation stabilization time (approximately 250 µs) has elapsed. Setting the bit to "1" while in SNOOZE state starts the PLL60 oscillation and switches to ACTIVE60 state after the PLL oscillation stabilization time (approximately 250 µs) has elapsed.

Setting the bit to "1" while in ACT_DEVICE state stops DevicePLL480 oscillation and switches to ACTIVE60 state. Setting the bit to "1" while in ACT_HOST state stops HostPLL480 oscillation and switches to ACTIVE60 state. Setting the bit to "1" while in ACT_ALL state stops the DevicePLL480 and HostPLL480 oscillation, then switches to ACTIVE60 state.

This bit is automatically cleared as soon as the switch is complete, regardless of the state switched from, and the MainIntStat.FinishedPM bit is set concurrently.

Bit4 ***GoActDevice***

This starts the switch to ACT_DEVICE state from states other than ACT_DEVICE.

Setting this bit to "1" while in SLEEP state starts the oscillator oscillation, starts the PLL60 oscillation after the oscillator oscillation stabilization time has elapsed (time set in WakeupTim_H, L), starts the DevicePLL480 oscillation after the PLL60 oscillation stabilization time (approximately 250 µs) has elapsed, then switches to ACT_DEVICE state after the PLL480 oscillation stabilization time (approximately 250 µs) has elapsed. Setting the bit to "1" while in SNOOZE state starts the PLL60 oscillation, starts the DevicePLL480 oscillation after the PLL oscillation stabilization time (approximately 250 µs) has elapsed, then switches to ACT_DEVICE state after the PLL480 oscillation stabilization time (approximately 250 µs) has elapsed. Setting the bit to "1" while in ACTIVE60 state starts the DevicePLL480 oscillation and switches to ACT_DEVICE state after the PLL480 oscillation stabilization time (approximately 250 µs) has elapsed.

Setting the bit to "1" while in ACT_HOST state stops HostPLL480 oscillation and switches to ACTIVE60 state. DevicePLL480 oscillation is then started before switching to ACT_DEVICE state after the PLL480 oscillation stabilization time (approximately 250 µs) has elapsed.

Setting the bit to "1" while in ACT_ALL state stops HostPLL480 oscillation and switches to ACT_DEVICE state.

This bit is automatically cleared as soon as the switch is complete, regardless of the state switched from, and the MainIntStat.FinishedPM bit is set concurrently.

Bit3 ***GoActHost***

This starts the switch to ACT_HOST state from states other than ACT_HOST.

Setting this bit to "1" while in SLEEP state starts the oscillator oscillation, starts the PLL60 oscillation after the oscillator oscillation stabilization time has elapsed (time set in WakeupTim_H, L), starts the HostPLL480 oscillation after the PLL60 oscillation stabilization time (approximately 250 µs) has elapsed, then switches to ACT_HOST state after the PLL480 oscillation stabilization time (approximately 250 µs) has elapsed.

Setting the bit to "1" while in SNOOZE state starts the PLL60 oscillation, starts the HostPLL480 oscillation after the PLL oscillation stabilization time (approximately 250 µs) has elapsed, then switches to ACT_HOST state after the PLL480 oscillation stabilization time (approximately 250 µs) has elapsed.

Setting the bit to "1" while in ACTIVE60 state starts the HostPLL480 oscillation and switches to ACT_HOST state after the PLL480 oscillation stabilization time (approximately 250 µs) has elapsed.

Setting the bit to "1" while in ACT_DEVICE state stops DevicePLL480 oscillation and switches to ACTIVE60 state. HostPLL480 oscillation is then started before switching to ACT_HOST state after the PLL480 oscillation stabilization time (approximately 250 µs) has elapsed.

Setting the bit to "1" while in ACT_ALL state stops DevicePLL480 oscillation and switches to ACT_HOST state.

This bit is automatically cleared as soon as the switch is complete, regardless of the state switched from, and the MainIntStat.FinishedPM bit is set concurrently.

Bit2 ***GoActAllDev***

This starts the switch to ACT_ALL_DEV substate from substates other than ACT_ALL_DEV.

The ACT_ALL_DEV substate is defined as a substate to which such switch is performed after first switching to the ACT_ALL state and clearing the PM_Control_1.MonActFunc bit to "0."

Setting this bit to "1" while in SLEEP state starts the oscillator oscillation, starts the PLL60 oscillation after the oscillator oscillation stabilization time has elapsed (time set in WakeupTim_H, L), starts the DevicePLL480 and HostPLL480 oscillation after the PLL60 oscillation stabilization time (approximately 250 µs) has elapsed, then clears the PM_Control_1.MonActFunc bit to "0" and switches to the ACT_ALL_DEV substate after the PLL480 oscillation stabilization time (approximately 250 µs) has elapsed.

Setting the bit to "1" while in SNOOZE state starts the PLL60 oscillation, starts the DevicePLL480 and HostPLL480 oscillation after the PLL oscillation stabilization time (approximately 250 µs) has elapsed, then clears the PM_Control_1.MonActFunc bit to "0" and switches to the ACT_ALL_DEV substate after the PLL480 oscillation stabilization time (approximately 250 µs) has elapsed. Setting the bit to "1" while in ACTIVE60 state starts the DevicePLL480 and HostPLL480 oscillation, clears the PM_Control_1.MonActFunc bit to "0" and switches to the

ACT_ALL_DEV substate after the PLL480 oscillation stabilization time (approximately 250 µs) has elapsed.

Setting the bit to "1" while in ACT_DEVICE state starts the HostPLL480 oscillation, clears the PM_Control_1.MonActFunc bit to "0" and switches to the ACT_ALL_DEV substate after the PLL480 oscillation stabilization time (approximately 250 µs) has elapsed.

Setting the bit to "1" while in ACT_ALL_HOST substate clears the PM_Control_1.MonActFunc bit to "0" and switches to the ACT_ALL_DEV substate.

This bit is automatically cleared as soon as the switch is complete, regardless of the state switched from, and the MainIntStat.FinishedPM bit is set concurrently.

Both DevicePLL480 and HostPLL480 PLLs operate in the ACT_ALL_DEV substate, allowing use of both functions, but the data transfer can be used only for USB Device.

Bit1 ***GoActAllHost***

This starts the switch to ACT_ALL_HOST substate from substates other than ACT_ALL_HOST.

The ACT_ALL_HOST substate is defined as a substate to which such switch is performed after first switching to the ACT_ALL state and setting the PM_Control_1.MonActFunc bit to "1."

Setting this bit to "1" while in SLEEP state starts the oscillator oscillation, starts the PLL60 oscillation after the oscillator oscillation stabilization time has elapsed (time set in WakeupTim_H, L), starts the DevicePLL480 and HostPLL480 oscillation after the PLL60 oscillation stabilization time (approximately 250 µs) has elapsed, then sets the PM_Control_1.MonActFunc bit to "1" and switches to the ACT_ALL_HOST substate after the PLL480 oscillation stabilization time (approximately 250 µs) has elapsed.

Setting the bit to "1" while in SNOOZE state starts the PLL60 oscillation, starts the DevicePLL480 and HostPLL480 oscillation after the PLL oscillation stabilization time (approximately 250 µs) has elapsed, then sets the PM_Control_1.MonActFunc bit to "1" and switches to the ACT_ALL_HOST substate after the PLL480 oscillation stabilization time (approximately 250 µs) has elapsed.

Setting the bit to "1" while in ACTIVE60 state starts the DevicePLL480 and HostPLL480 oscillation, sets the PM_Control_1.MonActFunc bit to "1" and switches to the ACT_ALL_HOST substate after the PLL480 oscillation stabilization time (approximately 250 µs) has elapsed.

Setting the bit to "1" while in ACT_DEVICE state starts the HostPLL480 oscillation, sets the PM_Control_1.MonActFunc bit to "1" and switches to the ACT_ALL_HOST substate after the PLL480 oscillation stabilization time (approximately 250 µs) has elapsed.

Setting the bit to "1" while in ACT_ALL_DEV substate sets the PM_Control_1.MonActFunc bit to "1" and switches to the ACT_ALL_HOST substate.

This bit is automatically cleared as soon as the switch is complete, regardless of the state switched from, and the MainIntStat.FinishedPM bit is set concurrently.

Both DevicePLL480 and HostPLL480 PLLs operate in the ACT_ALL_HOST substate, allowing use of both functions, but the data transfer can be used only for Host.

* This LSI is masked to prevent assertion of the XINT signal in SNOOZE state due to an interrupt status (hereafter referred to as synchronous status) that cannot be accessed during SLEEP or SNOOZE states. However, the following processing should be performed via firmware to prevent assertion of the XINT terminal when the SNOOZE state is reset.

  <Before starting SLEEP or SNOOZE>

  Process and clear synchronous status (-IntStat).

  Disable synchronous status (-IntEnb).

  <After clearing SLEEP or SNOOZE >

  Clear synchronous status (-IntStat).

  Enable synchronous status (-IntEnb).

Bit0 Reserved

2.6.16 023h *PM_Control_1 (Power Management Control 1)*

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device / Host	023h	***PM_Control_1***	R	7: ***MonActFunc***	0: Device mode	1: Host mode	00h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
			R	3: ***PM_State[3]***	PM_State[3:0]		
				2: ***PM_State[2]***			
				1: ***PM_State[1]***			
				0: ***PM_State[0]***			

This sets the LSI power management related operations.

The register is enabled even in SLEEP and SNOOZE states.

Bit7 ***MonActFunc***

Indicates the USB functions that allow transfers in the ACT_ALL state.

0: Allows data transfers for USB Device

1: Allows data transfers for USB Host

Bit7-4 **Reserved**

Bit3-0 ***PM_State[3:0]***

Indicates the power management state.

0000: SLEEP state (OSC off, PLL60 off, DevicePLL480 off, HostPLL480 off)

0001: SNOOZE state (OSC on, PLL60 off, DevicePLL480 off, HostPLL480 off)

0011: ACTIVE60 state (OSC on, PLL60 on, DevicePLL480 off, HostPLL480 off)

0111: ACT_DEVICE state (OSC on, PLL60 on, DevicePLL480 on, HostPLL480 off)

1011: ACT_HOST state (OSC on, PLL60 on, DevicePLL480 off, HostPLL480 on)

1111: ACT_ALL state (OSC on, PLL60 on, DevicePLL480 on, HostPLL480 on)

Other: Not used

Note that this state should not be referenced, since its sequence to the corresponding state varies from when PM_Control_0.GoXXXX is set until when MainIntStat.FinishedPM interrupt status is set and the PM_Control_0.GoXXXX bit is cleared.

The ACT_ALL state in which the MonActFunc bit is cleared to “0” is defined as the ACT_ALL_DEV substate. The ACT_ALL state in which the MonActFunc bit is set to “1” is defined as the ACT_ALL_HOST substate.

2.6.17 024h *WakeupTim_H (Wakeup Time High)*

2.6.18 025h *WakeupTim_L (Wakeup Time Low)*

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device / Host	024h -025h	***WakeupTim_H*** ***WakeupTim_L***	R/W	***WakeupTim[15:0]***	Wakeup Time[15:0]	0000h

This sets the oscillator oscillation stabilization time when returning from SLEEP to SNOOZE state. This register can be accessed even in SLEEP state.

Writing "1" to the PM_Control_0.GoActDevice, PM_Control_0.GoActHost, PM_Control_0.GoActive60, or PM_Control_0.GoSNOOZE bit in SLEEP state enables the oscillator cell and initiates oscillator oscillation. This WakeupTim_H,L setting is loaded into the counter, and the OSC startup starts the countdown. Once the countdown ends, the internal OSCCLK gate is opened, and CLK is sent to the PLL and other circuits.

The oscillator oscillation stabilization time depends on factors such as oscillator, oscillator cell, circuit board, and load capacity. The internal SCLK must be stabilized to 60 MHz ±10% within 5.1 ms after USB RESET detection if dropping to SLEEP state for USB SUSPEND during device operations.

The total time for oscillator oscillation stabilization time + PLL60 stabilization time (within 250 µs) + PLL480 stabilization time (within 250 µs) must not exceed 5.1 ms.

2.6.19 026h *H_USB_Control (Host USB Control)*

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device / Host	026h	***H_USB_Control***	R/W	7: ***VBUS_Enb***	0: Disable	1: Enable	00h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

This sets the host-related operations.

This register is enabled even in SLEEP and SNOOZE states.

Bit7 ***VBUS_Enb***

Sets the VBUSEN_A terminal (output) state. The default is Low.

Bit6-0 **Reserved**

2.6.20 027h *H_XcvrControl (Host Xcvr Control)*

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device / Host	027h	***H_XcvrControl***	R/W	7: ***TermSelect***	0: HS Termination	1: FS Termination	91h
			R/W	6: ***RemovedRPD***	0: RPD ON	1: RPD OFF	
			R/W	5: ***XcvrSelect[1]***	XcvrSelect[1:0]		
			R/W	4: ***XcvrSelect[0]***			
				3:	0:	1:	
				2:	0:	1:	
			R/W	1: ***OpMode[1]***	OpMode[1:0]		
				0: ***OpMode[0]***			

This sets the host transceiver macro.

This register is enabled even in SLEEP and SNOOZE states.

Bit7 ***TermSelect***

Selects and enables the either HS or FS termination.

Do not set this bit manually. It is set automatically by the hardware when H_NegoControl_0.AutoMode is set.

Bit6 ***RemovedRPD***

This turns on/off the DP_A and DM_A internal pull-down resistors forming the host data line.

0: RPD ON

1: RPD OFF

Normally, leave this bit set at "0" (ON).

Always set at "0" (ON) for host operations (including SUSPEND) in particular. Note that changing the bit from "0" will alter the host data line characteristics and may cause USB malfunctions.

Bit5-4 ***XcvrSelect[1:0]***

Selects and enables the HS, FS, or LS transceiver.

00: High Speed transceiver

01: Full Speed transceiver

10: Reserved

11: Low Speed transceiver

Do not set this bit manually. It is set automatically by the hardware when H_NegoControl_0.AutoMode is set.

Bit3-2 **Reserved**

Bit1-0 ***OpMode***

This sets the HTM operating mode.

Do not set this bit manually. It is set automatically by the hardware when H_NegoControl_0.AutoMode is set.

If host port signal line change status is detected in a state other than ACT_HOST state, see “1.1.3.2 Using Signal Line Change Status” before setting this bit.

OpMode		
00	"Normal Operation"	Normal usage state
01	"Non-Driving"	Non-used state
10	"Disable Bitstuffing and NRZI encoding"	Bitstuffing and NRZI encoding functions disabled in normal use
11	"Power-Down"	Only single-end receiver used

2.6.21 028h *D_USB_Status (Device USB Status)*

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device / Host	028h	***D_USB_Status***	R	7: ***VBUS***	0: VBUS = L	1: VBUS = H	XXh
			R/W	6: FSxHS	0: HS mode	1: FS mode	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
			R	1: ***LineState[1]***	Line State[1:0]		
				0***: LineState[0]***			

This indicates the device status.

Bit7 ***VBUS***

This indicates the VBUS_B terminal status. This bit is enabled even in SLEEP and SNOOZE states.

Bit6 **FSxHS**

This indicates the current operating mode. It is automatically set when "HS Detection Handshake" (see "1.2.7.11.5 HS Detection Handshake") is run using the D_NegoControl.GoChirp bit. Writing this bit allows the operating mode to be overridden and changed, but this bit should be used only when it is necessary to switch the operating mode without using "HS Detection Handshake," such as for simulations.

Set to "FS(1)" when a cable is connected.

This bit can be read from in ACTIVE60, ACT_DEVICE, and ACT_HOST states, and can be written to in ACT_DEVICE state.

Bit5-2 **Reserved**

Bit1-0 ***LineState[1:0]***

This indicates the USB cable signal status. This bit is enabled even in SLEEP and SNOOZE states.

The DP/DM FS receiver received value is indicated if the XcvrSelect bit is "1" (with FS transceiver selected) when the D_XcvrControl register TermSelect bit is "1" (with FS termination selected). The HS receiver received value is indicated if XcvrSelect is "0" (with HS transceiver selected).

The USB activity is indicated when TermSelect is "0."

LineState		
TermSelect	**DP / DM**	**LineState[1:0]**
0	Don't Care	Bus activity
1	SE0	0b00
1	J	0b01
1	K	0b10
1	SE1	0b11

2.6.22 029h *H_USB_Status (Host USB Status)*

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device / Host	029h	***H_USB_Status***	R	7: ***VBUS_State***	0: VBUSFLG_A = High	1: VBUSFLG _A= Low	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
			R	1: ***LineState[1]***	Line State[1:0]		
				0***: LineState[0]***			

This indicates the host-related status.

Bit7 ***VBUS_State***

This indicates the VBUSFLG_A terminal status. This bit is enabled even in SLEEP and SNOOZE states.

Bit6-2 **Reserved**

Bit1-0 ***LineState[1:0]***

This indicates the USB cable signal status. This bit is enabled even in SLEEP and SNOOZE states.

The DP/DM FS receiver received value is indicated if the H_XcvrControl register XcvrSelect[1:0] is “01” (with FS transceiver selected), and the LS receiver received value is indicated if it is “11” (with LS transceiver selected).

The USB activity is indicated when XcvrSelect[1:0] is “00” (with HS transceiver selected).

LineState		
XcvrSelect[1:0]	**DP / DM**	**LineState[1:0]**
00	Don’t Care	Bus activity Activity present: 0b01 No activity: 0b00
01 or 11	SE0	0b00
01 or 11	J	0b01
01 or 11	K	0b10
01 or 11	SE1	0b11

Note: The XcvrSelect[1:0] = “10” code is reserved. The validity of associated operations is not guaranteed.

2.6.23 030h FIFO_Rd_0 (FIFO Read 0)

2.6.24 031h FIFO_Rd_1 (FIFO Read 1)

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device / Host	030h	FIFO_Rd_0	R	7: FIFO_Rd_0[7]	Endpoint n / Channel n / Media FIFO Read	XXh
				6: FIFO_Rd_0[6]		
				5: FIFO_Rd_0[5]		
				4: FIFO_Rd_0[4]		
				3: FIFO_Rd_0[3]		
				2: FIFO_Rd_0[2]		
				1: FIFO_Rd_0[1]		
				0: FIFO_Rd_0[0]		

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device / Host	031h	FIFO_Rd_1	R	7: FIFO_Rd_1[7]	Endpoint n / Channel n / Media FIFO Read	XXh
				6: FIFO_Rd_1[6]		
				5: FIFO_Rd_1[5]		
				4: FIFO_Rd_1[4]		
				3: FIFO_Rd_1[3]		
				2: FIFO_Rd_1[2]		
				1: FIFO_Rd_1[1]		
				0: FIFO_Rd_1[0]		

030h.Bit7-0, 031h.Bit7-0 FIFO_Rd_0[7:0], FIFO_Rd_1[7:0]

This can read the data from the endpoint FIFO setting the AREAx{x=0-5}Join_0.JoinCPU_Rd bit and the media FIFO setting the MediaFIFO_Join.JoinCPU_Rd bit.

If this register is read when there are byte boundaries in the FIFO, valid data is output to one side only. See “1.8.3.1.5 FIFO Access Fractional Number Processing” for detailed information.

To read out FIFO data using this register, be sure to first check the quantity of data that can be read using the FIFO_RdRemain_H,L register.

2.6.25 032h FIFO_Wr_0(FIFO Write 0)

2.6.26 033h FIFO_Wr_1(FIFO Write 1)

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device / Host	032h	FIFO_Wr_0	W	7: FIFO_Wr_0[7]	Endpoint n / Channel n / Media FIFO Write	XXh
				6: FIFO_Wr_0[6]		
				5: FIFO_Wr_0[5]		
				4: FIFO_Wr_0[4]		
				3: FIFO_Wr_0[3]		
				2: FIFO_Wr_0[2]		
				1: FIFO_Wr_0[1]		
				0: FIFO_Wr_0[0]		

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device / Host	033h	FIFO_Wr_1	W	7: FIFO_Wr_1[7]	Endpoint n / Channel n / Media FIFO Write	XXh
				6: FIFO_Wr_1[6]		
				5: FIFO_Wr_1[5]		
				4: FIFO_Wr_1[4]		
				3: FIFO_Wr_1[3]		
				2: FIFO_Wr_1[2]		
				1: FIFO_Wr_1[1]		
				0: FIFO_Wr_1[0]		

32h.Bit7-0, 33h.Bit7-0 FIFO_Wr_0[7:0], FIFO_Wr_1[7:0]

This can write the data to the endpoint FIFO setting the AREAx{x=0-5}Join_0.JoinCPU_Wr bit and the media FIFO setting the MediaFIFO_Join.JoinCPU_Wr bit.

If this register is written to when there are byte boundaries in the FIFO, data is written to one side only. See “1.8.3.1.5 FIFO Access Fractional Number Processing” for detailed information.

To write data to the FIFO using this register, be sure to first check the quantity of data that can be written using the FIFO_WrRemain_H,L register.

2.6.27 034h FIFO_RdRemain_H (FIFO Read Remain High)

2.6.28 035h FIFO_RdRemain_L (FIFO Read Remain Low)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device / Host	034h	FIFO_RdRemain_H	R	7: RdRemainValid	0:None	1: Read Remain Valid	00h
				6:	0:	1:	
				5:	0:	1:	
			R	4: RdRemain[12]	Endpoint n / Channel n / Media FIFO Read Remain High		
				3: RdRemain[11]			
				2: RdRemain[10]			
				1: RdRemain[9]			
				0: RdRemain[8]			

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device / Host	035h	FIFO_RdRemain_L	R	7: RdRemain[7]	Endpoint n / Channel n / Media FIFO Read Remain Low	00h
				6: RdRemain[6]		
				5: RdRemain[5]		
				4: RdRemain[4]		
				3: RdRemain[3]		
				2: RdRemain[2]		
				1: RdRemain[1]		
				0: RdRemain[0]		

34h.Bit7 RdRemainValid

This is set to “1” if an endpoint, channel, or media is joined to the CPU I/F by the AREAx{x=0-5}Join_0.JoinCPU_Rd or MediaFIFO_Join.JoinCPU_Rd bit and the FIFO_RdRemain value is valid. The RdRemain value is invalid if this bit is cleared.

34h.Bit6-5 Reserved

34h.Bit4-0, 35h.Bit7-0 RdRemain[12:0]

This indicates the quantity of readable data within the endpoint or channel FIFO connected to the CPU I/F by the AREAx{x=0-5}Join_0.JoinCPU_Rd or MediaFIFO_Join.JoinCPU_Rd bit. The FIFO_RdRemain_H and FIFO_RdRemain_L registers must be accessed as a pair to obtain the FIFO readable data quantity.

2.6.29 036h FIFO_WrRemain_H (FIFO Write Remain High)

2.6.30 037h FIFO_WrRemain_L (FIFO Write Remain Low)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device / Host	036h	WrRemain_H		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
			R	4: WrRemain[12]	Endpoint n / Channel n / Media FIFO Write Remain High		
				3: WrRemain[11]			
				2: WrRemain[10]			
				1: WrRemain[9]			
				0: WrRemain[8]			

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device / Host	037h	WrRemain_L	R	7: WrRemain[7]	Endpoint n / Channel n / Media FIFO Write Remain Low	00h
				6: WrRemain[6]		
				5: WrRemain[5]		
				4: WrRemain[4]		
				3: WrRemain[3]		
				2: WrRemain[2]		
				1: WrRemain[1]		
				0: WrRemain[0]		

36h.Bit7-5 Reserved

36h.Bit4-0, 37h.Bit7-0 WrRemain[12:0]

This indicates the free space in the endpoint FIFO connected to the CPU I/F by the AREAx{x=0-5}Join_0.JoinCPU_Wr bit or the channel FIFO connected to the CPU I/F by the MediaFIFO_Join.JoinCPU_Wr bit.

However, the FIFO free capacity cannot be checked accurately immediately after writing to the FIFO. Leave at least one CPU cycle before checking the FIFO free space. The FIFO_WrRemain_H and FIFO_WrRemain_L registers must be accessed as a pair to obtain the FIFO free capacity.

2.6.31 038h FIFO_ByteRd(FIFO Byte Read)

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device / Host	038h	FIFO_ByteRd	R	7: FIFO_ByteRd[7]	Endpoint n / Channel n / Media FIFO Byte Read	XXh
				6: FIFO_ByteRd[6]		
				5: FIFO_ByteRd[5]		
				4: FIFO_ByteRd[4]		
				3: FIFO_ByteRd[3]		
				2: FIFO_ByteRd[2]		
				1: FIFO_ByteRd[1]		
				0: FIFO_ByteRd[0]		

Bit7-0 FIFO_ByteRd[7:0]

This can read out the data in byte units from the endpoint FIFO setting the AREAx{x=0-5}Join_0.JoinCPU_Rd bit or the channel FIFO setting the MediaFIFO_Join.JoinCPU_Rd bit. To read out FIFO data using this register, be sure to first check the quantity of data that can be read using the FIFO_RdRemain_H,L register.

2.6.32 03Ah FIFO_ByteWr(FIFO Byte Write)

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device / Host	03Ah	FIFO_ByteWr	W	7: FIFO_ByteWr[7]	Endpoint n / Channel n / Media FIFO Byte Write	XXh
				6: FIFO_ByteWr[6]		
				5: FIFO_ByteWr[5]		
				4: FIFO_ByteWr[4]		
				3: FIFO_ByteWr[3]		
				2: FIFO_ByteWr[2]		
				1: FIFO_ByteWr[1]		
				0: FIFO_ByteWr[0]		

Bit7-0 FIFO_ByteWr[7:0]

This can write the data in byte units to the endpoint FIFO setting the AREAx{x=0-5}Join_0.JoinCPU_Wr bit or the FIFO setting the MediaFIFO_Join.JoinCPU_Wr bit. To write data to the FIFO using this register, be sure to first check the quantity of data that can be written using the FIFO_WrRemain_H,L register.

2.6.33 040h RAM_RdAdrs_H (RAM Read Address High)

2.6.34 041h RAM_RdAdrs_L (RAM Read Address Low)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device / Host	040h	RAM_RdAdrs_H		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
			R/W	4: RAM_RdAdrs[12]	RAM Read Address		
				3: RAM_RdAdrs[11]			
				2: RAM_RdAdrs[10]			
				1: RAM_RdAdrs[9]			
				0: RAM_RdAdrs[8]			

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device / Host	041h	RAM_RdAdrs_L	R/W	7: RAM_RdAdrs[7]	RAM Read Address	00h
				6: RAM_RdAdrs[6]		
				5: RAM_RdAdrs[5]		
				4: RAM_RdAdrs[4]		
				3: RAM_RdAdrs[3]		
				2: RAM_RdAdrs[2]		
				1:		
				0:		

040h.Bit7-5 Reserved

040h.Bit4-0, 041h.Bit7-2 RAM_RdAdrs[12:2]

This sets the initial address for RAM_Rd. After setting this address, set the RAM_RdCount, then set the RAM_RdControl register bit. This initiates RAM_Rd function. The register value varies, depending on the internal operation while the RAM_Rd function is operating. The register value should therefore not be read until the CPU_IntStat.RAM_RdCmp bit is set after the RAM_RdControl register bit has first been set and the RAM_Rd function initiated. Values cannot be guaranteed if this register is read while the RAM_Rd function is operating. Note that writing to this register while the RAM_Rd function is operating will result in malfunctions.

041h.Bit1-0 Reserved

2.6.35 042h RAM_RdControl (RAM Read Control)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device / Host	042h	RAM_RdControl	R/W	7: RAM_GoRdCBW_CSW	0: Do nothing	1: RAM Read CBW_CSW start	00h
			R/W	6: RAM_GoRd	0: Do nothing	1: RAM Read start	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

Bit7 RAM_GoRdCBW_CSW

This bit will initiate the RAM_Rd function to read out data received in the CBW area during device operations (ACT_DEVICE state) or the CSW area during host operations (ACT_HOST state).

Writing "1" to this bit during device operation initiates the RAM_Rd function and reads data from the CBW area. When the values of registers RAM_Rd_00 to RAM_Rd_1E become valid, the CPU_IntStat.RAM_RdCmp bit is set to "1," and the RAM_GoRdCBW_CSW bit is automatically cleared.

Writing "1" to this bit during host operation initiates the RAM_Rd function and reads data from the CSW area. When the values of registers RAM_Rd_00 to RAM_Rd_0C become valid, the CPU_IntStat.RAM_RdCmp bit is set to "1," and the RAM_GoRdCBW_CSW bit is automatically cleared.

In either case, it is not necessary to set the RAM_RdAdrs_H,L or RAM_RdCount register.

The function for this bit takes priority if set concurrently with the RAM_GoRd bit.

Bit6 RAM_GoRd

This bit will initiate the RAM_Rd function.

Setting the RAM_RdCount register and writing "1" to this bit after setting the initial address for RAM_Rd in the RAM_RdAdrs_H,L register initiates the RAM_Rd function. Once the specified count quantity of data has been read from the initial address specified and the RAM_Rd_xx{xx=00-1F} register value is enabled, the CPU_IntStat.RAM_RdCmp bit is set to "1," and the RAM_GoRd bit is automatically cleared.

The function for the RAM_GoRdCBW_CSW bit takes priority if set concurrently with the RAM_GoRdCBW_CSW bit.

Bit5-0 Reserved

2.6.36 043h RAM_RdCount (RAM Read Counter)

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device / Host	043h	RAM_RdCount	R/W	7:	RAM Read Counter	00h
				6:		
				5: RAM_RdCount[5]		
				4: RAM_RdCount[4]		
				3: RAM_RdCount[3]		
				2: RAM_RdCount[2]		
				1:		
				0:		

Bit7-0 RAM_RdCount[5:2]

This sets the quantity of data to be read in the RAM_Rd_xx{xx=00 - 1F} register using the RAM_Rd function. Start the RAM_Rd function by setting the RAM_RdAdrs_H,L register, setting this register, then setting the RAM_RdControl register bit. The register value varies, depending on the internal operation while the RAM_Rd function is operating. The register value should not be read until the CPU_IntStat.RAM_RdCmp bit is set after the RAM_RdControl register bit has first been set and the RAM_Rd function started. Values cannot be guaranteed if this register is read while the RAM_Rd function is operating. Note that writing to this register while the RAM_Rd function is operating will result in malfunctions.

Note that this register can be set to a maximum of 32 bytes. Setting a data quantity exceeding 32 bytes will result in malfunctions.

2.6.37 044h RAM_WrAdrs_H (RAM Write Address High)

2.6.38 045h RAM_WrAdrs_L (RAM Write Address Low)

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device / Host	044h	RAM_WrAdrs_H		7:		00h
				6:		
				5:		
			R/W	4: RAM_WrAdrs[12]	RAM Write Address High	
				3: RAM_WrAdrs[11]		
				2: RAM_WrAdrs[10]		
				1: RAM_WrAdrs[9]		
				0: RAM_WrAdrs[8]		

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device / Host	045h	RAM_WrAdrs_L	R/W	7: RAM_WrAdrs[7]	RAM Write Address Low	00h
				6: RAM_WrAdrs[6]		
				5: RAM_WrAdrs[5]		
				4: RAM_WrAdrs[4]		
				3: RAM_WrAdrs[3]		
				2: RAM_WrAdrs[2]		
				1: RAM_WrAdrs[1]		
				0: RAM_WrAdrs[0]		

This specifies the address for writing to RAM using the RAM_WrDoor_0,1 register.

44h.Bit7-5 Reserved

44h.Bit4-0, 45h.Bit7-0 RAM_WrAdrs[12:0]

This specifies the address for writing to RAM. The address is incremented depending on the number of bytes written to the RAM_WrDoor_0,1 register. Note that RAM_WrAdrs cannot be checked accurately immediately after writing to the RAM_WrDoor_0,1 register, and so a period of at least one CPU cycle should be left before checking RAM_WrAdrs. For detailed information on writing data, see “2.6.39 046h RAM_WrDoor_0 (RAM Write Door 0)” and “2.6.40 047h RAM_WrDoor_1 (RAM Write Door 1).”

2.6.39 046h RAM_WrDoor_0 (RAM Write Door 0)

2.6.40 047h RAM_WrDoor_1 (RAM Write Door 1)

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device / Host	046h	RAM_WrDoor_0	W	7: RAM_WrDoor_0[7]	RAM Write Door 0	XXh
				6: RAM_WrDoor_0[6]		
				5: RAM_WrDoor_0[5]		
				4: RAM_WrDoor_0[4]		
				3: RAM_WrDoor_0[3]		
				2: RAM_WrDoor_0[2]		
				1: RAM_WrDoor_0[1]		
				0: RAM_WrDoor_0[0]		

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device / Host	047h	RAM_WrDoor_1	W	7: RAM_WrDoor_1[7]	RAM Write Door 1	XXh
				6: RAM_WrDoor_1[6]		
				5: RAM_WrDoor_1[5]		
				4: RAM_WrDoor_1[4]		
				3: RAM_WrDoor_1[3]		
				2: RAM_WrDoor_1[2]		
				1: RAM_WrDoor_1[1]		
				0: RAM_WrDoor_1[0]		

046h.Bit7-0, 047h.Bit7-0 RAM_WrDoor_0[7:0], RAM_WrDoor1[7:0]

This is the access register when writing to RAM. It is a write-only register.

Set the initial address for writing RAM data to the RAM_WrAdrs_H,L register before starting to write data. Subsequently writing to this register writes in sequence and increments RAM_WrAdrs_H,L automatically by the number of bytes written.

Data can be written to the descriptor and CSW areas by the RAM_WrDoor_0,1 register in Device mode. Data written to the descriptor area by the RAM_WrDoor_0,1 register can be used repeatedly by the ReplyDescriptor function. In other words, the data cannot be deleted or overwritten by the ReplyDescriptor function. Note, however, that the area in which descriptor data is written can be overwritten if it overlaps an area retained by another endpoint.

In Host mode, data can be written to the CBW area by the RAM_WrDoor_0,1 register.

2.6.41 048h MediaFIFO_Control (Media FIFO Control)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device / Host	048h	MediaFIFO_Control		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
			W	0: MediaFIFO_Clr	0: Do nothing	1: Clear Media FIFO	

This sets the MediaFIFO operation. It is a write-only register.

Bit7-1 Reserved

Bit0 MediaFIFO_Clr

This clears the MediaFIFO.

This bit only clears the FIFO when set to "1" and does not retain the value set.

Do not set this bit to "1" when DMAx{x=0,1} is joined to the MediaFIFO and the corresponding DMA is running (while DMA_Running bit is "1").

2.6.42 049h ClrAllMediaFIFO_Join (Clear All Media FIFO Join)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device / Host	049h	ClrAllMediaFIFO_Join	W	7:ClrJoinIDE	0: Do nothing	1: Clear Join IDE	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
			W	3:ClrJoinDMA1	0: Do nothing	1: Clear Join DMA1	
			W	2:ClrJoinDMA0	0: Do nothing	1: Clear Join DMA0	
			W	1:ClrJoinCPU_Rd	0: Do nothing	1: Clear Join CPU_Rd	
			W	0:ClrJoinCPU_Wr	0: Do nothing	1: Clear Join CPU_Wr	

This clears the connection between the MediaFIFO and the corresponding port. It is a write-only register.

The register bit is automatically cleared to “0” after the connection has been cleared.

Do not set this register bit to “1” while the MediaFIFO is connected to a port (the corresponding MediaFIFO_Join register bit is set to “1”) and each port is running. Doing so will cause malfunctions.

2.6.43 04Ah MediaFIFO_Join (Media FIFO Join)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device / Host	04Ah	MediaFIFO_Join	R/W	7:JoinIDE	0: Do nothing	1: Join MediaFIFO to IDE	00h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
			R/W	3: JoinDMA1	0: Do nothing	1: Join MediaFIFO to DMA1	
			R/W	2: JoinDMA0	0: Do nothing	1: Join MediaFIFO to DMA0	
			R/W	1: JoinCPU_Rd	0: Do nothing	1: Join MediaFIFO to CPU_Rd	
			R/W	0: JoinCPU_Wr	0: Do nothing	1: Join MediaFIFO to CPU_Wr	

This specifies the port for data transfer to and from the MediaFIFO.

Bit7 JoinIDE

This performs DMA1 transfer for MediaFIFO. The transfer direction depends on the IDE_Control.Dir bit setting.

Bit6-4 Reserved

Bit3 JoinDMA1

This performs DMA1 transfer for MediaFIFO. The transfer direction depends on the DMA1_Control.Dir bit setting.

Bit2 JoinDMA0

This performs DMA0 transfer for MediaFIFO. The transfer direction depends on the DMA0_Control.Dir bit setting.

Bit1 JoinCPU_Rd

This performs CPU register access read transfer for MediaFIFO. Data is read from the Media FIFO for FIFO_Rd_0,1 or FIFO_ByteRd register reading.

Bit0 JoinCPU_Wr

This performs CPU register access write transfer for MediaFIFO. Data is written to the Media FIFO for FIFO_Wr_0,1 or FIFO_ByteWr register writing.

If the JoinDMAx {x=0,1} bit is set, the remaining data quantity can be checked by the DMAx{x=0,1}_Remain_H,L register when the DMA0_Control.Dir bit is 1, and free space can be checked when it is 0.

If the JoinCPU_Rd and JoinCPU_Wr bits are set, data can be read from or written to the FIFO_Rd_0,1, FIFO_ByteRd, FIFO_Wr_0,1, and FIFO_ByteWr registers by checking FIFO_RdRemain_H,L and FIFO_WrRemain_H,L.

Only one of the JoinDMA1, JoinDMA0, JoinCPU_Rd, and JoinCPU_Wr bits can be set to “1” at the same time. If multiple bits are set to “1” concurrently, the upper bit is valid.

2.6.44 050h RAM_Rd_00 (RAM Read 00)
2.6.45 051h RAM_Rd_01 (RAM Read 01)
2.6.46 052h RAM_Rd_02 (RAM Read 02)
2.6.47 053h RAM_Rd_03 (RAM Read 03)
2.6.48 054h RAM_Rd_04 (RAM Read 04)
2.6.49 055h RAM_Rd_05 (RAM Read 05)
2.6.50 056h RAM_Rd_06 (RAM Read 06)
2.6.51 057h RAM_Rd_07 (RAM Read 07)
2.6.52 058h RAM_Rd_08 (RAM Read 08)
2.6.53 059h RAM_Rd_09 (RAM Read 09)
2.6.54 05Ah RAM_Rd_0A (RAM Read 0A)
2.6.55 05Bh RAM_Rd_0B (RAM Read 0B)
2.6.56 05Ch RAM_Rd_0C (RAM Read 0C)
2.6.57 05Dh RAM_Rd_0D (RAM Read 0D)
2.6.58 05Eh RAM_Rd_0E (RAM Read 0E)
2.6.59 05Fh RAM_Rd_0F (RAM Read 0F)
2.6.60 060h RAM_Rd_10 (RAM Read 10)
2.6.61 061h RAM_Rd_11 (RAM Read 11)
2.6.62 062h RAM_Rd_12 (RAM Read 12)
2.6.63 063h RAM_Rd_13 (RAM Read 13)
2.6.64 064h RAM_Rd_14 (RAM Read 14)
2.6.65 065h RAM_Rd_15 (RAM Read 15)
2.6.66 066h RAM_Rd_16 (RAM Read 16)
2.6.67 067h RAM_Rd_17 (RAM Read 17)
2.6.68 068h RAM_Rd_18 (RAM Read 18)
2.6.69 069h RAM_Rd_19 (RAM Read 19)
2.6.70 06Ah RAM_Rd_1A (RAM Read 1A)
2.6.71 06Bh RAM_Rd_1B (RAM Read 1B)
2.6.72 06Ch RAM_Rd_1C (RAM Read 1C)
2.6.73 06Dh RAM_Rd_1D (RAM Read 1D)
2.6.74 06Eh RAM_Rd_1E (RAM Read 1E)
2.6.75 06Fh RAM_Rd_1F (RAM Read 1F)

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device / Host	050h -06Fh	RAM_Rd_00 ~ RAM_Rd_1F	R	7: RAM_Rd_xx[7]	RAM Read	00h
				6: RAM_Rd_xx[6]		
				5: RAM_Rd_xx[5]		
				4: RAM_Rd_xx[4]		
				3: RAM_Rd_xx[3]		
				2: RAM_Rd_xx[2]		
				1: RAM_Rd_xx[1]		
				0: RAM_Rd_xx[0]		

50h-6Fh.Bit7-0 RAM_Rd_xx[7:0]

This register stores data read from RAM using the RAM_Rd function. Start the RAM_Rd function by setting RAM_RdAdrs_H,L and RAM_RdCount registers and using the RAM_RdControl register bit. The FIFO_IntStat.RAM_RdCmp bit is set to "1" when this register value becomes valid. If the value set in the RAM_RdCount register is less than 32 bytes, the data read from RAM is stored in sequence from RAM_Rd_00. Register values above the count set in the RAM_RdCount register (e.g. RAM_Rd_10 to RAM_Rd_1F when the count setting is "16") become invalid.

2.6.76 071h DMA0_Config (DMA0 Config)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device / Host	071h	DMA0_Config	R/W	7: FreeRun	0: Count mode	1: FreeRun mode	00h
			R/W	6: DMA_Mode	0: Normal mode	1: Address Decode mode	
				5:	0:	1:	
				4:	0:	1:	
			R/W	3: ActiveDMA	0: DMA0 Inactive	1: DMA0 Active	
				2:	0:	1:	
			R/W	1: ReqAssertCount[1]	Request Assert Count		
				0: ReqAssertCount[0]			

This sets the DMA0 operating mode.

Bit7 FreeRun

Sets the DMA0 operating mode.

0: Count mode

1: Free-run mode

Bit6 DMA_Mode

Sets the DMA0 mode.

0: Operates the DMA using the XDACK from the host as an acknowledgement.

1: Operates the DMA using the access to the DMA0_RdData/DMA0_WrData register from the host as an acknowledgement.

Bit5-4 Reserved

Bit3 ActiveDMA

Enables DMA0 access.

0: DMA0 access disabled

1: DMA0 access enabled

Bit2 Reserved

Bit1-0 ReqAssertCount[1:0]

This bit sets the REQ assert count option to support CPU burst-reading and burst-writing.

Sets the XDREQ0 assert count (number of transfer bytes). XDREQ0 is asserted if the free space for writing or data for reading in the FIFO is at least equal to the assert count set.

Once the DMA transfer ends for the assert count set, XDREQ0 is first negated, then asserted again once the free space or data is confirmed as being at least equal to the assert count.

In other words, transfers for the assert count number set are guaranteed for a single XDREQ0 assert.

However, if count mode is set and the remaining count for DMA0_Count_HH,HL,LH,LL is smaller than the assert count, the DMA0_Count_HH,HL,LH,LL count takes priority and XDREQ0 is asserted if the free space or data in the FIFO is at least equal to the DMA0_Count_HH,HL,LH,LL count.

The table below shows the correlation between DMA0_Count_HH,HL,LH,LL (Count in the table), ReqAssertCount (Req in the table), and FIFO free space/data (Ready in the table), with the XDREQ0 signal and transferable quantity.

The DMA0_Count_HH,HL,LH,LL remaining count must be at least "1."

	Count≥Req		**Count<Req**	
	Ready≥Req	**Ready<Req**	**Ready≥Count**	**Ready<Count**
XDREQ0	Assert	Negate	Assert	Negate
Transferable quantity	Req	-	Req	-

ReqAssertCount[1:0]	**Mode**
	16bit mode
0b00	Normal
0b01	16Byte (8Count)
0b10	32Byte (16Count)
0b11	64Byte (32Count)

The REQ assert count option is not used for the 00 (Normal) setting.

2.6.77 072h DMA0_Control (DMA0 Control)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device / Host	072h	DMA0_Control	R	7: DMA_Running	0: DMA is not running	1: DMA is running	00h
				6:	0:	1:	
				5:	0:	1:	
			W	4: CounterClr	0: Do nothing	1: Clear DMA counter	
			R/W	3: Dir	0: CPU-IF → FIFO RAM	1: CPU-IF ← FIFO RAM	
				2:	0:	1:	
			W	1: DMA_Stop	0: Do nothing	1: Finish DMA	
			W	0: DMA_Go	0: Do nothing	1: Start DMA	

This controls DMA0 and indicates the status.

Bit7 DMA_Running

This bit is set to "1" during DMA0 transfers. The AREAx{x=0-5}Join_0.JoinDMA0 and MediaFIFO_Join.JoinDMA0 bits cannot be changed while this bit is "1."

Bit6-5 Reserved

Bit4 CounterClr

Setting this bit to "1" clears the DMA0_Count_HH,HL,LH,LL registers to 0x00000000. Writes to this bit are ignored if the DMA_Running bit is set to "1."

Bit3 Dir

This sets the DMA0 transfer direction.

0: CPU-IF → FIFO RAM (DMA writing)

1: CPU-IF ← FIFO RAM (DMA reading)

Bit2 Reserved

Bit1 DMA_Stop

Setting this bit to "1" ends the DMA0 transfer. Stopping the DMA0 transfer clears the DMA_Running bit to "0" and sets the CPU_IntStat register DMA0_Cmp bit to "1." To resume the DMA0 transfer, first check that the DMA has ended using the DMA0_Running bit or DMA0_Cmp bit.

Bit0 DMA_Go

Setting this bit to "1" starts the DMA0 transfer.

2.6.78 074h DMA0_Remain_H (DMA0 FIFO Remain High)

2.6.79 075h DMA0_Remain_L (DMA0 FIFO Remain Low)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device / Host	074h	DMA0_Remain_H		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
			R	4: DMA_Remain[12]	DMA FIFO Remain High		
				3: DMA_Remain[11]			
				2: DMA_Remain[10]			
				1: DMA_Remain[9]			
				0: DMA_Remain[8]			

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device / Host	075h	DMA0_Remain_L	R	7: DMA_Remain[7]	DMA FIFO Remain Low	00h
				6: DMA_Remain[6]		
				5: DMA_Remain[5]		
				4: DMA_Remain[4]		
				3: DMA_Remain[3]		
				2: DMA_Remain[2]		
				1: DMA_Remain[1]		
				0: DMA_Remain[0]		

074h.Bit7-5 Reserved

074h.Bit4-0, 075h.Bit7-0 DMA_Remain[12:0]

For reading, this indicates the quantity of data remaining in the endpoint FIFO connected to the DMA by the AREAx{x=0-5}Join_0.JoinDMA0 bit and in the MediaFIFO connected to the DMA by the MediaFIFO_Join.JoinDMA0 bit.

For writing, this indicates the amount of free space in the endpoint FIFO connected to the DMA by the AREAx{x=0-5}Join_0.JoinDMA0 bit and in the MediaFIFO connected to the DMA by the MediaFIFO_Join.JoinDMA0 bit. Note that the correct FIFO free space cannot be checked using this register immediately after a DMA write. Allow at least one CPU cycle to pass before checking FIFO free space.

2.6.80 078h DMA0_Count_HH (DMA0 Transfer Byte Counter High/High)

2.6.81 079h DMA0_Count_HL (DMA0 Transfer Byte Counter High/Low)

2.6.82 07Ah DMA0_Count_LH (DMA0 Transfer Byte Counter Low/High)

2.6.83 07Bh DMA0_Count_LL (DMA0 Transfer Byte Counter Low/Low)

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device / Host	078h	DMA0_Count_HH	R/W	7: DMA_Count[31]	DMA Transfer Byte Counter High-HIgh	00h
				6: DMA_Count[30]		
				5: DMA_Count[29]		
				4: DMA_Count[28]		
				3: DMA_Count[27]		
				2: DMA_Count[26]		
				1: DMA_Count[25]		
				0: DMA_Count[24]		

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device / Host	079h	DMA0_Count_HL	R/W	7: DMA_Count[23]	DMA Transfer Byte Counter High-Low	00h
				6: DMA_Count[22]		
				5: DMA_Count[21]		
				4: DMA_Count[20]		
				3: DMA_Count[19]		
				2: DMA_Count[18]		
				1: DMA_Count[17]		
				0: DMA_Count[16]		

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device / Host	07Ah	DMA0_Count_LH	R/W	7: DMA_Count[15]	DMA Transfer Byte Counter Low-High	00h
				6: DMA_Count[14]		
				5: DMA_Count[13]		
				4: DMA_Count[12]		
				3: DMA_Count[11]		
				2: DMA_Count[10]		
				1: DMA_Count[9]		
				0: DMA_Count[8]		

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device / Host	07Bh	DMA0_Count_LL	R/W	7: DMA_Count[7]	DMA Transfer Byte Counter Low-Low	00h
				6: DMA_Count[6]		
				5: DMA_Count[5]		
				4: DMA_Count[4]		
				3: DMA_Count[3]		
				2: DMA_Count[2]		
				1: DMA_Count[1]		
				0: DMA_Count[0]		

These set the DMA0 transfer data length in bytes when in count mode. Up to 0xFFFF_FFFF bytes can be set. The value set is used to start the countdown. After setting the transfer quantities for these registers, set the DMA0_Control.DMA_Go bit to "1" and start the DMA transfer. The DMA transfer ends when the number of transfer bytes set in these registers has been transferred.

In free-run mode, the value set is used to start the count-up.

The CPU_IntStat register DMA0_CountUp bit is set to "1" if the DMA0_Count_HH,HL,LH,LL register values overflow. The count continues even after the overflow. The DMA transfer quantity can be checked in this mode.

Note that the count cannot be accurately checked using these registers immediately after a DMA write. Allow at least one CPU cycle to pass before checking the count. These registers should be read out using the DMA0_Count_HH,HL,LH,LL sequence.

2.6.84 07Ch DMA0_RdData_0 (DMA0 Read Data 0)

2.6.85 07Dh DMA0_RdData_1 (DMA0 Read Data 1)

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device / Host	07Ch	DMA0_RdData_0	R	7: DMA_RdData_0[7]	DMA Read Data 0	XXh
				6: DMA_RdData_0[6]		
				5: DMA_RdData_0[5]		
				4: DMA_RdData_0[4]		
				3: DMA_RdData_0[3]		
				2: DMA_RdData_0[2]		
				1: DMA_RdData_0[1]		
				0: DMA_RdData_0[0]		

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device / Host	07Dh	DMA0_RdData_1	R	7: DMA_RdData_1[7]	DMA Read Data 1	XXh
				6: DMA_RdData_1[6]		
				5: DMA_RdData_1[5]		
				4: DMA_RdData_1[4]		
				3: DMA_RdData_1[3]		
				2: DMA_RdData_1[2]		
				1: DMA_RdData_1[1]		
				0: DMA_RdData_1[0]		

07Ch.Bit7-0, 07Dh.Bit7-0 DMA_RdData_0[7:0], DMA_RdData_1[7:0]

When the DMA0_Config.DMA_Mode bit is set to "1," the FIFO data can be read for the endpoint, channel, or MediaFIFO connected to the DMA by the AREAx{x=0-5}Join_0.JoinDMA0 or MediaFIFO_Join.JoinDMA0 bit. The DMA0_Control.Dir bit here must be set to DMA read.

2.6.86 07Eh DMA0_WrData_0 (DMA0 Write Data 0)

2.6.87 07Fh DMA0_WrData_1 (DMA0 Write Data 1)

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device / Host	07Eh	DMA0_WrData_0	W	7: DMA_WrData_0[7]	DMA Write Data 0	XXh
				6: DMA_WrData_0[6]		
				5: DMA_WrData_0[5]		
				4: DMA_WrData_0[4]		
				3: DMA_WrData_0[3]		
				2: DMA_WrData_0[2]		
				1: DMA_WrData_0[1]		
				0: DMA_WrData_0[0]		

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device / Host	07Fh	DMA0_WrData_1	W	7: DMA_WrData_1[7]	DMA Write Data 1	XXh
				6: DMA_WrData_1[6]		
				5: DMA_WrData_1[5]		
				4: DMA_WrData_1[4]		
				3: DMA_WrData_1[3]		
				2: DMA_WrData_1[2]		
				1: DMA_WrData_1[1]		
				0: DMA_WrData_1[0]		

07Eh.Bit7-0, 07Fh.Bit7-0 DMA_WrData_0[7:0], DMA_WrData_1[7:0]

When the DMA0_Config.DMA_Mode bit is set to "1," FIFO data can be written to for the endpoint, channel, or MediaFIFO connected to the DMA by the AREAx{x=0-5}Join_0.JoinDMA0 or MediaFIFO_Join.JoinDMA0 bit. The DMA0_Control.Dir bit here must be set to DMA write.

2.6.88 081h DMA1_Config (DMA1 Config)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device / Host	081h	DMA1_Config	R/W	7: FreeRun	0: Count mode	1: FreeRun mode	00h
			R/W	6: DMA_Mode	0: Normal mode	1: Address Decode mode	
				5:	0:	1:	
				4:	0:	1:	
			R/W	3: ActiveDMA	0: DMA1 Inactive	1: DMA1 Active	
				2:	0:	1:	
			R/W	1: ReqAssertCount[1]	Request Assert Count		
				0: ReqAssertCount[0]			

This sets the DMA1 operating mode.

Bit7 FreeRun

Sets the DMA1 operating mode.

0: Count mode

1: Free-run mode

Bit6 DMA_Mode

Sets the DMA0 mode.

0: Operates the DMA using the XDACK from the host as an acknowledgement.

1: Operates the DMA using the access to the DMA0_RdData/DMA0_WrData register from the host as an acknowledgement.

Bit5-4 Reserved

Bit3 ActiveDMA

Enables DMA1 access.

0: DMA1 access disabled

1: DMA1 access enabled

Bit2 Reserved

Bit1-0 ReqAssertCount[1:0]

This bit sets the REQ assert count option to support CPU burst-reading and burst-writing.

Sets the XDREQ1 assert count (number of transfer bytes). XDREQ1 is asserted if the free space for writing or data for reading in the FIFO is at least equal to the assert count set.

Once the DMA transfer ends for the assert count set, XDREQ1 is first negated, then asserted again once the free space or data is confirmed to be at least equal to the assert count.

In other words, transfers for the assert count number set are guaranteed for a single XDREQ1 assert.

However, if count mode is set and the remaining count for DMA1_Count_HH,HL,LH,LL is less than the assert count, the DMA1_Count_HH,HL,LH,LL count takes precedence, and XDREQ1 is asserted if the free space or data in the FIFO is at least equal to the DMA1_Count_HH,HL,LH,LL count.

The table below shows the correlation between DMA1_Count_HH,HL,LH,LL (Count in the table), ReqAssertCount (Req in the table), and FIFO free space/data (Ready in the table), with the XDREQ1 signal and transferable quantity.

The DMA1_Count_HH,HL,LH,LL remaining count must be at least "1."

	Count≥Req		**Count<Req**	
	Ready≥Req	**Ready<Req**	**Ready≥Count**	**Ready<Count**
XDREQ1	Assert	Negate	Assert	Negate
Transferable quantity	Req	-	Req	-

ReqAssertCount[1:0]	**Mode**
	16bit mode
0b00	Normal
0b01	16Byte(8Count)
0b10	32Byte(16Count)
0b11	64Byte(32Count)

The REQ assert count option is not used for the 00 (Normal) setting.

2.6.89 082h DMA1_Control (DMA1 Control)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device / Host	082h	DMA1_Control	R	7: DMA_Running	0: DMA is not running	1: DMA is running	00h
				6:	0:	1:	
				5:	0:	1:	
			W	4: CounterClr	0: Do nothing	1: Clear DMA counter	
			R/W	3: Dir	0: CPU-IF → FIFO RAM	1: CPU-IF ← FIFO RAM	
				2:	0:	1:	
			W	1: DMA_Stop	0: Do nothing	1: Finish DMA	
			W	0: DMA_Go	0: Do nothing	1: Start DMA	

This controls DMA1 and indicates the status.

Bit7 DMA_Running

This bit is set to "1" during DMA1 transfers. The AREAx{x=0-5}Join_0.JoinDMA1 and MediaFIFO_Join.JoinDMA1 bits cannot be changed while this bit is "1."

Bit6-5 Reserved

Bit4 CounterClr

Setting this bit to "1" clears the DMA1_Count_HH,HL,LH,LL registers to 0x00000000. Writes to this bit are ignored if the DMA_Running bit is set to "1."

Bit3 Dir

This sets the DMA1 transfer direction.

0: CPU-IF → FIFO RAM (DMA writing)

1: CPU-IF ← FIFO RAM (DMA reading)

Bit2 Reserved

Bit1 DMA_Stop

Setting this bit to "1" ends the DMA1 transfer. Stopping the DMA1 transfer clears the DMA_Running bit to "0." The CPU_IntStat register DMA1_Cmp bit is also set to "1." To resume the DMA1 transfer, first check that the DMA has ended using the DMA1_Running bit or DMA1_Cmp bit.

Bit0 DMA_Go

Setting this bit to "1" starts the DMA1 transfer.

2.6.90 084h DMA1_Remain_H (DMA1 FIFO Remain High)

2.6.91 085h DMA1_Remain_L (DMA1 FIFO Remain Low)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device / Host	084h	DMA1_Remain_H		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
			R	4: DMA_Remain[12]	DMA FIFO Remain High		
				3: DMA_Remain[11]			
				2: DMA_Remain[10]			
				1: DMA_Remain[9]			
				0: DMA_Remain[8]			

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device / Host	085h	DMA1_Remain_L	R	7: DMA_Remain[7]	DMA FIFO Remain Low	00h
				6: DMA_Remain[6]		
				5: DMA_Remain[5]		
				4: DMA_Remain[4]		
				3: DMA_Remain[3]		
				2: DMA_Remain[2]		
				1: DMA_Remain[1]		
				0: DMA_Remain[0]		

084h.Bit7-5 Reserved

084h.Bit4-0, 085h.Bit7-0 DMA_Remain[12:0]

For reading, this indicates the quantity of data remaining in the endpoint FIFO connected to the DMA by the AREAx{x=0-5}Join_0.JoinDMA1 bit and in the MediaFIFO connected to the DMA by the MediaFIFO_Join.JoinDMA1 bit.

For writing, this indicates the amount of free space in the endpoint FIFO connected to the DMA by the AREAx{x=0-5}Join_0.JoinDMA1 bit and in the MediaFIFO connected to the DMA by the MediaFIFO_Join.JoinDMA1 bit. Note that the correct FIFO free space cannot be checked using this register immediately after a DMA write. Allow at least one CPU cycle to pass before checking FIFO free space.

2.6.92 088h DMA1_Count_HH (DMA1 Transfer Byte Counter High/High)

2.6.93 089h DMA1_Count_HL (DMA1 Transfer Byte Counter High/Low)

2.6.94 08Ah DMA1_Count_LH (DMA1 Transfer Byte Counter Low/High)

2.6.95 08Bh DMA1_Count_LL (DMA1 Transfer Byte Counter Low/Low)

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device / Host	088h	DMA1_Count_HH	R/W	7: DMA_Count[31]	DMA Transfer Byte Counter High-High	00h
				6: DMA_Count[30]		
				5: DMA_Count[29]		
				4: DMA_Count[28]		
				3: DMA_Count[27]		
				2: DMA_Count[26]		
				1: DMA_Count[25]		
				0: DMA_Count[24]		

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device / Host	089h	DMA1_Count_HL	R/W	7: DMA_Count[23]	DMA Transfer Byte Counter High-Low	00h
				6: DMA_Count[22]		
				5: DMA_Count[21]		
				4: DMA_Count[20]		
				3: DMA_Count[19]		
				2: DMA_Count[18]		
				1: DMA_Count[17]		
				0: DMA_Count[16]		

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device / Host	08Ah	DMA1_Count_LH	R/W	7: DMA_Count[15]	DMA Transfer Byte Counter Low-High	00h
				6: DMA_Count[14]		
				5: DMA_Count[13]		
				4: DMA_Count[12]		
				3: DMA_Count[11]		
				2: DMA_Count[10]		
				1: DMA_Count[9]		
				0: DMA_Count[8]		

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device / Host	08Bh	DMA1_Count_LL	R/W	7: DMA_Count[7]	DMA Transfer Byte Counter Low-Low	00h
				6: DMA_Count[6]		
				5: DMA_Count[5]		
				4: DMA_Count[4]		
				3: DMA_Count[3]		
				2: DMA_Count[2]		
				1: DMA_Count[1]		
				0: DMA_Count[0]		

088h-08Bh.Bit7-0 DMA_Count[31:0]

This sets the DMA1 transfer data length in bytes when in count mode. Up to 0xFFFF_FFFF bytes can be set. The value set is used to start the countdown. After setting the transfer quantities for these registers, set the DMA1_Control.DMA_Go bit to "1" and start the DMA transfer. The DMA transfer ends when the number of transfer bytes set in these registers has been transferred.

In free-run mode, the value set is used to start the count-up.

The CPU_IntStat register DMA1_CountUp bit is set to "1" if the DMA1_Count_HH,HL,LH,LL register values overflow. The count continues even after the overflow. The DMA transfer quantity can be checked in this mode.

Note that the count cannot be accurately checked using these registers immediately after a DMA write. Allow at least one CPU cycle to pass before checking the count. These registers should be read out using the DMA1_Count_HH,HL,LH,LL sequence.

2.6.96 08Ch DMA1_RdData_0 (DMA1 Read Data 0)

2.6.97 08Dh DMA1_RdData_1 (DMA1 Read Data 1)

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device / Host	08Ch	DMA1_RdData_0	R	7: DMA_RdData_0[7]	DMA Read Data 0	XXh
				6: DMA_RdData_0[6]		
				5: DMA_RdData_0[5]		
				4: DMA_RdData_0[4]		
				3: DMA_RdData_0[3]		
				2: DMA_RdData_0[2]		
				1: DMA_RdData_0[1]		
				0: DMA_RdData_0[0]		

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device / Host	08Dh	DMA1_RdData_1	R	7: DMA_RdData_1[7]	DMA Read Data 1	XXh
				6: DMA_RdData_1[6]		
				5: DMA_RdData_1[5]		
				4: DMA_RdData_1[4]		
				3: DMA_RdData_1[3]		
				2: DMA_RdData_1[2]		
				1: DMA_RdData_1[1]		
				0: DMA_RdData_1[0]		

08Ch.Bit7-0, 08Dh.Bit7-0 DMA_RdData_0[7:0], DMA_RdData_1[7:0]

When the DMA1_Config.DMA_Mode bit is set to "1," the FIFO data can be read for the endpoint, channel, or MediaFIFO connected to the DMA by the AREAx{x=0-5}Join_0.JoinDMA1 or MediaFIFO_Join.JoinDMA1 bit. The DMA1_Control.Dir bit here must be set to DMA read.

2.6.98 08Eh DMA1_WrData_0 (DMA1 Write Data 0)

2.6.99 08Fh DMA1_WrData_1 (DMA1 Write Data 1)

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device / Host	08Eh	DMA1_WrData_0	W	7: DMA_WrData_0[7]	DMA Write Data 0	XXh
				6: DMA_WrData_0[6]		
				5: DMA_WrData_0[5]		
				4: DMA_WrData_0[4]		
				3: DMA_WrData_0[3]		
				2: DMA_WrData_0[2]		
				1: DMA_WrData_0[1]		
				0: DMA_WrData_0[0]		

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device / Host	08Fh	DMA1_WrData_1	W	7: DMA_WrData_1[7]	DMA Write Data 1	XXh
				6: DMA_WrData_1[6]		
				5: DMA_WrData_1[5]		
				4: DMA_WrData_1[4]		
				3: DMA_WrData_1[3]		
				2: DMA_WrData_1[2]		
				1: DMA_WrData_1[1]		
				0: DMA_WrData_1[0]		

08Eh.Bit7-0, 08Fh.Bit7-0 DMA_WrData_0[7:0], DMA_WrData_1[7:0]

When the DMA1_Config.DMA_Mode bit is set to "1," FIFO data can be written to the endpoint, channel, or MediaFIFO connected to the DMA by the AREAx{x=0-5}Join_0.JoinDMA1 or MediaFIFO_Join.JoinDMA1 bit. The DMA1_Control.Dir bit here must be set to DMA write.

2.6.100 090h IDE_Status (IDE Status)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device / Host	090h	IDE_Status	R	7: DMARQ	0:HDMARQ Not Asserted	1:HDMARQ Asserted	00h
			R	6: DMACK	0:XHDMACK Not Asserted	1:XHDMACK Asserted	
			R	5: INTRQ	0:HINTRQ Not Asserted	1:HINTRQ Asserted	
			R	4: IORDY	0:HIORDY Not Asserted	1:HIORDY Asserted	
				3:	0:	1:	
				2:	0:	1:	
			R	1: PDIAG	0:xHPDIAG Not Asserted	1:xHPDIAG Asserted	
			R	0: DASP	0:xHDASP Not Asserted	1:xHDASP Asserted	

This indicates the IDE bus signal status. “1” can be read when the individual signals are asserted.

Note that the XHDMACK, XHPDIAG, and XHDASP negative logic signals will be read as “1” when the voltage level is “0.”

Bit3-2 is reserved and is normally read as “0.”

2.6.101 091h IDE_Control (IDE Control)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device / Host	091h	IDE_Control		7:	0:	1:	00h
			W	6: IDE_Clr	0: None	1: Clear IDE Circuit	
				5:	0:	1:	
				4:	0:	1:	
			R/W	3: Dir	0: IDE → FIFO RAM	1: IDE ← FIFO RAM	
				2:	0:	1:	
				1:	0:	1:	
			R/W	0: IDE_Go	0: None	1: IDE DMA Go	

This controls the IDE DMA.

Bit7 Reserved

Bit6 IDE_Clr

Setting this bit to "1" resets the IDE circuit to the initial state. The contents of the register set do not change. This bit must not be set during IDE DMA or during the register access sequence.

Bit5-4 Reserved

Bit3 Dir

This sets the IDE transfer direction.

0: IDE → FIFO RAM (IDE reading)

1: IDE ← FIFO RAM (IDE writing)

Bit2-1 Reserved

Bit0 IDE_Go

Setting this bit to "1" launches the IDE DMA. Once the DMA ends, the IDE_IntStat register IDE_Cmp bit assumes the value of "1."

This bit is set to "1" during DMA transfer. It is cleared to "0" when the DMA ends.

Writing "0" while this bit is "1" aborts the DMA transfer in progress, but the IDE_IntStat register IDE_Cmp bit is not set.

2.6.102 092h IDE_Config_0 (IDE Configuration 0)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device / Host	092h	IDE_Config_0	R/W	7: IDE_BusReset	0: None	1: XHRESET Asserted	00h
			R/W	6: IDE_LongBusReset	0: None	1: XHRESET Asserted	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
			R/W	1: Ultra	0: Non Ultra mode	1: Ultra mode	
			R/W	0: DMA	0: Non DMA mode	1: DMA mode	

This controls the IDE DMA.

Bit7 IDE_BusReset

Setting this bit to "1" asserts the IDE XHRESET signal for 50 µs. Setting the bit again to "1" while "1" is already indicated for the bit asserts the IDE XHRESET signal for a further 50 µs. "1" is read from this bit with a XHRESET signal assert from either Bit7 or Bit6.

Bit6 IDE_LongBusReset

Setting this bit to "1" asserts the IDE XHRESET signal for 400 µs. Setting the bit again to "1" while "1" is already indicated for the bit asserts the IDE XHRESET signal for a further 400 µs. "1" is read from this bit with a XHRESET signal assert from either Bit7 or Bit6.

Bit5-2 Reserved

Bit1 Ultra

Setting this bit to "1" simultaneously with Bit0:DMA switches the IDE DMA launched by the IDE_Control register to ultra mode.

This bit cannot be changed during DMA transfers. The table below shows the IDE DMA transfer modes set by this bit.

Bit0 DMA

Setting this bit to "1" switches the IDE DMA launched by the IDE_Control register to multi-word DMA mode.

This bit cannot be changed during DMA transfers. The table below shows the IDE DMA transfer modes set by this bit.

Bit1-0	"00"	"01"	"10"	"11"
	PIO	Multiword DMA	Setting prohibited	Ultra

2.6.103 093h IDE_Config_1 (IDE Configuration 1)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device / Host	093h	IDE_Config_1	R/W	7: ActiveIDE	0: InActivated IDE Bus	1: Activate IDE Bus	04h
			R/W	6: DelayStrobe	0: Not Delay Strobe Signal	1: Delay Strobe Signal	
				5:	0:	1:	
			R/W	4: InterLock	0: None	1: DMA InterLock	
				3:	0:	1:	
			R/W	2: Swap	0: Data Swap	1: None	
				1:	0:	1:	
				0:	0:	1:	

This controls the IE bus state.

Bit7 ActiveIDE

Setting this bit to "1" enables the IDE output signal. The bit must first be set to "1" before issuing register read/write commands to the IDE bus or running IDE-DMA.

When the bit is set to "0," all IDE signals are set to input mode.

Bit6 DelayStrobe

Setting this bit to "1" reserves the setup time for the two system clock period from the XHDMACK assert to XHIOR/XHIOW strobe signal assert (approximately 33 ns) for IDE-DMA multi-word DMA transfer. When the bit is "0," the XHDMACK assert and XHIOR/XHIOW strobe signal assert are simultaneous (approximately 0 ns) for IDE-DMA multi-word DMA transfers.

Bit5 Reserved

Bit4 InterLock

Setting this bit to "1" maintains the IDE bus and waits until the internal data is ready without negating XHDMACK due to the inability to transfer data inside the LSI for IDE-DMA multi-word DMA transfers. When the bit is "0," XHDMACK is temporarily freed when internal data cannot be prepared.

Bit3 Reserved

Bit2 Swap

Clearing this bit to "0" inputs and outputs the IDE bus data with the first 8 bits swapped with the last 8 bits. This bit should normally be set to "1." See "Appendix A IDE_Config_1.Swap Bit Settings" for detailed information on the swap bit.

Bit1-0 Reserved

2.6.104 094h IDE_Rmod (IDE Register Mode)

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device / Host	094h	IDE_Rmod	R/W	7: RegisterAssertPulseWidth[3]	Register Assert Pulse Width	00h
				6: RegisterAssertPulseWidth[2]		
				5: RegisterAssertPulseWidth[1]		
				4: RegisterAssertPulseWidth[0]		
			R/W	3: RegisterNegatePulseWidth[3]	Register Negate Pulse Width	
				2: RegisterNegatePulseWidth[2]		
				1: RegisterNegatePulseWidth[1]		
				0: RegisterNegatePulseWidth[0]		

This sets the XHIOR/XHIOW assert/negate strobe width when accessing the IDE bus in register mode.

The appropriate value must be selected to match the IDE transfer mode.

Bit7-4 RegisterAssertPulseWidth[3:0]

Assumes a value [RegisterAssertPulseWidth + 4] times the system clock (60 Hz) cycle.

E.g. 0000: 4 x 16.67 ns = 67 ns

0001: 5 x 16.67 ns = 83 ns

Bit3-0 RegisterNegatePulseWidth[3:0]

Assumes a value [RegisterNegatePulseWidth + 4] times the system clock (60 Hz) cycle.

E.g. 0000: 4 x 16.67 ns = 67 ns

0001: 5 x 16.67 ns = 83 ns

2.6.105 095h IDE_Tmod (IDE Transfer Mode)

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device / Host	095h	IDE_Tmod	R/W	7: TransferAssertPulseWidth[3]	Transfer Assert Pulse Width	00h
				6: TransferAssertPulseWidth[2]		
				5: TransferAssertPulseWidth[1]		
				4: TransferAssertPulseWidth[0]		
			R/W	3: TransferNegatePulseWidth[3]	Transfer Negate Pulse Width	
				2: TransferNegatePulseWidth[2]		
				1: TransferNegatePulseWidth[1]		
				0: TransferNegatePulseWidth[0]		

This sets the XHIOR/XHIOW assert/negate strobe width when accessing the IDE bus in PIO mode.

The appropriate value must be selected to match the IDE transfer mode.

Bit7-4 TransferAssertPulseWidth[3:0]

Assumes a value [TransferAssertPulseWidth + 4] times the system clock (60 Hz) cycle.

E.g. 0000: 4 x 16.67 ns = 67 ns

0001: 5 x 16.67 ns = 83 ns

Bit3-0 TransferNegatePulseWidth[3:0]

Assumes a value [TransferNegatePulseWidth + 4] times the system clock (60 Hz) cycle.

E.g. 0000: 4 x 16.67 ns = 67 ns

0001: 5 x 16.67 ns = 83 ns

2.6.106 096h IDE_Umod (IDE Ultra-DMA Transfer Mode)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device / Host	096h	IDE_Umod		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
			R/W	3: UltraDMA_Cycle[3]	UltraDMA_Cycle		
				2: UltraDMA_Cycle[2]			
				1: UltraDMA_Cycle[1]			
				0: UltraDMA_Cycle[0]			

This sets the access cycle width when accessing the IDE bus in ultra mode.

The appropriate value must be selected to match the IDE transfer mode.

Bit7-4 Reserved

Bit3-0 UltraDMA_Cycle[3:0]

Assumes a value [UltraDMA_Cycle + 2] times the system clock (60 Hz) cycle.

E.g. 0000: 2 x 16.67 ns = 33 ns

0001: 3 x 16.67 ns = 50 ns

2.6.107 09Ah IDE_CRC_H (IDE CRC High)

2.6.108 09Bh IDE_CRC_L (IDE CRC Low)

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device / Host	09Ah	IDE_CRC_H	R	7: IDE_CRC[15]	IDE_CRC[15:8]	00h
				6: IDE_CRC[14]		
				5: IDE_CRC[13]		
				4: IDE_CRC[12]		
				3: IDE_CRC[11]		
				2: IDE_CRC[10]		
				1: IDE_CRC[9]		
				0: IDE_CRC[8]		

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device / Host	09Bh	IDE_CRC_L	R	7: IDE_CRC[7]	IDE_CRC[7:0]	00h
				6: IDE_CRC[6]		
				5: IDE_CRC[5]		
				4: IDE_CRC[4]		
				3: IDE_CRC[3]		
				2: IDE_CRC[2]		
				1: IDE_CRC[1]		
				0: IDE_CRC[0]		

These indicate the CRC calculation results in sequence for DMA transfers in IDE ultra mode. The IDE_CRC_H and IDE_CRC_L registers must be accessed as a pair for readouts.

2.6.109 09Dh IDE_Count_H (IDE Transfer Byte Counter High)

2.6.110 09Eh IDE_Count_M (IDE Transfer Byte Counter Middle)

2.6.111 09Fh IDE_Count_L (IDE Transfer Byte Counter Low)

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device / Host	09Dh	IDE_Count_H	R/W	7: IDE_Count[23]	IDE_Count[23:16]	00h
				6: IDE_Count[22]		
				5: IDE_Count[21]		
				4: IDE_Count[20]		
				3: IDE_Count[19]		
				2: IDE_Count[18]		
				1: IDE_Count[17]		
				0: IDE_Count[16]		

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device / Host	09Eh	IDE_Count_M	R/W	7: IDE_Count[15]	IDE_Count[15:8]	00h
				6: IDE_Count[14]		
				5: IDE_Count[13]		
				4: IDE_Count[12]		
				3: IDE_Count[11]		
				2: IDE_Count[10]		
				1: IDE_Count[9]		
				0: IDE_Count[8]		

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device / Host	09Fh	IDE_Count_L	R/W	7: IDE_Count[7]	IDE_Count[7:1]		00h
				6: IDE_Count[6]			
				5: IDE_Count[5]			
				4: IDE_Count[4]			
				3: IDE_Count[3]			
				2: IDE_Count[2]			
				1: IDE_Count[1]			
				0:	0:	1:	

These set the number of transfer bytes for IDE DMA transfers. The DMA launch is ignored if launched when this register is set to 0 bytes. The IDE_Count_H, IDE_Count_M, and IDE_Count_L registers must be accessed in pairs for readouts. The IDE_CRC_H register should be accessed first. Note that the last bit of the IDE_Count_L register normally has the value “0.”

2.6.112 0A0h IDE_RegAdrs (IDE Register Address)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device / Host	0A0h	IDE_RegAdrs	R/W	7: IDE_WrReg	0: None	1: IDE Register Write Go	00h
			R/W	6: IDE_RdReg	0: None	1: IDE Register Read Go	
				5:	0:	1:	
				4:	0:	1:	
			R/W	3: IDE_RegAddress[3]	IDE_RegAddress[3:0]		
				2: IDE_RegAddress[2]			
				1: IDE_RegAddress[1]			
				0: IDE_RegAddress[0]			

This controls register access to the IDE bus by the CPU.

Bit7 IDE_WrReg

Setting this bit to "1" writes the IDE register in PIO or register mode to the IDE bus using the contents of the IDE_WrRegValue_0,1 registers set beforehand. This bit has the value "1" during operation and reverts to "0" when the operation is complete and the IDE_IntStat register IDE_RegCmp bit is set. The address to the IDE bus must be set in IDE_RegAddress beforehand or at the same time. The IDE_Rmod and IDE_Tmod registers must be set to the appropriate modes beforehand.

Bit6 IDE_RdReg

Setting this bit to "1" reads the IDE register in PIO or register mode to the IDE bus and sets the value read to the IDE_RdRegValue_0,1 registers. This bit has the value "1" during operation and reverts to "0" when the operation is complete and the IDE_IntStat register IDE_RegCmp bit is set. The address to the IDE bus must be set in IDE_RegAddress beforehand or at the same time. The IDE_Rmod and IDE_Tmod registers must be set to the appropriate modes beforehand.

Bit5-4 Reserved

Bit3-0 IDE_RegAddress[3:0]

This sets the address for register access to the IDE bus using the IDE_WrReg and IDE_RdReg bits. The table below shows the corresponding addresses output to the IDE bus.

IDE_RegAddress[3]	0:XHCS0=0	1:XHCS1=0
IDE_RegAddress[2]	0:HDA2=0	1:HDA2=1
IDE_RegAddress[1]	0:HDA1=0	1:HDA1=1
IDE_RegAddress[0]	0:HDA0=0	1:HDA0=1

2.6.113 0A2h IDE_RdRegValue_0 (IDE Register Read Value 0)

2.6.114 0A3h IDE_RdRegValue_1 (IDE Register Read Value 1)

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device / Host	0A2h	IDE_RdRegValue_0	R	7: IDE_RdRegValue[15]	IDE RdRegValue 0	00h
				6: IDE_RdRegValue[14]		
				5: IDE_RdRegValue[13]		
				4: IDE_RdRegValue[12]		
				3: IDE_RdRegValue[11]		
				2: IDE_RdRegValue[10]		
				1: IDE_RdRegValue[9]		
				0: IDE_RdRegValue[8]		

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device / Host	0A3h	IDE_RdRegValue_1	R	7: IDE_RdRegValue[7]	IDE RdRegValue1	00h
				6: IDE_RdRegValue[6]		
				5: IDE_RdRegValue[5]		
				4: IDE_RdRegValue[4]		
				3: IDE_RdRegValue[3]		
				2: IDE_RdRegValue[2]		
				1: IDE_RdRegValue[1]		
				0: IDE_RdRegValue[0]		

The value read in IDE register reading to the IDE bus by the IDE_RegAdrs register IDE_RdReg bit is set to these registers. The value read in IDE_RegConfig register automatic status register reading is also set to these registers. The IDE_RdRegValue_0 and IDE_RdRegValue_1 registers must be accessed as a pair when reading.

2.6.115 0A4h IDE_WrRegValue_0 (IDE Register Write Value 0)

2.6.116 0A5h IDE_WrRegValue_1 (IDE Register Write Value 1)

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device / Host	0A4h	IDE_WrRegValue_0	R/W	7: IDE_WrRegValue[15]	IDE WrRegValue 0	00h
				6: IDE_WrRegValue[14]		
				5: IDE_WrRegValue[13]		
				4: IDE_WrRegValue[12]		
				3: IDE_WrRegValue[11]		
				2: IDE_WrRegValue[10]		
				1: IDE_WrRegValue[9]		
				0: IDE_WrRegValue[8]		

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device / Host	0A5h	IDE_WrRegValue_1	R/W	7: IDE_WrRegValue[7]	IDE WrRegValue 1	00h
				6: IDE_WrRegValue[6]		
				5: IDE_WrRegValue[5]		
				4: IDE_WrRegValue[4]		
				3: IDE_WrRegValue[3]		
				2: IDE_WrRegValue[2]		
				1: IDE_WrRegValue[1]		
				0: IDE_WrRegValue[0]		

The data to be written is set here beforehand when writing the IDE register to the IDE bus using the IDE_RegAdrs register IDE_WrReg bit.

2.6.117 0A6h IDE_SeqWrRegControl (IDE Sequential Register Write Control)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device / Host	0A6h	IDE_SeqWrRegControl	R/W	7: IDE_SeqWrReg	0:	1: IDE Sequence Write Go	00h
			W	6: IDE_SeqWrRegClr	0:	1: Clear IDE Sequence Write	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

This controls the register sequence writing operation to the IDE bus by the CPU.

Bit7 IDE_SeqWrReg

Setting this bit to "1" writes up to 16 sets of the address data set in IDE_SeqWrRegAdrs and IDE_SeqWrRegValue beforehand to the IDE bus in the sequence set. The IDE_IntStat register IDE_SeqWrRegCmp bit is set to "1" once this ends.

This bit is set to "1" during the sequence operation and reverts to "0" when the operation is complete.

Bit6 IDE_SeqWrRegClr

Setting this bit to "1" discards up to 16 sets of the address data set in IDE_SeqWrRegAdrs and IDE_SeqWrRegValue beforehand and returns to the initial state. This bit must not be set during the sequence operation.

Bit5-0 Reserved

2.6.118 0A7h IDE_SeqWrRegCnt (IDE Sequential Register Write Counter)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device / Host	0A7h	IDE_SeqWrRegCnt		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
			R	4: IDE_SeqWrRegCnt[4]	IDE_SeqWrRegCnt[4:0]		
				3: IDE_SeqWrRegCnt[3]			
				2: IDE_SeqWrRegCnt[2]			
				1: IDE_SeqWrRegCnt[1]			
				0: IDE_SeqWrRegCnt[0]			

This indicates the data quantity written to the IDE_SeqWrRegValue register for up to 10h. The value decreases as sequence writing to the IDE bus proceeds, reverting to “0” when all data written to the IDE_SeqWrRegValue register has been written to the IDE bus. It reverts to “0” even when “1” is written to the IDE_SeqWrRegControl register IDE_SeqWrRegClr bit.

2.6.119 0A8h IDE_SeqWrRegAdrs (IDE Sequential Register Write Address FIFO)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device / Host	0A8h	IDE_SeqWrRegAdrs		7:	0:	1:	XXh
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
			W	3: IDE_SeqRegAddress[3]	IDE_SeqRegAddress[3:0]		
				2: IDE_SeqRegAddress[2]			
				1: IDE_SeqRegAddress[1]			
				0: IDE_SeqRegAddress[0]			

This sets the address output to the IDE bus together with the IDE_SeqWrRegValue register data for sequence write operations to the IDE bus using the IDE_SeqWrRegControl register. If the same address is repeated, there is no need to set it a second time after the address is first set. The relationship between the bit and the address output to the IDE bus is the same as for the IDE_RegAdrs register IDE_RegAddress bit.

2.6.120 0A9h IDE_SeqWrRegValue (IDE Sequential Register Write Value FIFO)

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device / Host	0A9h	IDE_SeqWrRegValue	W	7: IDE_SeqWrRegValue[7]	IDE_SeqWrRegValue[7:0]	XXh
				6: IDE_SeqWrRegValue[6]		
				5: IDE_SeqWrRegValue[5]		
				4: IDE_SeqWrRegValue[4]		
				3: IDE_SeqWrRegValue[3]		
				2: IDE_SeqWrRegValue[2]		
				1: IDE_SeqWrRegValue[1]		
				0: IDE_SeqWrRegValue[0]		

This sets the data output to the IDE bus together with the IDE_SeqWrRegAdrs register address in sequence for sequence write operation to the IDE bus using the IDE_SeqWrRegControl register. Up to 16 sets of data can be set. Operations that write data exceeding this are disregarded. Since 16-bit access is used for the IDE when IDE_SeqWrRegAdrs is “0” (writing to XHCS=0 and HDA=0 data ports), this register must be set twice (in the sequence of lower bytes followed by upper bytes). In this case, two of the 16 sets are used. Since 8-bit access is used for the IDE for all other addresses, this register will be set once each time data is written to the IDE.

2.6.121 0ACh IDE_RegConfig (IDE Register Configuration)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device / Host	0ACh	IDE_RegConfig	R/W	7: EnAutoStsRd	0: None	1: Auto Status Read Enable	00h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

This controls automatic status read operations using the IDE bus HINTRQ interrupt.

Bit7 EnAutoStsRd

Setting this bit to "1" reads the IDE bus status register (XHCS0=0 and HDA=7) automatically when an IDE bus HINTRQ interrupt occurs. The value read to the IDE_RdRegValue register is set when this ends, and the IDE_IntStat register CompleteINTRQ bit is set to "1." This bit remains set even on completion.

Bit6-0 Reserved

2.6.122 0B1h *HostDeviceSel (Host Device Select)*

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device / Host	0B1h	***HostDeviceSel***		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
			R/W	0: ***HOSTxDEVICE***	0: Device mode	1: Host mode	

This sets the register map device mode or host mode. The USB operating mode is set when in 1-port mode. See "1.5.2 1-port Mode (ClkSelect.Port1x2 Set to 1)" for detailed information.

Bit7-1 Reserved

Bit0 *HOSTxDEVICE*

This sets device mode or host mode.

0: Device mode

1: host mode

Changing the settings for this bit will not clear the register settings.

This bit can be accessed even in SLEEP and SNOOZE states.

2.6.123 0B3h *ModeProtect (Mode Protection)*

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device / Host	B3h	***ModeProtect***	R/W	7: ***ModeProtect[7]***	Mode Protection	56h
				6: ***ModeProtect[6]***		
				5***: ModeProtect[5]***		
				4: ***ModeProtect[4]***		
				3: ***ModeProtect[3]***		
				2: ***ModeProtect[2]***		
				1: ***ModeProtect[1]***		
				0: ***ModeProtect[0]***		

Bit7-0 ***ModeProtect[7:0]***

This protects the ChipConfig register and ClkSelect.ClkSelect bit values. Writing 56h to this register enables write access to the ChipConfig register and ClkSelect.ClkSelect bit.

For normal use, the ChipConfig register and ClkSelect.ClkSelect bit should be set as required before setting this register to a value other than 56h (e.g. 00h) to protect the ChipConfig register and ClkSelect.ClkSelect bit settings.

This bit can be accessed even in SLEEP and SNOOZE states.

2.6.124 0B5h *ClkSelect (Clock Select)*

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device / Host	0B5h	***ClkSelect***	R/W	7: ***xActIDE_Term***	0: Termination ON	1: Termination OFF	41h
			R/W	6: ***xActIDE_DD_Term***	0: Termination ON	1: Termination OFF	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
			R/W	1: ***PORT1x2***	0: 2 Port	1: 1 Port	
			R/W	0: ***ClkSelect***	0: 12MHz	1: 24MHz	

Bit7 ***xActIDE_Term***

This turns termination on/off for the HDMARQ, HIORDY, HINTRQ, XHDASP, and HDD7 IDE port terminals.

This bit can be accessed even in SLEEP and SNOOZE states.

0: Termination ON

1: Termination OFF

Bit6 ***xActIDE_DD_Term***

This turns termination on/off for the HDD15 to HDD8 and HDD6 to HDD0 IDE port terminals.

This bit can be accessed even in SLEEP and SNOOZE states.

0: Termination ON

1: Termination OFF

Bit6-2 **Reserved**

Bit1 ***PORT1x2***

This selects the USB port mode used by the LSI.

When set to 2-port mode, the USB host is assigned to port A, and the USB device is assigned to port B. Either function is selected for use.

When set to 1-port mode, only port B can be used, and either function of USB host or USB device is selected for use with port B.

See “Appendix C 1-Port Mode” for detailed information on 1-port mode.

This bit can be accessed even in SLEEP and SNOOZE states.

0: 2-port

1: 1-port

Bit0 ***ClkSelect***

This selects the clock used by the LSI. This bit can be accessed even in SLEEP and SNOOZE states.

0: 12 MHz

1: 24 MHz

2.6.125 0B7h *ChipConfig (Chip Configuration)*

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device / Host	0B7h	***ChipConfig***	R/W	7: ***IntLevel***	0: Low Active	1: High Active	0Xh
			R/W	6: ***IntMode***	0: 1/0 mode	1: Hi-z/0 mode	
			R/W	5: ***DREQ_Level***	0: Low Active	1: High Active	
			R/W	4: ***DACK_Level***	0: Low Active	1: High Active	
			R/W	3: ***CS_Mode***	0: DACK mode	1: CS mode	
			R	2: ***CPU_Endian***	0: Do nothing	1: Bus Swap	
			R	1: ***BusMode***	0: XWRH/L mode	1: XBEH/L mode	
			R	0: ***Initialized***	0: Uninitialized	1: Initialized	

This sets the LSI operating mode.

Bit7 ***IntLevel***

This sets the XINT logic level. This bit can be accessed even in SLEEP and SNOOZE states.

0: Negative logic

1: Positive logic

Bit6 ***IntMode***

This sets the XINT output mode. This bit can be accessed even in SLEEP and SNOOZE states.

0: 1/0 mode

1: Hi-z/0 mode

Bit5 ***DREQ_Level***

This sets the XDREQ0,1 logic level. This bit can be accessed even in SLEEP and SNOOZE states.

0: Negative logic

1: Positive logic

Bit4 ***DACK_Level***

This sets the XDACK0,1 logic level. This bit can be accessed even in SLEEP and SNOOZE states.

0: Negative logic

1: Positive logic

Bit3 ***CS_Mode***

This sets the DMA0,1 operating mode. This bit can be accessed even in SLEEP and SNOOZE states.

0: Operates with DMA access enabled when XDACK0,1 are asserted.

1: Operates with DMA access enabled when XCS and XDACK0,1 are asserted.

Bit2 ***CPU_Endian***

Indicates the CPU bus setting. This bit can be accessed even in SLEEP and SNOOZE states.

0: BigEndian mode. Sets even addresses first and odd addresses last.

1: LittleEndian mode. Sets even addresses last and odd addresses first.

See “Appendix B Connection to Little-endian CPU” for detailed information and register maps for little-endian CPU connections.

Bit1 ***BusMode***

This sets the CPU operating mode. This bit can be accessed even in SLEEP and SNOOZE states.

0: Strobe mode

1: BE mode

Bit0 ***Initialized***

This flag indicates initialization has been performed. The value is normally “1.”

2.6.126 0BDh TimerConfig (Timer Configuration)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device / Host	0BDh	TimerConfig	W	7: TimerStop	0: Low Active	1: High Active	00h
			R	6: TimerRunning	0: 1/0 mode	1: Hi-z/0 mode	
				5:	0:	1:	
			R/W	4: FreeRun	0: Single Mode	1: Free Run Mode	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

This sets the timer.

Bit7 TimerStop

Writing "1" to this bit stops the timer. This bit is cleared once the timer has been stopped.

Bit6 TimerRunning

This indicates the timer operation state.

0: Timer stopped

1: Timer operating

Bit5 Reserved

Bit4 FreeRun

Setting this bit to "1" allows the timer to be used in free-run mode.

0: Single mode. The timer count is stopped once the countdown counter reaches 0x0.

1: Free-run mode. The TimerSet_H,L register values are loaded and the count continues once the countdown counter reaches 0x0.

Bit3-0 Reserved

2.6.127 0BEh TimerSet_H (Timer Set High)

2.6.128 0BFh TimerSet_L (Timer Set Low)

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device / Host	0BEh	TimerSet_H	R/W	TimerSet[15:8]	Timer Set[15:8]	FFh

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device / Host	0BFh	TimerSet_L	R/W	TimerSet[7:0]	Timer Set[7:0]	FFh

These set the countdown value. Writing to this register loads the value to the countdown counter and starts timer operations. The value is not loaded at this point if this register is written to while the timer is running. A new value is loaded and countdown performed once the countdown value reaches 0x0 in free-run mode.

The timer cycle is calculated by following formula.

(60 MHz x 65536) x (Register setting + 1)

2.6.129 0C0h AREAnIntStat (AREAn Interrupt Status)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device / Host	0C0h	AREAnIntStat		7:	0:	1:	00h
				6:	0:	1:	
			R	5: AREA5IntStat	0: None	1: AREA5 Interrupts	
			R	4: AREA4IntStat	0: None	1: AREA4 Interrupts	
			R	3: AREA3IntStat	0: None	1: AREA3 Interrupts	
			R	2: AREA2IntStat	0: None	1: AREA2 Interrupts	
			R	1: AREA1IntStat	0: None	1: AREA1 Interrupts	
			R	0: AREA0IntStat	0: None	1: AREA0 Interrupts	

This indirectly instructs the FIFO area interrupt factors.

Bit7-6 Reserved

Bit5 AREA5IntStat

Indirectly instructs interrupt factors.

Set to “1” when the AREA5IntStat register contains an interrupt factor and the AREA5IntEnb register bit corresponding to the interrupt factor is enabled.

Bit4 AREA4IntStat

Indirectly instructs interrupt factors.

Set to “1” when the AREA4IntStat register contains an interrupt factor and the AREA4IntEnb register bit corresponding to the interrupt factor is enabled.

Bit3 AREA3IntStat

Indirectly instructs interrupt factors.

Set to “1” when the AREA3IntStat register contains an interrupt factor and the AREA3IntEnb register bit corresponding to the interrupt factor is enabled.

Bit2 AREA2IntStat

Indirectly instructs interrupt factors.

Set to “1” when the AREA2IntStat register contains an interrupt factor and the AREA2IntEnb register bit corresponding to the interrupt factor is enabled.

Bit1 AREA1IntStat

Indirectly instructs interrupt factors.

Set to “1” when the AREA1IntStat register contains an interrupt factor and the AREA1IntEnb register bit corresponding to the interrupt factor is enabled.

Bit0 AREA0IntStat

Indirectly instructs interrupt factors.

Set to “1” when the AREA0IntStat register contains an interrupt factor and the AREA0IntEnb register bit corresponding to the interrupt factor is enabled.

2.6.130 0C1h AREA0IntStat (AREA0 Interrupt Status)
2.6.131 0C2h AREA1IntStat (AREA1 Interrupt Status)
2.6.132 0C3h AREA2IntStat (AREA2 Interrupt Status)
2.6.133 0C4h AREA3IntStat (AREA3 Interrupt Status)
2.6.134 0C5h AREA4IntStat (AREA4 Interrupt Status)
2.6.135 0C6h AREA5IntStat (AREA5 Interrupt Status)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device	0C1h	AREA0IntStat		7:	0:	1:	01h
/ Host	0C2h	AREA1IntStat		6:	0:	1:	
	0C3h	AREA2IntStat		5:	0:	1:	
	0C4h	AREA3IntStat		4:	0:	1:	
	0C5h	AREA4IntStat		3:	0:	1:	
	0C6h	AREA5IntStat	R (W)	2: FIFO_NotEmpty	0: None	1: FIFO NotEmpty	
			R (W)	1: FIFO_Full	0: None	1: FIFO Full	
			R (W)	0: FIFO_Empty	0: None	1: FIFO Empty	

Indicates the interrupt status for FIFO area x{x=0-5}.

Writing "1" to all bits clears the interrupt factors.

Bit7-3 Reserved

Bit2 FIFO_NotEmpty

Directly instructs interrupt factors.

Set to "1" when the FIFO area x{x=0-5} contains data (i.e., NotEmpty).

Bit1 FIFO_Full

Directly instructs interrupt factors.

Set to "1" when the FIFO area x{x=0-5} is full.

Bit0 FIFO_Empty

Directly instructs interrupt factors.

Set to "1" when the FIFO area x{x=0-5} is empty.

2.6.136 0C8h AREAnIntEnb (AREAn Interrupt Enable)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device / Host	0C8h	AREAnIntEnb		7:	0:	1:	00h
				6:	0:	1:	
			R/W	5: EnAREA5IntStat	0: Disable	1: Enable	
			R/W	4: EnAREA4IntStat	0: Disable	1: Enable	
			R/W	3: EnAREA3IntStat	0: Disable	1: Enable	
			R/W	2: EnAREA2IntStat	0: Disable	1: Enable	
			R/W	1: EnAREA1IntStat	0: Disable	1: Enable	
			R/W	0: EnAREA0IntStat	0: Disable	1: Enable	

This permits or prohibits assertion of the MainIntStat register AREAnIntStat bit using the AREAnIntStat register interrupt factors.

2.6.137 0C9h AREA0IntEnb (AREA0 Interrupt Enable)
2.6.138 0CAh AREA1IntEnb (AREA1 Interrupt Enable)
2.6.139 0CBh AREA2IntEnb (AREA2 Interrupt Enable)
2.6.140 0CCh AREA3IntEnb (AREA3 Interrupt Enable)
2.6.141 0CDh AREA4IntEnb (AREA4 Interrupt Enable)
2.6.142 0CEh AREA5IntEnb (AREA5 Interrupt Enable)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device	0C9h	AREA0IntEnb		7:	0:	1:	00h
/ Host	0CAh	AREA1IntEnb		6:	0:	1:	
	0CBh	AREA2IntEnb		5:	0:	1:	
	0CCh	AREA3IntEnb		4:	0:	1:	
	0CDh	AREA4IntEnb		3:	0:	1:	
	0CEh	AREA5IntEnb	R/W	2: EnFIFO_NotEmpty	0: Disable	1: Enable	
			R/W	1: EnFIFO_Full	0: Disable	1: Enable	
			R/W	0: EnFIFO_Empty	0: Disable	1: Enable	

These permit or prohibit assertion of the AREAnIntStat register AREAx{x=0-5}IntStat bit using the AREAx{x=0-5}IntStat register interrupt factors.

2.6.143 0D0h AREA0Join_0 (AREA 0 Join 0)

2.6.144 0D2h AREA1Join_0 (AREA 1 Join 0)

2.6.145 0D4h AREA2Join_0 (AREA 2 Join 0)

2.6.146 0D6h AREA3Join_0 (AREA3 Join 0)

2.6.147 0D8h AREA4Join_0 (AREA 4 Join 0)

2.6.148 0DAh AREA5Join_0 (AREA 5 Join 0)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device	0D0h	AREA0Join_0	R/W	7: JoinIDE	0: Do nothing	1: Join to IDE	00h
/ Host	0D2h	AREA1Join_0	R/W	6: JoinFIFO_Stat	0: Do nothing	1: Join to FIFO Status	
	0D4h	AREA2Join_0		5:	0:	1:	
	0D6h	AREA3Join_0		4:	0:	1:	
	0D8h	AREA4Join_0	R/W	3: JoinDMA1	0: Do nothing	1: Join to DMA1	
	0DAh	AREA5Join_0	R/W	2: JoinDMA0	0: Do nothing	1: Join to DMA0	
			R/W	1: JoinCPU_Rd	0: Do nothing	1: Join to CPU Read	
			R/W	0: JoinCPU_Wr	0: Do nothing	1: Join to CPU Write	

These set the port connected to the FIFO area AREAx{x=0-5}.

Bit7 JoinIDE

Performs IDE transfer using the FIFO area AREAx{x=0-5} FIFO. The transfer direction is set by the IDE_Control.Dir bit.

Bit6 JoinFIFO_Stat

Allows the FIFO area AREAx{x=0-5} FIFO full and empty states to be monitored using D_FIFO_IntStat.FIFO_NotEmpty, D_FIFO_IntStat.FIFO_Full and D_FIFO_IntStat.FIFO_Empty or H_FIFO_IntStat.FIFO_NotEmpty, H_FIFO_IntStat.FIFO_Full and H_FIFO_IntStat.FIFO_Empty.

Bit5-4 Reserved

Bit3 JoinDMA1

Performs DMA1 transfer using the FIFO area AREAx{x=0-5} FIFO. The transfer direction is set by the DMA1_Control.Dir bit.

Bit2 JoinDMA0

Performs DMA0 transfer using the FIFO area AREAx{x=0-5} FIFO. The transfer direction is set by the DMA0_Control.Dir bit.

Bit1 JoinCPU_Rd

Performs CPU register access read transfer using the FIFO area AREAx{x=0-5} FIFO. In other words, data is read out from this FIFO area if the FIFO_Rd_0,1 or FIFO_ByteRd register is read.

Bit0 JoinCPU_Wr

Performs CPU register access write transfer using the FIFO area AREAx{x=0-5} FIFO. In other words, data is written to this FIFO area if the FIFO_Wr_0,1 or FIFO_ByteWr register is written to.

Setting the JoinDMAx{x=0,1} bit enables checking of the remaining data quantity via the DMAx{x=0,1}_Remain_H,L register if the DMAx{x=0,1}_Control.Dir bit is 1 or identification of free space if the bit is 0.

If the JoinCPU_Rd, JoinCPU_Wr bit is set, FIFO_RdRemain_H,L and FIFO_WrRemain_H,L can be checked before reading or writing data to or from the FIFO_Rd_0,1, FIFO_ByteRd, FIFO_Wr_0,1 and FIFO_ByteWr registers.

Only one of the JoinDMAx{x=0,1}, JoinCPU_Rd, and JoinCPU_Wr bits should be set to "1" at any given time. Writing "1" to more than one bit simultaneously may destabilize operations.

Exclusive conditions exist for the FIFO area settings. See "1.6.5.6 FIFO Access Restrictions" for detailed information.

2.6.149 0D1h AREA0Join_1 (AREA 0 Join 1)

2.6.150 0D3h AREA1Join_1 (AREA 1 Join 1)

2.6.151 0D5h AREA2Join_1 (AREA 2 Join 1)

2.6.152 0D7h AREA3Join_1 (AREA 3 Join 1)

2.6.153 0D9h AREA4Join_1 (AREA 4 Join 1)

2.6.154 0DBh AREA5Join_1 (AREA 5 Join 1)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device / Host	0D5h	AREA2Join_1		7:	0:	1:	00h
				6:	0:	1:	
			R/W	5: JoinEPeCHe	0: Do nothing	1: Join to EPe / CHe	
			R/W	4: JoinEPdCHd	0: Do nothing	1: Join to EPd / CHd	
			R/W	3: JoinEPcCHc	0: Do nothing	1: Join to EPc / CHc	
			R/W	2: JoinEPbCHb	0: Do nothing	1: Join to EPb / CHb	
			R/W	1: JoinEPaCHa	0: Do nothing	1: Join to EPa / CHa	
			R/W	0: JoinEP0CH0	0: Do nothing	1: Join to EP0 / CH0	

These join endpoints and channels connecting to the FIFO area AREAx{x=0-5}.

Bit7-6 Reserved

Bit5 JoinEPeCHe

This connects endpoint EPe or channel CHe to the FIFO area AREAx{x=0-5}. This connection enables transactions related to data transfers using endpoint EPe or channel EPe.

Bit4 JoinEPdCHd

This connects endpoint EPd or channel CHd to the FIFO area AREAx{x=0-5}. This connection enables transactions related to data transfers using endpoint EPd or channel EPd.

Bit3 JoinEPcCHc

This connects endpoint EPc or channel CHc to the FIFO area AREAx{x=0-5}. This connection enables transactions related to data transfers using endpoint EPc or channel EPc.

Bit2 JoinEPbCHb

This connects endpoint EPb or channel CHb to the FIFO area AREAx{x=0-5}. This connection enables transactions related to data transfers using endpoint EPb or channel EPb.

Bit1 JoinEPaCHa

This connects endpoint EPa or channel CHa to the FIFO area AREAx{x=0-5}. This connection enables transactions related to data transfers using endpoint EPa or channel EPa.

Bit0 JoinEP0CH0

This connects endpoint EP0 or channel CH0 to the FIFO area AREAx{x=0-5}. This connection enables transactions related to data transfers using endpoint EP0 or channel EP0.

Caution is required when setting multiple JoinEPxCHx{x=0,a-e} bits simultaneously to the same FIFO area, as unforeseen operations may result, depending on the transaction order. Normally, we recommend against setting JoinEPxCHx{x=0,a-e} bits to the same FIFO area.

Restrictions exist for FIFO joining. See “1.2.1 Endpoints” and “1.3.1 Channels” for detailed information.

2.6.155 0DEh ClrAREAnJoin_0 (Clear AREA n Join 0)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device / Host	0DEh	ClrAREAnJoin_0	W	7: ClrJoinIDE	0: Do nothing	1: Clear JoinIDE	00h
			W	6: ClrJoinFIFO_Stat	0: Do nothing	1: Clear JoinFIFO_Stat	
				5:	0:	1:	
				4:	0:	1:	
			W	3: ClrJoinDMA1	0: Do nothing	1: Clear JoinDMA1	
			W	2: ClrJoinDMA0	0: Do nothing	1: Clear JoinDMA0	
			W	1: ClrJoinCPU_Rd	0: Do nothing	1: Clear JoinCPU_Rd	
			W	0: ClrJoinCPU_Wr	0: Do nothing	1: Clear JoinCPU_Wr	

This clears the port connection corresponding to each FIFO area. This is a write-only register.

This register bit is automatically cleared to “0” after the connection is cleared.

Do not set this register bit to “1” while the FIFO area is connected to the port (the bit corresponding to the AREAx{x=0-5}Join_0 register is set to “1”) and each port is running. Doing so will result in malfunctions.

2.6.156 0DFh ClrAREAnJoin_1 (Clear AREA 1 Join 1)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device / Host	0DFh	ClrAREAnJoin_1		7:	0:	1:	00h
				6:	0:	1:	
			W	5: ClrJoinEPeCHe	0: Do nothing	1: Clear JoinEPeCHe	
			W	4: ClrJoinEPdCHd	0: Do nothing	1: Clear JoinEPdCHd	
			W	3: ClrJoinEPcCHc	0: Do nothing	1: Clear JoinEPcCHc	
			W	2: ClrJoinEPbCHb	0: Do nothing	1: Clear JoinEPbCHb	
			W	1: ClrJoinEPaCHa	0: Do nothing	1: Clear JoinEPaCHa	
			W	0: ClrJoinEP0CH0	0: Do nothing	1: Clear JoinEP0CH0	

This clears the endpoint and channel connections corresponding to each FIFO area. This is a write-only register.

This register bit is automatically cleared to “0” after the connection is cleared.

Do not set this register bit to “1” while the FIFO area is connected to the endpoint or channel (the bit corresponding to the AREAx{x=0-5}Join_1 register is set to “1”) and each endpoint and channel is running. Doing so will result in malfunctions.

2.6.157 180h AREA0StartAdrs_H (AREA 0 Start Address High)
2.6.158 181h AREA0StartAdrs_L (AREA 0 Start Address Low)
2.6.159 184h AREA1StartAdrs_H (AREA 1 Start Address High)
2.6.160 185h AREA1StartAdrs_L (AREA 1 Start Address Low)
2.6.161 188h AREA2StartAdrs_H (AREA2 Start Address High)
2.6.162 189h AREA2StartAdrs_L (AREA2 Start Address Low)
2.6.163 18Ch AREA3StartAdrs_H (AREA3 Start Address High)
2.6.164 18Dh AREA3StartAdrs_L (AREA3 Start Address Low)
2.6.165 190h AREA4StartAdrs_H (AREA4 Start Address High)
2.6.166 191h AREA4StartAdrs_L (AREA4 Start Address Low)
2.6.167 194h AREA5StartAdrs_H (AREA5 Start Address High)
2.6.168 195h AREA5StartAdrs_L (AREA5 Start Address Low)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device	180h	AREA0StartAdrs_H		7:	0:	1:	00h
/ Host	184h	AREA1StartAdrs_H		6:	0:	1:	
	188h	AREA2StartAdrs_H		5:	0:	1:	
	18Ch	AREA3StartAdrs_H	R/W	4: StartAdrs[12]	AREAx{x=0-5} Start Address High		
	190h	AREA4StartAdrs_H		3: StartAdrs[11]			
	194h	AREA5StartAdrs_H		2: StartAdrs[10]			
				1: StartAdrs[9]			
				0: StartAdrs[8]			

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device	181h	AREA0StartAdrs_L	R/W	7: StartAdrs[7]	AREAx{x=0-5} Start Address Low	00h
/ Host	185h	AREA1StartAdrs_L		6: StartAdrs[6]		
	189h	AREA2StartAdrs_L		5: StartAdrs[5]		
	18Dh	AREA3StartAdrs_L		4: StartAdrs[4]		
	191h	AREA4StartAdrs_L		3: StartAdrs[3]		
	195h	AREA5StartAdrs_L		2: StartAdrs[2]		
				1:		
				0:		

These set the FIFO areas used by AREAx{x=0-5}.

XX0h.Bit7-5 Reserved

XX0h.Bit4-0, XX1h.Bit7-2 StartAdrs[12:2]

This sets the initial address of the FIFO assigned to the FIFO area AREAx{x=0-5}.

The address is set with 12 to 2 upper bits and is specified in 4-byte blocks.

The area assigned to the FIFO area AREAx{x=0-5} extends up to the byte preceding the address set by EndAdrs.

The AREAnFIFO_Clr register ClrAREAx{x=0-5} bit must always be set to “1” and the FIFO area AREAx{x=0-5} FIFO cleared after StartAdrs and EndAdrs are set.

Note that the LSI will not operate correctly if the MaxSize for the USB device/host joined exceeds the area set here. The same applies if the FIFO area AREAx{x=0-5} overlaps another FIFO area.

This LSI has 4.5 kB of internal RAM and supports addresses up to 0x1200.

XX1h.Bit1-0 Reserved

2.6.169 182h AREA0EndAdrs_H (AREA 0 End Address High)
2.6.170 183h AREA0EndAdrs_L (AREA 0 End Address Low)
2.6.171 186h AREA1EndAdrs_H (AREA1 End Address High)
2.6.172 187h AREA1EndAdrs_L (AREA1 End Address Low)
2.6.173 18Ah AREA2EndAdrs_H (AREA2 End Address High)
2.6.174 18Bh AREA2EndAdrs_L (AREA2 End Address Low)
2.6.175 18Eh AREA3EndAdrs_H (AREA3 End Address High)
2.6.176 18Fh AREA3EndAdrs_L (AREA3 End Address Low)
2.6.177 192h AREA4EndAdrs_H (AREA4 End Address High)
2.6.178 193h AREA4EndAdrs_L (AREA4 End Address Low)
2.6.179 196h AREA5EndAdrs_H (AREA5 End Address High)
2.6.180 197h AREA5EndAdrs_L (AREA5 End Address Low)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device	182h	AREA0EndAdrs_H		7:	0:	1:	00h
/ Host	186h	AREA1EndAdrs_H		6:	0:	1:	
	18Ah	AREA2EndAdrs_H		5:	0:	1:	
	18Eh	AREA3EndAdrs_H	R/W	4: EndAdrs[12]	AREAx{x=0-5} End Address High		
	192h	AREA4EndAdrs_H		3: EndAdrs[11]			
	196h	AREA5EndAdrs_H		2: EndAdrs[10]			
				1: EndAdrs[9]			
				0: EndAdrs[8]			

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device	183h	AREA0EndAdrs_L	R/W	7: EndAdrs[7]	AREAx{x=0-5} End Address Low	00h
/ Host	187h	AREA1EndAdrs_L		6: EndAdrs[6]		
	18Bh	AREA2EndAdrs_L		5: EndAdrs[5]		
	18Fh	AREA3EndAdrs_L		4: EndAdrs[4]		
	193h	AREA4EndAdrs_L		3: EndAdrs[3]		
	197h	AREA5EndAdrs_L		2: EndAdrs[2]		
				1:		
				0:		

These set the FIFO areas used by AREAx{x=0-5}.

XX0h.Bit7-5 Reserved

XX0h.Bit4-0, XX1h.Bit7-2 EndAdrs[12:2]

This sets the subsequent byte of the final address of the FIFO assigned to the FIFO area AREAx{x=0-5}.

The address is set with 12 to 2 upper bits and is specified in 4-byte blocks.

The area assigned to the FIFO area AREAx{x=0-5} extends up to the byte preceding the address set by EndAdrs.

The AREAnFIFO_Clr register ClrAREAx{x=0-5} bit must always be set to “1” and the FIFO area AREAx{x=0-5} FIFO cleared after StartAdrs and EndAdrs are set.

Note that the LSI will not operate correctly if the MaxSize for the USB device/host joined exceeds the area set here. The same applies if the FIFO area AREAx{x=0-5} overlaps another FIFO area.

This LSI has 4.5 kB of internal RAM and supports addresses up to 0x1200.

XX1h.Bit1-0 Reserved

2.6.181 19Fh AREAnFIFO_Clr (AREAn FIFO Clear)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device / Host	19Fh	AREAnFIFO_Clr		7:	0:	1:	XXh
				6:	0:	1:	
			W	5: AREA5FIFO_Clr	0: Do nothing	1: Clear AREA5 FIFO	
			W	4: AREA4FIFO_Clr	0: Do nothing	1: Clear AREA4 FIFO	
			W	3: AREA3FIFO_Clr	0: Do nothing	1: Clear AREA3 FIFO	
			W	2: AREA2FIFO_Clr	0: Do nothing	1: Clear AREA2 FIFO	
			W	1: AREA1FIFO_Clr	0: Do nothing	1: Clear AREA1 FIFO	
			W	0: AREA0FIFO_Clr	0: Do nothing	1: Clear AREA0FIFO	

This clears the corresponding FIFO area AREAx{x=0-5} FIFO. This is a write-only register.

Setting each bit of this register to “1” only clears the FIFO; it will not retain the values set.

Do not set the corresponding endpoint bit to “1” when the DMA is joined to the FIFO area AREAx{x=0-5} and the corresponding DMA is running (i.e., while the DMA_Running bit is “1”).

This register only initializes the data retention information; it does not write or clear the data itself. The bit will not clear data in RAM.

2.7 Device Register Details

2.7.1 0E0h *D_SIE_IntStat (Device SIE Interrupt Status)*

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device	0E0h	***D_SIE_IntStat***		7:	0:	1:	00h
			R (W)	6: ***NonJ***	0: None	1: Detect Non J state	
			R (W)	5: RcvSOF	0: None	1: Received SOF	
			R (W)	4: DetectRESET	0: None	1: Detect USB Reset	
			R (W)	3: DetectSUSPEND	0: None	1: Detect USB Suspend	
			R (W)	2: ChirpCmp	0: None	1: Chirp Complete	
			R (W)	1: RestoreCmp	0: None	1: Restore Complete	
			R (W)	0: SetAddressCmp	0: None	1: AutoSetAddress Complete	

This indicates the device SIE-related interrupts.

Writing "1" to all bits clears the interrupt factors.

Bit7 Reserved

Bit6 *NonJ*

Directly instructs interrupt factors.

Set to "1" when a state other than J-state is detected on the USB. This bit is enabled when the LSI is in SNOOZE state (PM_Control register InSNOOZE bit is "1") and when the USB_Control register InSUSPEND bit is set to "1" while the AutoNegotiation function is in use.

Bit5 RcvSOF

Directly instructs interrupt factors.

Set to "1" when an SOF token is received.

Bit4 DetectRESET

Directly instructs interrupt factors.

Set to "1" when USB RESET state is detected. The USB SUSPEND state cannot be detected (DetectSUSPEND cannot be set) while this bit is set.

This reset detection is enabled when the D_NegoControl register ActiveUSB bit is set to "1".

In "HS" operating mode, FS termination is automatically set to detect a USB reset/suspend if there is no bus activity for a preset duration. Detection of SE0 is assumed to be a reset, and this bit is set to "1".

If this bit is set to "1" when the AutoNegotiation function is not in use, set the D_NegoControl register DisBusDetect bit to "1" to disable USB RESET/SUSPEND state detection to prevent incorrect detection of continuous resets. The DisBusDetect bit should be cleared to "0" to enable USB RESET/SUSPEND state detection after reset processing has ended.

"HS Detection Handshake" can be started using the D_NegoControl register GoChirp bit on detection of a reset.

See the section on the D_NegoControl register EnAutoNego bit for detailed information on the AutoNegotiation function (see "2.7.22 102h D_NegoControl (Device Negotiation Control)").

Bit3 DetectSUSPEND

Directly instructs interrupt factors.

Set to "1" if the USB SUSPEND state is detected. The USB RESET state cannot be detected (DetectRESET cannot be set) while this bit is set.

In "HS" operating mode, "FS" operating mode is automatically set to detect a USB reset/suspend if there is no bus activity for a preset duration. Setting the PM_Control_0 register GoSNOOZE bit to "1" after detecting USB SUSPEND state allows the LSI to switch to SNOOZE mode (internal PLL oscillation stopped).

Bit2 ChirpCmp

Directly instructs interrupt factors.

Set to "1" when the "HS Detection Handshake" started by the D_NegoControl register GoChirp bit ends.

The current operating mode (FS or HS) can be determined by reading the D_USB_Status register FSxHS bit after an interrupt has occurred.

Bit1 RestoreCmp

Directly instructs interrupt factors.

Set to "1" when the Restore processing started by the D_NegoControl register RestoreUSB bit ends. Setting this bit to "1" returns the operating mode (FS or HS) to the state preceding Suspend.

Bit0 SetAddressCmp

Directly instructs interrupt factors.

The AutoSetAddress function (see "2.7.36 118h D_USB_Address (Device USB Address)") automatically performs control transfer processing on receipt of a SetAddress() request. The status is set to "1" after the status stage has been performed and the control transfer for the SetAddress() request is complete. The D_USB_Address register address is set simultaneously.

The synchronous bits (bits 5 to 0) can be read in ACTIVE60 and ACT_HOST states, but they cannot be written to (clearing interrupt factors).

The following processing should be performed by the firmware to prevent assertion of the interrupt signal by the interrupt status when switching from the ACT_DEVICE or ACT_ALL state to another state.

<When switching from ACT_DEVICE or ACT_ALL state>

1) Process and clear the interrupt status (D_SIE_IntStat.Bit5 to 0)

2） Disable the interrupt status (D_SIE_IntEnb.Bit5 to 0)

<When switching to ACT_DEVICE or ACT_ALL state>

3） Clear the interrupt status (D_SIE_IntStat.Bit5 to 0)

4） Enable the interrupt status (D_SIE_IntEnb.Bit5 to 0)

2.7.2 0E2h D_FIFO_IntStat (Device FIFO Interrupt Status)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device	0E2h	D_FIFO_IntStat	R (W)	7: DescriptorCmp	0: None	1: Descriptor Complete	00h
			R (W)	6: FIFO_IDE_Cmp	0: None	1: FIFO-IDE Complete	
			R (W)	5: FIFO1_Cmp	0: None	1: FIFO1 Complete	
			R (W)	4: FIFO0_Cmp	0: None	1: FIFO0 Complete	
				3:	0:	1:	
			R (W)	2: FIFO_NotEmpty	0: None	1: FIFO NotEmpty	
			R (W)	1: FIFO_Full	0: None	1: FIFO Full	
			R (W)	0: FIFO_Empty	0: None	1: FIFO Empty	

This indicates the device FIFO interrupt status.

Writing "1" to all bits clears the interrupt factors.

Bit7 DescriptorCmp

Directly instructs interrupt factors.

Set to "1" when the data quantity set in the DescriptorSize register is returned by the Descriptor response function.

Also set to "1" together with the D_EP0IntStat register OUT_TranNAK bit when switching to the status stage (on receipt of OUT token) before sending the quantity set in the DescriptorSize register.

Bit6 FIFO_IDE_Cmp

Directly instructs interrupt factors.

Set to "1" if the FIFO is empty after the IDE transfer ends when the endpoint joined to the IDE is in the IN direction. Set to "1" if the IDE transfer ends when the endpoint joined to the IDE is in the OUT direction.

Bit5 FIFO1_Cmp

Directly instructs interrupt factors.

Set to "1" if the FIFO is empty after the DMA1 transfer ends when the endpoint joined to the DMA1 is in the IN direction. Set to "1" if the DMA1 transfer ends when the endpoint joined to the DMA1 is in the OUT direction.

Bit4 FIFO0_Cmp

Directly instructs interrupt factors.

Set to "1" if the FIFO is empty after the DMA0 transfer ends when the endpoint joined to the DMA0 is in the IN direction. Set to "1" if the DMA0 transfer ends when the endpoint joined to the DMA0 is in the OUT direction.

Bit3 **Reserved**

Bit2 **FIFO_NotEmpty**

Directly instructs interrupt factors.

Set to “1” if the corresponding FIFO area contains data (i.e., NotEmpty) when the AREAx{x=0-5}Join.JoinFIFO_Stat bit is set to “1”.

Bit1 **FIFO_Full**

Directly instructs interrupt factors.

Set to “1” if the corresponding FIFO area is full when the AREAx{x=0-5}Join.JoinFIFO_Stat bit is set to “1”.

Bit0 **FIFO_Empty**

Directly instructs interrupt factors.

Set to “1” if the corresponding FIFO area is empty when the AREAx{x=0-5}Join.JoinFIFO_Stat bit is set to “1”.

2.7.3 0E3h D_BulkIntStat (Device Bulk Interrupt Status)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device	0E3h	D_BulkIntStat	R (W)	7: CBW_Cmp	0: None	1: CBW Complete	00h
			R (W)	6: CBW_LengthErr	0: None	1: CBW Length Error	
			R (W)	5: CBW_Err	0: None	1: CBW Transaction Error	
			R (W)	4:	0:	1:	
			R (W)	3: CSW_Cmp	0: None	1: CSW Complete	
			R (W)	2: CSW_Err	0: None	1: CSW Error	
				1:	0:	1:	
				0:	0:	1:	

This indicates the bulk transfer function interrupt status. Writing "1" to all bits clears the interrupt factors.

Bit7 CBW_Cmp

Directly instructs interrupt factors.

Set to "1" when the 31 CBW bytes are received correctly.

Bit6 CBW_LengthErr

Directly instructs interrupt factors.

Set to "1" when the received CBW packet length is not 31 bytes.

Bit5 CBW_Err

Directly instructs interrupt factors.

Set to "1" when a transaction error or other CRC error is detected in the CBW received.

Bit4 Reserved

Bit3 CSW_Cmp

Directly instructs interrupt factors.

Set to "1" when the 13 CSW bytes are sent correctly.

Bit2 CSW_Err

Directly instructs interrupt factors.

Set to "1" when an error (ACK is not returned) occurs during CSW transmission.

Bit1-0 Reserved

2.7.4 0E4h D_EPrIntStat (Device EPr Interrupt Status)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device	0E4h	D_EPrIntStat		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
			R	4: E_EPeIntStat	0: None	1: EPe Interrupt	
			R	3: D_EPdIntStat	0: None	1: EPd Interrupt	
			R	2: D_EPcIntStat	0: None	1: EPc Interrupt	
			R	1: D_EPbIntStat	0: None	1: EPb Interrupt	
			R	0: D_EPaIntStat	0: None	1: EPa Interrupt	

This indicates the endpoint EPr interrupts.

Bit7-5 Reserved

Bit4 D_EPeIntStat

Indirectly instructs interrupt factors.

Set to "1" when the D_EPeIntStat register contains an interrupt factor and the D_EPeIntEnb register bit corresponding to the interrupt factor is enabled.

Bit3 D_EPdIntStat

Indirectly instructs interrupt factors.

Set to "1" when the D_EPdIntStat register contains an interrupt factor and the D_EPdIntEnb register bit corresponding to the interrupt factor is enabled.

Bit2 D_EPcIntStat

Indirectly instructs interrupt factors.

Set to "1" when the D_EPcIntStat register contains an interrupt factor and the D_EPcIntEnb register bit corresponding to the interrupt factor is enabled.

Bit1 D_EPbIntStat

Indirectly instructs interrupt factors.

Set to "1" when the D_EPbIntStat register contains an interrupt factor and the D_EPbIntEnb register bit corresponding to the interrupt factor is enabled.

Bit0 D_EPaIntStat

Indirectly instructs interrupt factors.

Set to "1" when the D_EPaIntStat register contains an interrupt factor and the D_EPaIntEnb register bit corresponding to the interrupt factor is enabled.

2.7.5 0E5h D_EP0IntStat (Device EP0 Interrupt Status)
2.7.6 0E6h D_EPaIntStat (Device EPa Interrupt Status)
2.7.7 0E7h D_EPbIntStat (Device EPb Interrupt Status)
2.7.8 0E8h D_EPcIntStat (Device EPc Interrupt Status)
2.7.9 0E9h D_EPdIntStat (Device EPd Interrupt Status)
2.7.10 0EAh D_EPeIntStat (Device EPe Interrupt Status)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device	0E5h	D_EP0IntStat		7:	0:	1:	00h
	0E6h	D_EPaIntStat	R (W)	6: OUT_ShortACK	0: None	1: OUT Short-Packet ACK	
	0E7h	D_EPbIntStat	R (W)	5: IN_TranACK	0: None	1: IN Transaction ACK	
	0E8h	D_EPcIntStat	R (W)	4: OUT_TranACK	0: None	1: OUT Transaction ACK	
	0E9h	D_EPdIntStat	R (W)	3: IN_TranNAK	0: None	1: IN Transaction NAK	
	0EAh	D_EPeIntStat	R (W)	2: OUT_TranNAK	0: None	1: OUT Transaction NAK	
			R (W)	1: IN_TranErr	0: None	1: IN Transaction Error	
			R (W)	0: OUT_TranErr	0: None	1: OUT Transaction Error	

This indicates the interrupt status of endpoint EPx{x=0,a-e}. Writing "1" to all the bits clears the interrupt factors.

Bit7 Reserved

Bit6 OUT_ShortACK

Directly instructs interrupt factors.

This is set to "1" concurrently with OUT_TranACK when a short packet is received in an OUT transaction and an ACK is returned.

Bit5 IN_TranACK

Directly instructs interrupt factors.

This is set to "1" when ACK is received in an IN transaction.

Bit4 OUT_TranACK

Directly instructs interrupt factors.

This is set to "1" when ACK is returned in an OUT transaction.

Bit3 IN_TranNAK

Directly instructs interrupt factors.

This is set to "1" when NAK is returned in an IN transaction.

Bit2 OUT_TranNAK

Directly instructs interrupt factors.

This is set to "1" when NAK is returned to an OUT or PING transaction.

Bit1 **IN_TranErr**

Directly instructs interrupt factors.

This is set to “1” if STALL is returned in an IN transaction, if a packet error occurs, or if the handshake times out.

Bit0 **OUT_TranErr**

Directly instructs interrupt factors.

This is set to “1” if STALL is returned in an OUT transaction or if a packet error occurs.

2.7.11 0F0h *D_SIE_IntEnb (Device SIE Interrupt Enable)*

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device	0F0h	***D_SIE_IntEnb***		7:	0:	1:	00h
			R/W	6: ***EnNonJ***	0: Disable	1: Enable	
			R/W	5: EnRcvSOF	0: Disable	1: Enable	
			R/W	4: EnDetectRESET	0: Disable	1: Enable	
			R/W	3: EnDetectSUSPEND	0: Disable	1: Enable	
			R/W	2: EnChirpCmp	0: Disable	1: Enable	
			R/W	1: EnRestoreCmp	0: Disable	1: Enable	
			R/W	0: EnSetAddressCmp	0: Disable	1: Enable	

This permits or prohibits the assertion of the MainIntStat register D_SIE_IntStat bit using the D_SIE_IntStat register interrupt factors.

The EnNonJ bit is enabled even in SLEEP and SNOOZE states.

The synchronous bits (Bit5 to 0) can be read from in ACTIVE60 and ACT_HOST states but cannot be written to. See "2.7.1 0E0h *D_SIE_IntStat (Device SIE Interrupt Status)*" for detailed information on processing for changing from ACT_DEVICE or ACT_ALL state to another state.

2.7.12 0F2h D_FIFO_IntEnb (Device FIFO Interrupt Enable)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device	0F2h	D_FIFO_IntEnb	R/W	7: EnDescriptorCmp	0: Disable	1: Enable	00h
			R/W	6: EnFIFO_IDE_Cmp	0: Disable	1: Enable	
			R/W	5: EnFIFO1_Cmp	0: Disable	1: Enable	
			R/W	4: EnFIFO0_Cmp	0: Disable	1: Enable	
				3:	0:	1:	
			R/W	2: EnFIFO_NotEmpty	0: Disable	1: Enable	
			R/W	1: EnFIFO_Full	0: Disable	1: Enable	
			R/W	0: EnFIFO_Empty	0: Disable	1: Enable	

This permits or prohibits the assertion of the MainIntStat register D_FIFO_IntStat bit using the D_FIFO_IntStat register interrupt factors.

2.7.13 0F3h D_BulkIntEnb (Device Bulk Interrupt Enable)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device	0F3h	D_BulkIntEnb	R/W	7: EnCBW_Cmp	0: Disable	1: Enable	00h
			R/W	6: EnCBW_LengthErr	0: Disable	1: Enable	
			R/W	5: EnCBW_Err	0: Disable	1: Enable	
				4:	0:	1:	
			R/W	3: EnCSW_Cmp	0: Disable	1: Enable	
			R/W	2: EnCSW_Err	0: Disable	1: Enable	
				1:	0:	1:	
				0:	0:	1:	

This permits or prohibits the assertion of the MainIntStat register D_BulkIntStat bit using the D_BulkIntStat register interrupt factors.

2.7.14 0F4h D_EPrIntEnb (Device EPr Interrupt Enable)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device	0F4h	D_EPrIntEnb		7: (Reserved)	0:	1:	00h
				6: (Reserved)	0:	1:	
				5: (Reserved)	0:	1:	
			R/W	4: EnD_EPeIntStat	0: Disable	1: Enable	
			R/W	3: EnD_EPdIntStat	0: Disable	1: Enable	
			R/W	2: EnD_EPcIntStat	0: Disable	1: Enable	
			R/W	1: EnD_EPbIntStat	0: Disable	1: Enable	
			R/W	0: EnD_EPaIntStat	0: Disable	1: Enable	

This permits or prohibits the assertion of the MainIntStat register D_EPrIntStat bit using the D_EPrIntStat register interrupt factors.

2.7.15 0F5h D_EP0IntEnb (Device EP0 Interrupt Enable)

2.7.16 0F6h D_EPaIntEnb (Device EPa Interrupt Enable)

2.7.17 0F7h D_EPbIntEnb (Device EPb Interrupt Enable)

2.7.18 0F8h D_EPcIntEnb (Device EPc Interrupt Enable)

2.7.19 0F9h D_EPdIntEnb (Device EPd Interrupt Enable)

2.7.20 0FAh D_EPeIntEnb (Device EPe Interrupt Enable)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device	0F5h	D_EP0IntEnb		7:	0:	1:	00h
	0F6h	D_EPaIntEnb	R/W	6: EnOUT_ShortACK	0: Disable	1: Enable	
	0F7h	D_EPbIntEnb	R/W	5: EnIN_TranACK	0: Disable	1: Enable	
	0F8h	D_EPcIntEnb	R/W	4: EnOUT_TranACK	0: Disable	1: Enable	
	0F9h	D_EPdIntEnb	R/W	3: EnIN_TranNAK	0: Disable	1: Enable	
	0FAh	D_EPeIntEnb	R/W	2: EnOUT_TranNAK	0: Disable	1: Enable	
			R/W	1: EnIN_TranErr	0: Disable	1: Enable	
			R/W	0: EnOUT_TranErr	0: Disable	1: Enable	

These permit or prohibit the assertion of the MainIntStat register D_EP0IntStat bit using the D_EP0IntStat register interrupt factors.

They permit or prohibit the assertion of the D_EPrIntStat D_EPx{x=a-e}IntStat bit using the D_EPx{x=a-e}IntStat register interrupt factors.

2.7.21 100h *D_Reset (Device Reset)*

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device	100h	***D_Reset***		7:	0:	1:	01h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
			R/W	0: ***ResetDTM***	0: Do nothing	1: Reset DTM	

This resets the device transceiver macro.

It is enabled even in SLEEP and SNOOZE states.

Bit7-1 Reserved

Bit0 *ResetDTM*

Setting this bit to "1" initializes the LSI device transceiver macro.

To cancel the reset, clear this bit to "0."

2.7.22 102h D_NegoControl (Device Negotiation Control)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device	102h	D_NegoControl	R/W	7: DisBusDetect	0: Enable BusDetect	1: Disable BusDetect	00h
			R/W	6: EnAutoNego	0: Disable AutoNegotiation	1: Enable AutoNegotiation	
			R/W	5: InSUSPEND	0: Do nothing	1: Monitor NonJ	
			R/W	4: DisableHS	0: HS mode	1: Disable HS mode	
			R/W	3: SendWakeup	0: Do nothing	1:Send Remotewakeup Signal	
			R/W	2: RestoreUSB	0: Do nothing	1: Restore operation mode	
			R/W	1: GoChirp	0: Do nothing	1: Do Chirp sequence	
			R/W	0: ActiveUSB	0: Disactivate USB	1: Activate USB	

This sets operations related to device negotiation.

Bit7 DisBusDetect

Setting this bit to "1" disables USB RESET/SUSPEND state automatic detection. The USB activity is monitored to detect the USB RESET/SUSPEND state if this bit is cleared to "0."

If no bus activity is detected for more than 3 ms in "HS" mode, the mode automatically switches to "FS" mode. The USB is then determined to be in a RESET or SUSPEND state before setting the corresponding interrupt factor (DetectReset or DetectSuspend).

In "FS" mode, the USB is determined to be in a SUSPEND state if no bus activity is detected for more than 3 ms or is determined to be in a RESET state if "SE0" is detected for at least 2.5 µs. The corresponding interrupt factor is then set.

If the DetectReset or DetectSuspend bit is set to "1," set the DisBusDetect bit to "1" and disable detection while the USB remains in a RESET or SUSPEND state. Do not set this bit to "1" if using the AutoNegotiation function.

Bit6 EnAutoNego

This enables the AutoNegotiation function. The AutoNegotiation function automates the sequence from completion of speed negotiation to determining the speed mode when a RESET is detected.

See "1.2.7 Auto Negotiation Function" for detailed information on the AutoNegotiation function.

Bit5 InSUSPEND

This is automatically set to "1" to enable the NonJ state detection function on detection of the USB SUSPEND state when using the AutoNegotiation function. Clear this bit to "0" to reset from the USB SUSPEND state.

See "1.2.7 Auto Negotiation Function" for detailed information on the AutoNegotiation function.

Bit4 DisableHS

Setting this bit to "1" when GoChirp is set to "1" forces a switch to FS mode without sending a DeviceChirp, then issues a ChirpCmp interrupt.

Bit3 SendWakeup

Setting this bit to “1” outputs a RemoteWakeup signal (K) to the USB port.

Clear this bit to “0” to halt transmission between 1 ms and 15 ms after RemoteWakeup signal transmission begins.

Bit2 RestoreUSB

Setting this bit to “1” when resuming from USB SUSPEND state results in automatic switching to the operating mode (FS or HS) saved before entering USB SUSPEND and sets the corresponding interrupt factor (RestoreCmp).

This bit is automatically cleared to “0” once the operation ends.

Do not clear or set this bit when using the AutoNegotiation function; the bit function is controlled automatically.

Bit1 GoChirp

Setting this bit to “1” when the USB is in RESET state initiates an “HS Detection Handshake” with the host or hub and automatically sets the XcvrControl register TermSelect and XcvrSelect bits and the USB_Status register FSxHS bit. The interrupt factor (ChirpCmp) is set simultaneously when the operation ends.

This bit is automatically cleared to “0” when the operation ends. The “HS Detection Handshake” result can be checked by inspecting the USBStauts register FSxHS bit after the operation has ended.

Do not set or clear this bit when using the AutoNegotiation function; the bit function is controlled automatically.

Bit0 ActiveUSB

The LSI halts all USB device functions, since this bit is cleared to “0” after a hard reset. Setting this bit to “1” after setting the LSI allows operation as a USB device.

2.7.23 105h D_XcvrControl (Device Xcvr Control)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device	105h	D_XcvrControl	R/W	7: TermSelect	0: HS Termination	1: FS Termination	41h
			R/W	6: XcvrSelect	0: HS Transceiver	1: FS Transceiver	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
			R/W	1: OpMode[1]	OpMode[1:0]		
				0: OpMode[0]			

This sets the device transceiver macros.

Bit7 TermSelect

This selects and enables either FS or HS termination. This bit is set automatically if "HS detection handshake" is performed using the USB_Control register GoChirp bit or if the D_NegoControl register EnAutoNego bit is set to perform the AutoNegotiation function.

Bit6 XcvrSelect

This selects and enables either the FS or HS transceiver. This bit is set automatically if "HS detection handshake" is performed using the D_NegoControl register GoChirp bit or if the D_NegoControl register EnAutoNego bit is set to perform the AutoNegotiation function.

Bit5-2 Reserved

Bit1-0 OpMode

This sets the UTM operating mode.

Normally, this does not need to be set unless the USB cable has been disconnected(*) or the USB is in SUSPEND state or test mode.

OpMode		
00	"Normal Operation"	Normal usage state
01	"Non-Driving"	Set to this state when the USB cable has been disconnected.
10	"Disable Bitstuffing and NRZI encoding"	Set to this state when in USB test mode.
11	"Power-Down"	Set to this state when in USB SUSPEND state.

* We recommend setting this register to "41h" when the USB cable has been disconnected.

2.7.24 106h D_USB_Test (Device USB_Test)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device	106h	D_USB_Test	R/W	7: EnHS_Test	0: Do nothing	1: EnHS_Test	00h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
			R/W	3: Test_SE0_NAK	0: Do nothing	1: Test_SE0_NAK	
			R/W	2: Test_J	0: Do nothing	1: Test_J	
			R/W	1: Test_K	0: Do nothing	1: Test_K	
			R/W	0: Test_Packet	0: Do nothing	1: Test_Packett	

This sets the device USB 2.0 test mode operations. Set the bit corresponding to the test mode specified by the SetFeature request, then set the EnHS_Test bit to "1" after the status stage ends to ensure operation using the test mode defined by the USB 2.0 standards.

Bit7 EnHS_Test

Setting this bit to "1" switches to the test mode corresponding to this bit when any one of the last four bits in the D_USB_Test register is set to "1." In test mode, the D_NegoControl register DisBusDetect bit must be set to "1" to prevent USB SUSPEND and RESET detection. Clear the D_NegoControl register EnAutoNego bit to "0" to disable the AutoNegotiation function.

Be careful to avoid switching to test mode until the status stage has ended for the SetFeature request.

Bit6-4 Reserved

Bit3 Test_SE0_NAK

Setting this bit to "1" and setting the EnHS_Test bit to "1" allows use of Test_SE0_NAK test mode.

Bit2 TEST_J

Setting this bit to "1" and setting the EnHS_Test bit to "1" allows use of Test_J test mode. Note that the XcvrControl register TermSelect and XcvrSelect bits should be set based on speed and OpMode should be set to "10" (Disable Bitstuffing and NRZI encoding) before setting the EnHS_Test bit to "1" in this test mode.

Bit1 TEST_K

Setting this bit to "1" and setting the EnHS_Test bit to "1" allows use of Test_K test mode. Note that the XcvrControl register TermSelect and XcvrSelect bits should be set based on speed and OpMode should be set to "10" (Disable Bitstuffing and NRZI encoding) before setting the EnHS_Test bit to "1" in this test mode.

Bit0 Test_Packet

Setting this bit to "1" and setting the EnHS_Test bit to "1" allows use of Test_Packet test mode.

Use the following settings, since this test mode can be used for any endpoint except EP0.

1) Set the endpoint EPx{x=a-e} MaxPacketSize to at least 64, set the transfer direction to IN, and set EndpointNumber to "0xF" and enable. Additionally, assign FIFO of at least 64 bytes for endpoint EPx{x=a-e}.
2) Make sure that the settings for other endpoints do not duplicate the above settings for EPx{x=a-e}. Clear all of the AREAx{x=0-5}Join_1 register bits.
3) Clear the EPx{x=a-e} FIFO, then write the following test packet data to this FIFO.
   Set the EPx{x=a-e}IntStat register IN_TranErr bit to "0."
4) The IN_TranErr status is set to "1" after each test packet transmission is completed.
   The following 53 bytes of data are written to the FIFO in packet transmission test mode.

   00h, 00h, 00h, 00h, 00h, 00h, 00h, 00h,

   00h, AAh, AAh, AAh, AAh, AAh, AAh, AAh,

   AAh, EEh, EEh, EEh, EEh, EEh, EEh, EEh,

   EEh, FEh, FFh, FFh, FFh, FFh, FFh, FFh,

   FFh, FFh, FFh, FFh, FFh, 7Fh, BFh, DFh,

   EFh, F7h, FBh, FDh, FCh, 7Eh, BFh, DFh,

   EFh, F7h, FBh, FDh, 7Eh

   Since PID and CRC are added by SIE when sending test packets, data written to the FIFO consists of data that follows DATA0 PID data to data other than CRC16 of the test packet data described in the USB standard Rev 2.0.

2.7.25 108h D_EPnControl (Device Endpoint Control)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device	108h	D_EPnControl	W	7: AllForceNAK	0: Do nothing	1: Set All ForceNAK	XXh
			W	6: EPrForceSTALL	0: Do nothing	1: Set EP's ForceSTALL	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

This sets the endpoint operation. It is a write-only register.

Bit7 AllForceNAK

This sets the ForceNAK bit to “1” for all endpoints.

Bit6 EPrForceSTALL

This sets the ForceSTALL bit to “1” for all endpoints except endpoint EP0.

Bit5-0 Reserved

2.7.26 10Ah D_BulkOnlyControl (Device BulkOnly Control)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device	10Ah	D_BulkOnlyControl	R/W	7:AutoForceNAK_CBW	0: None	1: AutoForceNAK after CBW	00h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
			R/W	2: GoCBW_Mode	0: None	1: Begin CBW Mode	
			R/W	1: GoCSW_Mode	0: None	1: Begin CSW Mode	
				0:	0:	1:	

This controls the bulk-only support function.

Bit7 AutoForceNAK_CBW

Setting this bit to "1" sets the ForceNAK bit to "1" for the corresponding endpoint once the OUT transaction received by the CBW using CBW support is complete.

Bit6-3 Reserved

Bit2 GoCBW_Mode

Set this bit to "1" to run CBW support for the corresponding endpoint. See the D_BulkOnlyConfig register section ("2.7.26 10Ah D_BulkOnlyControl (Device BulkOnly Control)") for detailed information on the endpoint running CBW support.

Bit1 GoCSW_Mode

Set this bit to "1" to run CSW support for the corresponding endpoint. See the D_BulkOnlyConfig register section ("2.7.26 10Ah D_BulkOnlyControl (Device BulkOnly Control)") for detailed information on the endpoint running CSW support.

Bit0 Reserved

2.7.27 10Bh D_BulkOnlyConfig (Device BulkOnly Configuration)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device	10Bh	D_BulkOnlyConfig		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
			R/W	4: EPeBulkOnly	0: None	1: Enable BulkOnly on EPe	
			R/W	3: EPdBulkOnly	0: None	1: Enable BulkOnly on EPd	
			R/W	2: EPcBulkOnly	0: None	1: Enable BulkOnly on EPc	
			R/W	1: EPbBulkOnly	0: None	1: Enable BulkOnly on EPb	
			R/W	0: EPaBulkOnly	0: None	1: Enable BulkOnly on EPa	

This enables the bulk-only support function.

Bit7-5 Reserved

Bit4 EPeBulkOnly

Set this bit to "1" to enable the bulk-only support function for endpoint EPe. With the bulk-only support function enabled, CBW support is implemented by setting the BulkOnlyControl.GoCBW_Mode bit when endpoint EPe is the OUT endpoint. Similarly, CSW support is implemented by setting the BulkOnlyControl.GoCSW_Mode bit when endpoint EPe is the IN endpoint.

Do not enable the bulk-only support function with more than one OUT endpoint.

Bit3 EPdBulkOnly

Set this bit to "1" to enable the bulk-only support function for endpoint EPd. With the bulk-only support function enabled, CBW support is implemented by setting the BulkOnlyControl.GoCBW_Mode bit when endpoint EPd is the OUT endpoint. Similarly, CSW support is implemented by setting the BulkOnlyControl.GoCSW_Mode bit when endpoint EPd is the IN endpoint.

Do not enable the bulk-only support function with more than one OUT endpoint.

Bit2 EPcBulkOnly

Set this bit to "1" to enable the bulk-only support function for endpoint EPc. With the bulk-only support function enabled, CBW support is implemented by setting the BulkOnlyControl.GoCBW_Mode bit when endpoint EPc is the OUT endpoint. Similarly, CSW support is implemented by setting the BulkOnlyControl.GoCSW_Mode bit when endpoint EPc is the IN endpoint.

Do not enable the bulk-only support function with more than one OUT endpoint.

Bit1 EPbBulkOnly

Set this bit to “1” to enable the bulk-only support function for endpoint EPb. With the bulk-only support function enabled, CBW support is implemented by setting the BulkOnlyControl.GoCBW_Mode bit when endpoint EPb is the OUT endpoint. Similarly, CSW support is implemented by setting the BulkOnlyControl.GoCSW_Mode bit when endpoint EPb is the IN endpoint.

Do not enable the bulk-only support function with more than one OUT endpoint.

Bit0 EPaBulkOnly

Set this bit to “1” to enable the bulk-only support function for endpoint EPa. With the bulk-only support function enabled, CBW support is implemented by setting the BulkOnlyControl.GoCBW_Mode bit when endpoint EPa is the OUT endpoint. Similarly, CSW support is implemented by setting the BulkOnlyControl.GoCSW_Mode bit when endpoint EPa is the IN endpoint.

Do not enable the bulk-only support function with more than one OUT endpoint.

2.7.28 110h D_EP0SETUP_0 (Device EP0 SETUP 0)
2.7.29 111h D_EP0SETUP_1 (Device EP0 SETUP 1)
2.7.30 112h D_EP0SETUP_2 (Device EP0 SETUP 2)
2.7.31 113h D_EP0SETUP_3 (Device EP0 SETUP 3)
2.7.32 114h D_EP0SETUP_4 (Device EP0 SETUP 4)
2.7.33 115h D_EP0SETUP_5 (Device EP0 SETUP 5)
2.7.34 116h D_EP0SETUP_6 (Device EP0 SETUP 6)
2.7.35 117h D_EP0SETUP_7 (Device EP0 SETUP 7)

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device	110h -117h	D_EP0SETUP_0 -D_EP0SETUP_7	R	7: EP0SETUP_n[7]	Endpoint 0 SETUP Data 0 -Endpoint 0 SETUP Data 7	00h
				6: EP0SETUP_n[6]		
				5: EP0SETUP_n[5]		
				4: EP0SETUP_n[4]		
				3: EP0SETUP_n[3]		
				2: EP0SETUP_n[2]		
				1: EP0SETUP_n[1]		
				0: EP0SETUP_n[0]		

The 8 bytes of data received in the endpoint EP0 setup stage are stored in sequence from EP0SETUP_0.

EP0SETUP_0

Set with BmRequestType.

EP0SETUP_1

Set with BRequest.

EP0SETUP_2

Set with the last 8 bits of Wvalue.

EP0SETUP_3

Set with the first 8 bits of Wvalue.

EP0SETUP_4

Set with the last 8 bits of WIndex.

EP0SETUP_5

Set with the first 8 bits of WIndex.

EP0SETUP_6

Set with the last 8 bits of WLength.

EP0SETUP_7

Set with the first 8 bits of WLength.

2.7.36 118h D_USB_Address (Device USB Address)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device	118h	D_USB_Address	R/W	7: SetAddress	0: None	1: Set USB Address	00h
			R (W)	6: USB_Address[6]	USB Address		
				5: USB_Address[5]			
				4: USB_Address[4]			
				3: USB_Address[3]			
				2: USB_Address[2]			
				1: USB_Address[1]			
				0: USB_Address[0]			

The USB address is set by the AutoSetAddress function.

The respective control transfer is performed automatically by the AutoSetAddress function on receipt of a SetAddress() request. The AutoSetAddress function issues SetAddressCmp status after the control transfer status stage for the SetAddress() request is complete and USB_Address has been set.

Bit7 SetAddress

Setting this bit to "1" on receipt of a SetAddress request sets USB_Address automatically when the request status stage is complete. This bit setting is enabled if automatic address setting mode is disabled. See "Appendix D SetAddress Request Response (for TS only)" for detailed information.

Bit6-0 USB_Address

This sets the USB address.

This is written to automatically by the AutoSetAddress function.

Writing is possible, but any data written will be automatically rewritten on receipt of a SetAddress() request.

2.7.37 11Ah D_SETUP_Control(Device SETUP Control)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device	11Ah	D_SETUP_Control		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
			R/W	0: ProtectEP0	0: None	1: Protect EP0	

This sets control transfer.

Bit7-1 Reserved

Bit0 ProtectEP0

This is set to “1” when the control transfer setup stage ends and the data received is stored in registers D_EP0SETUP_0 to D_EP0SETUP_7.

At the same time, the D_EP0ControlIN,D_EP0ControlOUT register ForceSTALL bit is automatically set to “0,” the ForceNAK bit is set to “1,” and the ToggleStat bit is set to “1.”

The ProtectEP0 bit is set when the SETUP transaction runs. Thus, it is also set for a SetAddress() request.

Setting this bit to “1” prevents changes in the EP0 ForceNAK and ForceSTALL bit settings.

2.7.38 11Eh D_FrameNumber_H (Device FrameNumber High)

2.7.39 11Fh D_FrameNumber_L (Device FrameNumber Low)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device	11Eh	D_FrameNumber_H	R	7: FnInvalid	0: Frame number is valid	1: Frame number is not valid	80h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
			R	2: FrameNumber[10]	Frame Number High		
				1: FrameNumber[9]			
				0: FrameNumber[8]			

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device	11Fh	D_FrameNumber_L	R	7: FrameNumber[7]	Frame Number Low	00h
				6: FrameNumber[6]		
				5: FrameNumber[5]		
				4: FrameNumber[4]		
				3: FrameNumber[3]		
				2: FrameNumber[2]		
				1: FrameNumber[1]		
				0: FrameNumber[0]		

This indicates the USB frame number updated each time an SOF token is received. The FrameNumber_H and FrameNumber_L registers must be accessed as a pair when obtaining the frame number.

11Eh.Bit7 FnInvalid

This bit is set to “1” if an error occurs in the SOF packet received.

11Eh.Bit6-3 Reserved

11Eh.Bit2-0, 11Fh.Bit7-0 FrameNumber[10:0]

This indicates FrameNumber for the SOF packet received.

2.7.40 120h D_EP0MaxSize (Device EP0 Max Packet Size)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device	120h	D_EP0MaxSize		7:	0:	1:	40h
			R/W	6: EP0MaxSize[6]	Endpoint[0] Max Packet Size		
				5: EP0MaxSize[5]			
				4: EP0MaxSize[4]			
				3: EP0MaxSize[3]			
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

This sets endpoint EP0.

Bit7 Reserved

Bit6-3 EP0MaxSize[6:3]

This sets MaxPacketSize for endpoint EP0.

This endpoint can be used with any of the sizes shown below selected.

FS: 8, 16, 32, or 64 bytes

HS: 64 bytes

Bit2-0 Reserved

2.7.41 121h D_EP0Control (Device EP0 Control)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device	121h	D_EP0Control	R/W	7: INxOUT	0: OUT	1: IN	00h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
			R/W	0: ReplyDescriptor	0: Do nothing	1: Reply Descriptor	

This sets endpoint EP0.

Bit7 INxOUT

This sets the endpoint EP0 transfer direction.

Identify the request received in the setup stage, then set this bit to that value.

If a data stage is present, set this bit to the transfer direction for the data stage. Since the D_EP0ControlIN and D_EP0ControlOUT register ForceNAK bit is set when the setup stage is complete, this should be cleared when running the data or status stage.

Set this bit once again to suit the status stage direction once the data stage has ended. This bit should be set to "0" if the data stage transfer direction is IN, as the status stage will be in the OUT direction. Similarly, if the data stage transfer direction is OUT or if there is no data stage, the status stage will be in the IN direction, in which case the endpoint EP0 FIFO should be cleared and this bit set to "1."

A NAK response is returned to any IN or OUT transaction with directions differing from this bit setting. However, a STALL response is returned if the D_EP0ControlIN or D_EP0ControlOUT register ForceSTALL bit is set corresponding to the transaction direction.

Bit6-1 Reserved

Bit0 ReplyDescriptor

This runs the descriptor reply function.

Set this bit to "1" to return descriptor data corresponding to the MaxPacketSize size from the FIFO in response to the endpoint EP0 IN transaction. The descriptor data refers to the data corresponding to the size set by the DescSize_H,L registers starting with the addresses set in the DescAdrs_H,L registers. These settings should be set each time the ReplyDescriptor bit is set, since they are updated while the descriptor reply function is being run.

The DescAdrs_H,L registers are incremented by the volume of data sent for each transaction. The DescSize_H,L registers are decremented by the volume of data sent.

When the volume of data set in DescSize_H,L has been sent or if a transaction other than an IN transaction is performed, the descriptor reply function ends, the ReplyDescriptor bit is cleared to "0," and the D_FIFO_IntStat register DescriptorCmp bit and D_EP0IntStat register IN_TranACK bit are set to "1."

See the "1.2.3.4 Descriptor Return Function" operation description for additional details.

2.7.42 122h D_EP0ControlIN (Device EP0 Control IN)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device	122h	D_EP0ControlIN		7:	0:	1:	00h
			R/W	6: EnShortPkt	0: Do nothing	1: Enable short Packet	
				5:	0:	1:	
			R	4: ToggleStat	Toggle sequence bit		
			W	3: ToggleSet	0: Do nothing	1: Set Toggle sequence bit	
			W	2: ToggleClr	0: Do nothing	1: Clear Toggle sequence bit	
			R/W	1: ForceNAK	0: Do nothing	1: Force NAK	
			R/W	0: ForceSTALL	0: Do nothing	1: Force STALL	

This indicates the operation settings and status for endpoint EP0 IN transactions.

Bit7 Reserved

Bit6 EnShortPkt

Setting this bit to "1" allows FIFO data smaller than MaxPacketSize to be sent as a short packet to the EP0 IN transaction. This bit is automatically cleared to "0" upon completion of the IN transaction sending the short packet. This bit is not cleared if a MaxPacketSize packet is sent.

Setting this bit to "1" when there is no data in the FIFO allows a zero-length packet to be sent to the IN token from the host. If data is written to the corresponding FIFO while this bit is set and packets are being sent, that data may also be sent, depending on the precise timing. Do not write data to the FIFO until the packet has been sent and this bit has been cleared.

Bit5 Reserved

Bit4 ToggleStat

This indicates the endpoint EP0 IN transaction toggle sequence bit state.

Bit3 ToggleSet

This sets the endpoint EP0 IN transaction toggle sequence bit to "1." If set concurrently with the ToggleClr bit, the ToggleClr bit function takes precedence.

Bit2 ToggleClr

This clears the endpoint EP0 IN transaction toggle sequence bit to "0." If set concurrently with the ToggleSet bit, this bit function takes precedence.

Bit1 ForceNAK

Set this bit to "1" to return an NAK response to the endpoint EP0 IN transaction regardless of FIFO data quantity.

The MainIntStat register RcvEP0SETUP bit is set to "1" upon completion of the setup stage. This bit is then set to "1" and cannot be cleared to "0" while the RcvEP0SETUP bit is "1." Similarly,

this bit is set to “1” upon completion of the IN transaction sending a short packet.

If the transaction is already underway when this bit is set to “1,” the bit is not set until the transaction ends. It is then set to “1” once the transaction ends. The bit is immediately set to “1” if the transaction is not underway.

Bit0 ForceSTALL

Set this bit to “1” to return a STALL response to the endpoint EP0 IN transaction. This bit takes precedence over the ForceNAK bit setting.

If the DeviceIntStat register RcvEP0SETUP bit is set to “1” upon completion of the setup stage, this bit is cleared to “0” and cannot be set to “1” while the RcvEP0SETUP bit is “1.”

If a transaction is underway, this bit setting will be enabled from the subsequent transaction for a preset timeframe after the start of the transaction.

2.7.43 123h D_EP0ControlOUT (Device EP0 Control OUT)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device	123h	D_EP0ControlOUT	R/W	7: AutoForceNAK	0: Do nothing	1: Auto Force NAK	00h
				6:	0:	1:	
				5:	0:	1	
			R	4: ToggleStat	Toggle sequence bit		
			W	3: ToggleSet	0: Do nothing	1: Set Toggle sequence bit	
			W	2: ToggleClr	0: Do nothing	1: Clear Toggle sequence bit	
			R/W	1: ForceNAK	0: Do nothing	1: Force NAK	
			R/W	0: ForceSTALL	0: Do nothing	1: Force STALL	

This indicates the operation settings and status for endpoint EP0 OUT transactions.

Bit7 AutoForceNAK

This sets the register ForceNAK bit to "1" once the endpoint EP0 OUT transaction ends normally.

Bit6-5 Reserved

Bit4 ToggleStat

This indicates the endpoint EP0 OUT transaction toggle sequence bit state.

Bit3 ToggleSet

This sets the endpoint EP0 OUT transaction toggle sequence bit to "1." If set concurrently with the ToggleClr bit, the ToggleClr bit function takes precedence.

Bit2 ToggleClr

This clears the endpoint EP0 OUT transaction toggle sequence bit to "0." If set concurrently with the ToggleSet bit, this bit function takes precedence.

Bit1 ForceNAK

Set this bit to "1" to return an NAK response to the endpoint EP0 OUT transaction regardless of FIFO free space.

The DeviceIntStat register RcvEP0SETUP bit is set to "1" upon completion of the setup stage. This bit is then set to "1" and cannot be cleared to "0" while the RcvEP0SETUP bit is "1."

If the transaction is already underway when this bit is set to "1," the bit is not set until the transaction ends. It is then set to "1" once the transaction ends. The bit is immediately set to "1" if the transaction is not underway.

Bit0 ForceSTALL

Set this bit to “1” to return a STALL response to the endpoint EP0 OUT transaction. This bit takes precedence over the ForceNAK bit setting.

If the DeviceIntStat register RcvEP0SETUP bit is set to “1” upon completion of the setup stage, this bit is cleared to “0” and cannot be set to “1” while the RcvEP0SETUP bit is “1.”

If a transaction is underway, this bit setting will be enabled from the subsequent transaction for a preset timeframe after the start of the transaction.

2.7.44 130h D_EPaMaxSize_H (Device EPa Max Packet Size High)
2.7.45 131h D_EPaMaxSize_L (Device EPa Max Packet Size Low)
2.7.46 140h D_EPbMaxSize_H (Device EPb Max Packet Size High)
2.7.47 141h D_EPbMaxSize_L (Device EPb Max Packet Size Low)
2.7.48 150h D_EPcMaxSize_H (Device EPc Max Packet Size High)
2.7.49 151h D_EPcMaxSize_L (Device EPc Max Packet Size Low)
2.7.50 1A0h D_EPdMaxSize_H (Device EPd Max Packet Size High)
2.7.51 1A1h D_EPdMaxSize_L (Device EPd Max Packet Size Low)
2.7.52 1B0h D_EPeMaxSize_H (Device EPe Max Packet Size High)
2.7.53 1B1h D_EPeMaxSize_L (Device EPe Max Packet Size Low)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device	130h	D_EPaMaxSize_H		7:	0:	1:	00h
	140h	D_EPbMaxSize_H		6:	0:	1:	
	150h	D_EPcMaxSize_H		5:	0:	1:	
	1A0h	D_EPdMaxSize_H		4:	0:	1:	
	1B0h	D_EPeMaxSize_H		3:	0:	1:	
			R/W	2: MaxSize[10]	Max Packet Size		
				1: MaxSize[9]			
				0: MaxSize[8]			

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device	131h	D_EPaMaxSize_L	R/W	7: MaxSize[7]	Max Packet Size	00h
	141h	D_EPbMaxSize_L		6: MaxSize[6]		
	151h	D_EPcMaxSize_L		5: MaxSize[5]		
	1A1h	D_EPdMaxSize_L		4: MaxSize[4]		
	1B1h	D_EPeMaxSize_L		3: MaxSize[3]		
				2: MaxSize[2]		
				1: MaxSize[1]		
				0: MaxSize[0]		

This sets MaxPacketSize.

XX0h.Bit7-3 Reserved

XX0h.Bit2-0, XX1h.Bit7-0 MaxSize[10:0]

This sets the endpoint EPx{x=a-e} MaxPacketSize.

Set to one of the following when this endpoint is used for bulk transfer.

FS: 8, 16, 32, 64 bytes

HS: 512 bytes

Transfer rate can be set as shown below when this endpoint is used for interrupt transfer.

FS: Up to 64 bytes

HS: Up to 1,024 bytes

Transfer rate can be set as shown below when this endpoint is used for isochronous transfer.

FS: Up to 1,023 bytes

HS: Up to 1,024 bytes

XX1h.Bit2-0 Reserved

2.7.54 132h D_EPaConfig_0 (Device EPa Configuration 0)

2.7.55 142h D_EPbConfig_0 (Devie EPb Configuration 0)

2.7.56 152h D_EPcConfig_0 (Device EPc Configuration 0)

2.7.57 1A2h D_EPdConfig_0 (Device EPd Configuration 0)

2.7.58 1B2h D_EPeConfig_0 (Device EPe Configuration 0)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device	132h	D_EPaConfig_0	R/W	7: INxOUT	0: OUT	1: IN	00h
	142h	D_EPbConfig_0	R/W	6: IntEP_Mode	0: Normal	1: Special	
	152h	D_EPcConfig_0		5:	0:	1:	
	1A2h	D_EPdConfig_0	R/W	4: ISO	0: Other than Isochronous	1: Isochronous	
	1B2h	D_EPeConfig_0	R/W	3: EndpointNumber[3]	Endpoint Number		
				2: EndpointNumber[2]			
				1: EndpointNumber[1]			
				0: EndpointNumber[0]			

This sets endpoint EPx{x=a-e}.

Set to ensure that the EndpointNumber and INxOUT combination does not duplicate other endpoints.

Bit7 INxOUT

This sets the endpoint transfer direction.

Bit6 IntEP_Mode

This sets interrupt transfer.

Do not set this bit to “1” for bulk endpoints.

This bit setting will vary depending on the endpoint direction (IN or OUT) (the endpoint direction is set by Bit7 “INxOUT”).

The toggle sequence bit operating mode is set for the IN direction (INxOUT = 1). The toggle sequence operating mode depends on the application. Select the operating mode for the Interrupt IN endpoint.

0: Normal toggle: Performs the normal toggle sequence.

1: Always toggle: Toggles normally for each transaction. See section 5.7.5 of the USB 2.0 standards manual for detailed information on this mode.

Whether or not PING flow control is used for this endpoint is set for the OUT direction (INxOUT = 0). Set this bit to “1” for the Interrupt OUT endpoint.

0: Bulk OUT: Set for Bulk OUT endpoint.

1: Interrupt OUT: Set for Interrupt OUT endpoint.

Bit5 Reserved

Bit4 ISO

Set this bit to “1” to run the Isochronous transaction using this endpoint.

Bit3-0 EndpointNumber

This sets an endpoint number between 0x1 and 0xF.

2.7.59 134h D_EPaControl (Device EPa Control)
2.7.60 144h D_EPbControl (Device EPb Control)
2.7.61 154h D_EPcControl (Device EPc Control)
2.7.62 1A4h D_EPdControl (Device EPd Control)
2.7.63 1B4h D_EPeControl (Device EPe Control)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device	134h	D_EPaControl	R/W	7: AutoForceNAK	0: Do nothing	1: Auto Force NAK	00h
	144h	D_EPbControl	R/W	6: EnShortPkt	0: Do nothing	1: Enable Short Packet	
	154h	D_EPcControl	R/W	5: DisAF_NAK_Short	0: Auto Force NAK Short	1 Disable Auto Force	
	1A4h	D_EPdControl	R	4: ToggleStat	Toggle sequence bit		
	1B4h	D_EPeControl	W	3: ToggleSet	0: Do nothing	1: Set Toggle sequence bit	
			W	2: ToggleClr	0: Do nothing	1: Clear Toggle sequence bit	
			R/W	1: ForceNAK	0: Do nothing	1: Force NAK	
			R/W	0: ForceSTALL	0: Do nothing	1: Force STALL	

This sets the endpoint EPx{x=a-e} operation.

Bit7 AutoForceNAK

This sets the register ForceNAK bit to “1” once the endpoint EPx{x=a-e} transaction ends normally.

Bit6 EnShortPkt

Setting this bit to “1” allows FIFO data smaller than MaxPacketSize to be sent as a short packet to the endpoint EPx{x=a-e} IN transaction. This bit is automatically cleared to “0” upon completion of the IN transaction sending the short packet. This bit is not cleared if a MaxPacketSize packet is sent.

Setting this bit to “1” when there is no data in the FIFO allows a zero-length packet to be sent to the IN token from the host. If data is written to the corresponding FIFO while this bit is set and packets are being sent, that data may also be sent, depending on the precise timing. Do not write data to the FIFO until the packet has been sent and this bit has been cleared.

Bit5 DisAF_NAK_Short

This sets whether the Auto Force NAK Short (referred to as AF_NAK_Short* hereafter) function is enabled or disabled.

* This automatically sets the ForceNAK bit to “1” if a short packet is received at the end of a normal OUT transaction.

The default setting is AF_NAK_Short function enabled.

Setting this bit to “1” disables the AF_NAK_Short function.

If the AutoForceNAK bit is set to “1,” the AutoForceNAK bit takes precedence.

Bit4 ToggleStat

This indicates the endpoint EPx{x=a-e} toggle sequence bit state.

Bit3 ToggleSet

This sets the endpoint EPx{x=a-e} toggle sequence bit to "1." If set concurrently with the ToggleClr bit, the ToggleClr bit function takes precedence.

Bit2 ToggleClr

This clears the endpoint EPx{x=a-e} toggle sequence bit to "0." If set concurrently with the ToggleSet bit, this bit function takes precedence.

Bit1 ForceNAK

Set this bit to "1" to return an NAK response to the endpoint EPx{x=a-e} transaction regardless of FIFO data quantity or free space.

If the transaction is already underway when this bit is set to "1," the bit is not set until the transaction ends, after which it set to "1." The bit is set to "1" immediately if the transaction is not underway.

Bit0 ForceSTALL

Set this bit to "1" to return a STALL response to the endpoint EPx{x=a-e} transaction. This bit takes precedence over the ForceNAK bit setting.

If a transaction is underway, this bit setting will be enabled from the subsequent transaction for a preset timeframe after the start of the transaction.

2.7.64 160h D_DescAdrs_H (Device Descriptor Address High)

2.7.65 161h D_DescAdrs_L (Device Descriptor Address Low)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device	160h	D_DescAdrs_H		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
			R/W	4: DescAdrs[12]	Descriptor Address		
				3: DescAdrs[11]			
				2: DescAdrs[10]			
				1: DescAdrs[9]			
				0: DescAdrs[8]			

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device	161h	D_DescAdrs_L	R/W	7: DescAdrs[7]	Descriptor Address	00h
				6: DescAdrs[6]		
				5: DescAdrs[5]		
				4: DescAdrs[4]		
				3: DescAdrs[3]		
				2: DescAdrs[2]		
				1: DescAdrs[1]		
				0: DescAdrs[0]		

This specifies the descriptor address.

160h.Bit7-5 Reserved

160h.Bit4-0, 161h.Bit7-0 DescAdrs[12:0]

This specifies the initial FIFO address for the descriptor reply function when starting the descriptor reply operation.

The descriptor address does not assign FIFO areas to the descriptor reply function. The descriptor address can specify all areas of the FIFO from 0x000 to 0x11FF (4.5 kbytes) regardless of FIFO area settings.

DescAdrs is updated by the volume of data sent each time an IN transaction is completed for endpoint EP0 when returning the descriptor reply. See the D_EP0Control register ReplyDescriptor section (“2.7.41 121h D_EP0Control (Device EP0 Control)”) for detailed information on the descriptor reply function.

Avoid duplication with the FIFO for other endpoints using D_DescAdrs_H,L and D_DescSize_H,L register specifications. The FIFO area for the descriptor reply function is not assigned explicitly.

2.7.66 162h D_DescSize_H (Device Descriptor Size High)

2.7.67 163h D_DescSize_L (Device Descriptor Size Low)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device	162h	D_DescSize_H		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
			R/W	1: DescSize[9]	Descriptor Size		
				0: DescSize[8]			

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Device	163h	D_DescSize_L	R/W	7: DescSize[7]	DescriptorSize	00h
				6: DescSize[6]		
				5: DescSize[5]		
				4: DescSize[4]		
				3: DescSize[3]		
				2: DescSize[2]		
				1: DescSize[1]		
				0: DescSize[0]		

This specifies the descriptor size.

162h.Bit7-2 Reserved

162h.Bit1-0, 163h.Bit7-0 DescSize[9:0]

This specifies the total data quantity returned to descriptor size by the descriptor reply function.

See EP0Control register ReplyDescriptor bit section (“2.7.41 121h D_EP0Control (Device EP0 Control)”) for detailed information on the descriptor reply function.

Descriptor size can specify values from 0x000 to 0x3FF regardless of FIFO size or area settings. DescSize is updated by the volume of data sent each time an IN transaction is completed for endpoint EP0 when returning the descriptor reply.

Avoid duplication with the FIFO for other endpoints using DescAdrs_H,L and DescSize_H,L register specifications. The FIFO area for the descriptor reply function is not assigned explicitly.

2.7.68 170h D_DMA0_FIFO_Control (Device DMA0 FIFO Control)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device	170h	D_ DMA0_FIFO_Control	R	7: FIFO_Running	0: FIFO is not running	1: FIFO is running	00h
			R/W	6: AutoEnShort	0: Do nothing	1: Auto Enable Short Packet	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

This indicates and sets the FIFO state during DMA0 transfer.

Bit7 FIFO_Running

This indicates that the endpoint FIFO connected to DMA0 is operating. This is set to “1” when DMA0 is started and is cleared to “0” when the FIFO becomes empty after DMA0 ends.

Bit6 AutoEnShort

This sets the corresponding endpoint EnShortPkt bit to “1” if the volume of data remaining in the FIFO is less than the maximum packet size after DMA0 ends.

It is enabled when the endpoint connected to DMA0 is in the IN direction.

Bit5-0 Reserved

2.7.69 172h D_DMA1_FIFO_Control (Device DMA1 FIFO Control)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device	172h	D_ DMA1_FIFO_Control	R	7: FIFO_Running	0: FIFO is not running	1: FIFO is running	00h
			R/W	6: AutoEnShort	0: Do nothing	1: Auto Enable Short Packet	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

This indicates and sets the FIFO state during DMA1 transfer.

Bit7 FIFO_Running

This indicates that the endpoint FIFO connected to DMA1 is operating. This is set to “1” when DMA1 is started and is cleared to “0” when the FIFO becomes empty after DMA1 ends.

Bit6 AutoEnShort

This sets the corresponding endpoint EnShortPkt bit to “1” if the volume of data remaining in the FIFO is smaller than the maximum packet size after DMA1 ends.

It is enabled when the endpoint connected to DMA1 is in the IN direction.

Bit5-0 Reserved

2.7.70 1BEh *DTM_Config (Device Transceiver Macro Config)*

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device	1BEh	***DTM_Config***		7:	0:	1:	XXh
				6:	0:	1:	
			R/W	5: ***DTM_SlopeValue[1]***	DTM Slope Value[1:0]		
			R/W	4: ***DTM_SlopeValue[0]***			
				3:	0:	1:	
				2:	0:	1:	
			R/W	1: ***DTM_TermValue[1]***	DTM Termination Value[1:0]		
			R/W	0: ***DTM_TermValue[0]***			

This register is used to set transceiver macro adjustment values.

Bit7-6 Reserved

Bit5-4 *DTM_SlopeValue[1:0]*

This adjusts the HS transmitter through-rate to one of four levels.

00: Slow

01: ↑

10: ↓

11: Fast

Bit3-2 Reserved

Bit1-0 *DTM_TermValue[1:0]*

This adjusts the HS transfer line termination to one of four levels.

00: High

01: ↑

10: ↓

11: Low

2.7.71 1E1h D_ModeControl (Address Mode Control)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Device	1E1h	D_ModeControl	W	7: (Reserved)	Don't set "1"		XXh
			W	6: (Reserved)	Don't set "1"		
			W	5: (Reserved)	Don't set "1"		
			W	4: SetAddressMode	0: Auto mode	1: Manual mode	
			W	3: (Reserved)	Don't set "1"		
			W	2: (Reserved)	Don't set "1"		
			W	1: (Reserved)	Don't set "1"		
			W	0: (Reserved)	Don't set "1"		

Bit7-5 Reserved

Bit4 SetAddressMode

This disables the automatic address setup function.

Bit3-0 Reserved

See “Appendix D SetAddress Request Response (for TS only)” for detailed information.

2.8 Host Register Details

2.8.1 0E0h H_SIE_IntStat_0 (Host SIE Interrupt Status 0)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Host	0E0h	H_SIE_IntStat_0		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
			R (W)	4: DetectCon	0: None	1: Detect Connect	
			R (W)	3: DetectDiscon	0: None	1: Detect Disconnect	
			R (W)	2: DetectRmtWkup	0: None	1: Detect Remote WakeUp	
			R (W)	1: DetectDevChirpOK	0: None	1: Detect Device Chirp OK	
			R (W)	0: DetectDevChirpNG	0: None	1: Detect Device Chirp NG	

This indicates host SIE-related interrupts.

All bits can be set to "1" to clear the interrupt factors.

Bit7-5 Reserved

Bit4 DetectCon

Directly instructs interrupt factors.

This is set to "1" if the USB cable is identified as being connected.

Bit3 DetectDiscon

Directly instructs interrupt factors.

This is set to "1" if the USB cable is detected as being disconnected.

Bit2 DetectRmtWkup

Directly instructs interrupt factors.

This is set to "1" if a Remote WakeUp signal is detected from the device in SUSPEND state.

Bit1 DetectDevChirpOK

Directly instructs interrupt factors.

This is set to "1" if a chirp signal is received normally from the device.

The hardware uses this bit to determine the presence of a chirp signal from the device when the host is in RESET state.

When this bit is set to "1," H_NegoControl_1.PortSpeed is set to "00b(HS)" and processing is subsequently performed for the HS device. This bit must always be cleared to "0" when the device is disconnected.

Bit0 DetectDevChirpNG

Directly instructs interrupt factors.

This is set to “1” if an error chirp signal is received from the device.

In 2-port mode, the synchronous bits (Bit4 to 0) can be read in ACTIVE60 or ACT_DEVICE state but cannot be written to (to clear interrupt factors). In 1-port mode, however, the synchronous bits (Bit4 to 0) can be read in ACTIVE60 state but cannot be written to (to clear interrupt factors).

The firmware should perform the following processing to prevent assertion of the interrupt signal XINT by the interrupt status when switching from ACT_HOST or ACT_ALL state to another state in 2-port mode or from ACT_DEVICE state to another state in 1-port mode.

<Switching from ACT_HOST or ACT_ALL state (in 2-port mode) or from ACT_DEVICE state (in 1-port mode)>

1) Process and clear the interrupt status (H_SIE_IntStat_0.Bit4 to 0)

2) Disable the interrupt status (H_SIE_IntEnb_0.Bit4 to 0)

<Switching to ACT_HOST or ACT_ALL state (in 2-port mode), or to ACT_DEVICE state (in 1-port mode)>

3) Clear the interrupt status (H_SIE_IntStat_0.Bit4 to 0)

4) Enable the interrupt status (H_SIE_IntEnb_0.Bit4 to 0)

2.8.2 0E1h H_SIE_IntStat_1 (SIE Host Interrupt Status 1)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Host	0E1h	H_SIE_IntStat_1		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1	
				4:	0:	1:	
			R (W)	3: DisabledCmp	0: None	1: Disabled Complete	
			R (W)	2: ResumeCmp	0: None	1: Resume Complete	
			R (W)	1: SuspendCmp	0: None	1: Suspend Cmplete	
			R (W)	0: ResetCmp	0: None	1: Reset Cmplete	

This indicates host SIE-related interrupts. All bits can be set to "1" to clear the interrupt factors.

Bit7-4 Reserved

Bit3 DisabledCmp

Directly instructs interrupt factors.

This is set to "1" upon completion of the switch to DISABLED state when H_NegoControl_0.AutoMode[3:0] is set to GoDISABLED and the state management function is run.

Bit2 ResumeCmp

Directly instructs interrupt factors.

This is set to "1" upon normal completion of the switch to RESUME state when H_NegoControl_0.AutoMode[3:0] is set to GoRESUME and the state management function is run.

Bit1 SuspendCmp

Directly instructs interrupt factors.

This is set to "1" upon completion of the switch to SUSPEND state when H_NegoControl_0.AutoMode[3:0] is set to GoSUSPEND and the state management function is run.

Bit0 ResetCmp

Directly instructs interrupt factors.

This is set to "1" upon normal completion of the switch to USB RESET state when H_NegoControl_0.AutoMode[3:0] is set to GoRESET and the state management function is run.

In 2-port mode, the synchronous bits (Bit3 to 0) can be read in ACTIVE60 or ACT_DEVICE state but cannot be written to (to clear interrupt factors). In 1-port mode, however, the synchronous bits (Bit4 to 0) can be read in ACTIVE60 state but cannot be written to (to clear interrupt factors).

The firmware should perform the following processing to prevent the assertion of the interrupt signal XINT by

the interrupt status when switching from ACT_HOST or ACT_ALL state to another state in 2-port mode, or from ACT_DEVICE state to another state in 1-port mode.

<Switching from ACT_HOST or ACT_ALL state (in 2-port mode), or from ACT_DEVICE state (in 1-port mode)>

1) Process and clear the interrupt status (H_SIE_IntStat_1.Bit3 to 0)

2) Disable the interrupt status (H_SIE_IntEnb_1.Bit3 to 0)

<Switching to ACT_HOST or ACT_ALL state (in 2-port mode), or to ACT_DEVICE state (in 1-port mode)>

3) Clear the interrupt status (H_SIE_IntStat_1.Bit3 to 0)

4) Enable the interrupt status (H_SIE_IntEnb_1.Bit3 to 0)

2.8.3 0E2h H_FIFO_IntStat (Host FIFO Interrupt Status)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Host	0E2h	H_FIFO_IntStat		7:	0:	1:	00h
			R (W)	6: FIFO_IDE_Cmp	0: None	1: FIFO-IDE Complete	
			R (W)	5: FIFO1_Cmp	0: None	1: FIFO1 Complete	
			R (W)	4: FIFO0_Cmp	0: None	1: FIFO0 Complete	
				3:	0:	1:	
			R (W)	2: FIFO_NotEmpty	0: None	1: FIFO NotEmpty	
			R (W)	1: FIFO_Full	0: None	1: FIFO Full	
			R (W)	0: FIFO_Empty	0: None	1: FIFO Empty	

This indicates host FIFO-related interrupts.

All bits can be set to “1” to clear the interrupt factors.

Bit7 Reserved

Bit6 FIFO_IDE_Cmp

Directly instructs interrupt factors.

This is set to “1” if the FIFO becomes empty after the IDE transfer ends when the channel joined to the FIFO area joined to the IDE is in the OUT direction. This is set to “1” if the IDE transfer ends when the channel joined to the FIFO area joined to the IDE is in the IN direction.

Bit5 FIFO1_Cmp

Directly instructs interrupt factors.

This is set to “1” if the FIFO becomes empty after the DMA1 transfer ends when the channel joined to the FIFO area joined to DMA1 is in the OUT direction. This is set to “1” if the DMA1 transfer ends when the channel joined to the FIFO area joined to DMA1 is in the IN direction.

Bit4 FIFO0_Cmp

Directly instructs interrupt factors.

This is set to “1” if the FIFO becomes empty after the DMA0 transfer ends when the channel joined to the FIFO area joined to DMA0 is in the OUT direction. This is set to “1” if the DMA0 transfer ends when the channel joined to the FIFO area joined to DMA0 is in the IN direction.

Bit3 Reserved

Bit2 FIFO_NotEmpty

Directly instructs interrupt factors.

This is set to “1” if the corresponding FIFO area contains data (i.e., NotEmpty) when the AREAx{x=0-5}Join_0.JoinFIFO_Stat bit is set to “1.”

Bit1 **FIFO_Full**

Directly instructs interrupt factors.

This is set to “1” if the corresponding FIFO area is full when the AREAx{x=0-5}Join_0.JoinFIFO_Stat bit is set to “1.”

Bit0 **FIFO_Empty**

Directly instructs interrupt factors.

This is set to “1” if the corresponding FIFO area is empty when the AREAx{x=0-5}Join_0.JoinFIFO_Stat bit is set to “1.”

2.8.4 0E3h H_FrameIntStat (Host Frame Interrupt Status)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Host	0E3h	H_FrameIntStat	R (W)	7: TriggerFrame	0: None	1: Trigger Frame	00h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
			R (W)	2: PortErr	0: None	1: Port Error	
			R (W)	1: FrameNumOver	0: None	1: Frame Number Over	
			R (W)	0: SOF	0: None	1: SOF	

This indicates host frame-related interrupts.

All bits can be set to "1" to clear the interrupt factors.

Bit7 TriggerFrame

Directly instructs interrupt factors.

This is set to "1" if the trigger frame number (H_TriggerFrameNum_H,L) matches the frame number (H_FameNumber_H,L) when the host controller sends an SOF token.

Bit6-3 Reserved

Bit2 PortErr

Directly instructs interrupt factors.

This is set to "1" if a port error is detected during host operations.

Bit1 FrameNumOver

Directly instructs interrupt factors.

This is set to "1" when the frame number counter overflows (FrameNumber_H register MSb (bit 2) changes from "1" to "0"). Count this interrupt to offset insufficient numbers of count digits in the FrameNumber_H,L registers.

Bit0 SOF

Directly instructs interrupt factors.

This is set to "1" in the cases shown below, depending on transfer speed.

HS: When the host controller sends a micro frame 0 SOF token

FS: When the host controller sends an SOF token

LS: When the host controller sends keepalive

2.8.5 0E4h H_CHrIntStat (Host CHr Interrupt Status)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Host	0E4h	H_CHrIntStat		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
			R	4: H_CHeIntStat	0: None	1: CHe Interrupt	
			R	3: H_CHdIntStat	0: None	1: CHd Interrupt	
			R	2: H_CHcIntStat	0: None	1: CHc Interrupt	
			R	1: H_CHbIntStat	0: None	1: CHb Interrupt	
			R	0: H_CHaIntStat	0: None	1: CHa Interrupt	

This indicates the channel CHx{x=a-e} interrupt.

Bit7-5 Reserved

Bit4 H_CHeIntStat

Indirectly instructs interrupt factors.

This is set to "1" when the H_CHeIntStat register contains an interrupt factor and the H_CHeIntEnb register bit corresponding to that interrupt factor is enabled.

Bit3 H_CHdIntStat

Indirectly instructs interrupt factors.

This is set to "1" when the H_CHdIntStat register contains an interrupt factor and the H_CHdIntEnb register bit corresponding to that interrupt factor is enabled.

Bit2 H_CHcIntStat

Indirectly instructs interrupt factors.

This is set to "1" when the H_CHcIntStat register contains an interrupt factor and the H_CHcIntEnb register bit corresponding to that interrupt factor is enabled.

Bit1 H_CHbIntStat

Indirectly instructs interrupt factors.

This is set to "1" when the H_CHbIntStat register contains an interrupt factor and the H_CHbIntEnb register bit corresponding to that interrupt factor is enabled.

Bit0 H_CHaIntStat

Indirectly instructs interrupt factors.

This is set to "1" when the H_CHaIntStat register contains an interrupt factor and the H_CHaIntEnb register bit corresponding to that interrupt factor is enabled.

2.8.6 0E5h H_CH0IntStat (Host CH0 Interrupt Status)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Host	0E5h	H_CH0IntStat	R (W)	7: TotalSizeCmp	0: None	1: TotalSize Complete	00h
			R (W)	6: TranACK	0: None	1: Tran ACK	
			R (W)	5: TranErr	0: None	1: Tran Error	
			R (W)	4: ChangeCondition	0: None	1: Change Condition	
				3:	0:	1:	
				2:	0:	1:	
			R (W)	1: CTL_SupportCmp	0: None	1: CTL_Support Cmplete	
			R (W)	0: CTL_SupportStop	0: None	1: CTL_Support Stop	

This indicates the channel CH0 interrupt status.

All bits can be set to "1" to clear the interrupt factors.

Bit7 TotalSizeCmp

Directly instructs interrupt factors.

This bit is set to "1" when the IRP packet transfer has been completed normally.

When the control transfer support function is operating, this bit is set to "1" once the setup stage, data stage, and status stage have ended normally.

Bit6 TranACK

Directly instructs interrupt factors.

This bit is set to "1" when the number of transactions set in the H_CH0Config_0 register ACK_Cnt have ended normally. When the control transfer support function is operating, only the data stage transactions are counted.

Bit5 TranErr

Directly instructs interrupt factors.

This bit is set to "1" when any one of the transactions ends in a retry error, namely time-out error, CRC error, bit stuffing error, PID error (including unforeseen PID), or toggle mismatch error.

Bit4 ChangeCondition

Directly instructs interrupt factors.

This bit is set to "1" when a retry error occurs three times in succession for a transaction, including condition code stall, data overrun, and data underrun.

This bit is also set to "1" when the H_CH0Config_0.TranGo bit is raised by the firmware. In this case, ConditionCode indicates the final transaction results.

Bit3-2 Reserved

Bit1 **CTL_SupportCmp**

Directly instructs interrupt factors.

This bit is set to “1” when all stages of the control transfer end normally using the control transfer support function.

This bit is also set to “1” if the status stage ends normally and stop processing is completed for control transfer support function stop processing due to clearing of the H_CTL_SupportControl register CTL_SupportGo bit.

Bit0 **CTL_SupportStop**

Directly instructs interrupt factors.

This bit is set to “1” when the control transfer is aborted due to an error by the control transfer support function.

This bit is also set to “1” if the stop processing ends for a stage other than the status stage or if the status stage ends with a transaction error for the control transfer support function stop processing due to the clearing of the H_CTL_SupportControl register CTL_SupportGo bit.

2.8.7 0E6h H_CHaIntStat (Host CHa Interrupt Status)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Host	0E6h	H_CHaIntStat	R (W)	7: TotalSizeCmp	0: None	1: TotalSize Complete	00h
			R (W)	6: TranACK	0: None	1: Tran ACK	
			R (W)	5: TranErr	0: None	1: Tran Error	
			R (W)	4: ChangeCondition	0: None	1: Change Condition	
				3:	0:	1:	
				2:	0:	1:	
			R (W)	1: BO_SupportCmp	0: None	1: BO Support Cmplete	
			R (W)	0: BO_SupportStop	0: None	1: BO Support Stop	

This indicates the channel CHa interrupt status.

All bits can be set to "1" to clear the interrupt factors.

Bit7 TotalSizeCmp

Directly instructs interrupt factors.

This bit is set to "1" when the IRP transfer has been completed normally.

When the bulk-only support function is operating, this bit is set to "1" once the CBW transport, data transport, and CSW transport have ended normally.

Bit6 TranACK

Directly instructs interrupt factors.

This bit is set to "1" when the number of transactions set in the H_CHaConfig_0 register ACK_Cnt have ended normally. When the bulk-only support function is operating, only the data transport transactions are counted.

Bit5 TranErr

Directly instructs interrupt factors.

This bit is set to "1" when any one of the transactions ends in a retry error, namely time-out error, CRC error, bit stuffing error, PID error (including unforeseen PID), or toggle mismatch error.

Bit4 ChangeCondition

Directly instructs interrupt factors.

This bit is set to "1" when a retry error occurs three times in succession for a transaction, including condition code stall, data overrun, or data underrun.

This bit is also set to "1" when the H_CHaConfig_0.TranGo bit is raised by the firmware. In this case, ConditionCode indicates the final transaction results.

Bit3-2 Reserved

Bit1 BO_SupportCmp

Directly instructs interrupt factors.

This bit is set to “1” when the status transport ends normally using the bulk-only support function.

This bit is also set to “1” if the CSW transport ends normally and stop processing is completed for the bulk-only support function stop processing due to the clearing of the H_BO_SupportControl register BO_SupportGo bit.

Bit0 BO_SupportStop

Directly instructs interrupt factors.

This bit is set to “1” when any transfer ends with an error by the bulk-only support function.

This bit is also set to “1” if the stop processing ends for a transport other than the CSW transport during the bulk-only support function stop processing due to clearing of the H_BO_SupportControl register BO_SupportGo bit, or if an error is detected in the CSW transport.

2.8.8 0E7h H_CHbIntStat (Host CHb Interrupt Status)

2.8.9 0E8h H_CHcIntStat (Host CHc Interrupt Status)

2.8.10 0E9h H_CHdIntStat (Host CHd Interrupt Status)

2.8.11 0EAh H_CHeIntStat (Host CHe Interrupt Status)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Host	0E7h	H_ChbIntStat	R (W)	7: TotalSizeCmp	0: None	1: TotalSize Complete	00h
	0E8h	H_CHcIntStat	R (W)	6: TranACK	0: None	1: Tran ACK	
	0E9h	H_CHdIntStat	R (W)	5: TranErr	0: None	1: Tran Error	
	0EAh	H_CHeIntStat	R (W)	4: ChangeCondition	0: None	1: Change Condition	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

This indicates the interrupt status of channel CHx{x=b-e}.

Writing "1" to all bits clears the interrupt factors.

Bit7 TotalSizeCmp

Directly instructs interrupt factors.

This bit is set to "1" when the transfer is completed normally using IRP transfer.

Bit6 TranACK

Directly instructs interrupt factors.

This bit is set to "1" when the number of transactions set in the H_CHx{x=b-e}Config_0 register ACK_Cnt are completed normally.

Bit5 TranErr

Directly instructs interrupt factors. This bit is set to "1" when any one of the transactions ends in a retry error (e.g., time-out error, CRC error, bit stuffing error, PID error (including unforeseen PID), or toggle mismatch error).

This bit is also set to "1" when a buffer overrun or buffer underrun occurs during isochronous transfers.

Bit4 ChangeCondition

Directly instructs interrupt factors.

This bit is set to “1” when a condition code stall, data overrun, or data underrun occurs during a transaction.

This bit is also set to “1” when a retry error occurs three times during bulk or interrupt transfers.

Also set to “1” when the H_Chx{x=b-e}Config_0.TranGo bit is raised by the firmware. In this case, ConditionCode indicates the final transaction results.

Bit3-0 Reserved

2.8.12 0F0h H_SIE_IntEnb_0 (Host SIE Interrupt Enable)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Host	0F0h	H_SIE_IntEnb_0		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
			R/W	4: EnDetectCon	0: Disable	1: Enable	
			R/W	3: EnDetectDiscon	0: Disable	1: Enable	
			R/W	2: EnDetectRmtWkup	0: Disable	1: Enable	
			R/W	1: EnDetectDevChirpOK	0: Disable	1: Enable	
			R/W	0: EnDetectDevChirpNG	0: Disable	1: Enable	

This permits or prohibits assertion of the MainIntStat register H_SIE_IntStat_0 bit using H_SIE_IntStat_0 register interrupt factors.

The synchronous bits (Bit4 to 0) can be read from in ACTIVE60 and ACT_DEVICE states but cannot be written to. See “2.8.1 0E0h H_SIE_IntStat_0 (Host SIE Interrupt Status 0)” for detailed information on processing to change from ACT_HOST or ACT_ALL state to another state.

2.8.13 0F1h H_SIE_IntEnb_1 (SIE Host Interrupt Enable 1)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Host	0F1h	H_SIE_IntEnb_1		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1	
				4:	0:	1: F	
			R/W	3: EnDisabledCmp	0: Disable	1: Enable	
			R/W	2: EnResumeCmp	0: Disable	1: Enable	
			R/W	1: EnSuspendCmp	0: Disable	1: Enable	
			R/W	0: EnResetCmp	0: Disable	1: Enable	

This permits or prohibits assertion of the MainIntStat register H_SIE_IntStat_1 bit using H_SIE_IntStat_1 register interrupt factors.

The synchronous bits (Bit3 to 0) can be read from in ACTIVE60 and ACT_DEVICE states but cannot be written to. See “2.8.2 0E1h H_SIE_IntStat_1 (SIE Host Interrupt Status 1)” for detailed information on processing to change from ACT_HOST or ACT_ALL state to another state.

2.8.14 0F2h H_FIFO_IntEnb (Host FIFO Interrupt Enable)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Host	0F2h	H_FIFO_IntEnb		7:	0:	1:	00h
			R/W	6: EnFIFO_IDE_Cmp	0: Disable	1: Enable	
			R/W	5: EnFIFO1_Cmp	0: Disable	1: Enable	
			R/W	4: EnFIFO0_Cmp	0: Disable	1: Enable	
				3:	0:	1:	
			R/W	2: EnFIFO_NotEmpty	0: Disable	1: Enable	
			R/W	1: EnFIFO_Full	0: Disable	1: Enable	
			R/W	0: EnFIFO_Empty	0: Disable	1: Enable	

This permits or prohibits assertion of the MainIntStat register H_FIFO_IntStat bit using H_FIFO_IntStat register interrupt factors.

2.8.15 0F3h H_FrameIntEnb (Host Frame Interrupt Enable)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Host	0F3h	H_FrameIntEnb	R/W	7: EnTriggerFrame	0: Disable	1: Enable	00h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
			R/W	2: EnPortErr	0: Disable	1: Enable	
			R/W	1: EnFrameNumOver	0: Disable	1: Enable	
			R/W	0: EnSOF	0: Disable	1: Enable	

This permits or prohibits assertion of the MainIntStat register H_FrameIntStat bit using H_FrameIntStat register interrupt factors.

2.8.16 0F4h H_CHrIntEnb (Host CHr Interrupt Enable)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Host	0F4h	H_CHrIntEnb		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
			R/W	4: EnH_CHeIntStat	0: Disable	1: Enable	
			R/W	3: EnH_CHdIntStat	0: Disable	1: Enable	
			R/W	2: EnH_CHcIntStat	0: Disable	1: Enable	
			R/W	1: EnH_CHbIntStat	0: Disable	1: Enable	
			R/W	0: EnH_CHaIntStat	0: Disable	1: Enable	

This permits or prohibits assertion of the MainIntStat register H_CHrIntStat bit using H_CHrIntStat register interrupt factors.

2.8.17 0F5h H_CH0IntEnb (Host CH0 Interrupt Enable)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Host	0F5h	H_CH0IntEnb	R/W	7: EnTotalSizeCmp	0: Disable	1: Enable	00h
			R/W	6: EnTranACK	0: Disable	1: Enable	
			R/W	5: EnTranErr	0: Disable	1: Enable	
			R/W	4: EnChangeCondition	0: Disable	1: Enable	
				3:	0:	1:	
				2:	0:	1:	
			R/W	1: EnCTL_SupportCmp	0: Disable	1: Enable	
			R/W	0: EnCTL_SupportStop	0: Disable	1: Enable	

This permits or prohibits assertion of the MainIntStat register H_CH0IntStat bit using H_CH0IntStat register interrupt factors.

2.8.18 0F6h H_CHaIntEnb (Host CHa Interrupt Enable)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Host	0F6h	H_CHaIntEnb	R/W	7: EnTotalSizeCmp	0: Disable	1: Enable	00h
			R/W	6: EnTranACK	0: Disable	1: Enable	
			R/W	5: EnTranErr	0: Disable	1: Enable	
			R/W	4: EnChangeCondition	0: Disable	1: Enable	
				3:	0:	1:	
				2:	0: Disable	1:	
			R/W	1: EnBO_Support_Cmp	0: Disable	1: Enable	
			R/W	0: EnBO_Support_Stop	0: Disable	1: Enable	

This permits or prohibits assertion of the H_CHrIntStat register CHaIntStat bit using H_CHaIntStat register interrupt factors.

2.8.19 0F7h H_CHbIntEnb (Host CHb Interrupt Enable)

2.8.20 0F8h H_CHcIntEnb (Host CHc Interrupt Enable)

2.8.21 0F9h H_CHdIntEnb (Host CHd Interrupt Enable)

2.8.22 0FAh H_CHeIntEnb (Host CHe Interrupt Enable)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Host	0F7h	H_CHbIntEnb	R/W	7: EnTotalSizeCmp	0: Disable	1: Enable	00h
	0F8h	H_CHcIntEnb	R/W	6: EnTranACK	0: Disable	1: Enable	
	0F9h	H_CHdIntEnb	R/W	5: EnTranErr	0: Disable	1: Enable	
	0FAh	H_CHeIntEnb	R/W	4: EnChangeCondition	0: Disable	1: Enable	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

These permit or prohibit assertion of the H_CHrIntStat register CHx{x=b-e}IntStat bit using H_CHx{x=b-e}IntStat register interrupt factors.

2.8.23 100h *H_Reset (Host Reset)*

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Host	100h	***H_Reset***		7:	0:	1:	01h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
			R/W	0: ***ResetHTM***	0: Do nothing	1: Reset HTM	

This resets the host transceiver macro.

It can be accessed even in SLEEP and SNOOZE states.

Bit7-1 Reserved

Bit0 *ResetHTM*

Setting this bit to "1" initializes the LSI host transceiver macro.

To cancel the reset, clear the bit to "0."

2.8.24 102h H_NegoControl_0 (Host NegoControl 0)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Host	102h	H_NegoControl_0	R/W	7: AutoModeCancel	0: None	1: Cancel	1Xh
			R	6: HostState[2]	HostState[2:0]		
			R	5: HostState[1]			
			R	4: HostState[0]			
			W	3: AutoMode[3]	AutoMode[3:0]		
			W	2: AutoMode[2]			
			W	1: AutoMode[1]			
			W	0: AutoMode[0]			

This sets host negotiation operations.

Bit7 AutoModeCancel

Setting this bit to "1" cancels the host state management support function and maintains that state (H_NegoControl_0.AutoMode and H_XcvrControl settings are retained, signal line status is retained, internal timer is stopped, and connection, disconnection, device chirp, remote wakeup detection functions are turned off).

The host state management support function must be canceled using this bit before performing the operations listed below.

- When switching the host state to IDLE state
- When switching the host state to DISABLED state without waiting for the reset completion status (H_SIE_IntStat_1.ResetCmp) to be issued when a chirp error has been detected from the device
- When running test mode using H_USB_Test.EnHS_Test
  Setting this bit to "1" initiates host state management support function stop processing and sets this bit to "0" once the stop processing is completed (requires approximately 6 cycles for a 60 MHz clock). In this case, check that the bit has been changed to "0" before setting H_NegoControl_0.AutoMode GoIDLE or GoDISABLED or before setting H_USB_Test.EnHS_Test.

Bit6-4 HostState[2:0]

This indicates the current host state when running the host state management support function. The state is given as one of the following.

000: Reserved

001: IDLE

010: WAIT_CONNECT

011: DISABLED

100: USB_RESET

101: USB_OPERATIONAL

110: USB_SUSPEND

111: USB_RESUME

Bit3-0 AutoMode[3:0]

This sets the new host state when running the host state management support function.

This register is write-only and sets any of the following.

0001: GoIDLE (switches to IDLE state)

0010: GoWAIT_CONNECT (switches to WAIT_CONNECT state)

0011: GoDISABLED (switches to DISABLED state)

0100: GoRESET (switches to RESET state)

0101: GoOPERATIONAL (switches to OPERATIONAL state)

0110: GoSUSPEND (switches to SUSPEND state)

0111: GoRESUME (switches to RESUME state)

1001: GoWAIT_CONNECTtoDIS (switches continuously from WAIT_CONNECT to DISABLED state)

1010: GoWAIT_CONNECTtoOP (switches continuously from WAIT_CONNECT to OPERATIONAL state)

1100: GoRESETtoOP (switches continuously from RESET to OPERATIONAL state)

1110: GoSUSPENDtoOP (switches continuously from SUSPEND to OPERATIONAL state)

1111: GoRESUMEtoOP (switches continuously from RESUME to OPERATIONAL state)

All others: Reserved

Use the following procedure to switch from any state to IDLE state (using GoIDLE).

- Write 0x80 to the H_NegoControl_0 register.
- Confirm that the H_NegoControl_0.AutoModeCancel bit has changed to 0.
- Write 0x01 to the H_NegoControl_0 register.

2.8.25 104h H_NegoControl_1 (Host NegoControl 1)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Host	104h	H_NegoControl_1		7:	0:	1:	10h
				6:	0:	1:	
			R/W	5: PortSpeed[1]	PortSpeed[1:0]		
			R/W	4: PortSpeed[0]			
				3:	0:	1:	
				2:	0:	1:	
			R/W	1: DisChirpFinish	0: Normal	1: DisableChirpFinish	
			R/W	0: RmtWkupDetEnb	0: Disable	1: Enable	

This sets host negotiation operations.

Note: The reset value for this register is the value that can be read in ACT_HOST and ACT_ALL states when in 2-port mode and in ACT_DEVICE state when in 1-port mode. The reset value is read as 00h in all other states.

Bit7-6 Reserved

Bit5-4 PortSpeed[1:0]

This indicates and sets the transfer speed. It is normally read-only, but can be written to if the H_Protect.PortSpeedWrEnb bit is set to "1". There is no need to write to this bit except when running Test_Force_Enable test mode.

00: High Speed

01: Full Speed

10: Reserved

11: Low Speed

Bit3-2 Reserved

Bit1 DisChirpFinish

This sets the operating mode when the device chirp does not end within the stipulated timeframe.

0: Ends the USB Reset by sending a USB Reset within the stipulated timeframe after lifting the device chirp error status.

1: Waits for the device chirp to end after lifting the device chirp error status and ends the USB Reset after running the host chirp upon completion of the device chirp.

Bit0 RmtWkupDetEnb

Enables/disables the remote wakeup detection function.

2.8.26 106h H_USB_Test (Host USB_Test)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Host	106h	H_USB_Test	R/W	7: EnHS_Test	0: Do nothing	1: EnHS_Test	00h
				6:	0:	1:	
				5:	0:	1:	
			R/W	4: Test_Force_Enable	0: Do nothing	1: Test_Force_Enable	
			R/W	3: Test_SE0_NAK	0: Do nothing	1: Test_SE0_NAK	
			R/W	2: Test_J	0: Do nothing	1: Test_J	
			R/W	1: Test_K	0: Do nothing	1: Test_K	
			R/W	0: Test_Packet	0: Do nothing	1: Test_Packett	

This sets host USB2.0 test mode operations.

Test mode can be run in WAIT_CONNECT, DISABLED, or SUSPEND state.

The respective state processing must be stopped before switching from these states to test mode. Use the following procedure when switching to test mode.

- Set the TranGo bit (H_CHx{x=0,a-e}Config_0.TranGo), H_CTL_SupportControl.CTL_SupportGo, and H_BO_SupportControl.BOSupportGo to "0" for all channels.
- Write 0x80 to the H_NegoControl_0 register.
- Confirm that the H_NegoControl_0.AutoModeCancel bit has changed to "0."
- Set any one of the last five bits of the register to "1" simultaneously with the EnHS_Test bit.

Write 0x00 to this register when switching from one test mode to another test mode or when ending test mode. This ends test mode, and the host state switches to IDLE.

Bit7 EnHS_Test

Setting any one of the last five bits of the H_USB_Test register to "1" simultaneously with this bit switches to the test mode corresponding to the bit.

Bit6-5 Reserved

Bit4 Test_Force_Enable

This bit is used when running Test_Force_Enable test mode.
Perform the following processing after the procedures for switching to the previously mentioned test mode to run Test_Force_Enable test mode.

First set H_Protect.PortSpeedWrEnb to "1". Then clear H_Protect.PortSpeedWrEnb to "0" after setting H_NegoControl_1.PortSpeed = "HS(0b00)." Next set H_NegoControl_0.AutoMode to "GoOPERATIONAL".

Setting the Test_Force_Enable and EnHS_Test bits simultaneously to "1" after performing the above processing allows switching to TestForceEnable test mode. This test mode enables the host port to send SOF in HS mode and detect disconnection.

Bit3 Test_SE0_NAK

Setting this bit to "1" simultaneously with the EnHS_Test bit allows switching to Test_SE0_NAK test mode. This test mode enables the host port to receive data in HS mode.

Bit2 TEST_J

Setting this bit to "1" simultaneously with the EnHS_Test bit allows switching to Test_J test mode. This test mode enables the host port to send "J" in HS mode.

Bit1 TEST_K

Setting this bit to "1" simultaneously with the EnHS_Test bit allows switching to Test_K test mode. This test mode enables the host port to send "K" in HS mode.

Bit0 Test_Packet

Setting this bit to "1" simultaneously with the EnHS_Test bit allows switching to Test_Packet test mode. This test mode can be used with CH0 only. Set the FIFO area joined to CH0 to 64 bytes before switching to this test mode, clear the FIFO area, and write the test packet data shown below to the FIFO area.

The following 53 bytes are written to the FIFO in packet transmission test mode:

00h, 00h, 00h, 00h, 00h, 00h, 00h, 00h,

00h, AAh, AAh, AAh, AAh, AAh, AAh, AAh,

AAh, EEh, EEh, EEh, EEh, EEh, EEh, EEh,

EEh, FEh, FFh, FFh, FFh, FFh, FFh, FFh,

FFh, FFh, FFh, FFh, FFh, 7Fh, BFh, DFh,

EFh, F7h, FBh, FDh, FCh, 7Eh, BFh, DFh,

EFh, F7h, FBh, FDh, 7Eh

Since PID and CRC are added by SIE when sending test packets, of the test packet data described in the USB standard Rev 2.0, data written to the FIFO consists of data that follows DATA0 PID data to data other than CRC16.

2.8.27 110h H_CH0SETUP_0 (Host CH0 SETUP 0)
2.8.28 111h H_CH0SETUP_1 (Host CH0 SETUP 1)
2.8.29 112h H_CH0SETUP_2 (Host CH0 SETUP 2)
2.8.30 113h H_CH0SETUP_3 (Host CH0 SETUP 3)
2.8.31 114h H_CH0SETUP_4 (Host CH0 SETUP 4)
2.8.32 115h H_CH0SETUP_5 (Host CH0 SETUP 5)
2.8.33 116h H_CH0SETUP_6 (Host CH0 SETUP 6)
2.8.34 117h H_CH0SETUP_7 (Host CH0 SETUP 7)

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Host	110h -117h	H_CH0SETUP_0 -H_CH0SETUP_7	R/W	7: CH0SETUP_n[7]	Channel 0 SETUP Data 0 - Channel 0 SETUP Data 7	00h
				6: CH0SETUP_n[6]		
				5: CH0SETUP _n[5]		
				4: CH0SETUP _n[4]		
				3: CH0SETUP _n[3]		
				2: CH0SETUP _n[2]		
				1: CH0SETUP _n[1]		
				0: CH0SETUP _n[0]		

These registers set the 8 bytes of data in sequence for sending in the channel CH0 setup stage.

CH0SETUP_0

Sets bmRequestType.

CH0SETUP_1

Sets bRequest.

CH0SETUP_2

Sets the last 8 bits of wValue.

CH0SETUP_3

Sets the first 8 bits of wValue.

CH0SETUP_4

Sets the last 8 bits of wIndex.

CH0SETUP_5

Sets the first 8 bits of wIndex.

CH0SETUP_6

Sets the last 8 bits of wLength.

CH0SETUP_7

Sets the first 8 bits of wLength.

2.8.35 11Eh H_FrameNumber_H (Host FrameNumber High)

2.8.36 11Fh H_FrameNumber_L (Host FrameNumber Low)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Host	11Eh	H_FrameNumber_H		7:	0:	1:	07h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
			R/W	2: FrameNumber[10]	Frame Number High		
				1: FrameNumber[9]			
				0: FrameNumber[8]			

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Host	11Fh	H_FrameNumber_L	R/W	7: FrameNumber[7]	Frame Number Low	FFh
				6: FrameNumber[6]		
				5: FrameNumber[5]		
				4: FrameNumber[4]		
				3: FrameNumber[3]		
				2: FrameNumber[2]		
				1: FrameNumber[1]		
				0: FrameNumber[0]		

These indicate the USB frame number updated each time an SOF token is sent. The H_FrameNumber_H and H_FrameNumber_L registers must be accessed as a pair when obtaining the frame number.

Note: The reset value for this register is the value that can be read in ACT_HOST and ACT_ALL states when in 2-port mode and in ACT_DEVICE state when in 1-port mode. The reset value is read as 00h in all other states.

11Eh.Bit7-3 Reserved

11Eh.Bit2-0, 11Fh.Bit7-0 FrameNumber[10:0]

Indicates FrameNumber for the SOF packet to be sent.

2.8.37 120h H_CH0Config_0 (Host Channel 0 Configuration0)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Host	120h	H_CH0Config_0	R/W	7: ACK_Cnt[3]	ACK Count		00h
				6: ACK_Cnt[2]			
				5: ACK_Cnt[1]			
				4: ACK_Cnt[0]			
			R/W	3: SpeedMode[1]	Speed Mode		
				2: SpeedMode[0]			
			R/W	1: Toggle	0: Toggle0	1: Toggle1	
			R/W	0: TranGo	0: Stand by	1: Transaction Start	

This sets basic channel CH0 settings for host operations.

Bit7-4 ACK_Cnt[3:0]

This sets the ACK count number for transfers performed using channel CH0.

The H_CH0IntStat register TranACK bit is set once the ACK number set is counted.

0000: 16 ACK counts

0001 to 1111: 1 to 15 ACK counts

When the control transfer support function is running, only data stage transactions are counted. Setup stage and status stage transactions are not counted.

Bit3-2 SpeedMode[1:0]

This sets the operating mode for the device performing the transfer using channel CH0.

00: HS mode – Use this setting for HS device.

01: FS mode – Use this setting for FS device.

10: Reserved – Use of this value is prohibited.

11: LS mode – Use this setting for LS device.

Bit1 Toggle

This sets initial value of the toggle sequence bit when starting a transaction. It also indicates the toggle sequence bit state after the transaction has been run or completed.

0: Toggle 0

1: Toggle 1

Bit0 TranGo

Setting this bit to “1” initiates the channel CH0 transactions. The transaction process can be stopped after being started by clearing the bit to “0.” This bit also indicates whether transactions are running using channel CH0.

0: Stops transactions (Transactions are stopped)

1: Starts transactions (Transactions are running)

The H_CH0IntStat register TotalSizeCmp bit is set to “1” as soon as the number of bytes set in registers H_CH0TotalSize_H to L have been transferred. This bit returns automatically to “0”. It is reset to “0” if the H_CH0IntStat register ChangeCondition bit has been set. In this case, the cause is set to the H_CH0ConditionCode register to permit review.

The H_CH0IntStat register ChangeCondition bit is set as soon as the transactions underway end when stopped by clearing the bit. The data in the FIFO, (remaining) total size, and channel-related settings remain unchanged even when the transactions are halted. The transactions can be resumed from the point at which it was stopped by setting this bit to “1” once again. (To perform new transactions, clear the FIFO and reset the channel information.)

This bit does not need to be set when using the control transfer support function.

2.8.38 121h H_CH0Config_1 (Host Channel 0 Configuration1)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Host	121h	H_CH0Config_1	R/W	7: TID[1]	Transaction ID		00h
				6: TD[0]			
				5:	0:		
				4:	0:		
				3:	0:		
				2:	0:		
				1:	0:		
				0:	0:		

This sets basic channel CH0 settings for host operations.

Bit7-6 TID[1:0]

This sets the transaction type (SETUP/OUT/IN) to be issued using channel CH0. This bit setting is disabled when the CTL_SupportControl register CTL_SupportGo bit is set to "1" and the transaction is initiated.

00: SETUP – Issues a SETUP token.

01: OUT – Issues an OUT token.

10: IN – Issues an IN token.

11: Reserved – Use of this value is prohibited.

This bit does not need to be set when using the control transfer support function.

Bit5-0 Reserved

2.8.39 123h H_CH0MaxPktSize (Host Channel 0 Max Packet Size)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Host	123h	H_CH0MaxPktSize		7:	0:	1:	00h
			R/W	6: MaxPktSize[6]	Max Packet Size		
				5: MaxPktSize[5]			
				4: MaxPktSize[4]			
				3: MaxPktSize[3]			
				2: MaxPktSize[2]			
				1: MaxPktSize[1]			
				0: MaxPktSize[0]			

This sets channel CH0 MaxPacketSize for host operations.

Bit7 Reserved

Bit6-0 MaxPktSize[6:0]

This sets channel CH0 MaxPacketSize.

Set to one of the following.

LS: 8 bytes

FS: 8, 16, 32, 64 bytes

HS: 64 bytes

All other settings are prohibited.

2.8.40 126h H_CH0TotalSize_H (Host Channel 0 Total Size High)

2.8.41 127h H_CH0TotalSize_L (Host Channel 0 Total Size Low)

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Host	126h	H_CH0TotalSize_H	R/W	7: TotalSize[15]	Total Size High	00h
				6: TotalSize[14]		
				5: TotalSize[13]		
				4: TotalSize[12]		
				3: TotalSize[11]		
				2: TotalSize[10]		
				1: TotalSize[9]		
				0: TotalSize[8]		

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Host	127h	H_CH0TotalSize_L	R/W	7: TotalSize[7]	Total Size Low	00h
				6: TotalSize[6]		
				5: TotalSize[5]		
				4: TotalSize[4]		
				3: TotalSize[3]		
				2: TotalSize[2]		
				1: TotalSize[1]		
				0: TotalSize[0]		

This sets TotalSize for the data to be transferred using channel CH0 for host operations.

118h.Bit7-0, 119h.Bit7-0 TotalSize[15:0]

This sets the total size in bytes of the data to be transferred using channel CH0 (max. 65,535 bytes = approx. 64 Kbytes).

This register is updated each time a transaction ends normally after the transaction has been started by the H_CH0Config_0 register TranGo bit.

When TotalSize = 0, a zero-length packet is sent when an OUT transaction is started by the H_CH0Config_0 register TranGo bit.

This register does not need to be set when performing a SETUP transaction or when using the control transfer support function.

2.8.42 128h H_CH0HubAdrs (Host Channel 0 Hub Address)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Host	128h	H_CH0HubAdrs	R/W	7: HubAdrs[3]	Hub Address		00h
				6: HubAdrs[2]			
				5: HubAdrs[1]			
				4: HubAdrs[0]			
				3:	0:	1:	
			R/W	2: Port[2]	Port Number		
				1: Port[1]			
				0: Port[0]			

This sets hub connecting to channel CH0 for host operations.

Bit7-4 HubAdrs[3:0]

This sets the USB address of the hub to which the function performing the transfer using channel CH0 connects.

It can be set as desired to any value from 0 to 15.

Bit3 Reserved

Bit2-0 Port[2:0]

This sets the port number of the hub to which the function performing the transfer using channel CH0 connects.

It can be set as desired to any value from 0 to 7.

2.8.43 129h H_CH0FuncAdrs (Host Channel 0 Function Address)

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Host	129h	H_CH0FuncAdrs	R/W	7: FuncAdrs[3]	Function Address	00h
				6: FuncAdrs[2]		
				5: FuncAdrs[1]		
				4: FuncAdrs[0]		
			R/W	3: EP_Number[3]	Endpoint Number	
				2: EP_Number[2]		
				1: EP_Number[1]		
				0: EP_Number[0]		

This sets the address of the function performing the transfer using channel CH0 for host operations.

Bit7-4 FuncAdrs[3:0]

This sets the USB address of the function including the endpoint managed by channel CH0.

It can be set as desired to any value from 0 to 15.

Bit3-0 EP_Number[3:0]

This sets the endpoint number for the transfer using channel CH0.

It can be set as desired to any value from 0 to 15.

2.8.44 12Bh H_CTL_SupportControl (Host ControlTransfer Support Control)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Host	12Bh	H_CTL_Support-		7:	0:	1:	00h
		Control		6:	0:	1:	
			R/W	5: CTL_SupportState[1]	Control Transfer Support State		
				4: CTL_SupportState[0]			
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
			R/W	0: CTL_SupportGo	0: Stand by	1: Control Transfer Go	

This sets the support function for control transfer using channel CH0 for host operations.

Bit7-6 Reserved

Bit5-4 CTL_SupportState[1:0]

This sets the CTL_SupportGo to "1" and indicates which stage is being run during a transfer using the control support function.

00: Idle – Indicates that the transfer has not been started or was completed normally.

01: Setup Stage – Indicates that the setup stage is running.

10: Data Stage – Indicates that the data stage is running.

11: Status Stage – Indicates that the status stage is running.

Bit3-1 Reserved

Bit0 CTL_SupportGo

Setting this bit to "1" automatically initiates the setup stage (and data stage) to status stage with the control transfer support function using channel CH0.

In the setup stage, a SETUP token is automatically issued, and the requests set in H_CH0SETUP_0 to 7 are sent.

If a subsequent data stage exists, the transaction will run automatically with the specified direction and size.

Finally, in the status stage, an appropriate PID token is automatically issued and a zero-length packet sent and received based on the presence of a data stage and the direction.

The H_CH0IntStat register CTL_SupportCmp bit is set once the transactions above, and the stage sequence ends normally. If a packet error is detected midway during the sequence, the H_CH0IntStat register CTL_SupportStop bit is set, and the transaction stops. The cause is set to the ConditionCode register in this case to permit review.

This bit is automatically cleared once the control transfer ends (either normally or in error).

The control transfer can be stopped by clearing this bit while running the control transfer support function. The CTL_SupportCmp bit is set if the control transfer ends normally in the status stage. In all other cases, the CTL_SupportStop is set. Refer to CTL_SupportState for the stage in which the control transfer stopped.

2.8.45 12Eh H_CH0ConditionCode (Host Channel 0 Condition Code)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Host	11Eh	H_CH0ConditionCode		7:	0:	1:	00h
			R	6: ConditionCode[2]	Condition Code		
				5: ConditionCode[1]			
				4: ConditionCode[0]			
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

This indicates channel CH0 transfer completion results for host operations.

Bit7 Reserved

Bit6-4 ConditionCode[2:0]

This indicates results when the transfer is complete using channel CH0.

Code	Meaning	Description
000	NOERROR	Transaction completed without error.
001	STALL	Endpoint returned Stall PID.
010	DATAOVERRUN	• Data packet received exceeding maximum packet size. * Handled as a retry error if CRC error and bit stuffing error are detected simultaneously. • Data size received exceeding IRP (TotalSize). * Handled as a retry error if CRC error and bit stuffing error are detected simultaneously. * Handled as a toggle mismatch and not a data overrun if the data packet is smaller than maximum packet size and the data toggle included in the data packet does not match the expected value.
011	DATAUNDERRUN	• A data packet smaller than maximum packet size is received, and data size is smaller than IRP (TotalSize). * Handled as a retry error if CRC error and bit stuffing error are detected simultaneously.
100	RETRYERROR	• Device does not respond to token (IN) or does not send a handshake (OUT) within the specified timeframe. • Data packet from endpoint includes CRC error. • Data packet from endpoint includes bit stuffing error. • PID inspection bit from endpoint failed in data PID (IN) or handshake (OUT). • PID received is invalid or PID value is undefined. • Data toggle included in data packet from endpoint does not match the expected value (toggle mismatch).
Other	Reserved	

Bit3-0 Reserved

2.8.46 130h H_CHaConfig_0 (Host Channel a Configuration0)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Host	130h	H_CHaConfig_0	R/W	7: ACK_Cnt[3]	ACK Count		00h
				6: ACK_Cnt[2]			
				5: ACK_Cnt[1]			
				4: ACK_Cnt[0]			
			R/W	3: SpeedMode[1]	Speed Mode		
				2: SpeedMode[0]			
			R/W	1: Toggle	0: Toggle0	1: Toggle1	
			R/W	0: TranGo	0: Stand by	1: Transaction Start	

This sets the basic channel CHa settings for host operations.

Bit7-4 ACK_Cnt[3:0]

This sets the ACK count number for transfers performed using channel CHa.

The H_CHaIntStat register TranACK bit is set once the set ACK number is counted.

0000: 16 ACK counts

0001 to 1111: 1 to 15 ACK counts

When the bulk-only support function is running, only data transport transactions are counted. CBW and CSW transport transactions are not counted.

Bit3-2 SpeedMode[1:0]

This sets the operating mode for the device performing the transfer using channel CHa.

00: HS mode – Use this setting for HS device.

01: FS mode – Use this setting for FS device.

10-11: Reserved – Use of this value is prohibited.

Bit1 Toggle

This sets the initial value of the toggle sequence bit when starting a transaction. It also indicates the toggle sequence bit state after the transaction has been run or completed.

0: Toggle 0

1: Toggle 1

This bit does not need to be set when using the bulk-only support function.

Bit0 TranGo

Setting this bit to “1” initiates the channel CHa transaction. The transaction process can be stopped after being started by clearing the bit to “0.” This bit also indicates whether a transaction is running using channel CHa.

0: Stops transaction (Transaction is stopped)

1: Starts transaction (Transaction is running)

The H_CHaIntStat register TotalSizeCmp bit is set to “1” as soon as the number of bytes set in registers H_CHaTotalSize_HH to LL have been transferred. This bit returns automatically to “0”. It is reset to “0” if the H_CHaIntStat register ChangeCondition bit has been set. The cause is set to the H_CHaConditionCode register in this case to permit review.

The H_CHaIntStat register ChangeCondition bit is set as soon as the transaction underway ends when stopped by clearing the bit. The data in the FIFO, (remaining) total size, and channel-related settings remain unchanged even when the transaction is halted. The transaction can be resumed from the point at which it was stopped by setting this bit to “1” once again. (To perform a new transaction, clear the FIFO and reset the channel information.)

This bit does not need to be set when using the bulk-only support function.

2.8.47 131h H_CHaConfig_1 (Host Channel a Configuration1)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Host	131h	H_CHaConfig_1	R/W	7: TID[1]	Transaction ID		00h
				6: TID[0]			
				5:	0:	1:	
				4:	0:	1:	
			R/W	3: AutoZerolen	0: Do nothing	1: Add Zerolen	
				2:	0:	1:	
				1:	0:	1:	
			R/W	0: TotalSizeFree	0: Do nothing	1: Total Size Free	

This sets the basic channel CHa settings for host operations.

Bit7-6 TID[1:0]

This sets transaction type (OUT/IN) to be issued using channel CHa. This bit setting is disabled when the H_BO_SupportControl register BO_SupportGo bit is set to "1" and the transaction is initiated.

00: Reserved – Use of this value is prohibited.

01: OUT – Issues an OUT token.

10: IN – Issues an IN token.

11: Reserved – Use of this value is prohibited.

This bit does not need to be set when using the bulk-only support function.

Bit5-4 Reserved

Bit3 AutoZerolen

Setting this bit to "1" automatically adds a zero-length packet after the size set in the H_CHaTotalSizeHH to LL registers ends at exactly the Max Packet Size. This bit is enabled only for OUT transfers.

Bit2-1 Reserved

Bit0 TotalSizeFree

Setting this bit to "1" cancels any restrictions on transfer size regardless of the H_CHaTotalSizeHH to LL register settings.

2.8.48 132h H_CHaMaxPktSize_H (Host Channel a Max Packet Size High)

2.8.49 133h H_CHaMaxPktSize_L (Host Channel a Max Packet Size Low)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Host	132h	H_CHaMaxPktSize_H		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
			R/W	1: MaxPktSize[9]	Max Packet Size High		
				0: MaxPktSize[8]			

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Host	133h	H_CHaMaxPktSize_L	R/W	7: MaxPktSize[7]	Max Packet Size Low	00h
				6: MaxPktSize[6]		
				5: MaxPktSize[5]		
				4: MaxPktSize[4]		
				3: MaxPktSize[3]		
				2: MaxPktSize[2]		
				1: MaxPktSize[1]		
				0: MaxPktSize[0]		

These set the channel CHa MaxPacketSize for host operations.

132h.Bit7-2 Reserved

Do not write “1” to reserved bits.

132h.Bit1-0, 133h.Bit7-0 MaxPktSize[9:0]

These set the channel CHa MaxPacketSize.

Set to one of the following.

FS: 8, 16, 32, 64 bytes (32 or 64 bytes when using bulk-only support function)

HS: 512 bytes

All other settings are prohibited.

2.8.50 134h H_CHaTotalSize_HH (Host Channel a Total Size High-High)

2.8.51 135h H_CHaTotalSize_HL (Host Channel a Total Size High-Low)

2.8.52 136h H_CHaTotalSize_LH (Host Channel a Total Size Low-High)

2.8.53 137h H_CHaTotalSize_LL (Host Channel a Total Size Low-Low)

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Host	134h	H_CHaTotalSize_HH	R/W	7: TotalSize[31]	Total Size High-High	00h
				6: TotalSize[30]		
				5: TotalSize[29]		
				4: TotalSize[28]		
				3: TotalSize[27]		
				2: TotalSize[26]		
				1: TotalSize[25]		
				0: TotalSize[24]		

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Host	135h	H_CHaTotalSize_HL	R/W	7: TotalSize[23]	Total Size High-Low	00h
				6: TotalSize[22]		
				5: TotalSize[21]		
				4: TotalSize[20]		
				3: TotalSize[19]		
				2: TotalSize[18]		
				1: TotalSize[17]		
				0: TotalSize[16]		

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Host	136h	H_CHaTotalSize_LH	R/W	7: TotalSize[15]	Total Size Low-High	00h
				6: TotalSize[14]		
				5: TotalSize[13]		
				4: TotalSize[12]		
				3: TotalSize[11]		
				2: TotalSize[10]		
				1: TotalSize[9]		
				0: TotalSize[8]		

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Host	137h	H_CHaTotalSize_LL	R/W	7: TotalSize[7]	Total Size Low-Low	00h
				6: TotalSize[6]		
				5: TotalSize[5]		
				4: TotalSize[4]		
				3: TotalSize[3]		
				2: TotalSize[2]		
				1: TotalSize[1]		
				0: TotalSize[0]		

These set the Total Size of the data transferred using channel CHa for host operations.

134h.Bit7-0, 135h.Bit7-0, 136h.Bit7-0, 137h.Bit7-0 TotalSize[31:0]

These set the total number of bytes of the data to be transferred using channel CHa (max. 4,294,967,295 bytes = approx. 4 Gbytes).

This register is updated each time a transaction ends normally after the transaction has been started by the H_CHaConfig_0 register TranGo bit.

When TotalSize = 0, a zero-length packet is sent when an OUT transaction is started by the H_CHaConfig_0 register TranGo bit.

This register does not need to be set when using the bulk-only support function.

2.8.54 138h H_CHaHubAdrs (Host Channel a Hub Address)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Host	138h	H_CHaHubAdrs	R/W	7: HubAdrs[3]	Hub Address		00h
				6: HubAdrs[2]			
				5: HubAdrs[1]			
				4: HubAdrs[0]			
				3:	0:	1:	
			R/W	2: Port[2]	Channel a Port Number		
				1: Port[1]			
				0: Port[0]			

This sets hub connecting to channel CHa for host operations.

Bit7-4 HubAdrs[3:0]

This sets the USB address of the hub to which the function performing the transfer using channel CHa connects.

It can be set as desired to any value from 0 to 15.

Bit3 Reserved

Bit2-0 Port[2:0]

This sets the port number of the hub to which the function performing the transfer using channel CHa connects.

It can be set as desired to any value from 0 to 7.

2.8.55 139h H_CHaFuncAdrs (Host Channel a Function Address)

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Host	139h	H_CHaFuncAdrs	R/W	7: FuncAdrs[3]	Function Address	00h
				6: FuncAdrs[2]		
				5: FuncAdrs[1]		
				4: FuncAdrs[0]		
			R/W	3: EP_Number[3]	Endpoint Number	
				2: EP_Number[2]		
				1: EP_Number[1]		
				0: EP_Number[0]		

This sets the address of the function performing the transfer using channel CHa for host operations.

Bit7-4 FuncAdrs[3:0]

This sets the USB address of the function including the endpoint managed by channel CHa.

It can be set as desired to any value from 0 to 15.

Bit3-0 EP_Number[3:0]

This sets the endpoint number for the transfer using channel CHa.

It can be set as desired to any value from 0 to 15.

This bit does not need to be set when using the bulk-only support function.

2.8.56 13Ah H_CHaBO_SupporotControl (Host Bulk Only Transfer Supporot Control)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Host	13Ah	H_CHaBO_Supporot-Control		7:	0:	1:	00h
				6:	0:	1:	
			R/W	5: BO_TransportState[1]	Bulk Only Transfer Transport State		
				4: BO_TransportState[0]			
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
			R/W	0: BO_SupportGo	0: Stand by	1: BO Transfer Go	

This sets the channel CHa bulk-only support function for host operations.

Bit7-6 Reserved

Bit5-4 BO_TransportState[1:0]

This sets the BO_SupportGo bit to "1" and indicates which transport is being run during a transfer using the bulk-only support function.

00: Idle – Indicates the transfer has not been started or is completed normally.

01: CBW Transport – Indicates the CBW transport is running.

10: Data Transport – Indicates the data transport is running.

11: CSW Transport – Indicates the CSW transport is running.

Bit3-1 Reserved

Bit0 BO_SupportGo

Setting this bit to "1" automatically initiates CBW transport (and data transport) to CSW transport with the bulk-only support function using channel CHa.

In CBW transport, an OUT token is automatically issued, and the data set in the FIFO CBW area is sent.

If a subsequent data stage exists, the data transport will be run automatically with the specified direction and size.

Finally, in CSW transport, an IN token is automatically sent and data is received in the FIFO CSW area.

The H_CHaIntStat register BO_SupportCmp bit is set once the above transports end normally. If a packet error is detected midway during the transport or if the CSW value is incorrect, the H_CHaIntStat register BO_SupportStop bit is set and the transaction stops. In this case, the cause is set to the H_CHaConditionCode register to permit review. If the ConditionCode value is "000" when the H_CHaIntStat register BO_SupportStop bit is set to "1", the CSW value is incorrect.

This bit is automatically cleared once the transport sequence ends (either normally or in error).

The transport can be stopped by clearing this bit while running the bulk-only support function. The BO_SupportCmp bit is set if the CSW transport ends normally here, and the BO_SupportStop bit is set in all other cases. Refer to BO_TransportState for the transport which was stopped.

2.8.57 13Bh H_CHaBO_CSW_RcvDataSize (Host CSW Receive Data Size)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Host	13Bh	H_CHaBO_CSW_RcvDataSize		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
			R	3: CSW_RcvDataSize[3]	CSW Resceive Data Size		
				2: CSW_RcvDataSize[2]			
				1: CSW_RcvDataSize[1]			
				0: CSW_RcvDataSize[0]			

This indicates the data size received when running CSW transport using the channel CHa bulk-only support function for host operations.

Bit7-4 Reserved

Bit3-0 CSW_RcvDataSize[3:0]

This indicates the CSW received data size.

The amount of data received can be checked using this register when less than 13 bytes of data are received in CSW transport.

This register value has no meaning if a handshake is received in CSW transport or for other than CSW transport.

2.8.58 13Ch H_CHaBO_OUT_EP_Control (Host OUT Endpoint Control)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Host	13Ch	H_CHaBO_OUT_EP_Control		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
			R/W	4: OUT_Toggle	0: Toggle0	1: Toggle1	
			R/W	3: OUT_EP_Number[3]	OUT EP Number		
				2: OUT_EP_Number[2]			
				1: OUT_EP_Number[1]			
				0: OUT_EP_Number[0]			

This sets the channel CHa bulk-only support function for host operations.

Bit7-5 Reserved

Bit4 OUT_Toggle

This sets the H_CHaBO_SupportControl register BO_SupportGo bit to "1" and sets the initial value for the toggle sequence bit for OUT-direction transfer (CBW transport or Data OUT transport) using the bulk-only support function.

0: Toggle 0

1: Toggle 1

The toggle sequence bit is automatically retained at this bit if the OUT-direction transport ends normally.

Bit3-0 OUT_EP_Number[3:0]

This sets the H_CHaBO_SupportControl register BO_SupportGo bit to "1" and sets the endpoint number of the transfer destination device for OUT-direction transfer (CBW transport or Data OUT transport) using the bulk-only support function.

It can be set as desired to any value from 0 to 15.

2.8.59 13Dh H_CHaBO_IN_EP_Control (Host IN Endpoint Control)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Host	13Dh	H_CHaBO_IN_EP_Control		7:	0:	1:	00h
				6:	0:	1:	
				5:	0:	1:	
			R/W	4: IN_Toggle	0: Toggle0	1: Toggle1	
			R/W	3: IN_EP_Number[3]	IN EP Number		
				2: IN_EP_Number[2]			
				1: IN_EP_Number[1]			
				0: IN_EP_Number[0]			

This sets channel CHa bulk-only support function for host operations.

Bit7-5 Reserved

Bit4 IN_Toggle

This sets the H_CHaBO_SupportControl register BO_SupportGo bit to "1" and sets the initial value for the toggle sequence bit for IN-direction transfer (CBW transport or Data IN transport) using the bulk-only support function.

0: Toggle 0

1: Toggle 1

The toggle sequence bit is automatically retained at this bit if the IN-direction transport ends normally.

Bit3-0 IN_EP_Number[3:0]

This sets the H_CHaBO_SupportControl register BO_SupportGo bit to "1" and sets the endpoint number of the transfer destination device for IN-direction transfer (CSW transport or Data IN transport) using the bulk-only support function.

It can be set as desired to any value from 0 to 15.

2.8.60 13Eh H_CHaConditionCode (Host Channel a Condition Code)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Host	13Eh	H_CHaConditionCode		7:	0:	1:	00h
			R	6: ConditionCode[2]	Condition Code		
				5: ConditionCode[1]			
				4: ConditionCode[0]			
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

This indicates channel CHa transfer completion results for host operations.

Bit7 Reserved

Bit6-4 ConditionCode[2:0]

This indicates results when the transfer is completed using channel CHa.

Code	Meaning	Description
000	NOERROR	Transaction completed without error.
001	STALL	Endpoint returned Stall PID.

010	DATAOVERRUN	• Data packet received exceeding maximum packet size.
		  * Handled as a retry error if CRC error and bit stuffing error are detected simultaneously.
		• Data size received exceeding IRP (TotalSize).
		  * Handled as a retry error if CRC error and bit stuffing error are detected simultaneously.
		  * Handled as a toggle mismatch rather than data overrun if data packet is smaller than maximum packet size and data toggle included in data packet does not match the expected value.

011	DATAUNDERRUN	• Data packet smaller than maximum packet size is received and data size is smaller than IRP (TotalSize). * Handled as a retry error if CRC error and bit stuffing error are detected simultaneously.
100	RETRYERROR	• Device does not respond to a token (IN) or does not send a handshake (OUT) within the specified timeframe. • Data packet from endpoint includes CRC error. • Data packet from endpoint includes bit stuffing error. • PID inspection bit from endpoint failed in data PID (IN) or handshake (OUT). • PID received is invalid or PID value is undefined. • Data toggle included in data packet from endpoint does not match the expected value (toggle mismatch).
Other	Reserved	

Bit3-0 Reserved

2.8.61 140h H_CHbConfig_0 (Host Channel b Configuration0)

2.8.62 150h H_CHcConfig_0 (Host Channel c Configuration0)

2.8.63 160h H_CHdConfig_0 (Host Channel d Configuration0)

2.8.64 170h H_CHeConfig_0 (Host Channel e Configuration0)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Host	140h 150h 160h 170h	H_CHbConfig_0 H_CHcConfig_0 H_CHdConfig_0 H_CHeConfig_0	R/W	7: ACK_Cnt[3]	ACK Count		00h
				6: ACK_Cnt[2]			
				5: ACK_Cnt[1]			
				4: ACK_Cnt[0]			
			R/W	3: SpeedMode[1]	Speed Mode		
				2: SpeedMode[0]			
			R/W	1: Toggle	0: Toggle0	1: Toggle1	
			R/W	0: TranGo	0: Stand by	1: Transaction Start	

This sets the basic channel CHx{x=b-e} settings for host operations.

Bit7-4 ACK_Cnt[3:0]

This sets the ACK count number for transfers performed using channel CHx{x=b-e}.

The H_CHx{x=b-e}IntStat register TranACK bit is set once the ACK number set is counted.

0000: 16 ACK counts

0001 to 1111: 1 to 15 ACK counts

Bit3-2 SpeedMode[1:0]

This sets the operating mode for the device performing the transfer using channel CHx{x=b-e}.

00: HS mode – Use this setting for HS device.

01: FS mode – Use this setting for FS device.

10: Reserved – Use of this value is prohibited.

11: LS mode – Use this setting for LS device.

Bit1 Toggle

This sets the initial value of the toggle sequence bit when starting a transaction. It also indicates the toggle sequence bit state after the transaction has been run or completed.

0: Toggle 0

1: Toggle 1

Bit0 TranGo

Setting this bit to “1” initiates the channel CHx{x=b-e} transaction. The transaction process can be stopped after being started by clearing the bit to “0.” This bit also indicates whether a transaction is running using channel CHx{x=b-e}.

0: Stops transaction (Transaction is stopped)

1: Starts transaction (Transaction is running)

The H_CHx{x=b-e}IntStat register TotalSizeCmp bit is set to “1” as soon as the number of bytes set in registers H_CHx{x=b-e}TotalSize_HH to LL have been transferred. This bit returns automatically to “0”. It is reset to “0” if the H_CHx{x=b-e}IntStat register ChangeCondition bit has been set. In this case, the cause is set to the H_CHx{x=b-e}ConditionCode register to permit review.

The H_CHx{x=b-e}IntStat register ChangeCondition bit is set as soon as the transaction underway ends when stopped by clearing the bit. The data in the FIFO, (remaining) total size and channel-related settings remain unchanged, even when the transaction is halted. The transaction can be resumed from the point at which it was stopped by setting this bit to “1” once again. (To perform a new transaction, clear the FIFO and reset the channel information.)

2.8.65 141h H_CHbConfig_1 (Host Channel b Configuration1)

2.8.66 151h H_CHcConfig_1 (Host Channel c Configuration1)

2.8.67 161h H_CHdConfig_1 (Host Channel d Configuration1)

2.8.68 171h H_CHeConfig_1 (Host Channel e Configuration1)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Host	141h 151h 161h 171h	H_CHbConfig_1 H_CHcConfig_1 H_CHdConfig_1 H_CHeConfig_1	R/W	7: TID[1]	Transaction ID		00h
				6: TID[0]			
			R/W	5: TranType[1]	Transfer Type		
				4: TranType[0]			
			R/W	3: AutoZerolen	0: Do nothing	1: Add Zerolen	
			R/W	2: Audio441	0: Disable	1: Enable	
			R/W	1: TotalSizeFree[1]	TotalSizeFree[1:0]		
				0: TotalSizeFree[0]			

This sets the basic channel CHx{x=b-e} settings for host operations.

Bit7-6 TID[1:0]

This sets the transaction type (OUT/IN) to be issued using channel CHx{x=b-e}.

00: Reserved – Use of this value is prohibited.

01: OUT – Issues an OUT token.

10: IN – Issues an IN token.

11: Reserved – Use of this value is prohibited.

Bit5-4 TranType[1:0]

Sets the transfer type performed using channel CHx{x=b-e}.

00: Reserved – Use of this value is prohibited.

01: Isochronous – Uses isochronous transfer.

10: Bulk – Uses bulk transfer.

11: Interrupt – Uses interrupt transfer.

Bit3 AutoZerolen

Setting this bit to "1" automatically adds a zero-length packet if the size of the data set in the registers H_CHx{x=b-e}TotalSizeHH to LL and transferred was exactly the same as the MaxPacketSize. This bit is enabled only for OUT transfers.

Bit2 Audio441

Setting this bit to “1” enables the audio-class assist function. The audio-class assist function is used when sending 16-bit 2-channel PCM data with a sampling frequency of 44.1 kHz and a 1 ms cycle using isochronous transfers. The size of the data packets sent is automatically adjusted when the audio-class assist function is enabled. In this case, 9 continuous transactions are performed with a data packet size of 176 bytes, followed by one transaction with a data packet size of 180 bytes.

Bit1-0 TotalSizeFree[1:0]

This sets transfer control using the H_CHx{x=b-e}TotalSize_HH to LL registers.

00: The TranGo bit is cleared and the transfer ended when the size set in registers H_CHx{x=b-e}TotalSize_HH to LL has been transferred. In this case, a TotalSizeCmp interrupt occurs. For OUT transfers, the last data packet size is the smaller of MaxPktSize and TotalSize. For IN transfers, the expected last data packet size is the smaller of MaxPktSize and TotalSize.

01: Transfers are performed regardless of the H_CHx{x=b-e}TotalSize_HH to LL register values. For OUT transfers, the data packet size is the MaxPktSize value. For IN transfers, the expected data packet size is the MaxPktSize value. For this setting, the TranGo bit is not cleared, and a TotalSizeCmp interrupt does not occur, even if the size set in registers H_CHx{x=b-e}TotalSize_HH to LL has been transferred.

10 : A TotalSizeCmp interrupt occurs when the size set in registers H_CHx{x=b-e}TotalSize_HH to LL has been transferred. The TranGo bit is not cleared here, but the issuing of transactions for this channel is stopped while the TotalSize value is “0.” For OUT transfers, the last data packet size is the smaller of MaxPktSize and TotalSize. For IN transfers, the expected last data packet size is the smaller of MaxPktSize and TotalSize.

11: Reserved

2.8.69 142h H_CHbMaxPktSize_H (Host Channel b Max Packet Size High)
2.8.70 143h H_CHbMaxPktSize_L (Host Channel b Max Packet Size Low)
2.8.71 152h H_CHcMaxPktSize_H (Host Channel c Max Packet Size High)
2.8.72 153h H_CHcMaxPktSize_L (Host Channel c Max Packet Size Low)
2.8.73 162h H_CHdMaxPktSize_H (Host Channel d Max Packet Size High)
2.8.74 163h H_CHdMaxPktSize_L (Host Channel d Max Packet Size Low)
2.8.75 172h H_CHeMaxPktSize_H (Host Channel e Max Packet Size High)
2.8.76 173h H_CHeMaxPktSize_L (Host Channel e Max Packet Size Low)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Host	142h	H_CHbMaxPktSize_H		7:	0:	1:	00h
	152h	H_CHcMaxPktsize_H		6:	0:	1:	
	162h	H_CHdMaxPktSize_H		5:	0:	1:	
	172h	H_CHeMaxPktSize_H		4:	0:	1:	
				3:	0:	1:	
			R/W	2: MaxPktSize[10]	Max Packet Size High		
				1: MaxPktSize[9]			
				0: MaxPktSize[8]			

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Host	143h	H_CHbMaxPktSize_L	R/W	7: MaxPktSize[7]	Max Packet Size Low	00h
	153h	H_CHcMaxPktSize_L		6: MaxPktSize[6]		
	163h	H_CHdMaxPktSize_L		5: MaxPktSize[5]		
	173h	H_CHeMaxPktSize_L		4: MaxPktSize[4]		
				3: MaxPktSize[3]		
				2: MaxPktSize[2]		
				1: MaxPktSize[1]		
				0: MaxPktSize[0]		

These set the channel CHx{x=b-e} MaxPacketSize for host operations.

142h.Bit7-3 Reserved

Do not write "1" to reserved bits.

142h.Bit2-0, 143h.Bit7-0 MaxPktSize[10:0]

These set the channel CHx{x=b-e} MaxPacketSize.

Set to one of the following when using this channel for bulk transfers.

FS: 8, 16, 32, 64 bytes

HS: 512 bytes

The transfer size can be set as follows when using this channel for interrupt transfers.

LS: Up to 8 bytes

FS: Up to 64 bytes

HS: Up to 512 bytes

The transfer size can be set as follows when using this channel for isochronous transfers.

FS: Up to 1,023 bytes

HS: Up to 1,024 bytes

All other settings are prohibited.

2.8.77 144h H_CHbTotalSize_HH (Host Channel b Total Size High-High)
2.8.78 145h H_CHbTotalSize_HL (Host Channel b Total Size High-Low)
2.8.79 146h H_CHbTotalSize_LH (Host Channel b Total Size Low-High)
2.8.80 147h H_CHbTotalSize_LL (Host Channel b Total Size Low-Low)
2.8.81 154h H_CHcTotalSize_HH (Host Channel c Total Size High-High)
2.8.82 155h H_CHcTotalSize_HL (Host Channel c Total Size High-Low)
2.8.83 156h H_CHcTotalSize_LH (Host Channel c Total Size Low-High)
2.8.84 157h H_CHcTotalSize_LL (Host Channel c Total Size Low-Low)
2.8.85 164h H_CHdTotalSize_HH (Host Channel d Total Size High-High)
2.8.86 165h H_CHdTotalSize_HL (Host Channel d Total Size High-Low)
2.8.87 166h H_CHdTotalSize_LH (Host Channel d Total Size Low-High)
2.8.88 167h H_CHdTotalSize_LL (Host Channel d Total Size Low-Low)
2.8.89 174h H_CHeTotalSize_HH (Host Channel e Total Size High-High)
2.8.90 175h H_CHeTotalSize_HL (Host Channel e Total Size High-Low)
2.8.91 176h H_CHeTotalSize_LH (Host Channel e Total Size Low-High)
2.8.92 177h H_CHeTotalSize_LL (Host Channel e Total Size Low-Low)

See next page.

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Host	144h	H_CHbTotalSize_HH	R/W	7: TotalSize[31]	Total Size High-High	00h
	154h	H_CHcTotalSize_HH		6: TotalSize[30]		
	164h	H_CHdTotalSize_HH		5: TotalSize[29]		
	174h	H_CHeTotalSize_HH		4: TotalSize[28]		
				3: TotalSize[27]		
				2: TotalSize[26]		
				1: TotalSize[25]		
				0: TotalSize[24]		

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Host	145h	H_CHbTotalSize_HL	R/W	7: TotalSize[23]	Total Size High-Low	00h
	155h	H_CHcTotalSize_HL		6: TotalSize[22]		
	165h	H_CHdTotalSize_HL		5: TotalSize[21]		
	175h	H_CHeTotalSize_HL		4: TotalSize[20]		
				3: TotalSize[19]		
				2: TotalSize[18]		
				1: TotalSize[17]		
				0: TotalSize[16]		

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Host	146h	H_CHbTotalSize_LH	R/W	7: TotalSize[15]	Total Size Low-High	00h
	156h	H_CHcTotalSize_LH		6: TotalSize[14]		
	166h	H_CHdTotalSize_LH		5: TotalSize[13]		
	176h	H_CHeTotalSize_LH		4: TotalSize[12]		
				3: TotalSize[11]		
				2: TotalSize[10]		
				1: TotalSize[9]		
				0: TotalSize[8]		

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Host	147h	H_CHbTotalSize_LL	R/W	7: TotalSize[7]	Total Size Low-Low	00h
	157h	H_CHcTotalSize_LL		6: TotalSize[6]		
	167h	H_CHdTotalSize_LL		5: TotalSize[5]		
	177h	H_CHeTotalSize_LL		4: TotalSize[4]		
				3: TotalSize[3]		
				2: TotalSize[2]		
				1: TotalSize[1]		
				0: TotalSize[0]		

These set the Total Size of the data to be sent using channel CHx{x=b-e} for host operations.

1X4h.Bit7-0, 1X5h.Bit7-0, 1X6h.Bit7-0, 1X7h.Bit7-0 TotalSize[31:0]

These set the total number of bytes of the data to be transferred using channel CHx{x=b-e} (max. 4,294,967,295 bytes = approx. 4 Gbytes).

This register is updated each time a transaction ends normally after the transaction has been started by the H_CHx{x=b-e}Config_0 register TranGo bit.

When TotalSize = 0, a zero-length packet is sent when an OUT transaction is started by the H_CHx{x=b-e}Config_0 register TranGo bit.

2.8.93 148h H_CHbHubAdrs (Host Channel b Hub Address)

2.8.94 158h H_CHcHubAdrs (Host Channel c Hub Address)

2.8.95 168h H_CHdHubAdrs (Host Channel d Hub Address)

2.8.96 178h H_CHeHubAdrs (Host Channel e Hub Address)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Host	148h 158h 168h 178h	H_CHbHubAdrs H_CHcHubAdrs H_CHdHubAdrs H_CHeHubAdrs	R/W	7: HubAdrs[3]	Hub Address		00h
				6: HubAdrs[2]			
				5: HubAdrs[1]			
				4: HubAdrs[0]			
				3:	0:	1:	
			R/W	2: Port[2]	Port Number		
				1: Port[1]			
				0: Port[0]			

These set the hub connecting to channel CHx{x=b-e} for host operations.

Bit7-4 HubAdrs[3:0]

This sets the USB address of the hub to which the function performing the transfer using channel CHx{x=b-e} connects.

It can be set as desired to any value from 0 to 15.

Bit3 Reserved

Bit2-0 Port[2:0]

This sets the port number of the hub to which the function performing the transfer using channel CHx{x=b-e} connects.

It can be set as desired to any value from 0 to 7.

2.8.97 149h H_CHbFuncAdrs (Host Channel b Function Address)

2.8.98 159h H_CHcFuncAdrs (Host Channel c Function Address)

2.8.99 169h H_CHdFuncAdrs (Host Channel d Function Address)

2.8.100 179h H_CHeFuncAdrs (Host Channel e Function Address)

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Host	149h 159h 169h 179h	H_CHbFuncAdrs H_CHcFuncAdrs H_CHdFuncAdrs H_CHeFuncAdrs	R/W	7: FuncAdrs[3]	Function Address	00h
				6: FuncAdrs[2]		
				5: FuncAdrs[1]		
				4: FuncAdrs[0]		
			R/W	3: EP_Number[3]	Endpoint Number	
				2: EP_Number[2]		
				1: EP_Number[1]		
				0: EP_Number[0]		

These set the address of the function performing the transfer using channel CHx{x=b-e} for host operations.

Bit7-4 FuncAdrs[3:0]

This sets the USB address of the function including the endpoint managed by channel CHx{x=b-e}.

It can be set as desired to any value from 0 to 15.

Bit3-0 EP_Number[3:0]

This sets the endpoint number for the transfer using channel CHx{x=b-e}.

It can be set as desired to any value from 0 to 15.

2.8.101 14Ah H_CHbInterval_H (Host Channel b Interval High)
2.8.102 14Bh H_CHbInterval_L (Host Channel b Interval Low)
2.8.103 15Ah H_CHcInterval_H (Host Channel c Interval High)
2.8.104 15Bh H_CHcInterval_L (Host Channel c Interval Low)
2.8.105 16Ah H_CHdInterval_H (Host Channel d Interval High)
2.8.106 16Bh H_CHdInterval_L (Host Channel d Interval Low)
2.8.107 17Ah H_CHeInterval_H (Host Channel e Interval High)
2.8.108 17Bh H_CHeInterval_L (Host Channel e Interval Low)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Host	14Ah	H_CHbInterval_H		7:	0:	1:	00h
	15Ah	H_CHcInterval_H		6:	0:	1:	
	16Ah	H_CHdInterval_H		5:	0:	1:	
	17Ah	H_CHeInterval_H		4:	0:	1:	
				3:	0:	1:	
			R/W	2: Interval[10]	Interrupt Transfer Interval High		
				1: Interval[9]			
				0: Interval[8]			

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Host	14Bh	H_CHbInterval_L	R/W	7: Interval[7]	Interrupt Transfer Interval Low	00h
	15Bh	H_CHcInterval_L		6: Interval[6]		
	16Bh	H_CHdInterval_L		5: Interval[5]		
	17Bh	H_CHeInterval_L		4: Interval[4]		
				3: Interval[3]		
				2: Interval[2]		
				1: Interval[1]		
				0: Interval[0]		

These set the interval for interrupt or isochronous transfers using channel CHx{x=b-e} for host operations.

1XAh.Bit7-3 Reserved

1XAh.Bit2-0, 1XBh.Bit7-0 Interval[10:0]

The interrupt and isochronous transfer token issuing interval (cycle) is specified by this register. The last 3 bits specify the interval in microframes (125 µs), while the first 7 bits specify the interval in frames (ms). This register setting is enabled when the H_CHx{x=b-e}Config1 register TranType bit is "11" (interrupt transfer) or when the TranType bit is "01" (isochronous transfer).

The "0" is disabled for this register.

The interval set in this register is also used when resending transactions.

Interval[2:0] µFrame – Specifies the interval in 125 µs units. Set to 1, 2, or 4 microframes. Settings to any other values are prohibited. Interval[10:3] must always be set to "0" when setting this bit.

Interval[10:3] Frame – Specifies the interval in ms units. It can be set to any value from 1 to 255 frames. Interval[2:0] must always be set to "0" when setting this bit.

2.8.109 14Ch H_CHbTranPause(Host Channel b Transaction Pause)

2.8.110 15Ch H_CHcTranPause(Host Channel c Transaction Pause)

2.8.111 16Ch H_CHdTranPause(Host Channel d Transaction Pause)

2.8.112 17Ch H_CHeTranPause(Host Channel e Transaction Pause)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Host	14Ch	H_CHbTranPause		7:	0:	1:	00h
	15Ch	H_CHcTranPause		6:	0:	1:	
	16Ch	H_CHdTranPause		5:	0:	1:	
	17Ch	H_CHeTranPause		4:	0:	1:	
				3:	0:	1:	
				2:	0:	1:	
			R/W	1: EnTranPause	0: Disable	1: Enable	
			R/W	0: TranPause	0: Do nothing	1: Transaction Pause	

Bit7-2 Reserved

Bit1 EnTranPause

Setting this bit to "1" sets the H_CHx{x=b-e}TranPause register TranPause bit to "1" simultaneously if the H_CHx{x=b-e}IntStat register TranACK bit is set.

Bit0 TranPause

The transaction cannot be processed with this channel if this bit is "1," even if the H_CHx{x=b-e}Config_0 TranGo bit has been set to "1." The transfer cycle will be retained if this channel is set to interrupt transfer or isochronous transfer, even if the transaction is not processed, since this bit is "1."

2.8.113 14Eh H_CHbConditionCode (Host Channel b Condition Code)

2.8.114 15Eh H_CHcConditionCode (Host Channel c Condition Code)

2.8.115 16Eh H_CHdConditionCode (Host Channel d Condition Code)

2.8.116 17Eh H_CHeConditionCode (Host Channel e Condition Code)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Host	14Eh	H_CHbConditionCode		7:	0:	1:	00h
	15Eh	H_CHcConditionCode	R	6: ConditionCode[2]	Condition Code		
	16Eh	H_CHdConditionCode		5: ConditionCode[1]			
	17Eh	H_CHeConditionCode		4: ConditionCode[0]			
				3:	0:	1:	
				2:	0:	1:	
				1:	0:	1:	
				0:	0:	1:	

These indicate the channel CHx{x=b-e} transfer completion results for host operations.

Bit7 Reserved

Bit6-4 ConditionCode[2:0]

This indicates the results when the transfer is completed using channel CHx{x=b-e}.

Code	Meaning	Description
000	NoError	Transaction completed without error.
001	Stall	Endpoint returned Stall PID.

010	DataOverrun	• Data packet received exceeding maximum packet size.
		  * Handled as a retry error if CRC error and bit stuffing error are detected simultaneously.
		• Data size received exceeding IRP (TotalSize).
		  * Handled as a retry error if CRC error and bit stuffing error are detected simultaneously.
		  * Handled as a toggle mismatch rather than a data overrun if the data packet is smaller than maximum packet size and the data toggle in the data packet does not match the expected value.

011	DataUnderrun	• Data packet of less than maximum packet size is received, and data size is smaller than IRP (TotalSize). * Handled as a retry error if CRC error and bit stuffing error are detected simultaneously.
100	RetryError	• Device does not respond to token (IN) or does not send handshake (OUT) within specified timeframe. • Data packet from endpoint includes CRC error. • Data packet from endpoint includes bit stuffing error. • PID inspection bit from endpoint failed in data PID (IN) or handshake (OUT). • PID received is invalid or PID value is undefined. • Data toggle included in data packet from endpoint does not match the expected value (toggle mismatch).
110	BufferOverrun	• No transaction was performed because the FIFO free space was smaller than the maximum packet size in an isochronous transfer.
111	BufferUnderrun	• No transaction was performed due to insufficient FIFO valid data in an isochronous transfer.

Bit3-0 Reserved

2.8.117 1B0h H_TriggerFrameNum_H (Host Trigger Frame Number High)

2.8.118 1B1h H_TriggerFrameNum_L (Host Trigger Frame Number Low)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Host	1B0h	H_TriggerFrameNum_H	R/W	7: UseTriggerFrame	0: Don't Use	1: Use	00h
				6:	0:	1:	
				5:	0:	1:	
				4:	0:	1:	
				3:	0:	1:	
			R/W	2: TriggerFrameNum[10]	TriggerFrameNum[10:8]		
				1: TriggerFrameNum[9]			
				0: TriggerFrameNum[8]			

Mode	Address	Register Name	R/W	Bit Symbol	Description	Reset
Host	1B1h	H_TriggerFrameNum_L	R/W	7: TriggerFrameNum[7]	TriggerFrameNum[7:0]	00h
				6: TriggerFrameNum[6]		
				5: TriggerFrameNum[5]		
				4: TriggerFrameNum[4]		
				3: TriggerFrameNum[3]		
				2: TriggerFrameNum[2]		
				1: TriggerFrameNum[1]		
				0: TriggerFrameNum[0]		

These set the frame number set by the H_FrameIntStat.TriggerFrame factor.

1B0h.Bit7 UseTriggerFrame

When this bit is set to "1," H_FrameIntStat.TriggerFrame is set to "1" if the frame number (H_FrameNumber_H,L) and trigger frame number (H_TriggerFrameNum_H,L) match when an SOF token is sent by the host controller.

1B0h.Bit6-3 Reserved

1B0h.Bit2-0, 1B1h.Bit7-0 TriggerFrameNum[10:0]

This specifies the frame number generating an H_FrameIntStat.TriggerFrame interrupt.

2.8.119 1BEh *HTM_Config (Host Transceiver Macro Config)*

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Host	1BEh	***HTM_Config***		7:	0:	1:	XXh
				6:	0:	1:	
			R/W	5: ***HTM_SlopeValue[1]***	HTM Slope Value[1:0]		
			R/W	4: ***HTM_SlopeValue[0]***			
				3:	0:	1:	
				2:	0:	1:	
			R/W	1: ***HTM_TermValue[1]***	HTM Termination Value[1:0]		
			R/W	0: ***HTM_TermValue[0]***			

This register is used to set transceiver macro adjustment values.

Bit7-6 Reserved

Bit5-4 *HTM_SlopeValue[1:0]*

This adjusts the HS transmitter through-rate to one of four levels:

00: Slow

01: ↑

10: ↓

11: Fast

Bit3-2 Reserved

Bit1-0 *HTM_TermValue[1:0]*

This adjusts the HS transfer line termination to one of four levels:

00: High

01: ↑

10: ↓

11: Low

2.8.120 1F5h H_Protect (Host Protect)

Mode	Address	Register Name	R/W	Bit Symbol	Description		Reset
Host	1F5h	H_Protect		7:	Don't change values		XXh
				6:			
				5:			
				4:			
			R/W	3: PortSpeedWrEnb	0: Protect	1: Free	
				2:	Don't change values		
				1:			
				0:			

This register is used to set transceiver macro adjustment values.

Bit7-4 Reserved

Bit3 PortSpeedWrEnb

Setting this to “1” allows H_NegoControl_1.PortSpeed to be overwritten.

Clearing this to “0” protects H_NegoControl_1.PortSpeed from overwriting.

Bit2-0 Reserved

Appendix A IDE_Config_1.Swap Bit Settings

The S2R72C05 internal bus has a big-endian configuration, with the [15:8] side forming the first byte. By comparison, the IDE I/F is little-endian, with the [7:0] side forming the first byte. The S2R72C05 allows connections to be switched between the S2R72C05 internal bus and the IDE I/F and data bus using the IDE_Config_1.Swap bit.

The hardware operations for the IDE_Config_1.Swap bit settings are described below.

A ⇒ B in the table below indicates that B is updated to match value A.

● Data DMA transfer using IDE_Control.IDE_Go bit

Swap	HDD[15:0]	
	IDE read	IDE write
0	HDD [15:0] ⇒ Internal bus [15:0]	Internal bus [15:0] ⇒ HDD [15:0]
1 (initial value)	HDD [15:0] ⇒ {Internal bus [7:0], Internal bus [15:8]}	Internal bus [15:0] ⇒ {HDD [7:0], HDD [15:8]}

● IDE data register access

Swap	IDE data register HDD [15:0]			
	IDE read		IDE write	
	IDE_RegAdrs. IDE_RdReg	IDE_RegConfig. EnAutoStsRd	IDE_RegAdrs. IDE_WrReg	IDE_SeqWrRegControl. IDE_SeqWrReg
0	HDD[15:8] ⇒ IDE_RdRegValue_0 HDD[7:0] ⇒ IDE_RdRegValue_1	None	IDE_WrRegValue_0 ⇒ HDD[15:8] IDE_WrRegValue_1 ⇒ HDD[7:0]	IDE_SeqWrRegValue(1st) ⇒ HDD[7:0] IDE_SeqWrRegValue(2nd) ⇒ HDD[15:8]
1 (initial value)	HDD[15:8] ⇒ IDE_RdRegValue_1 HDD[7:0] ⇒ IDE_RdRegValue_0	None	IDE_WrRegValue_0 ⇒ HDD[7:0] IDE_WrRegValue_1 ⇒ HDD[15:8]	IDE_SeqWrRegValue(1st) ⇒ HDD[15:8] IDE_SeqWrRegValue(2nd) ⇒ HDD[7:0]

● IDE task file register access

Swap	IDE task file register HDD [7:0]			
	IDE read		IDE write	
	IDE_RegAdrs. IDE_RdReg	IDE_RegConfig. EnAutoStsRd	IDE_RegAdrs. IDE_WrReg	IDE_SeqWrRegControl. IDE_SeqWrReg
0	HDD[7:0] ⇒ IDE_RdRegValue_0 HDD[7:0] ⇒ IDE_RdRegValue_1	As above	IDE_WrRegValue_1 ⇒ HDD[7:0]	IDE_SeqWrRegValue ⇒ HDD[7:0]
1 (initial value)	As above	As above	IDE_WrRegValue_0 ⇒ HDD[7:0]	As above

Appendix B Connection to Little-endian CPU

While the S2R72C05 internal bus has a big-endian configuration, with even addresses as the first bytes and odd addresses as the last bytes, this section describes methods for connecting to a little-endian CPU.

<Circuit board>

The little-endian CPU and 72C05 terminal should be connected as per the terminal names for data bus and write control signals. In other words, connect 72C05 CD15 to CD8 to the CPU data bus bits 15 to 8 (first bytes) and connect 72C05 CD7 to CD0 to the CPU data bus bits 7 to 0 (last bits). The write control signal terminals should be connected to high and low in the usual manner.

Note that write signal specifications themselves will differ, based on the CPU used.

<Firmware>

Make the following settings before initializing to operate the LSI with a little-endian CPU.

(1) Set the CPUIF_MODE.CPU_Endian bit to "1"

The CPUIF_MODE register can be accessed only while uninitialized immediately after the LSI has been hard-reset. Accessing the 72C05 while uninitialized will access this register regardless of the address. See "1.8.2 CPUIF Mode Setup" for more information.

Setting CPU_Endian as described above switches the bus first and last bytes and sets the even address as the last bytes and odd address as the first bytes. Note that the odd and even addresses are switched by the CPU_Endian setting for registers with 16-bit significance (those with names ending in *_H, _M, _L, *_HH, *_HL, *_LH, or *_LL).

	Big Endian(CPU_Endian=0)		Little Endian(CPU_Endian=1)	
	Even Address	Odd Address	Even Address	Odd Address
	CD[15:8]	CD[7:0]	CD[7:0]	CD[15:8]
Example 1: 8-bit register	MainIntStat	DeviceIntStat	MainIntStat	DeviceIntStat
Example 2: 8-bit register	FIFO_Rd_0	FIFO_Rd_1	FIFO_Rd_0	FIFO_Rd_1
Example 3: 16-bit register	FIFO_RdRemain_H	FIFO_RdRemain_L	FIFO_RdRemain_L	FIFO_RdRemain_H
Example 4: 16-bit register	DMA0Count_HH	DMA0Count_HL	DMA0Count_HL	DMA0Count_HH
Example 5: 16-bit register	DMA0Count_LH	DMA0Count_LL	DMA0Count_LL	DMA0Count_LH

Using the above settings to switch when initialized enables Char or Short to access all internal registers. They are also set to the appropriate endian for access using the CPU DMAC. (See the table below.)

Example: Accessing FIFO_Rd_0/1 register when receiving data from USB in the sequence 01_02_03_04_05_06

Access by Short	CPU access method			
	Big-endian		Little-endian	
	CD[15:8]	CD[7:0]	CD[15:8]	CD[7:0]
1st	01	02	02	01
2nd	03	04	04	03
3rd	05	06	06	05

For registers larger than Short, use divided access with Short and cast in the CPU memory for use.

Appendix C 1-Port Mode

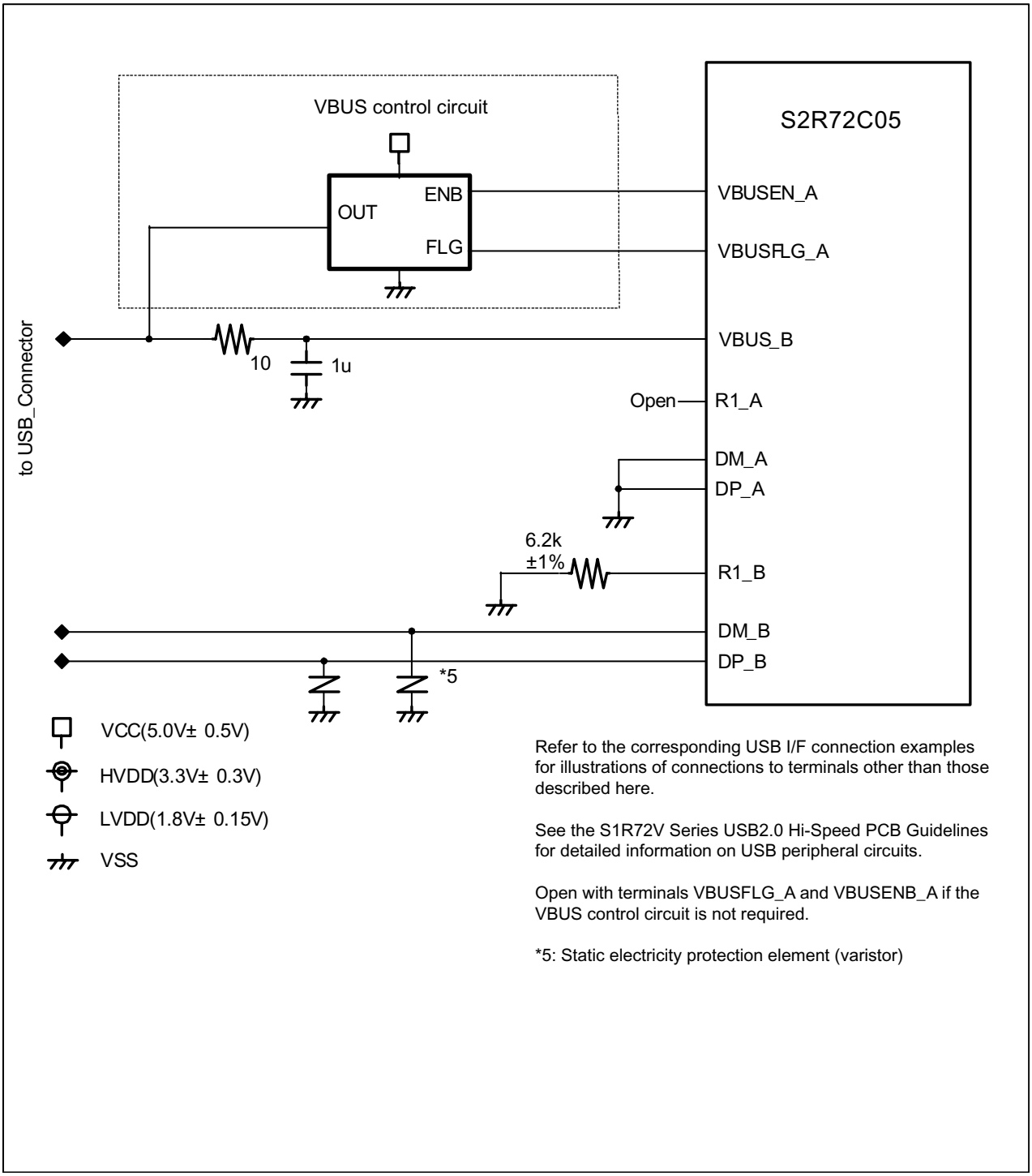
The S2R72C05 can be set to either 2-port mode (ClkSelect.PORT1x2 = “0”) or 1-port mode (ClkSelect.PORT1x2 = “1”) by setting the ClkSelect.PORT1x2 bit in initial setup. In 2-port mode, port A is used for the host function and port B is used for the device function. In 1-port mode, only port B is used for both host and device functions.

If 1-port mode is selected, the free port A should be treated as shown in the diagram below. Also, when using the host function, note that VBUS control signals (VBUSEN_A, VBUSFLG_A) should be used with the VBUS control circuit, regardless of the “_A” at the end.

When using 1-port mode, avoid using the ACT_HOST state and ACT_ALL state power management functions. The PM_Control_0.GoActHost, PM_Control_0.GoActAllDevice, and PM_Control_0.GoActAllHost bits are masked to prevent use in 1-port mode and cannot be set. The ACT_DEVICE state should therefore be used as the power management function in 1-port mode, even when running the host function or device function.

Both the D_Reset.ResetDTM and H_Reset.ResetHTM bits should be set to “0.”

<Typical connection>



Refer to the corresponding USB I/F connection examples for illustrations of connections to terminals other than those described here.

See the S1R72V Series USB2.0 Hi-Speed PCB Guidelines for detailed information on USB peripheral circuits.

Open with terminals VBUSFLG_A and VBUSENB_A if the VBUS control circuit is not required.

*5: Static electricity protection element (varistor)

Appendix D SetAddress Request Response (for TS only)

RcvEP0SETUP interrupt status is not issued on receiving a request if bmRequestType is not 0 (standard request) and bRequest is 0x05 with the automatic address setup function enabled.

This problem arises because the RcvEP0SETUP interrupt status is masked by the bRequest value when the automatic address setup function automatically processes the SetAddress request (bmRequestType==0, bRequest==0x05).

Use one of the following approaches to resolve this problem:

1. Restrict vendor and class requests

   No special procedures are necessary if no vendor requests or class requests in the form bRequest==0x05 are used.

2. Disable the automatic address setup function

   This problem can be resolved by disabling the automatic address setup function. In this case, since the function performing the status stage is automatically disabled after receiving the SetAddress request, the status stage must be performed by the firmware in the same way as for other requests. Note that the setting of address values to the D_USB_Address register can be automated using part of the automatic address setup function (D_USB_Address.SetAddress).

   The settings for disabling the automatic address setup function and the control sequence when the function is disabled are described below. To allow comparison, the control for when the automatic address setup function is enabled is also described.

<Processing to disable the automatic address setup function>

Event/processing	Automatic address setup function = enabled	Automatic address setup function = disabled
(1) Disable automatic address setup function	–	Firmware sets D_ModeControl.SetAddressMode = “1.”

(1) Disable automatic address setup function

Set D_ModeControl.SetAddressMode = “1.”

This setup processing is performed once after resetting and does not need to be set again.

<SetAddress request processing>

Event/processing	Automatic address setup function = enabled	Automatic address setup function = disabled
(1) Receive SetAddress request	–	Hardware issues RcvEP0SETUP interrupt status.
(2) Confirm request	–	Firmware checks using EP0SETUP0 and EP0SETUP1.
(3) Instruct address setting	–	Firmware sets D_USB_Address.SetAddress = "1."
(4) Prepare the status stage response	–	The firmware sets the following. D_SETUP_Control.ProtectEP0 = "0" D_EP0Control.INxOUT = "1" D_EP0ControlIN = "0x40"* * ForceNAK = "0," EnShortPkt = "1"
(5) Perform the status stage	Hardware issues SetAddressCmp interrupt status.	Hardware issues SetAddressCmp interrupt status.

(1) Receive SetAddress request

The hardware issues the RcvEP0SETUP interrupt status on receiving a request.

Notification of receipt of the SETUP transaction is provided by this status in the same way as for other requests, by disabling the automatic address setup function, even for SetAddress requests.

(2) Confirm request

The firmware checks bmRequestType and bRequest using the contents of registers D_EP0SETUP0,1.

A SetAddress request is determined if bmRequest == 0 and bRequest == 0x05.

(3) Instruct address setting

The firmware sets USB_Address.SetAddress = "1."

The address instructed by the SetAddress request is overwritten by the hardware in the D_USB_Address register once the status stage has been completed as a set.

(4) Prepare status stage response

Processing is performed to return a zero-length packet in the same way as for IN-direction status stage for other requests.

- D_SETUP_Control.ProtectEP0 = "0"
- D_EP0Control.INxOUT = "1"
- D_EP0ControlIN = "0x40" (ForceNAK = "0," EnShortPkt = "1")

(5) Perform status stage

The hardware issues SetAddressCmp interrupt status when the status stage (IN transaction) is performed.

Revision History (Rev. 1.00)

Date	Description of revision			
	Rev.	Section (old version)	Class	Details
06/05/2007	1.00	All	New	Newly created.

EPSON

International Sales Operations

AMERICA

EPSON ELECTRONICS AMERICA, INC.
HEADQUARTERS
2580 Orchard Parkway
San Jose , CA 95131,USA
Phone: +1-800-228-3964 FAX: +1-408-922-0238

SALES OFFICES
Northeast
301 Edgewater Place, Suite 210
Wakefield, MA 01880, U.S.A.
Phone: +1-800-922-7667 FAX: +1-781-246-5443

EUROPE

EPSON EUROPE ELECTRONICS GmbH
HEADQUARTERS
Riesstrasse 15
80992 Munich, GERMANY
Phone: +49-89-14005-0 FAX: +49-89-14005-110

ASIA

EPSON (CHINA) CO., LTD.
23F, Beijing Silver Tower 2# North RD DongSanHuan
ChaoYang District, Beijing, CHINA
Phone: +86-10-6410-6655 FAX: +86-10-6410-7320

SHANGHAI BRANCH
7F, High-Tech Bldg., 900, Yishan Road,
Shanghai 200233, CHINA
Phone: +86-21-5423-5522 FAX: +86-21-5423-5512

EPSON HONG KONG LTD.
20/F., Harbour Centre, 25 Harbour Road
Wanchai, Hong Kong
Phone: +852-2585-4600 FAX: +852-2827-4346
Telex: 65542 EPSCO HX

EPSON Electronic Technology Development (Shenzhen) LTD.
12/F, Dawning Mansion, Keji South 12th Road,
Hi- Tech Park, Shenzhen
Phone: +86-755-2699-3828 FAX: +86-755-2699-3838

EPSON TAIWAN TECHNOLOGY & TRADING LTD.
14F, No. 7, Song Ren Road,
Taipei 110
Phone: +886-2-8786-6688 FAX: +886-2-8786-6660

EPSON SINGAPORE PTE., LTD.
1 HarbourFront Place,
#03-02 HarbourFront Tower One, Singapore 098633
Phone: +65-6586-5500 FAX: +65-6271-3182

SEIKO EPSON CORPORATION
KOREA OFFICE
50F, KLI 63 Bldg., 60 Yoido-dong
Youngdeungpo-Ku, Seoul, 150-763, KOREA
Phone: +82-2-784-6027 FAX: +82-2-767-3677

GUMI OFFICE
2F, Grand B/D, 457-4 Songjeong-dong,
Gumi-City, KOREA
Phone: +82-54-454-6027 FAX: +82-54-454-6093

SEIKO EPSON CORPORATION
SEMICONDUCTOR OPERATIONS DIVISION

IC Sales Dept.
IC International Sales Group
421-8, Hino, Hino-shi, Tokyo 191-8501, JAPAN
Phone: +81-42-587-5814 FAX: +81-42-587-5117

Document Code: 410609603
First Issue June 2007
Printed July 2007 in JAPAN