



USB 3.0 Super-Speed Graphics Controller with VGA, HDMI/DVI, and Digital RGB Interfaces

PRODUCT FEATURES

Databook

Highlights

- Single-Chip Super-Speed USB 3.0 Graphics Adapter
- USB 3.0 and 2.0 Device Controllers with Integrated USB 3.0 and 2.0 PHYs
- Highly Efficient Compression Algorithm
- Supports Uncompressed HD Quality Content in USB 3.0 Mode
- HDMI/DVI Display Connectivity via Integrated HDMI/DVI Controller/PHY
- VGA Display Connectivity via Integrated Video DAC
- Support for External Display Interface IC's via Digital RGB Interface
- High Performance DDR2 SDRAM Controller with Integrated DDR2 PHY

Target Applications

- USB to Video Adapters
- Docking Stations, USB Port Replicators
- Thin Clients
- USB Monitors and Projectors
- Embedded Systems

Features

- USB 3.0 and 2.0 Device Controllers
 - Fully compliant with Universal Serial Bus Specification Revision 3.0
 - Operates in SS (5 Gbps) and HS (480 Mbps) modes
 - Supports Control, Bulk-Out, and Interrupt-In endpoints
 - Supports vendor specific commands
 - Integrated USB 3.0 and 2.0 PHYs
 - Integrated USB termination pull-up/pull-down resistors
 - Short circuit protection of USB differential signals

- Graphics Subsystem
 - Integrated HDMI/DVI Controller and PHY
 - Complies with DVI specification v1.0
 - Complies with HDMI specification v1.3
 - S/PDIF and I²S inputs for HDMI audio (2-channel uncompressed PCM)
 - Master I²C interface for DDC connection
 - Integrated Triple 10-bit Video DAC with VGA output
 - Digital RGB Interface
 - 12/15-bit double data rate digital RGB
 - 24-bit single data rate digital RGB
 - Supports up to 2048x1152 (QWXGA) with 32-bit color
 - 8-bit and 16-bit color support
 - Supports display cloning and extending
 - Standard and wide screen aspect ratios
 - Complies with VESA auto display identification
 - Gamma correction
 - Color Look-Up Table (CLUT)
 - Triple-buffered animations
 - Graphics Engine
 - Optimized algorithms for static and dynamic content
 - I²C controller
- DDR2 SDRAM Controller
 - 16-bit data bus, 13-bit address bus
 - JEDEC DDR2 compliant (JESD79-2E)
 - Integrated DDR2 SDRAM PHY
- Power
 - Reduced power operating modes
 - Supports bus-powered and self-powered operation
- Miscellaneous Features
 - Optional EEPROM controller
 - IEEE 1149.1 (JTAG) boundary scan TAP controller
- Software
 - Microsoft Windows® XP/Vista/7 drivers
- Packaging & Environmental
 - 225-ball LFBGA, lead-free RoHS compliant package
 - Commercial temperature range (0°C to +70°C)



Order Number:

UFX7000-VE FOR 225-BALL LFBGA Pb-FREE ROHS COMPLIANT PACKAGE (0 TO +70°C TEMP RANGE)

This product meets the halogen maximum concentration values per IEC61249-2-21

For RoHS compliance and environmental information, please visit www.smSC.com/rohs



80 ARKAY DRIVE, HAUPPAUGE, NY 11788 (631) 435-6000, FAX (631) 273-3123

Copyright © 2013 SMSC or its subsidiaries. All rights reserved.

Circuit diagrams and other information relating to SMSC products are included as a means of illustrating typical applications. Consequently, complete information sufficient for construction purposes is not necessarily given. Although the information has been checked and is believed to be accurate, no responsibility is assumed for inaccuracies. SMSC reserves the right to make changes to specifications and product descriptions at any time without notice. Contact your local SMSC sales office to obtain the latest specifications before placing your product order. The provision of this information does not convey to the purchaser of the described semiconductor devices any licenses under any patent rights or other intellectual property rights of SMSC or others. All sales are expressly conditional on your agreement to the terms and conditions of the most recently dated version of SMSC's standard Terms of Sale Agreement dated before the date of your order (the "Terms of Sale Agreement"). The product may contain design defects or errors known as anomalies which may cause the product's functions to deviate from published specifications. Anomaly sheets are available upon request. SMSC products are not designed, intended, authorized or warranted for use in any life support or other application where product failure could cause or contribute to personal injury or severe property damage. Any and all such uses without prior written approval of an Officer of SMSC and further testing and/or modification will be fully at the risk of the customer. Copies of this document or other SMSC literature, as well as the Terms of Sale Agreement, may be obtained by visiting SMSC's website at <http://www.smSC.com>. SMSC is a registered trademark of Standard Microsystems Corporation ("SMSC"). Product names and company names are the trademarks of their respective holders.

SMSC DISCLAIMS AND EXCLUDES ANY AND ALL WARRANTIES, INCLUDING WITHOUT LIMITATION ANY AND ALL IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE, AND AGAINST INFRINGEMENT AND THE LIKE, AND ANY AND ALL WARRANTIES ARISING FROM ANY COURSE OF DEALING OR USAGE OF TRADE. IN NO EVENT SHALL SMSC BE LIABLE FOR ANY DIRECT, INCIDENTAL, INDIRECT, SPECIAL, PUNITIVE, OR CONSEQUENTIAL DAMAGES; OR FOR LOST DATA, PROFITS, SAVINGS OR REVENUES OF ANY KIND; REGARDLESS OF THE FORM OF ACTION, WHETHER BASED ON CONTRACT; TORT; NEGLIGENCE OF SMSC OR OTHERS; STRICT LIABILITY; BREACH OF WARRANTY; OR OTHERWISE; WHETHER OR NOT ANY REMEDY OF BUYER IS HELD TO HAVE FAILED OF ITS ESSENTIAL PURPOSE, AND WHETHER OR NOT SMSC HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Table of Contents

Chapter 1 General Terms and Conventions

| | | |
|-----|---------------------------------|----|
| 1.1 | General Terms | 14 |
| 1.2 | Register Nomenclature | 16 |

Chapter 2 Overview

| | | |
|------|-------------------------------------|----|
| 2.1 | USB Controllers and PHYs | 17 |
| 2.2 | Graphics Engine (GPH) | 18 |
| 2.3 | DDR2 Controller and PHY | 20 |
| 2.4 | Display Controller (DISP) | 20 |
| 2.5 | Pixel PLL (PIX) | 20 |
| 2.6 | System PLL (SYS) | 20 |
| 2.7 | Video DAC (DAC) | 21 |
| 2.8 | Audio Controller (AUDIO) | 21 |
| 2.9 | HDMI/DVI Controller | 21 |
| 2.10 | I2C | 21 |
| 2.11 | EEPROM | 21 |
| 2.12 | TAP Controller | 21 |
| 2.13 | Performance | 21 |
| 2.14 | System Memory Map | 22 |

Chapter 3 Pin Description and Configuration

| | | |
|-----|---------------------------|----|
| 3.1 | Pin Assignments | 40 |
| 3.2 | Buffer Types | 43 |

Chapter 4 USB Controller

| | | |
|-------|---|----|
| 4.1 | Overview | 44 |
| 4.2 | USB Control Endpoint | 44 |
| 4.2.1 | USB Vendor Commands | 44 |
| 4.3 | USB Bulk Out Endpoint | 47 |
| 4.4 | USB Interrupt Endpoint | 48 |
| 4.5 | USB Descriptors | 49 |
| 4.5.1 | USB Descriptors - UFX7000 | 49 |
| 4.6 | Control and Status Registers | 67 |
| 4.6.1 | USB Configuration Register (USB_CFG) | 68 |
| 4.6.2 | USB Interrupt Endpoint Status Register (USB_INT_STS) | 70 |
| 4.6.3 | USB Interrupt Endpoint Control Register (USB_INT_CTL) | 71 |
| 4.6.4 | USB PHY Parameters 0 Register (USB_PHY_PARAMS0) | 72 |
| 4.6.5 | USB PHY Parameters 1 Register (USB_PHY_PARAMS1) | 74 |
| 4.6.6 | USB PHY MPLL Control Register (USB_PHY_MPLL_CTL) | 75 |

Chapter 5 Clocks and Power Management

| | | |
|-------|--------------------------------------|----|
| 5.1 | Overview | 76 |
| 5.2 | Power States | 76 |
| 5.2.1 | Power State Definitions | 76 |
| 5.2.2 | VBUS Detection | 77 |
| 5.2.3 | Power State Mapping Tables | 78 |
| 5.3 | Clock Generation | 79 |
| 5.4 | Pixel and System PLLs | 81 |
| 5.4.1 | Overview | 81 |
| 5.4.2 | PLL Programming | 81 |
| 5.4.3 | PLL Programming Examples | 82 |

| | | |
|-------|--|----|
| 5.4.4 | DDR2 Clock Frequency Considerations | 82 |
| 5.5 | USB Suspend | 83 |
| 5.6 | USB Resume | 86 |
| 5.7 | LPM Support | 87 |
| 5.8 | U0, U1, U2, U3 States | 87 |
| 5.9 | Control and Status Registers | 88 |
| 5.9.1 | Pixel Clock PLL Control Register (PIX_PLL_CTL) | 89 |
| 5.9.2 | Pixel Clock PLL Stage 0 Configuration Register (PIX_PLL0_CFG) | 91 |
| 5.9.3 | Pixel Clock PLL Stage 1 Configuration Register (PIX_PLL1_CFG) | 92 |
| 5.9.4 | System Clock PLL Control Register (SYS_PLL_CTL) | 93 |
| 5.9.5 | System Clock PLL Stage 0 Configuration Register (SYS_PLL0_CFG) | 95 |
| 5.9.6 | System Clock PLL Stage 1 Configuration Register (SYS_PLL1_CFG) | 96 |
| 5.9.7 | CPM Control Register (CPM_CTL) | 97 |

Chapter 6 Reset Generator

| | | |
|-----|-----------------------|----|
| 6.1 | Description | 98 |
|-----|-----------------------|----|

Chapter 7 Memory Controller

| | | |
|--------|--|-----|
| 7.1 | Overview | 101 |
| 7.2 | Description of Controller Functions | 101 |
| 7.2.1 | Address Mapper | 101 |
| 7.2.2 | ODT Controls | 103 |
| 7.2.3 | Refresh Controls | 104 |
| 7.2.4 | DLL Calibration | 105 |
| 7.2.5 | Power Saving Features | 106 |
| 7.2.6 | DRAM Initialization Sequence | 108 |
| 7.3 | DDR2 Controller Configuration | 109 |
| 7.3.1 | Auto Initialization | 109 |
| 7.3.2 | Configuration Example | 110 |
| 7.4 | Operating Modes | 111 |
| 7.4.1 | Moving To Power Down | 111 |
| 7.4.2 | Moving To Self Refresh | 111 |
| 7.4.3 | List of Dynamic Registers | 112 |
| 7.4.4 | Refresh Related Signals | 112 |
| 7.4.5 | Changing Clock Frequencies | 112 |
| 7.5 | Control and Status Registers | 113 |
| 7.5.1 | DDR2 Dynamic Register (DDR_DYN_REG) | 114 |
| 7.5.2 | DDR2 Bank/CS Address Mapping Register (DDR_AD_MAP_BA_CS) | 115 |
| 7.5.3 | DDR2 Column Address Mapping Register (DDR_AD_MAP_COL) | 116 |
| 7.5.4 | DDR2 Row Address Mapping Register (DDR_AD_MAP_ROW) | 119 |
| 7.5.5 | DDR2 Timing Register 0 (DDR_TIMING_0) | 121 |
| 7.5.6 | DDR2 Timing Register 1 (DDR_TIMING_1) | 122 |
| 7.5.7 | DDR2 Timing Register 2 (DDR_TIMING_2) | 123 |
| 7.5.8 | DDR2 Timing Register 3 (DDR_TIMING_3) | 124 |
| 7.5.9 | DDR2 Timing Register 4 (DDR_TIMING_4) | 125 |
| 7.5.10 | DDR2 Refresh Register (DDR_REFRESH) | 126 |
| 7.5.11 | DDR2 Control Register 0 (DDR_CONTROL_0) | 127 |
| 7.5.12 | DDR2 Control Register 1 (DDR_CONTROL_1) | 129 |
| 7.5.13 | DDR2 Control Register 2 (DDR_CONTROL_2) | 130 |
| 7.5.14 | DDR2 Control Register 3 (DDR_CONTROL_3) | 131 |
| 7.5.15 | DDR2 Control Register 4 (DDR_CONTROL_4) | 132 |
| 7.5.16 | DDR2 Control Register 5 (DDR_CONTROL_5) | 133 |
| 7.5.17 | DDR2 Control Register 6 (DDR_CONTROL_6) | 134 |
| 7.5.18 | DDR2 Control Register 7 (DDR_CONTROL_7) | 135 |
| 7.5.19 | DDR2 Control Register 8 (DDR_CONTROL_8) | 136 |

| | | |
|--------|---|-----|
| 7.5.20 | DDR2 Control Register 9 (DDR_CONTROL_9) | 137 |
| 7.5.21 | DDR2 Control Register 10 (DDR_CONTROL_10) | 138 |
| 7.5.22 | DDR2 Control Register 11 (DDR_CONTROL_11) | 139 |
| 7.5.23 | DDR2 Control Register 12 (DDR_CONTROL_12) | 140 |
| 7.5.24 | DDR2 Control Register 13 (DDR_CONTROL_13) | 141 |
| 7.5.25 | DDR2 PHY Control Register 0 (DDR_PHY_CTL_0) | 142 |
| 7.5.26 | DDR2 PHY Control Register 1 (DDR_PHY_CTL_1) | 144 |
| 7.5.27 | DDR2 PHY Debug Register 0 (DDR_PHY_DEBUG_0) | 145 |
| 7.5.28 | DDR2 PHY Debug Register 1 (DDR_PHY_DEBUG_1) | 146 |
| 7.5.29 | DDR2 PHY Debug Register 2 (DDR_PHY_DEBUG_2) | 147 |
| 7.5.30 | DDR2 PHY Debug Register 3 (DDR_PHY_DEBUG_3) | 148 |
| 7.5.31 | DDR2 PHY Debug Register 4 (DDR_PHY_DEBUG_4) | 149 |
| 7.5.32 | DDR2 PHY Debug Register 5 (DDR_PHY_DEBUG_5) | 150 |
| 7.5.33 | DDR2 PHY Debug Register 6 (DDR_PHY_DEBUG_6) | 151 |
| 7.5.34 | DDR2 PHY Debug Register 7 (DDR_PHY_DEBUG_7) | 152 |
| 7.5.35 | DDR2 PHY Debug Register 8 (DDR_PHY_DEBUG_8) | 153 |

Chapter 8 EEPROM

| | | |
|-------|---|-----|
| 8.1 | Description | 154 |
| 8.1.1 | Command Register Format | 155 |
| 8.1.2 | Data Register Format | 155 |
| 8.2 | EEPROM Format | 155 |
| 8.3 | EEPROM Defaults | 163 |
| 8.4 | Customized Operation Without EEPROM | 164 |
| 8.4.1 | Initialization of SCSR Elements in Lieu of EEPROM Load | 164 |
| 8.4.2 | Attribute Register Initialization | 165 |
| 8.4.3 | Descriptor RAM Initialization | 165 |
| 8.4.4 | Enable Descriptor RAM and Attribute Registers as Source | 168 |
| 8.4.5 | Inhibit Reset of Select SCSR Elements | 168 |
| 8.5 | Control and Status Registers | 169 |
| 8.5.1 | EEPROM Command Register (E2P_CMD) | 170 |
| 8.5.2 | EEPROM Data Register (E2P_DATA) | 173 |

Chapter 9 HDMI

| | | |
|-----|--------------------|-----|
| 9.1 | Overview | 174 |
|-----|--------------------|-----|

Chapter 10 Display Controller

| | | |
|---------|---|-----|
| 10.1 | Introduction | 175 |
| 10.2 | Block Diagram | 175 |
| 10.3 | Configuration | 175 |
| 10.3.1 | Controller Reset / Enable / Status | 175 |
| 10.3.2 | Horizontal and Vertical Timing Configuration | 176 |
| 10.3.3 | Display Blanking | 178 |
| 10.3.4 | Display Frame Buffer Processing | 179 |
| 10.3.5 | Display Data Interpretation | 179 |
| 10.3.6 | Display Cursor | 181 |
| 10.3.7 | Combining Cursor and Display Frame Data | 184 |
| 10.3.8 | Color Lookup/Gamma Correction Via The Color Lookup Table | 185 |
| 10.3.9 | Border and Blanking Insert, Color Lookup/Gamma Correction | 185 |
| 10.3.10 | Interface Control | 186 |
| 10.4 | Display Controller Programming Outline | 187 |
| 10.4.1 | Initialization | 187 |
| 10.4.2 | System Operation | 188 |
| 10.5 | Control and Status Registers | 189 |
| 10.5.1 | Display Control Register (DISPLAY_CTRL) | 190 |

| | | |
|---------|--|-----|
| 10.5.2 | Display Status Register (DISPLAY_STATUS) | 192 |
| 10.5.3 | Display Horizontal Size Register (DISPLAY_H_SIZE) | 194 |
| 10.5.4 | Display Horizontal Blank Register (DISPLAY_H_BLANK) | 195 |
| 10.5.5 | Display Horizontal Sync Register (DISPLAY_H_SYNC) | 196 |
| 10.5.6 | Display Vertical Size Register (DISPLAY_V_SIZE) | 197 |
| 10.5.7 | Display Vertical Blank Register (DISPLAY_V_BLANK) | 198 |
| 10.5.8 | Display Vertical Sync Register (DISPLAY_V_SYNC) | 199 |
| 10.5.9 | Display Frame Buffer Base Address Register (DISPLAY_FRAME_BASE_ADDR) | 200 |
| 10.5.10 | Display Border Color Register (DISPLAY_BORDER_COLOR) | 201 |
| 10.5.11 | Display Frame Buffer Length Register (DISPLAY_FRAME_LEN) | 202 |
| 10.5.12 | Display Frame Base Address Changed Count Register (DISPLAY_FRAME_BASE_ADDR_CHANGED_CNT) | 203 |
| 10.5.13 | Display Interface Control 0 Register (DISPLAY_INTERFACE_CTRL_0) | 204 |
| 10.5.14 | Display Interface Control 1 Register (DISPLAY_INTERFACE_CTRL_1) | 205 |
| 10.5.15 | Display Interface Control 2 Register (DISPLAY_INTERFACE_CTRL_2) | 206 |
| 10.5.16 | Display Cursor Control Register (DISPLAY_CURSOR_CTRL) | 208 |
| 10.5.17 | Display Cursor Status Register (DISPLAY_CURSOR_STATUS) | 210 |
| 10.5.18 | Display Cursor Base Address Register (DISPLAY_CURSOR_BASE_ADDR) | 211 |
| 10.5.19 | Display Cursor Current Base Address Register (DISPLAY_CURSOR_CUR_BASE_ADDR) | 212 |
| 10.5.20 | Display Cursor Position Register (DISPLAY_CURSOR_POSITION) | 213 |
| 10.5.21 | Display Cursor Current Position Register (DISPLAY_CURSOR_CUR_POSITION) | 214 |
| 10.5.22 | Display Cursor Color 0 Register (DISPLAY_CURSOR_COLOR_0) | 215 |
| 10.5.23 | Display Cursor Current Color 0 Register (DISPLAY_CURSOR_CUR_COLOR_0) | 216 |
| 10.5.24 | Display Cursor Color 1 Register (DISPLAY_CURSOR_COLOR_1) | 217 |
| 10.5.25 | Display Cursor Current Color 1 Register (DISPLAY_CURSOR_CUR_COLOR_1) | 218 |
| 10.5.26 | Display Cursor Color 2 Register (DISPLAY_CURSOR_COLOR_2) | 219 |
| 10.5.27 | Display Cursor Current Color 2 Register (DISPLAY_CURSOR_CUR_COLOR_2) | 220 |
| 10.5.28 | Display Cursor Color 3 Register (DISPLAY_CURSOR_COLOR_3) | 221 |
| 10.5.29 | Display Cursor Current Color 3 Register (DISPLAY_CURSOR_CUR_COLOR_3) | 222 |
| 10.5.30 | Display Color Lookup Table (DISPLAY_CLUT) | 223 |

Chapter 11 Graphics Engine

| | | |
|--------|---|-----|
| 11.1 | Overview | 224 |
| 11.2 | FIFO (FCT) Interface | 225 |
| 11.3 | Command Parser | 225 |
| 11.3.1 | Error Checking and Handling | 225 |
| 11.3.2 | Aborting Command | 226 |
| 11.3.3 | Graphics Engine Commands | 226 |
| 11.3.4 | Command Parser Implemented Commands | 226 |
| 11.4 | Direct RAM Write Co-processor | 228 |
| 11.4.1 | Error Checking and Handling | 229 |
| 11.4.2 | Aborting Command | 229 |
| 11.5 | BLT Co-processor | 229 |
| 11.5.1 | Error Checking and Handling | 234 |
| 11.5.2 | Aborting Command | 234 |
| 11.5.3 | Writing to Display Frame Buffer | 235 |
| 11.5.4 | DRAW_FILLED_RECTANGLE | 235 |
| 11.5.5 | DRAW_RAW_RECTANGLE | 235 |
| 11.5.6 | ROP_RECTANGLE | 236 |
| 11.5.7 | ROP_RAW_RECTANGLE | 237 |
| 11.6 | JPEG Co-processor | 238 |
| 11.6.1 | Error Checking and Handling | 241 |
| 11.6.2 | Aborting Command | 241 |
| 11.6.3 | Block Diagram | 241 |

| | | |
|----------|--|-----|
| 11.6.4 | FCT Reader / FIFO | 242 |
| 11.6.5 | Command Reader | 242 |
| 11.6.6 | Data Feeder | 242 |
| 11.6.7 | JPEG Decoder | 242 |
| 11.6.8 | YCbCr Interpolation and Conversion | 243 |
| 11.7 | Hextile Co-processor | 243 |
| 11.7.1 | Error Checking and Handling | 244 |
| 11.7.2 | Aborting Command | 245 |
| 11.7.3 | Block Diagram | 245 |
| 11.7.4 | FCT Reader / FIFO | 246 |
| 11.7.5 | Hextile Stream Processor | 246 |
| 11.8 | JPEG-LS Co-processor | 247 |
| 11.8.1 | Error Checking and Handling | 249 |
| 11.8.2 | Aborting Command | 249 |
| 11.8.3 | Block Diagram | 250 |
| 11.8.4 | Command Reading | 250 |
| 11.8.5 | Data Feeding | 250 |
| 11.8.6 | Header Processing | 250 |
| 11.8.7 | Data Segment Decoder | 251 |
| 11.8.8 | Writing to Display Frame Buffer | 252 |
| 11.9 | General Error Checking and Handling | 252 |
| 11.10 | Pixel Retriever | 252 |
| 11.10.1 | Pixel Retrieving | 252 |
| 11.10.2 | Pixel Reading | 252 |
| 11.10.3 | Pixel Color Data | 252 |
| 11.10.4 | Error Checking and Handling | 252 |
| 11.11 | Pixel to QWORD Converter | 253 |
| 11.11.1 | Memory Address | 253 |
| 11.11.2 | Pixel Color Data | 253 |
| 11.12 | Control and Status Registers | 253 |
| 11.12.1 | Graphic Engine Control Register (GE_CTRL) | 254 |
| 11.12.2 | Graphic Engine Command Status Register (GE_CMD_STATUS) | 255 |
| 11.12.3 | Graphic Engine Command Info Register (GE_CMD_INFO) | 257 |
| 11.12.4 | Graphic Engine Command Length Register (GE_CMD_LEN) | 258 |
| 11.12.5 | Graphic Engine Command Offset Register (GE_CMD_OFFSET) | 259 |
| 11.12.6 | Graphic Engine Previous Command Info Register (GE_PREV_CMD_INFO) | 260 |
| 11.12.7 | Graphic Engine Previous Command Length Register (GE_PREV_CMD_LEN) | 261 |
| 11.12.8 | Graphic Engine Previous Command Offset Register (GE_PREV_CMD_OFFSET) | 262 |
| 11.12.9 | Graphic Engine Chroma Key High Register (GE_CHROMA_KEY_HIGH) | 263 |
| 11.12.10 | Graphic Engine Chroma Key Low Register (GE_CHROMA_KEY_LOW) | 264 |
| 11.13 | Error Recovery | 265 |
| 11.13.1 | USB 3-Strike Error Recovery | 265 |

Chapter 12 I2C Controller

| | | |
|--------|-------------------------|-----|
| 12.1 | I2C Controller Overview | 267 |
| 12.2 | I2C Sub-Modules | 268 |
| 12.2.1 | Control Block | 268 |
| 12.2.2 | Interrupt Controller | 268 |
| 12.2.3 | I2C Mode Logic | 269 |
| 12.2.4 | I2C Data Control Logic | 269 |
| 12.2.5 | Clock Generator | 269 |
| 12.3 | I2C Terminology | 269 |
| 12.3.1 | I2C Bus Terms | 269 |
| 12.3.2 | I2C Bus Transfer Terms | 270 |
| 12.4 | I2C Operation | 270 |

| | | |
|---------|--|-----|
| 12.4.1 | Slave Mode Operation | 270 |
| 12.4.2 | Master Mode Operation | 272 |
| 12.4.3 | Prioritizing Slave-Transmit Operation Over Master-Receiver Operation | 273 |
| 12.4.4 | Disabling the I2C Controller | 273 |
| 12.4.5 | I2CSCL Frequency Configuration | 274 |
| 12.4.6 | Interrupt Operation | 274 |
| 12.5 | I2C Register Descriptions | 275 |
| 12.5.1 | Control Register (I2C_CONTROL) | 278 |
| 12.5.2 | Target Address Register (I2C_TAR) | 280 |
| 12.5.3 | Slave Address Register (I2C_SAR) | 281 |
| 12.5.4 | Rx/Tx Data Buffer and Command Register (I2C_DATA_CMD) | 282 |
| 12.5.5 | SCL Clock High Count Register (I2C_SCL_HCNT) | 283 |
| 12.5.6 | SCL Clock Low Count Register (I2C_SCL_LCNT) | 284 |
| 12.5.7 | Masked Interrupt Status Register (I2C_MIS) | 285 |
| 12.5.8 | Interrupt Mask Register (I2C_INTR_MASK) | 286 |
| 12.5.9 | Raw Interrupt Status Register (I2C_RIS) | 287 |
| 12.5.10 | Receive FIFO Threshold Register (I2C_RX_TL) | 289 |
| 12.5.11 | Transmit FIFO Threshold Register (I2C_TX_TL) | 290 |
| 12.5.12 | Clear Interrupt Register (I2C_CLR_INTR) | 291 |
| 12.5.13 | Clear RX_UNDER Interrupt Register (I2C_CLR_RX_UNDER) | 292 |
| 12.5.14 | Clear RX_OVER Interrupt Register (I2C_CLR_RX_OVER) | 293 |
| 12.5.15 | Clear TX_OVER Interrupt Register (I2C_CLR_TX_OVER) | 294 |
| 12.5.16 | Clear RD_REQ Interrupt Register (I2C_CLR_RD_REQ) | 295 |
| 12.5.17 | Clear TX_ABORT Interrupt Register (I2C_CLR_TX_ABORT) | 296 |
| 12.5.18 | Clear RX_DONE Interrupt Register (I2C_CLR_RX_DONE) | 297 |
| 12.5.19 | Clear ACTIVITY Interrupt Register (I2C_CLR_ACTIVITY) | 298 |
| 12.5.20 | Clear STOP_DET Interrupt Register (I2C_CLR_STOP_DET) | 299 |
| 12.5.21 | Clear START_DET Interrupt Register (I2C_CLR_START_DET) | 300 |
| 12.5.22 | Clear GEN_CALL Interrupt Register (I2C_CLR_GEN_CALL) | 301 |
| 12.5.23 | I2C Enable Register (I2C_ENABLE) | 302 |
| 12.5.24 | Status Register (I2C_STATUS) | 303 |
| 12.5.25 | Transmit FIFO Level Register (I2C_TXFLR) | 305 |
| 12.5.26 | Receive FIFO Level Register (I2C_RXFLR) | 306 |
| 12.5.27 | Transmit Abort Source Register (I2C_TX_ABORT_SRC) | 307 |
| 12.5.28 | SDA Setup Register (I2C_SDA_SETUP) | 309 |
| 12.5.29 | ACK General Call Register (I2C_ACK_GEN_CALL) | 310 |
| 12.5.30 | Enable Status Register (I2C_ENABLE_STATUS) | 311 |
| 12.5.31 | Access Module Register (I2C_ACCESS) | 312 |
| 12.5.32 | I2C Pin Control Register (I2C_PIN) | 314 |
| 12.5.33 | Sequential Data 0 Register (I2C_SEQ_DATA_0) | 315 |
| 12.5.34 | Sequential Data 1 Register (I2C_SEQ_DATA_1) | 316 |
| 12.5.35 | Sequential Data 2 Register (I2C_SEQ_DATA_2) | 317 |
| 12.5.36 | Sequential Data 3 Register (I2C_SEQ_DATA_3) | 318 |
| 12.5.37 | Sequential Data 4 Register (I2C_SEQ_DATA_4) | 319 |
| 12.5.38 | Sequential Data 5 Register (I2C_SEQ_DATA_5) | 320 |
| 12.5.39 | Sequential Data 6 Register (I2C_SEQ_DATA_6) | 321 |
| 12.5.40 | Sequential Data 7 Register (I2C_SEQ_DATA_7) | 322 |
| 12.5.41 | Sequential Data 8 Register (I2C_SEQ_DATA_8) | 323 |
| 12.5.42 | Sequential Data 9 Register (I2C_SEQ_DATA_9) | 324 |
| 12.5.43 | Sequential Data 10 Register (I2C_SEQ_DATA_10) | 325 |
| 12.5.44 | Sequential Data 11 Register (I2C_SEQ_DATA_11) | 326 |
| 12.5.45 | Sequential Data 12 Register (I2C_SEQ_DATA_12) | 327 |
| 12.5.46 | Sequential Data 13 Register (I2C_SEQ_DATA_13) | 328 |
| 12.5.47 | Sequential Data 14 Register (I2C_SEQ_DATA_14) | 329 |
| 12.5.48 | Sequential Data 15 Register (I2C_SEQ_DATA_15) | 330 |

Chapter 13 System Control

| | | |
|--------|--|-----|
| 13.1 | Overview | 331 |
| 13.2 | I2C Selection | 331 |
| 13.3 | Control and Status Registers | 332 |
| 13.3.1 | Device ID (ID_REV) | 333 |
| 13.3.2 | FPGA (FPGA_REV) | 334 |
| 13.3.3 | Hardware Configuration Register (HW_CFG) | 335 |
| 13.3.4 | I2C Interface Selection Register (I2C_SEL) | 337 |

Chapter 14 Miscellaneous

| | | |
|---------|---|-----|
| 14.1 | Overview | 338 |
| 14.2 | LED Configuration | 338 |
| 14.3 | Audio Configuration | 342 |
| 14.3.1 | I2S Audio | 342 |
| 14.3.2 | SDPIF Audio | 342 |
| 14.4 | Control and Status Registers | 343 |
| 14.4.1 | Data Port Select Register (DP_SEL) | 344 |
| 14.4.2 | Data Port Command Register (DP_CMD) | 346 |
| 14.4.3 | Data Port Address Register (DP_ADDR) | 347 |
| 14.4.4 | Data Port Data 0 Register (DP_DATA0) | 348 |
| 14.4.5 | Data Port Data 1 Register (DP_DATA1) | 349 |
| 14.4.6 | FIFO Status Register (FIFO_STATUS) | 350 |
| 14.4.7 | Video DAC Configuration Register (VDAC_CFG) | 351 |
| 14.4.8 | RGB Configuration Register (RGB_CFG) | 352 |
| 14.4.9 | External Reset Configuration Register (EXT_RST_CFG) | 353 |
| 14.4.10 | Audio Configuration Register (AUDIO_CFG) | 354 |
| 14.4.11 | Audio Frequency 0 Register (AUDIO_FREQ0) | 355 |
| 14.4.12 | Audio Frequency 1 Register (AUDIO_FREQ1) | 356 |
| 14.4.13 | HDMI Configuration Register (HDMI_CFG) | 357 |
| 14.4.14 | LED Configuration Register 0 (LED_CFG0) | 358 |
| 14.4.15 | LED Configuration Register 1 (LED_CFG1) | 359 |
| 14.4.16 | LED Configuration Register 2 (LED_CFG2) | 360 |
| 14.4.17 | GPIO Enable Register (GPIO_ENABLE) | 361 |
| 14.4.18 | GPIO Buffer Type Register (GPIO_BUF) | 362 |
| 14.4.19 | GPIO Direction Register (GPIO_DIR) | 363 |
| 14.4.20 | GPIO Data Register (GPIO_DATA) | 364 |
| 14.4.21 | BOS Attributes Register (BOS_ATTR) | 365 |
| 14.4.22 | SS Attributes Register (SS_ATTR) | 366 |
| 14.4.23 | HS Attributes Register (HS_ATTR) | 367 |
| 14.4.24 | FS Attributes Register (FS_ATTR) | 368 |
| 14.4.25 | String Attributes Register 0 (STRNG_ATTR0) | 369 |
| 14.4.26 | String Attributes Register 1 (STRNG_ATTR1) | 370 |

Chapter 15 Operational Characteristics

| | | |
|--------|--------------------------------|-----|
| 15.1 | Absolute Maximum Ratings* | 371 |
| 15.2 | Operating Conditions** | 371 |
| 15.3 | Package Thermal Specifications | 372 |
| 15.4 | Current Consumption | 372 |
| 15.4.1 | SUSPEND Power State | 372 |
| 15.4.2 | Operational | 373 |
| 15.5 | DC Specifications | 374 |
| 15.6 | AC Specifications | 375 |
| 15.6.1 | Power Sequence Timing | 376 |
| 15.6.2 | Power-On Reset Timing | 377 |



| | | |
|--------|------------------------------|-----|
| 15.6.3 | Reset Timing | 378 |
| 15.6.4 | Video DAC Timing | 378 |
| 15.6.5 | Digital RGB Timing | 379 |
| 15.6.6 | EEPROM Timing | 381 |
| 15.6.7 | JTAG Timing | 382 |
| 15.7 | Clock Circuit | 383 |

Chapter 16 Package Outline

| | | |
|------|-----------------------------|-----|
| 16.1 | 225-LFBGA Package | 384 |
|------|-----------------------------|-----|

Chapter 17 Power Connections

Chapter 18 Databook Revision History

List Of Tables

| | | |
|------------|--|-----|
| Table 1.1 | Register Bit Types | 16 |
| Table 2.1 | Graphics Engine Commands | 18 |
| Table 2.2 | Graphics Engine Compression Algorithms | 19 |
| Table 2.3 | UFX7000 Memory Map | 22 |
| Table 3.1 | USB Pins | 24 |
| Table 3.2 | Digital RGB Pins | 25 |
| Table 3.3 | RGB / DDR Mode Mapping Table | 31 |
| Table 3.4 | VDAC Pins | 32 |
| Table 3.5 | DDR2 Memory Pins | 33 |
| Table 3.6 | HDMI Pins | 34 |
| Table 3.7 | EEPROM Pins | 35 |
| Table 3.8 | JTAG Pins | 35 |
| Table 3.9 | Miscellaneous Pins | 36 |
| Table 3.10 | I/O Power Pins, Core Power Pins, and Ground Pins | 38 |
| Table 3.11 | No-Connect Pins | 39 |
| Table 3.12 | UFX7000 Bond Options | 40 |
| Table 3.13 | 225-LFBGA Package Pin Assignments | 40 |
| Table 3.14 | Buffer Types | 43 |
| Table 4.1 | Format of Register Write Setup Stage | 44 |
| Table 4.2 | Format of Register Write Data Stage | 44 |
| Table 4.3 | Format of Register Read Setup Stage | 45 |
| Table 4.4 | Format of Register Read Data Stage | 45 |
| Table 4.5 | Format of Register Read Setup Stage | 45 |
| Table 4.6 | Format of Get Statistics Setup Stage | 46 |
| Table 4.7 | Format of Get Statistics Data Stage | 46 |
| Table 4.8 | Interrupt Packet Format | 48 |
| Table 4.9 | Device Descriptor | 49 |
| Table 4.10 | Configuration Descriptor | 50 |
| Table 4.11 | Interface Descriptor 0 | 51 |
| Table 4.12 | Endpoint 1 Descriptor | 52 |
| Table 4.13 | Super-Speed Endpoint Companion Descriptor (Bulk Out Endpoint) | 53 |
| Table 4.14 | Endpoint 2 Descriptor | 56 |
| Table 4.15 | Super-Speed Endpoint Companion Descriptor (Interrupt Endpoint) | 57 |
| Table 4.16 | Other Speed Configuration Descriptor | 60 |
| Table 4.17 | Device Qualifier Descriptor | 61 |
| Table 4.18 | Binary Device Object Store Descriptor | 62 |
| Table 4.19 | USB 2.0 Extension Descriptor | 63 |
| Table 4.20 | Super-Speed USB Device Capabilities Descriptor | 64 |
| Table 4.21 | LANGID String Descriptor | 66 |
| Table 4.22 | String Descriptor (Indices 1-5) | 66 |
| Table 4.23 | USB Control and Status Register Map | 67 |
| Table 5.1 | Power States | 76 |
| Table 5.2 | Functionality to Power State Mapping - UFX7000 | 78 |
| Table 5.3 | Clocks | 79 |
| Table 5.4 | PLL Frequency Configurations | 82 |
| Table 5.5 | LPM States | 87 |
| Table 5.6 | LFPS States | 87 |
| Table 5.7 | Clocks and Power Management Control and Status Register Map | 88 |
| Table 6.1 | Chip Level Resets | 98 |
| Table 7.1 | Sample Programming Values for DDR2 | 110 |
| Table 7.2 | Memory Controller Control and Status Register Map | 113 |
| Table 8.1 | EEPROM Format | 156 |
| Table 8.2 | Configuration Flags 0 | 159 |



| | | |
|-------------|---|-----|
| Table 8.3 | Configuration Flags 1. | 160 |
| Table 8.4 | Configuration Flags 2. | 161 |
| Table 8.5 | Configuration Flags 3. | 162 |
| Table 8.6 | EEPROM Defaults. | 163 |
| Table 8.7 | USB Control and Status Register Map | 169 |
| Table 10.1 | Display Cursor Status/Control Register Pairs | 181 |
| Table 10.2 | Display Controller Control and Status Register Map | 189 |
| Table 10.3 | Cursor Modes | 209 |
| Table 11.1 | Graphics Engine Control and Status Register Map. | 253 |
| Table 12.1 | Clearing Method of I2C Interrupts | 275 |
| Table 12.2 | I2C Register Map. | 275 |
| Table 13.1 | System Control Register Map | 332 |
| Table 14.1 | Miscellaneous Control and Status Register Map. | 343 |
| Table 14.2 | Data Port Select Mapping Table | 345 |
| Table 15.1 | Package Thermal Parameters. | 372 |
| Table 15.2 | SUSPEND Supply Current | 372 |
| Table 15.3 | Typical Super-Speed Operational Supply Current (mA) | 373 |
| Table 15.4 | Typical High-Speed Operational Supply Current (mA) | 373 |
| Table 15.5 | I/O Buffer Characteristics. | 374 |
| Table 15.6 | Video DAC - DC Characteristics | 375 |
| Table 15.7 | Power-On Timing Values. | 376 |
| Table 15.8 | nRESET Power-On Timing Values | 377 |
| Table 15.9 | nRESET Timing Values. | 378 |
| Table 15.10 | Video DAC - AC Characteristics. | 378 |
| Table 15.11 | Digital RGB Timing Values - DDR Mode | 379 |
| Table 15.12 | Digital RGB Timing Values - SDR Mode | 380 |
| Table 15.13 | EEPROM Timing Values | 381 |
| Table 15.14 | JTAG Timing Values. | 382 |
| Table 15.15 | Crystal Specifications | 383 |
| Table 16.1 | UF7000 225-LFBGA Package Parameters. | 385 |
| Table 18.1 | Customer Revision History | 387 |

List of Figures

| | | |
|-------------|--|-----|
| Figure 2.1 | UFX7000 System Diagram | 17 |
| Figure 3.1 | Pin Assignments (TOP VIEW) | 23 |
| Figure 5.1 | Power States | 77 |
| Figure 5.2 | VBUS Detection | 77 |
| Figure 5.3 | USB Suspend Sequence | 83 |
| Figure 5.4 | Auto Suspend Actions | 85 |
| Figure 5.5 | USB Resume Sequence | 86 |
| Figure 7.1 | ODT Write Timing Diagram | 103 |
| Figure 7.2 | ODT Read Timing Diagram | 104 |
| Figure 8.1 | EEPROM Access Flow Diagram | 155 |
| Figure 8.2 | Descriptor RAM Example | 167 |
| Figure 10.1 | Display Controller Block Diagram | 175 |
| Figure 10.2 | Horizontal Timing | 176 |
| Figure 10.3 | Vertical Timing | 176 |
| Figure 10.4 | Combined Vertical and Horizontal Timing | 178 |
| Figure 10.5 | Quad Word to Pixel Mapping 24 bbp | 180 |
| Figure 10.6 | Quad Word to Pixel Mapping 16 bbp | 180 |
| Figure 10.7 | Quad Word to Pixel Mapping 8 bbp | 181 |
| Figure 11.1 | Graphics Engine Block Diagram | 224 |
| Figure 11.2 | Command Parser Wait for Event Logic | 227 |
| Figure 11.1 | JPEG Co-processor Block Diagram | 242 |
| Figure 11.2 | Hextile Co-processor Block Diagram | 246 |
| Figure 11.3 | JPEG-LS Co-processor Block Diagram | 250 |
| Figure 12.1 | I2C Block Diagram | 268 |
| Figure 13.1 | I2C Interface Selection | 331 |
| Figure 14.1 | LED Initial State Determination | 339 |
| Figure 14.2 | LED State Transitions | 340 |
| Figure 15.1 | Power-On Timing | 376 |
| Figure 15.1 | nRESET Power-On Timing | 377 |
| Figure 15.1 | nRESET Timing | 378 |
| Figure 15.2 | Digital RGB Timing - DDR Mode | 379 |
| Figure 15.3 | Digital RGB Timing - SDR Mode | 380 |
| Figure 15.4 | EEPROM Timing | 381 |
| Figure 15.5 | JTAG Timing | 382 |
| Figure 16.1 | UFX7000 225-LFBGA Package Definition | 384 |
| Figure 16.1 | UFX7000 225-LFBGA Package Ball Detail | 384 |
| Figure 16.2 | UFX7000 225-LFBGA Recommended PCB Land Pattern | 385 |
| Figure 17.1 | Power Connections | 386 |

Chapter 1 General Terms and Conventions

1.1 General Terms

Below is a list of the general terms used in this document:

| | |
|-----------------------------------|--|
| BYTE | 8-bits |
| AL | Posted CAS Additive Latency - Allows READ or WRITE commands to be issued prior to $t_{RCD(min)}$ with the requirement that $AL \leq t_{RCD(min)}$. The READ or WRITE command is held for the time of AL before it is issued internally to the DDR2 SDRAM. |
| BL | Burst Length - Maximum number of column locations that can be accessed with a given WRITE or READ command. |
| CL | CAS Latency - Delay, in clocks cycles, of registering the READ command by the DDR device to the availability of the first bit of the READ DATA. |
| CSR | Control and Status Registers |
| CWL | CAS Write Latency - Delay, in clock cycles, of registering the WRITE command by the DDR device to the registering of the first bit of Write Data. |
| DDC | Display Data Channel |
| DDRC | Double Data Rate Controller |
| DISP | Display Controller Module |
| DTS | Digital Theater Sound |
| DTS-HD | Digital Theater Sound High Definition |
| DVI | Digital Visual Interface |
| DWORD | 32-bits |
| EDID | Extended Display Identification Data |
| FIFO | First In First Out buffer |
| FSM | Finite State Machine |
| GPIO | General Purpose I/O |
| HDCP | High-bandwidth Digital Content Protection |
| HDMI | High-Definition Multimedia Interface |
| HOST | External system (Includes processor, application software, etc.) |
| I2C | Philips Inter-Integrated Circuit (I^2C) bus |
| I2S | Integrated Interchip Sound bus |
| Level-Triggered Sticky Bit | This type of status bit is set whenever the condition that it represents is asserted. The bit remains set until the condition is no longer true, and the status bit is cleared by writing a zero. |
| LFSR | Linear Feedback Shift Register |
| LPM | Link Power Management |

| | |
|-----------------|--|
| N/A | Not Applicable |
| ODT | On Die Termination |
| Packet | In the context of this document a packet refers to transfers on the USB interface. |
| POR | Power on Reset. |
| RESERVED | Refers to a reserved bit field or address. Unless otherwise noted, reserved bits must always be zero for write operations. Unless otherwise noted, values are not guaranteed when reading reserved bits. Unless otherwise noted, do not read or write to reserved addresses. |
| RL | Read Latency - The sum of CL and AL. |
| S/PDIF | Sony Philips Digital Interface |
| SCSR | System Control and Status Registers |
| URX | USB Bulk Out Packet Receiver. |
| USB | Universal Serial Bus |
| UTX | USB Bulk In Packet Transmitter. |
| WL | Write Latency - The sum of CWL and AL. |
| WORD | 16-bits |
| ZLP | Zero Length USB Packet |

1.2 Register Nomenclature

Table 1.1 describes the register bit attributes used throughout this document.

Table 1.1 Register Bit Types

| REGISTER BIT TYPE NOTATION | REGISTER BIT DESCRIPTION |
|----------------------------|--|
| R | Read: A register or bit with this attribute can be read. |
| W | Write: A register or bit with this attribute can be written. |
| RO | Read only: Read only. Writes have no effect. |
| RS | Read to Set: This bit is set on read. |
| WO | Write only: If a register or bit is write-only, reads will return unspecified data. |
| WC | Write One to Clear: writing a one clears the value. Writing a zero has no effect. |
| WAC | Write Anything to Clear: writing anything clears the value. |
| RC | Read to Clear: Contents is cleared after the read. Writes have no effect. |
| LL | Latch Low: Clear on read of register. |
| LH | Latch High: Clear on read of register. |
| SC | Self-Clearing: Contents is self-cleared after the being set. Writes of zero have no effect. Contents can be read. |
| RO/LH | Read Only, Latch High: This mode is used by the Ethernet PHY registers. Bits with this attribute will stay high until the bit is read. After it a read, the bit will remain high, but will change to low if the condition that caused the bit to go high is removed. If the bit has not been read the bit will remain high regardless of if its cause has been removed. |
| NASR | Not Affected by Software Reset. The state of NASR bits does not change on assertion of a software reset. |
| RESERVED | Reserved Field: Reserved fields must be written with zeros to ensure future compatibility. The value of reserved bits is not guaranteed on a read. |

modes. Both UDCs implement three USB endpoints: Bulk-Out (URX), Control (CTL), and Interrupt-In (INT). The received data is packed into 32-bit words which are passed to one of the supported endpoints via UDC's respective native interface.

The UDC 2.0 interfaces to the USB 2.0 PHY via a 8-bit 60 MHz UTMI compliant interface. The UDC 3.0 uses an 32-bit 125 MHz pipe interface to connect to the USB 3.0 PHY.

Both the USB 2.0 and 3.0 PHY include a common block. Depending on the USB version supported, the corresponding common block is enabled. In the interests of minimizing power consumption it is undesirable to always power both common blocks. In either the case, the enabled common block has the additional responsibility for generating the bulk of the system clocks. In USB 2.0 mode, system clocks are typically a multiple of 60 MHz, as opposed to USB 3.0 mode where they are multiples of 62.5 MHz.

The common blocks share a crystal oscillator which accepts a 25 MHz crystal. The oscillator can also accept a single-ended clock input.

The Bulk-Out endpoint is implemented by the URX module. This module is responsible for receiving compressed or uncompressed graphics data and drawing commands from the host.

The URX interfaces with the FIFO Controller (FCT). The FCT manages a 18 KB FIFO. The FIFO uses a store and forward architecture and supports flushing out errored Bulk-Out frames when the URX requests a rewind. In USB 2.0 mode, the USB 2.0 URX interfaces with the FCT via a 64-bit 60 MHz bus. In USB 3.0 mode, the USB 3.0 URX - FCT bus operates at 125 MHz.

The FCT also serves as a mechanism to cross clock domains in USB 2.0 mode, bridging the 60 MHz clock domain where the USB logic primarily operates on 167 MHz domain. In USB 3.0 mode, the read and write interfaces operate at 125 and 167 MHz respectively.

The Control module (CTL) manages traffic to/from the control endpoint that is not handled by the UDC. The CTL is responsible for handling select USB standard commands and all vendor specific commands (e.g., CSR reads and writes). The CTL also provides a direct path to the DDR2 SDRAM.

2.2 Graphics Engine (GPH)

The Graphics Engine module (GPH) receives a stream of commands, along with parameters and data, over the USB Bulk-Out endpoint. It executes the commands and writes the results into the display frame buffer in SDRAM via a DMA engine. The various commands range from simple memory write accesses to various decompression algorithms. The GPH supports the following commands.

Table 2.1 Graphics Engine Commands

| COMMAND NAME | DESCRIPTION |
|-----------------------|--|
| DRAW_FILLED_RECTANGLE | This command writes a rectangle of single solid color into the frame buffer. |
| DRAW_RAW_RECTANGLE | This command writes a rectangle into the frame buffer. The contents of the rectangle are specified in the command. |
| ROP_RECTANGLE | This command reads a source rectangle from the frame buffer, optionally performs a logic operation with the color specified in the command, and writes the rectangle into the display frame buffer. The location of the source and destination rectangles can be the same, overlapping or independent areas. |
| ROP_RAW_RECTANGLE | This command reads a source rectangle from the frame buffer, optionally performs a logic operation with the raw pixel data specified in the command, and writes the rectangle into the display frame buffer. The location of the source and destination rectangles can be the same, overlapping or independent areas. |
| DIRECT_RAM_WRITE | This command writes the data in the command's payload into the frame buffer. |

Table 2.1 Graphics Engine Commands (continued)

| | |
|----------------|--|
| WRITE_CSR | This command allows write access to the device's control and status registers. |
| NOP | This command is used to wait state the graphics engine. |
| WAIT_FOR_EVENT | This command instructs the GPH to pause until the event specified occurs (forced blanking status, base page pending, etc.) |
| DECOMP_HEXTILE | This commands performs hextile decompression on the data in the command's payload. It is ideal for font based applications (PDF, Word processing, etc.) |
| DECOMP_JPEG | This commands performs lossy JPEG decompression on the data in the command's payload. This algorithm is ideal for video. |
| DECOMP_JPEG_LS | This commands performs JPEG-LS decompression on the data in the command's payload. This algorithm is ideal for still images and computer generated graphics (Powerpoint, pictures, gaming, etc.) |
| PAD | This command is used to align subsequent commands to any DWORD boundary, if needed by the driver. Other than ones complement and sequence number checking, this command has no other affect. |
| BYPASS_JPEG | This command is used to accept raw video in YCrCb format and color space convert it into RGB. |

The device is coupled with a host PC which generates the graphics commands. These commands have been optimized for minimal CPU utilization, resulting in minimal negative impact on the user experience when the USB-to-video functionality is enabled. Optimization is particularly relevant for the decompression commands, as many popular compression algorithms are highly CPU intensive. The JPEG-D, JPEG-LS, and Hextile compression algorithms represent a reasonable trade-off between CPU load and compression performance. The expected compression ratios for the supported algorithms are detailed below.

Table 2.2 Graphics Engine Compression Algorithms

| ALGORITHM | APPROXIMATE COMPRESSION RATIO |
|------------|-------------------------------|
| Hextile | 8:1 to 15:1 |
| JPEG Lossy | 15:1 to 20:1 |
| JPEG-LS | 2:1 |

In order to implement the commands, the GPH includes several memories, the largest being 8 KB. The 8KB memory size, which can store 2048 pixels, was selected to enable the GPH to buffer an entire row of the frame buffer. This functionality is needed to execute rectangle copy commands such as ROP_RECTANGLE and ROP_RAW_RECTANGLE.

The GPH receives data via the read interface of the FCT. The GPH uses the FCT's 18 KB FIFO to absorb command processing latency. In the event that the GPH can not keep up with the Bulk-Out traffic, the FIFO detects that it is approaching maximum capacity which will cause the URX to back pressure the USB host. Such a scenario is common when decompressing data with a high compression ratio. The amount of data ultimately written into the frame buffer will greatly exceed the original compressed data received on the Bulk-Out pipe.

It should be noted that the theoretical maximum bandwidth on the USB 2.0 bus is 480 Mbps. Due to overhead, this number will can not be sustained in actual operation. The maximum sustained ingress rate expected to be handled by the GPH is 320 Mbps. In USB 3.0 mode, the compression is less important as there will be over 3 Gbps of bandwidth available to the Bulk Out pipe, but it will still be enabled. Therefore, a larger sustained ingress rate is possible. However, the maximum rate of decompressed data will be equivalent for USB 2.0 and USB 3.0.

The egress FCT interface, and the majority of the GPH, typically operates at 167 MHz, for maximum resolution.

The GPH accesses the frame buffer in SDRAM via a 64-bit 167MHz internal bus which interfaces to the DDR2 controller. The GPH operates off of the System PLL.

2.3 DDR2 Controller and PHY

The device features a DDR2 Memory Controller (DCTL) and integrated DDR2 PHY. The DCTL arbitrates between the GPH, DISP, and CTL modules via the 64-bit 167 MHz internal bus. The arbiter uses strict priority methodology, with the DISP always given highest priority. The DCTL aims to efficiently utilize DDR2 bandwidth when performing read and write burst accesses to DRAM.

The DCTL allows programming of various DDR2 timing parameters: TRTP, TWTR, TRP, TRCD TRAS, TRRD, TWR, TRFC and TREF. CAS latency, additive latency, burst length, and burst type. The pad strengths and pad delays are also programmable.

The DDR2 PHY interfaces to the DCTL via a 333 MHz 32-bit data bus and connects to external SDRAM with a 333 MHz 16-bit data bus. This allows for a total bandwidth, not accounting for refreshes and overhead, of 10.66 Gbps.

2.4 Display Controller (DISP)

The Display Controller (DISP) module implements a CRT/LCD Controller. The DISP reads the frame buffer from the SDRAM, via a DMA engine, and generates the video signals to the various display interfaces: HDMI/DVI, Video DAC, and the Digital RGB Interface.

The DISP is responsible for inserting borders and blanking. It has a programmable video resolution of up to 2048x1152 (QWXGA) and supports up to 60 frames per second (fps). It features 8, 16 and 24-bit color support, a 256 entry color lookup table, gamma correction, and a deep pixel FIFO.

The Pixel FIFO is used to store pixel data read from the SDRAM and helps prevent display underruns by buffering instantaneous bandwidth requirements. Under run prevention also requires the DISP to have priority access to the DCTL. The Pixel FIFO has a total buffer capacity of 3KB. Its ingress interface operates off of the core clock while the egress interface utilizes the pixel clock. The latter is required because the DISP logic that interfaces to the display operates off of the pixel clock.

Another notable component of the DISP is the Video Timing Generator. It is responsible for creating horizontal sync, vertical sync, composite sync, and video blank outputs. The timing of these video outputs is programmable. Additionally, the Video Timing Generator controls the read out timing of the Pixel FIFO.

The DISP provides video content off-chip through several mechanisms. The integrated Digital RGB Interface can be used for connection to an external video scaler or timing controller. Additionally, the DISP interfaces to the internally integrated HDMI/DVI Controller and Video DAC. The DISP can only drive a single video interface at any given point in time.

2.5 Pixel PLL (PIX)

The pixel clock used in the device is generated by the Pixel PLL. In order to minimize system cost, the Pixel PLL shares the 25 MHz crystal used by the USB PHY. The Pixel PLL supports the generation of a 200 MHz maximum pixel clock.

The PLL is programmable and allows support for a pixel clock at lower frequencies.

2.6 System PLL (SYS)

The DDR2 clock and system clock used in the device is generated by the System PLL. In order to minimize system cost, the System PLL shares the 25 MHz crystal used by the USB PHY.

In typical operation the System PLL is configured to provide a 333 MHz DDR2 clock and a 167 MHz system clock. As with the Pixel PLL, this PLL is programmable to support lower frequencies.

2.7 Video DAC (DAC)

The device integrates three 10-bit video DACs that are used to implement an analog RGB VGA interface. The video DACs are capable of running at frequencies up to 200 MHz, allowing for UXGA resolution support and a refresh rate of 60 Hz. The video DACs are driven by the DISP and utilize the pixel clock for timing generation.

2.8 Audio Controller (AUDIO)

The device audio controller forms accepts digital audio data via the S/PDIF input pin. The S/PDIF stream can carry 2-channel uncompressed PCM data (IEC 60958) or a compressed bit stream for multi-channel (IEC 61937) formats. The audio controller forms the audio data into packets according to the HDMI specification. The S/PDIF input supports rates from 32 to 192 kHz. Time stamping is accomplished via a separate master clock input (MCLK). MCLK is coherent with the S/PDIF input (both were created from the same clock source). There is no setup or hold timing requirement on an input with respect to MCLK.

Single channel I2S digital audio data is also accepted by the device. The I2SCLK is obtained from the [I2SCLKALT0](#) pin or the [I2SCLKALT1](#) pin, depending on the setting of the CSRs. The S/PDIF input pin of the chip may alternatively function as I2SDATA, when I2S digital data functionality is enabled. When I2S audio is being processed, the WS pin is used as the word select clock.

2.9 HDMI/DVI Controller

The device integrates an HDMI/DVI Controller which is compliant with revision 1.3 of the HDMI specification. It consists of both a transmitter and PHY. The transmitter is driven by the DISP module. DVI output is also supported.

Note: The HDMI/DVI Controller does not support HDCP.

2.10 I²C

The device's I²C controller features two I²C interfaces. Either one of these interfaces can be used to implement the Display Data Channel (DDC) interface for managing a Video DAC, HDMI, DVI, or external transmitter connected via digital RGB.

2.11 EEPROM

The device provides an integrated EEPROM controller that allows for customizing USB descriptors and configuration flags via an external EEPROM. The EEPROM controller contains a dual port RAM used for descriptor processing by both the USB3.0 and USB2.0 CTL modules.

2.12 TAP Controller

The integrated IEEE 1149.1 compliant TAP controller provides boundary scan via JTAG.

2.13 Performance

The following maximum resolutions are supported by the device.

- 1920x1200, 32-bit color, 60 fps
- 2048x1152, 32-bit color, 60 fps

In order to support these resolutions, the DCTL egress and DISP must be capable of supplying data at a maximum rate of 3.3 Gbps to the external video interfaces. The ingress bandwidth requirements are less, as the targeted update rate for the maximum resolution is 30 fps, or 1.65 Gbps.

For both USB 2.0 and 3.0 modes, the maximum resolution supported is the same. The main advantage of USB 3.0 mode is that the higher USB bus bandwidth makes it less likely that other devices on the shared USB medium would interfere with the video stream. For example, a USB 2.0 video stream can easily be degraded by an Ethernet controller.

The higher bandwidth of USB 3.0 also makes it possible for video to be transmitted without compression. However, uncompressed video transmission is not planned. The less bandwidth the device consumes, the more consistent the performance will be in the presence of other high bandwidth USB devices.

USB 3.0 does enable a larger ingress bandwidth for content that uses JPEG-LS compression. For an application such as computer gaming a rate of approximately 10 fps is possible at the maximum resolution with USB 2.0. USB 3.0 may allow 24 fps, the limiting factor being the capabilities of the JPEG-LS decompression engine.

2.14 System Memory Map

Table 2.3 defines the system memory map.

Table 2.3 UFX7000 Memory Map

| STARTING ADDRESS | ENDING ADDRESS | SIZE (BYTES) | DEVICE | CSR CLOCK DOMAIN Note 2.1 | DESCRIPTION |
|------------------|----------------|--------------|--------------------|------------------------------|-----------------------------|
| 0000h | 0FFFh | 4096 | DDR2 | sys_clk | DDR2 CSR |
| 1000h | 1FFFh | 4096 | I ² C | sys_clk | I ² C CSR |
| 2000h | 2FFFh | 4096 | Display Controller | sys_clk | Display Controller CSR |
| 3000h | 3FFFh | 4096 | System | clk_xtal | System Control CSR |
| 4000h | 4FFFh | 4096 | Graphics Engine | sys_clk | Graphics Engine CSR |
| 5000h | 5FFFh | 4096 | USB | sys_clk | USB Configuration |
| 6000h | 6FFFh | 4096 | EEPROM | clk_xtal | EEPROM CSR |
| 7000h | 7FFFh | 4096 | PMT | clk_xtal | Clocks and Power Management |
| 8000h | 8FFFh | 4096 | Miscellaneous | sys_clk | Miscellaneous CSR |
| 9000h | 9FFFh | 4096 | HDMI | sys_clk | HDMI Transmitter CSR |
| A000h | FFFFh | 24576 | - | sys_clk | RESERVED |

Note 2.1 CSRs on clk_xtal may be accessed before the System PLL is programmed.

Chapter 3 Pin Description and Configuration



































































































































































































































| SMSC UF7000 225-LFBGA TOP VIEW | | | | | | | | | | | | | | | | | |
|---|--|--|--|--|--|---|--|--|--|--|--|---|---|--|--|--|---|
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | | |
| A |  DDRFP0WE_IN |  DDRFP0WE_OUT |  DDRDQ7 |  nDDRDQS0 |  DDRDM0 |  DDRDQ1 |  nDDRWE |  DDRA10 |  DDRA3 |  DDRA9 |  DDRCK |  nDDRCAS |  DDRA2 |  DDRA4 |  DDRA8 | A | |
| B |  DDRVREF0 |  DDRDQ5 |  DDRDQ0 |  DDRDQS0 |  DDRDQ6 |  DDRDQ3 |  DDRCKE |  DDRBA0 |  DDRA7 |  DDRA12 |  nDDRCK |  nDDRCS |  DDRA6 |  DDRA11 |  VSS | B | |
| C |  DDRDQ9 |  DDRDQ11 |  DDRDQ12 |  DDRDQ2 |  VDD18DDR |  DDRDQ4 |  DDRBA1 |  DDRA1 |  DDRA5 |  nDDRRAS |  VDD18DDR |  DDRA0 |  DDRODT |  VSS |  EXTSWING | C | |
| D |  DDRDQS1 |  DDRDM1 |  DDRDQ14 |  VDD18DDR |  VDD18DDR |  DDRVREF2 |  VDD18DDR |  VDD18DDR |  VDD18DDR |  VDD18DDR |  VDD18DDR |  VDD18DDR |  VDD18DDR |  VDD18DDR |  TX2P |  TX2N | D |
| E |  nDDRDQS1 |  DDRDQ8 |  DDRDQ15 |  VDD18DDR |  VDD18DDR |  VDD12CORE |  VSS |  VDD12CORE |  VSS |  VDD12CORE |  VDD18DDR |  VSSHDMI |  VDD12HDMI |  TX1P |  TX1N | E | |
| F |  DDRDQ13 |  DDRDQ10 |  DDRVREF1 |  VDD18DDR |  VDD12CORE |  VSS |  VSS |  VSS |  VSS |  VSS |  VDD12CORE |  VSSHDMI |  VDD12HDMI |  TX0P |  TX0N | F | |
| G |  VSSUSB3 |  VSSUSB3 |  VDD33USB3 |  VSS |  VSS |  VSS |  VSS |  VSS |  VSS |  VSS |  VDD12CORE |  VSSHDMI |  VDD12HDMI |  TXCP |  TXCN | G | |
| H |  USB3TXDP |  USB3TXDM |  VDD12USB3 |  VDD12USB3 |  VDD12CORE |  VSS |  VSS |  VSS |  VSS |  VSS |  VSSVDAC |  IREF |  VSSVDAC |  VDACR |  nVDACR | H | |
| J |  USB3RXDP |  USB3RXDM |  VSSUSB3 |  REXT |  VSS |  VSS |  VSS |  VSS |  VSS |  VSS |  VDD33VDAC |  VDD33VDAC |  VDD33VDAC |  VDACG |  nVDACG | J | |
| K |  I2CSDA1/ GPIO27 |  I2CSDA0 |  I2CSC11/ GPIO28 |  I2CSC10 |  VDD12CORE |  VSS |  VSS |  VSS |  VSS |  VSS |  VDACREF |  VDAC_HS |  VDAC_VS |  VDACB |  nVDACB | K | |
| L |  USBDP |  USBDM |  VBUS_DET |  AUDIO_DIS/ GPIO30 |  VDD33IO |  VDD12CORE |  VSS |  VDD12CORE |  VSS |  VDD12CORE |  VDD33IO |  VDD33IO |  WS/GPIO29 |  MCLK/ GPIO25 |  SPDIF/ I2SDATA/ GPIO26 | L | |
| M |  VDD33USB |  NC |  HPD |  VDATAB4/ VD3/GPIO20 |  VDD33IO |  VDD33IO |  VDD33IO |  VDATAG4/ VD9/GPIO12 |  VDD33IO |  VDATAR5/ GPIO5 |  VDD33IO |  TDI |  INT |  EECS |  EECLK | M | |
| N |  XI |  VDD12USBPLL |  USBRBIAS |  VDATAB3/ VD4/GPIO19 |  VDATAB0/ GPIO16 |  nBLANK |  VDATAG7/ GPIO15 |  VDATAG3/ GPIO11 |  VDATAG0/ VD11/GPIO8 |  VDATAR4/ VD13/GPIO4 |  VDATAR1/ GPIO1 |  TDO |  nEXTRST |  NC |  NC | N | |
| P |  XO |  SYSPLL6 |  VDATAB6/ VD1/GPIO22 |  VDATAB2/ VD5/GPIO18 |  HSYNC |  nVCLK |  VDATAG6/ VD7/GPIO14 |  VDATAG2/ GPIO10 |  VDATAR7/ VD12/GPIO7 |  VDATAR3/ VD14/GPIO3 |  VDATAR0/ I2SCLKALT/ GPIO0 |  TCK |  nSW_MODE |  nRESET |  EEDI | P | |
| R |  SYSPLLP |  VDATAB7/ VD0/GPIO23 |  VDATAB5/ VD2/GPIO21 |  VDATAB1/ VD6/GPIO17 |  VSYNC |  VCLK |  VDATAG5/ VD8/GPIO13 |  VDATAG1/ VD10/GPIO9 |  VDATAR6/ GPIO6 |  VDATAR2/ VD15/GPIO2 |  nTRST |  TMS |  NC |  LED/ I2SCLKALT/ GPIO24 |  EEDO | R | |
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | | |

Figure 3.1 Pin Assignments (TOP VIEW)

Table 3.1 USB Pins

| NUM PINS | NAME | SYMBOL | BUFFER TYPE | DESCRIPTION |
|----------|------------------------------------|----------|-------------|--|
| 1 | USB DMINUS | USBDM | AIO | USB Data Minus. Note: The functionality of this pin may be swapped to USB DPLUS via the Port Swap (CFG0_PORT_SWAP) bit of Configuration Flags 0 of the EEPROM. |
| 1 | USB DPLUS | USBDP | AIO | USB Data Plus. Note: The functionality of this pin may be swapped to USB DMINUS via the Port Swap (CFG0_PORT_SWAP) bit of Configuration Flags 0 of the EEPROM. |
| 1 | External USB Bias Resistor | USBRBIAS | AI | Used for setting HS transmit current level and on-chip termination impedance. Connect to an external 12K 1.0% resistor to ground. |
| 1 | Crystal Input | XI | ICLK | External 25 MHz crystal input. Note: This pin can also be driven by a single-ended clock oscillator. When this method is used, XO should be left unconnected. (Note 3.1) |
| 1 | Crystal Output | XO | OCLK | External 25 MHz crystal output. |
| 1 | USB3 RX DMINUS | USB3RXDM | AIO | Super-Speed Differential Receive Minus. |
| 1 | USB3 RX DPLUS | USB3RXDP | AIO | Super-Speed Differential Receive Plus. |
| 1 | USB3 TX DMINUS | USB3TXDM | AIO | Super-Speed Differential Transmit Minus. |
| 1 | USB3 TX DPLUS | USB3TXDP | AIO | Super-Speed Differential Transmit Plus. |
| 1 | USB3 External Reference Resistance | REXT | AI | Connect to an external 200 ohm 1.0% resistor to ground. |

Note 3.1 A 25MHz oscillator, or other single-ended clock source that meets the specifications in [Section 15.5, "DC Specifications," on page 374](#) and [Section 15.7, "Clock Circuit," on page 383](#), is required when utilizing the Digital RGB interface. Do not use a crystal when operating in Digital RGB mode.

Table 3.2 Digital RGB Pins

| NUM PINS | NAME | SYMBOL | BUFFER TYPE | DESCRIPTION |
|----------|-------------------------------|--------|---|--|
| 1 | Video Clock High | VCLK | RGB | Active high video clock. |
| 1 | Video Clock Low | nVCLK | RGB | Active low MHz video clock. |
| 1 | Horizontal Sync | HSYNC | RGB | Video horizontal synchronization output. |
| 1 | Vertical Sync | VSYNC | RGB | Video vertical synchronization output. |
| 1 | Video Blanking | nBLANK | RGB | Active low video blanking signal. |
| 1 | Blue Pixel Data Channel Bit 7 | VATAB7 | RGB | Blue Pixel Video Data Bit 7, RGB Single Ended Mode. |
| | DDR RGB Data 0 | VD0 | RGB | Used in RGB DDR Mode, refer to Table 3.3 . |
| | General Purpose I/O 23 | GPIO23 | IS/O8/ OD8 (PU) Note 3.2 | This General Purpose I/O pin is fully programmable as either a push-pull output, an open-drain output, or a Schmitt-triggered input. |
| 1 | Blue Pixel Data Channel Bit 6 | VATAB6 | RGB | Blue Pixel Video Data Bit 6, RGB Single Ended Mode. |
| | DDR RGB Data 1 | VD1 | RGB | Used in RGB DDR Mode, refer to Table 3.3 . |
| | General Purpose I/O 22 | GPIO22 | IS/O8/ OD8 (PU) Note 3.2 | This General Purpose I/O pin is fully programmable as either a push-pull output, an open-drain output, or a Schmitt-triggered input. |
| 1 | Blue Pixel Data Channel Bit 5 | VATAB5 | RGB | Blue Pixel Video Data Bit 5, RGB Single Ended Mode. |
| | DDR RGB Data 2 | VD2 | RGB | Used in RGB DDR Mode, refer to Table 3.3 . |
| | General Purpose I/O 21 | GPIO21 | IS/O8/ OD8 (PU) Note 3.2 | This General Purpose I/O pin is fully programmable as either a push-pull output, an open-drain output, or a Schmitt-triggered input. |
| 1 | Blue Pixel Data Channel Bit 4 | VATAB4 | RGB | Blue Pixel Video Data Bit 4, RGB Single Ended Mode. |
| | DDR RGB Data 3 | VD3 | RGB | Used in RGB DDR Mode, refer to Table 3.3 . |
| | General Purpose I/O 20 | GPIO20 | IS/O8/ OD8 (PU) Note 3.2 | This General Purpose I/O pin is fully programmable as either a push-pull output, an open-drain output, or a Schmitt-triggered input. |

Table 3.2 Digital RGB Pins (continued)

| NUM PINS | NAME | SYMBOL | BUFFER TYPE | DESCRIPTION |
|----------|--------------------------------|--------|---|--|
| 1 | Blue Pixel Data Channel Bit 3 | VATAB3 | RGB | Blue Pixel Video Data Bit 3, RGB Single Ended Mode. |
| | DDR RGB Data 4 | VD4 | RGB | Used in RGB DDR Mode, refer to Table 3.3 . |
| | General Purpose I/O 19 | GPIO19 | IS/O8/ OD8 (PU) Note 3.2 | This General Purpose I/O pin is fully programmable as either a push-pull output, an open-drain output, or a Schmitt-triggered input. |
| 1 | Blue Pixel Data Channel Bit 2 | VATAB2 | RGB | Blue Pixel Video Data Bit 2, RGB Single Ended Mode. |
| | DDR RGB Data 5 | VD5 | RGB | Used in RGB DDR Mode, refer to Table 3.3 . |
| | General Purpose I/O 18 | GPIO18 | IS/O8/ OD8 (PU) Note 3.2 | This General Purpose I/O pin is fully programmable as either a push-pull output, an open-drain output, or a Schmitt-triggered input. |
| 1 | Blue Pixel Data Channel Bit 1 | VATAB1 | RGB | Blue Pixel Video Data Bit 1, RGB Single Ended Mode. |
| | DDR RGB Data 6 | VD6 | RGB | Used in RGB DDR Mode, refer to Table 3.3 . |
| | General Purpose I/O 17 | GPIO17 | IS/O8/ OD8 (PU) Note 3.2 | This General Purpose I/O pin is fully programmable as either a push-pull output, an open-drain output, or a Schmitt-triggered input. |
| 1 | Blue Pixel Data Channel Bit 0 | VATAB0 | RGB | Blue Pixel Video Data Bit 0, RGB Single Ended Mode. |
| | General Purpose I/O 16 | GPIO16 | IS/O8/ OD8 (PU) Note 3.2 | This General Purpose I/O pin is fully programmable as either a push-pull output, an open-drain output, or a Schmitt-triggered input. |
| 1 | Green Pixel Data Channel Bit 7 | VDTAG7 | RGB | Green Pixel Video Data Bit 7, RGB Single Ended Mode. |
| | General Purpose I/O 15 | GPIO15 | IS/O8/ OD8 (PU) Note 3.2 | This General Purpose I/O pin is fully programmable as either a push-pull output, an open-drain output, or a Schmitt-triggered input. |

Table 3.2 Digital RGB Pins (continued)

| NUM PINS | NAME | SYMBOL | BUFFER TYPE | DESCRIPTION |
|----------|--------------------------------|---------|---|--|
| 1 | Green Pixel Data Channel Bit 6 | VDATAG6 | RGB | Green Pixel Video Data Bit 6, RGB Single Ended Mode. |
| | DDR RGB Data 7 | VD7 | RGB | Used in RGB DDR Mode, refer to Table 3.3 . |
| | General Purpose I/O 14 | GPIO14 | IS/O8/ OD8 (PU) Note 3.2 | This General Purpose I/O pin is fully programmable as either a push-pull output, an open-drain output, or a Schmitt-triggered input. |
| 1 | Green Pixel Data Channel Bit 5 | VDATAG5 | RGB | Green Pixel Video Data Bit 5, RGB Single Ended Mode. |
| | DDR RGB Data 8 | VD8 | RGB | Used in RGB DDR Mode, refer to Table 3.3 . |
| | General Purpose I/O 13 | GPIO13 | IS/O8/ OD8 (PU) Note 3.2 | This General Purpose I/O pin is fully programmable as either a push-pull output, an open-drain output, or a Schmitt-triggered input. |
| 1 | Green Pixel Data Channel Bit 4 | VDATAG4 | RGB | Green Pixel Video Data Bit 4, RGB Single Ended Mode. |
| | DDR RGB Data 9 | VD9 | RGB | Used in RGB DDR Mode, refer to Table 3.3 . |
| | General Purpose I/O 12 | GPIO12 | IS/O8/ OD8 (PU) Note 3.2 | This General Purpose I/O pin is fully programmable as either a push-pull output, an open-drain output, or a Schmitt-triggered input. |
| 1 | Green Pixel Data Channel Bit 3 | VDATAG3 | RGB | Green Pixel Video Data Bit 3, RGB Single Ended Mode. |
| | General Purpose I/O 11 | GPIO11 | IS/O8/ OD8 (PU) Note 3.2 | This General Purpose I/O pin is fully programmable as either a push-pull output, an open-drain output, or a Schmitt-triggered input. |
| 1 | Green Pixel Data Channel Bit 2 | VDATAG2 | RGB | Green Pixel Video Data Bit 2, RGB Single Ended Mode. |
| | General Purpose I/O 10 | GPIO10 | IS/O8/ OD8 (PU) Note 3.2 | This General Purpose I/O pin is fully programmable as either a push-pull output, an open-drain output, or a Schmitt-triggered input. |

Table 3.2 Digital RGB Pins (continued)

| NUM PINS | NAME | SYMBOL | BUFFER TYPE | DESCRIPTION |
|----------|--------------------------------|---------|---|--|
| 1 | Green Pixel Data Channel Bit 1 | VDATAG1 | RGB | Green Pixel Video Data Bit 1, RGB Single Ended Mode. |
| | DDR RGB Data 10 | VD10 | RGB | Used in RGB DDR Mode, refer to Table 3.3 . |
| | General Purpose I/O 9 | GPIO9 | IS/O8/ OD8 (PU) Note 3.2 | This General Purpose I/O pin is fully programmable as either a push-pull output, an open-drain output, or a Schmitt-triggered input. |
| 1 | Green Pixel Data Channel Bit 0 | VDATAG0 | RGB | Green Pixel Video Data Bit 0, RGB Single Ended Mode. |
| | DDR RGB Data 11 | VD11 | RGB | Used in RGB DDR Mode, refer to Table 3.3 . |
| | General Purpose I/O 8 | GPIO8 | IS/O8/ OD8 (PU) Note 3.2 | This General Purpose I/O pin is fully programmable as either a push-pull output, an open-drain output, or a Schmitt-triggered input. |
| 1 | Red Pixel Data Channel Bit 7 | VDATAR7 | RGB | Red Pixel Video Data Bit 7, RGB Single Ended Mode. |
| | DDR RGB Data 12 | VD12 | RGB | Used in RGB DDR Mode, refer to Table 3.3 . |
| | General Purpose I/O 7 | GPIO7 | IS/O8/ OD8 (PU) Note 3.2 | This General Purpose I/O pin is fully programmable as either a push-pull output, an open-drain output, or a Schmitt-triggered input. |
| 1 | Red Pixel Data Channel Bit 6 | VDATAR6 | RGB | Red Pixel Video Data Bit 6, RGB Single Ended Mode. |
| | General Purpose I/O 6 | GPIO6 | IS/O8/ OD8 (PU) Note 3.2 | This General Purpose I/O pin is fully programmable as either a push-pull output, an open-drain output, or a Schmitt-triggered input. |
| 1 | Red Pixel Data Channel Bit 5 | VDATAR5 | RGB | Red Pixel Video Data Bit 5, RGB Single Ended Mode. |
| | General Purpose I/O 5 | GPIO5 | IS/O8/ OD8 (PU) Note 3.2 | This General Purpose I/O pin is fully programmable as either a push-pull output, an open-drain output, or a Schmitt-triggered input. |

Table 3.2 Digital RGB Pins (continued)

| NUM PINS | NAME | SYMBOL | BUFFER TYPE | DESCRIPTION |
|----------|------------------------------|---------|---|--|
| 1 | Red Pixel Data Channel Bit 4 | VDATAR4 | RGB | Red Pixel Video Data Bit 4, RGB Single Ended Mode. |
| | DDR RGB Data 13 | VD13 | RGB | Used in RGB DDR Mode, refer to Table 3.3 . |
| | General Purpose I/O4 | GPIO4 | IS/O8/ OD8 (PU) Note 3.2 | This General Purpose I/O pin is fully programmable as either a push-pull output, an open-drain output, or a Schmitt-triggered input. |
| 1 | Red Pixel Data Channel Bit 3 | VDATAR3 | RGB | Red Pixel Video Data Bit 3, RGB Single Ended Mode. |
| | DDR RGB Data 14 | VD14 | RGB | Used in RGB DDR Mode, refer to Table 3.3 . |
| | General Purpose I/O 3 | GPIO3 | IS/O8/ OD8 (PU) Note 3.2 | This General Purpose I/O pin is fully programmable as either a push-pull output, an open-drain output, or a Schmitt-triggered input. |
| 1 | Red Pixel Data Channel Bit 2 | VDATAR2 | RGB | Red Pixel Video Data Bit 2, RGB Single Ended Mode. |
| | DDR RGB Data 15 | VD15 | RGB | Used in RGB DDR Mode, refer to Table 3.3 . |
| | General Purpose I/O 2 | GPIO2 | IS/O8/ OD8 (PU) Note 3.2 | This General Purpose I/O pin is fully programmable as either a push-pull output, an open-drain output, or a Schmitt-triggered input. |
| 1 | Red Pixel Data Channel Bit 1 | VDATAR1 | RGB | Red Pixel Video Data Bit 1, RGB Single Ended Mode. |
| | General Purpose I/O 1 | GPIO1 | IS/O8/ OD8 (PU) Note 3.2 | This General Purpose I/O pin is fully programmable as either a push-pull output, an open-drain output, or a Schmitt-triggered input. |

Table 3.2 Digital RGB Pins (continued)

| NUM PINS | NAME | SYMBOL | BUFFER TYPE | DESCRIPTION |
|----------|------------------------------|------------|--|---|
| 1 | Red Pixel Data Channel Bit 0 | VATAR0 | RGB | Red Pixel Video Data Bit 0, RGB Single Ended Mode. |
| | I2S Clock Alternate Input 0 | I2SCLKALT0 | IS | <p>I²S Clock alternate input 0. The I²S clock input pin is selectable between the I2SCLKALT0 or I2SCLKALT1 pins.</p> <p>Note: If the single data rate RGB interface is enabled, I2SCLKALT1 should be used. I2SCLKALT0 should be used in all other cases.</p> <p>Refer to the Audio Configuration Register (AUDIO_CFG) on page 354 for further details.</p> |
| | General Purpose I/O 0 | GPIO0 | IS/O8/OD8 (PU) Note 3.2 | This General Purpose I/O pin is fully programmable as either a push-pull output, an open-drain output, or a Schmitt-triggered input. |

Note: A 25MHz oscillator, or other single-ended clock source that meets the specifications in [Section 15.5, "DC Specifications,"](#) on page 374 and [Section 15.7, "Clock Circuit,"](#) on page 383, is required when utilizing the Digital RGB interface. Do not use a crystal when operating in Digital RGB mode.

Note 3.2 The internal pull-up is disabled when the GPIO is configured as an O8 buffer type.

Table 3.3 RGB / DDR Mode Mapping Table

| SDR (24-BIT MODE) Note 3.3 | DDR NAME | DDR (12-BIT MODE) Note 3.3 | | DDR (15-BIT MODE) | |
|----------------------------------|----------|--|----------------------|---------------------|----------------------|
| | | VCLK RISING EDGE | VCLK FALLING EDGE | VCLK RISING EDGE | VCLK FALLING EDGE |
| VCLK | | | | | |
| nVCLK | | | | | |
| HSYNC | | | | | |
| VSYNC | | | | | |
| nBLANK | | | | | |
| VATAB7 | VD0 | - | - | BLUE0 | GREEN5 |
| VATAB6 | VD1 | - | - | BLUE1 | GREEN6 |
| VATAB5 | VD2 | BLUE0 | GREEN4 | BLUE2 | GREEN7 |
| VATAB4 | VD3 | BLUE1 | GREEN5 | BLUE3 | GREEN8 |
| VATAB3 | VD4 | BLUE2 | GREEN6 | BLUE4 | GREEN9 |
| VATAB2 | VD5 | BLUE3 | GREEN7 | BLUE5 | RED0 |
| VATAB1 | VD6 | BLUE4 | RED0 | BLUE6 | RED1 |
| VATAB0 | - | - | - | - | - |
| VDATAG7 | - | - | - | - | - |
| VDATAG6 | VD7 | BLUE5 | RED1 | BLUE7 | RED2 |
| VDATAG5 | VD8 | BLUE6 | RED2 | BLUE8 | RED3 |
| VDATAG4 | VD9 | BLUE7 | RED3 | BLUE9 | RED4 |
| VDATAG3 | - | - | - | - | - |
| VDATAG2 | - | - | - | - | - |
| VDATAG1 | VD10 | - | - | GREEN0 | RED5 |
| VDATAG0 | VD11 | - | - | GREEN1 | RED6 |
| VDATAR7 | VD12 | GREEN0 | RED4 | GREEN2 | RED7 |
| VDATAR6 | - | - | - | - | - |
| VDATAR5 | - | - | - | - | - |
| VDATAR4 | VD13 | GREEN1 | RED5 | GREEN3 | RED8 |
| VDATAR3 | VD14 | GREEN2 | RED6 | GREEN4 | RED9 |
| VDATAR2 | VD15 | GREEN3 | RED7 | - | - |
| VDATAR1 | - | - | - | - | - |
| VDATAR0 | - | - | - | - | - |

Note 3.3 For each color channel, the eight bits come from the upper bits of the [Display Color Lookup Table \(DISPLAY_CLUT\)](#).

Table 3.4 VDAC Pins

| NUM PINS | NAME | SYMBOL | BUFFER TYPE | DESCRIPTION |
|----------|------------------------------|------------|-------------|--|
| 1 | VDAC VSYNC | VDAC_VSYNC | RGB | VDAC vertical synchronization output. |
| 1 | VDAC HSYNC | VDAC_HSYNC | RGB | VDAC horizontal synchronization output. |
| 1 | Positive Red Analog Output | VDACR | AO | Positive Red VDAC analog output current. |
| 1 | Negative Red Analog Output | nVDACR | AO | Negative Red VDAC analog output current. |
| 1 | Positive Green Analog Output | VDACG | AO | Positive Green VDAC analog output current. |
| 1 | Negative Green Analog Output | nVDACG | AO | Negative Green VDAC analog output current. |
| 1 | Positive Blue Analog Output | VDACB | AO | Positive Blue VDAC analog output current. |
| 1 | Negative Blue Analog Output | nVDACB | AO | Negative Blue VDAC analog output current. |
| 1 | VDAC Reference Current | IREF | AI | VDAC reference current. Output current when using External Reference Resistor or Input Reference Current when using external current source. |

Table 3.5 DDR2 Memory Pins

| NUM PINS | NAME | SYMBOL | BUFFER TYPE | DESCRIPTION |
|----------|------------------------------------|-------------|-----------------|---|
| 13 | DDR2 Memory Address Bus | DDRA[12:0] | DDR2O | Bits 12:0 of the external DDR2 memory address bus. |
| 16 | DDR2 Memory Data Bus | DDRDQ[15:0] | DDR2I/ DDR2O | Bits 15:0 of the external DDR2 memory data bus. |
| 2 | DDR2 Memory Bank Address | DDRBA[1:0] | DDR2O | DDR2 memory bank address. |
| 1 | DDR2 Memory Clock High | DDRCK | DDR2O | Active high DDR2 clock. This clock is the complement of nDDRCK. |
| 1 | DDR2 Memory Clock Low | nDDRCK | DDR2O | Active low DDR2 clock. This clock is the complement of DDRCK. |
| 1 | DDR2 Memory Clock Enable Output | DDRCKE | DDR2O | DDR2 clock enable signal. |
| 1 | DDR2 Memory Chip Select | nDDRCS | DDR2O | Active low chip select |
| 1 | DDR2 Memory Row Address Strobe | nDDRRAS | DDR2O | Active low row address strobe. |
| 1 | DDR2 Memory Column Address Strobe | nDDRCAS | DDR2O | Active low column address strobe. |
| 1 | DDR2 Memory Write Enable | nDDRWE | DDR2O | Active low write enable. |
| 1 | DDR2 On Die Termination | DDRODT | DDR2O | DDR2 on die termination. |
| 1 | DDR2 Memory Lower Byte Mask | DDRDM0 | DDR2O | Mask bit for lower byte of DDR2 data word. |
| 1 | DDR2 Memory Upper Byte Mask | DDRDM1 | DDR2O | Mask bit for upper byte of DDR2 data word. |
| 1 | DDR2 Memory Lower Byte Strobe High | DDRDQS0 | DDR2I/ DDR2O | Active high data strobe for lower byte of DDR2 data word. |

Table 3.5 DDR2 Memory Pins (continued)

| NUM PINS | NAME | SYMBOL | BUFFER TYPE | DESCRIPTION |
|----------|------------------------------------|---------------|-------------|--|
| 1 | DDR2 Memory Lower Byte Strobe Low | nDDRQDS0 | DDR2I/DDR2O | Active low data strobe for upper byte of DDR2 data word. |
| 1 | DDR2 Memory Upper Byte Strobe High | DDRQDS1 | DDR2I/DDR2O | Active high data strobe for upper byte of DDR2 data word. |
| 1 | DDR2 Memory Upper Byte Strobe Low | nDDRQDS1 | DDR2I/DDR2O | Active low data strobe for upper byte of DDR2 data word. |
| 1 | DDR2 Memory Reference Voltage 0 | DDRVREF0 | AI | Reference voltage input pin for DDR2 Memory. DDRVREF0 must be half the VDD18DDR voltage. |
| 1 | DDR2 Memory Reference Voltage 1 | DDRVREF1 | AI | Reference voltage input pin for DDR2 Memory. DDRVREF1 must be half the VDD18DDR voltage. |
| 1 | DDR2 Memory Reference Voltage 2 | DDRVREF2 | AI | Reference voltage input pin for DDR2 Memory. DDRVREF2 must be half the VDD18DDR voltage. |
| 1 | DQS Enable Timing Match Input | DDRFIFOWE_IN | DDR2I | DQS enable input for timing match between DQS and system clock. |
| 1 | DQS Enable Timing Match Output | DDRFIFOWE_OUT | DDR2O | DQS enable output for timing match between DQS and system clock. |

Table 3.6 HDMI Pins

| NUM PINS | NAME | SYMBOL | BUFFER TYPE | DESCRIPTION |
|----------|---------------------|--------|-------------|---|
| 1 | TMDS Clock Positive | TXCP | AO | TMDS clock output differential positive signal. |
| 1 | TMDS Clock Negative | TXCN | AO | TMDS clock output differential negative signal. |
| 1 | TMDS Out0 Positive | TX0P | AO | TMDS Output 0 differential positive signal. |
| 1 | TMDS Out0 Negative | TX0N | AO | TMDS Output 0 differential negative signal. |
| 1 | TMDS Out1 Positive | TX1P | AO | TMDS Output 1 differential positive signal. |

Table 3.6 HDMI Pins (continued)

| NUM PINS | NAME | SYMBOL | BUFFER TYPE | DESCRIPTION |
|----------|----------------------|----------|-------------|---|
| 1 | TMDS Out1 Negative | TX1N | AO | TMDS Output 1 differential negative signal. |
| 1 | TMDS Out2 Positive | TX2P | AO | TMDS Output 2 differential positive signal. |
| 1 | TMDS Out2 Negative | TX2N | AO | TMDS Output 2 differential negative signal. |
| 1 | Voltage Swing Adjust | EXTSWING | AI | Connect this pin to an external resistor going to ground. The resistor determines the amplitude of the voltage swing. A low capacitive connection is allowed. A value of 5K is recommended. |
| 1 | Hot Plug Detect | HPD | IS_5V | Note: Hot plug detect signal. |

Table 3.7 EEPROM Pins

| NUM PINS | NAME | SYMBOL | BUFFER TYPE | DESCRIPTION |
|----------|--------------------|--------|-------------|--|
| 1 | EEPROM Data In | EEDI | IS (PD) | This pin is driven by the EEDO output of the external EEPROM. |
| 1 | EEPROM Data Out | EEDO | O8 | This pin drives the EEDI input of the external EEPROM. |
| 1 | EEPROM Chip Select | EECS | O8 | This pin drives the chip select output of the external EEPROM. Note: The EECS output may tri-state briefly during power-up. Some EEPROM devices may be prone to false selection during this time. When an EEPROM is used, an external pull-down resistor is recommended on this signal to prevent false selection. Refer to your EEPROM manufacturer's datasheet for additional information. |
| 1 | EEPROM Clock | EECLK | O8 | This pin drives the EEPROM clock of the external EEPROM. |

Table 3.8 JTAG Pins

| NUM PINS | NAME | SYMBOL | BUFFER TYPE | DESCRIPTION |
|----------|--------------------|--------|-------------|--|
| 1 | JTAG Test Data Out | TDO | O8 | JTAG data output. |
| 1 | JTAG Test Clock | TCK | IS | JTAG test clock. The maximum operating frequency of this clock is 10 MHz. |

Table 3.8 JTAG Pins (continued)

| NUM PINS | NAME | SYMBOL | BUFFER TYPE | DESCRIPTION |
|----------|-----------------------|--------|-------------|--|
| 1 | JTAG Test Mode Select | TMS | IS | JTAG test mode select. |
| 1 | JTAG Test Data Input | TDI | IS | JTAG data input. |
| 1 | JTAG Test Port Reset | nTRST | IS | Note: JTAG test port reset input. |

Table 3.9 Miscellaneous Pins

| NUM PINS | NAME | SYMBOL | BUFFER TYPE | DESCRIPTION |
|----------|-----------------------------|------------|---|--|
| 1 | LED | LED | O8/ OD8 (PU) | Can be used to provide device status. Alternatively, the LED can be configured for a fast or slow blink in accordance with the USB graphics data receive rate. |
| | I2S Clock Alternate Input 1 | I2SCLKALT1 | IS | I ² S Clock alternate input 1. The I ² S clock input pin is selectable between the I2SCLKALT0 or I2SCLKALT1 pins. Note: If the single data rate RGB interface is enabled, I2SCLKALT1 should be used. I2SCLKALT0 should be used in all other cases. Refer to the Audio Configuration Register (AUDIO_CFG) on page 354 for further details. |
| | General Purpose I/O 24 | GPIO24 | IS/O8/ OD8 (PU) Note 3.4 | This General Purpose I/O pin is fully programmable as either a push-pull output, an open-drain output, or a Schmitt-triggered input. |
| 1 | Interrupt | INT | IS | For use by external transmitter to signal an event requiring servicing. |
| 1 | External Reset | nEXTRST | O8 | Used to reset external transmitter. Note: The External Reset Configuration Register (EXT_RST_CFG) on page 353 controls the characteristics of the reset signal generated on this pin. |
| 1 | Switching Regulator Mode | nSW_MODE | O8 | When asserted, this pin can be used to place the external switching regulator into power saving mode. Note: The SW_MODE Polarity (CFG0_SW_MODE_POL) bit of Configuration Flags 0 controls the polarity of the pin. |

Table 3.9 Miscellaneous Pins (continued)

| NUM PINS | NAME | SYMBOL | BUFFER TYPE | DESCRIPTION |
|----------|----------------------------|----------|---|---|
| 1 | System Reset | nRESET | IS | This active-low pin allows external hardware to reset the device. |
| 1 | Detect Upstream VBUS Power | VBUS_DET | IS_5V | <p>Detects state of upstream bus power.</p> <p>For bus powered operation, this pin must be tied to VDD33IO.</p> <p>For self powered operation, refer to the device reference schematics.</p> <p>Note: The VBUS_DET signal is deglitched for a period of 10 ms.</p> |
| 1 | I ² C Data 0 | I2CSDA0 | IS_5V/ OD8 Note 3.5 | Bi-directional I ² C data 0 signal. |
| 1 | I ² C Clock 0 | I2CSCL0 | IS_5V/ OD8 Note 3.5 | Bi-directional I ² C clock 0 signal. All I ² C transactions are synchronous to the rising edge of this clock. The device supports I ² C standard mode operation (100 Kb/s). As an I ² C master, the device drives this clock. |
| 1 | I ² C Data 1 | I2CSDA1 | IS_5V/ OD8 Note 3.5 | Bi-directional I ² C data 1 signal. |
| | General Purpose I/O 27 | GPIO27 | IS_5V/ OD8 (PU) Note 3.4 | This General Purpose I/O pin is fully programmable as either a push-pull output, an open-drain output, or a Schmitt-triggered input. |
| 1 | I ² C Clock 1 | I2CSCL1 | IS_5V/ OD8 Note 3.5 | Bi-directional I ² C clock 1 signal. All I ² C transactions are synchronous to the rising edge of this clock. The device supports I ² C standard mode operation (100 Kb/s). As an I ² C master, the device drives this clock. |
| | General Purpose I/O 28 | GPIO28 | IS_5V/ OD8 (PU) Note 3.4 | This General Purpose I/O pin is fully programmable as either a push-pull output, an open-drain output, or a Schmitt-triggered input. |
| 1 | Audio Input Master Clock | MCLK | IS | Audio input master clock is coherent with the S/PDIF audio input. |
| | General Purpose I/O 25 | GPIO25 | IS/OD8/ (PU) Note 3.4 | This General Purpose I/O pin is fully programmable as either a push-pull output, an open-drain output, or a Schmitt-triggered input. |

Table 3.9 Miscellaneous Pins (continued)

| NUM PINS | NAME | SYMBOL | BUFFER TYPE | DESCRIPTION |
|----------|------------------------|-----------|--|--|
| 1 | S/PDIF Audio Input | SPDIF | IS | Digital audio interface input. Supports PCM, Dolby Digital, and DTS Digital audio transmission. Note: Usage of SPDIF requires the MCLK audio input master clock pin. |
| | I2S Data | I2SDATA | IS | I2S Data input. |
| | General Purpose I/O 26 | GPIO26 | IS/O8/OD8 (PU) Note 3.4 | This General Purpose I/O pin is fully programmable as either a push-pull output, an open-drain output, or a Schmitt-triggered input. |
| 1 | Audio Word Select | WS | IS | Specifies the I ² S word select input of the audio processor. |
| | General Purpose I/O 29 | GPIO29 | IS/O8/OD8 (PU) Note 3.4 | This General Purpose I/O pin is fully programmable as either a push-pull output, an open-drain output, or a Schmitt-triggered input. |
| 1 | Audio CODEC Disconnect | AUDIO_DIS | O8 | This pin is used for disconnecting an external USB audio CODEC. Refer to the UFX7000 reference schematic for additional details. |
| | General Purpose I/O 30 | GPIO30 | IS/O8/OD8 (PU) Note 3.4 | This General Purpose I/O pin is fully programmable as either a push-pull output, an open-drain output, or a Schmitt-triggered input. |

Note 3.4 The internal pull-up is disabled when the GPIO is configured as an O8 buffer type.

Note 3.5 If unused, this signal must be pulled to a valid state.

Table 3.10 I/O Power Pins, Core Power Pins, and Ground Pins

| NUM PINS | NAME | SYMBOL | BUFFER TYPE | DESCRIPTION |
|----------|-----------------------------|-----------|-------------|--|
| 1 | +3.3 V SYS PLL Power Input | SYSPLL P | P | +3.3 V system and pixel PLL power input. |
| 1 | SYS PLL Ground | SYSPLL G | P | System and pixel PLL ground. |
| 3 | +1.2 V HDMI Power Input | VDD12HDMI | P | +1.2 V HDMI power input. |
| 3 | HDMI Ground | VSSHDMI | P | HDMI ground. |
| 3 | +3.3 V VDAC Power Input | VDD33VDAC | P | +3.3 V Video DAC power input. (Note 3.6) |
| 1 | +3.3 V VDAC Reference Input | VDACREF | P | +3.3 V Video DAC reference voltage input. (Note 3.6) |

Table 3.10 I/O Power Pins, Core Power Pins, and Ground Pins (continued)

| NUM PINS | NAME | SYMBOL | BUFFER TYPE | DESCRIPTION |
|----------|---------------------------------|-------------|-------------|---|
| 2 | VDAC Ground | VSSVDAC | P | Video DAC ground. |
| 1 | +3.3 V USB3 Power Input | VDD33USB3 | P | +3.3 V USB3 power input. (Note 3.6) |
| 1 | +3.3 V USB Power Input | VDD33USB | P | +3.3 V USB power input. (Note 3.6) |
| 1 | +1.2 V USB PLL Supply Input | VDD12USBPLL | P | +1.2 V USB PLL supply input. (Note 3.6) |
| 2 | +1.2 V USB3 Power Input | VDD12USB3 | P | +1.2 V USB3 power input. (Note 3.6) |
| 15 | +1.8 V DDR2 Power Input | VDD18DDR | P | 1.8 V DDR2 power input. (Note 3.6) |
| 8 | +3.3 V I/O Power Input | VDD33IO | P | +3.3 V I/O power input. (Note 3.6) |
| 11 | +1.2 V Digital Core Power Input | VDD12CORE | P | +1.2 V digital core power input. (Note 3.6) |
| 3 | USB3 Ground | VSSUSB3 | P | USB3 Ground |
| 34 | Ground | VSS | P | Common Ground. |

Note 3.6 Refer to [Chapter 17, "Power Connections," on page 386](#) and the device reference schematics for additional power connection information.

Table 3.11 No-Connect Pins

| NUM PINS | NAME | SYMBOL | BUFFER TYPE | DESCRIPTION |
|----------|------------|--------|-------------|---|
| 4 | No Connect | NC | - | These pins must be left floating for normal device operation. |

3.1 Pin Assignments

Table 3.13 225-LFBGA Package Pin Assignments

| PIN NUM | PIN NAME | PIN NUM | PIN NAME | PIN NUM | PIN NAME | PIN NUM | PIN NAME |
|---------|---------------|---------|-----------|---------|-----------|---------|-----------|
| A1 | DDRFIFOWE_IN | B1 | DDRVREF0 | C1 | DDRDQ9 | D1 | DDRDQS1 |
| A2 | DDRFIFOWE_OUT | B2 | DDRDQ5 | C2 | DDRDQ11 | D2 | DDRDM1 |
| A3 | DDRDQ7 | B3 | DDRDQ0 | C3 | DDRDQ12 | D3 | DDRDQ14 |
| A4 | nDDRDQS0 | B4 | DDRDQS0 | C4 | DDRDQ2 | D4 | VDD18DDR |
| A5 | DDRDM0 | B5 | DDRDQ6 | C5 | VDD18DDR | D5 | VDD18DDR |
| A6 | DDRDQ1 | B6 | DDRDQ3 | C6 | DDRDQ4 | D6 | DDRVREF2 |
| A7 | nDDRWE | B7 | DDRCKE | C7 | DDRBA1 | D7 | VDD18DDR |
| A8 | DDRA10 | B8 | DDRBA0 | C8 | DDRA1 | D8 | VDD18DDR |
| A9 | DDRA3 | B9 | DDRA7 | C9 | DDRA5 | D9 | VDD18DDR |
| A10 | DDRA9 | B10 | DDRA12 | C10 | nDDRRAS | D10 | VDD18DDR |
| A11 | DDRCK | B11 | nDDRCK | C11 | VDD18DDR | D11 | VDD18DDR |
| A12 | nDDRCAS | B12 | nDDRCAS | C12 | DDRA0 | D12 | VDD18DDR |
| A13 | DDRA2 | B13 | DDRA6 | C13 | DDRODT | D13 | VDD18DDR |
| A14 | DDRA4 | B14 | DDRA11 | C14 | VSS | D14 | TX2P |
| A15 | DDRA8 | B15 | VSS | C15 | EXTSWING | D15 | TX2N |
| | | | | | | | |
| E1 | nDDRDQS1 | F1 | DDRDQ13 | G1 | VSSUSB3 | H1 | USB3TXDP |
| E2 | DDRDQ8 | F2 | DDRDQ10 | G2 | VSSUSB3 | H2 | USB3TXDM |
| E3 | DDRDQ15 | F3 | DDRVREF1 | G3 | VDD33USB3 | H3 | VDD12USB3 |
| E4 | VDD18DDR | F4 | VDD18DDR | G4 | VSS | H4 | VDD12USB3 |
| E5 | VDD18DDR | F5 | VDD12CORE | G5 | VSS | H5 | VDD12CORE |
| E6 | VDD12CORE | F6 | VSS | G6 | VSS | H6 | VSS |
| E7 | VSS | F7 | VSS | G7 | VSS | H7 | VSS |
| E8 | VDD12CORE | F8 | VSS | G8 | VSS | H8 | VSS |
| E9 | VSS | F9 | VSS | G9 | VSS | H9 | VSS |
| E10 | VDD12CORE | F10 | VSS | G10 | VSS | H10 | VSS |
| E11 | VDD18DDR | F11 | VDD12CORE | G11 | VDD12CORE | H11 | VSSVDAC |
| E12 | VSSHDMI | F12 | VSSHDMI | G12 | VSSHDMI | H12 | IREF |
| E13 | VDD12HDMI | F13 | VDD12HDMI | G13 | VDD12HDMI | H13 | VSSVDAC |
| E14 | TX1P | F14 | TX0P | G14 | TXCP | H14 | VDACR |
| E15 | TX1N | F15 | TX0N | G15 | TXCN | H15 | nVDACR |

Table 3.13 225-LFBGA Package Pin Assignments (continued)

| PIN NUM | PIN NAME | PIN NUM | PIN NAME | PIN NUM | PIN NAME | PIN NUM | PIN NAME |
|---------|-----------|---------|--------------------|---------|--------------------------|---------|-----------------------|
| J1 | USB3RXDP | K1 | I2CSDA1/ GPIO27 | L1 | USBDP | M1 | VDD33USB |
| J2 | USB3RXDM | K2 | I2CSDA0 | L2 | USBDM | M2 | NC |
| J3 | VSSUSB3 | K3 | I2CSCL1/ GPIO28 | L3 | VBUS_DET | M3 | HPD |
| J4 | REXT | K4 | I2CSCL0 | L4 | AUDIO_DIS/ GPIO30 | M4 | VATAB4/VD3/ GPIO20 |
| J5 | VSS | K5 | VDD12CORE | L5 | VDD33IO | M5 | VDD33IO |
| J6 | VSS | K6 | VSS | L6 | VDD12CORE | M6 | VDD33IO |
| J7 | VSS | K7 | VSS | L7 | VSS | M7 | VDD33IO |
| J8 | VSS | K8 | VSS | L8 | VDD12CORE | M8 | VATAG4/VD9/ GPIO12 |
| J9 | VSS | K9 | VSS | L9 | VSS | M9 | VDD33IO |
| J10 | VSS | K10 | VSS | L10 | VDD12CORE | M10 | VATAR5/ GPIO5 |
| J11 | VDD33VDAC | K11 | VDACREF | L11 | VDD33IO | M11 | VDD33IO |
| J12 | VDD33VDAC | K12 | VDAC_HSYNC | L12 | VDD33IO | M12 | TDI |
| J13 | VDD33VDAC | K13 | VDAC_VSYNC | L13 | WS/GPIO29 | M13 | INT |
| J14 | VDACG | K14 | VDACB | L14 | MCLK/GPIO25 | M14 | EECS |
| J15 | nVDACG | K15 | nVDACB | L15 | SPDIF/I2SDATA/ GPIO26 | M15 | EECLK |
| | | | | | | | |

Table 3.13 225-LFBGA Package Pin Assignments (continued)

| PIN NUM | PIN NAME | PIN NUM | PIN NAME | PIN NUM | PIN NAME | PIN NUM | PIN NAME |
|------------|------------------------|------------|----------------------------------|------------|-------------------------------|------------|----------|
| N1 | XI | P1 | XO | R1 | SYSPLL | | |
| N2 | VDD12USBPLL | P2 | SYSPLL | R2 | VDATAB7/VD0/ GPIO23 | | |
| N3 | USBRBIAS | P3 | VDATAB6/VD1/ GPIO22 | R3 | VDATAB5/VD2/ GPIO21 | | |
| N4 | VDATAB3/VD4/ GPIO19 | P4 | VDATAB2/VD5/ GPIO18 | R4 | VDATAB1/VD6/ GPIO17 | | |
| N5 | VDATAB0/ GPIO16 | P5 | HSYNC | R5 | VSYNC | | |
| N6 | nBLANK | P6 | nVCLK | R6 | VCLK | | |
| N7 | VDATAG7/ GPIO15 | P7 | VDATAG6/VD7/ GPIO14 | R7 | VDATAG5/VD8/ GPIO13 | | |
| N8 | VDATAG3/ GPIO11 | P8 | VDATAG2/ GPIO10 | R8 | VDATAG1/VD10/ GPIO9 | | |
| N9 | VDATAG0/VD11/ GPIO8 | P9 | VDATAR7/VD12/ GPIO7 | R9 | VDATAR6/ GPIO6 | | |
| N10 | VDATAR4/VD13/ GPIO4 | P10 | VDATAR3/VD14/ GPIO3 | R10 | VDATAR2/VD15/ GPIO2 | | |
| N11 | VDATAR1/ GPIO1 | P11 | VDATAR0/ I2SCLKALT0/ GPIO0 | R11 | nTRST | | |
| N12 | TDO | P12 | TCK | R12 | TMS | | |
| N13 | nEXTRST | P13 | nSW_MODE | R13 | NC | | |
| N14 | NC | P14 | nRESET | R14 | LED/ I2SCLKALT1/ GPIO24 | | |
| N15 | NC | P15 | EEDI | R15 | EEDO | | |

3.2 Buffer Types

Table 3.14 Buffer Types

| BUFFER TYPE | DESCRIPTION |
|-------------|---|
| IS | Schmitt-triggered Input |
| IS_5V | 5V Tolerant Schmitt-triggered Input |
| O8 | Output with 8mA sink and 8mA source |
| OD8 | Open-drain output with 8mA sink |
| O12 | Output with 12mA sink and 12mA source |
| OD12 | Open-drain output with 12mA sink |
| PU | 50uA (typical) internal pull-up. Unless otherwise noted in the pin description, internal pull-ups are always enabled. Note: Internal pull-up resistors prevent unconnected inputs from floating. Do not rely on internal resistors to drive signals external to the device. When connected to a load that must be pulled high, an external resistor must be added. |
| PD | 50uA (typical) internal pull-down. Unless otherwise noted in the pin description, internal pull-downs are always enabled. Note: Internal pull-down resistors prevent unconnected inputs from floating. Do not rely on internal resistors to drive signals external to the device. When connected to a load that must be pulled low, an external resistor must be added. |
| AI | Analog input |
| AO | Analog output |
| AIO | Analog bi-directional |
| DDR2I | DDR2 input |
| DDR2O | DDR2 output |
| RGB | RGB output |
| ICLK | Crystal oscillator input pin |
| OCLK | Crystal oscillator output pin |
| P | Power pin |

Chapter 4 USB Controller

4.1 Overview

UFX7000 complies with the USB 2.0 and USB 3.0 specifications. A compliant UTMI interface is used to interface to the USB 2.0 PHY on both devices, while a compliant PIPE 3.0 interface is used to interface to the USB3.0 PHY on the UFX7000 device. LPM is supported in USB 2.0 mode.

The device support three endpoints: Control, Bulk-Out and Interrupt (Endpoint0, Endpoint1, and Endpoint2). SS, HS, and FS operation is supported. LS is not supported. Although the device can enumerate in FS mode, the graphics engine is not operational.

There is one configuration, one interface, and no alternate settings. Bus powered and self powered operation is supported. Remote wakeup is not supported.

4.2 USB Control Endpoint

The Control endpoint is handled by the CTL (USB Control) module. The CTL module is responsible for handling USB standard commands as well as USB vendor commands. In order to support these commands the CTL must compile a variety of statistics and store the programmable portions of the USB descriptors.

4.2.1 USB Vendor Commands

The device implements several vendor specific commands in order to access CSRs and efficiently gather statistics. The vendor commands allow direct access to Systems CSRs and DDR2 memory.

4.2.1.1 Register Write Command

The commands allows the Host to write a single register. All writes are 32-bits.

Table 4.1 Format of Register Write Setup Stage

| OFFSET | FIELD | VALUE |
|--------|---------------|-------------------|
| 0h | bmRequestType | 40h |
| 1h | bRequest | A0h |
| 2h | wValue | 00h |
| 4h | wIndex | CSR Address[15:0] |
| 6h | wLength | 04h |

Table 4.2 Format of Register Write Data Stage

| OFFSET | FIELD |
|--------|----------------------------|
| 0h | Register Write Data [31:0] |

Special care needs to be taken when accessing the HDMI CSRs. The transmitter's CSRs are byte aligned. As a result the HDMI address must be converted into a dword boundary for use in the register write command.

Since, the HDMI CSRs have a 9-bit address the following equation is used to generate the value of CSR Address.

$$\text{CSR Address} = 9000\text{h} + (\text{HDMI Address}) * 4$$

Only bits 7:0 of the Register Write Data field are used when accessing the HDMI CSRs. All other bits are ignored.

4.2.1.2 Register Read Command

The commands allows the Host to read a single register. All reads return 32-bits.

Table 4.3 Format of Register Read Setup Stage

| OFFSET | FIELD | VALUE |
|--------|---------------|-------------------|
| 0h | bmRequestType | C0h |
| 1h | bRequest | A1h |
| 2h | wValue | 00h |
| 4h | wIndex | CSR Address[15:0] |
| 6h | wLength | 04h |

Table 4.4 Format of Register Read Data Stage

| OFFSET | FIELD |
|--------|---------------------------|
| 0h | Register Read Data [31:0] |

As with the Register Write Command additional step needs to be taken for generating the CSR Address when accessing the HDMI CSRs. See [Section 4.2.1.1](#) for a description.

Only bits 7:0 of the Register Read Data field are valid when accessing the HDMI CSRs. All other bits must be ignored.

4.2.1.3 DDR2 Memory Read Command

The commands allows the Host to read 128 bytes of DDR2 Memory starting at the specified address.

Table 4.5 Format of Register Read Setup Stage

| OFFSET | FIELD | VALUE |
|--------|---------------|--|
| 0h | bmRequestType | C0h |
| 1h | bRequest | A2h |
| 2h | wValue | DDR2 Address[15:0] Note 4.1 |
| 4h | wIndex | {7'b0, DDR2 Address[24:16]} |
| 6h | wLength | Note 4.2 |

Note 4.1 The lower 3 bits of the specified address are ignored, resulting in QUAD WORD aligned data being returned.

Note 4.2 Any length up to FFFFh may be specified. Length should be specified in multiples of 64 bytes. If it is not, then any extra bytes specified in excess of the 64 byte multiple will result in an additional 64 bytes being transferred.

4.2.1.4 Get Statistics Command

The Get Statistics Command returns the entire contents of the statistics counters.

Note: The contents of the statistics counters is snapshot when fulfilling the command request. The statistics counters rollover, hence the they are not cleared.

Table 4.6 Format of Get Statistics Setup Stage

| OFFSET | FIELD | VALUE |
|--------|---------------|-------|
| 0h | bmRequestType | C0h |
| 1h | bRequest | A3h |
| 2h | wValue | 00h |
| 4h | wIndex | 00h |
| 6h | wLength | 2Ch |

Table 4.7 Format of Get Statistics Data Stage

| OFFSET | FIELD | DESCRIPTION |
|--------|--|--|
| 00h | Bulk Out Packets Received | <ul style="list-style-type: none"> Only data stage counted Only successfully received packets are counted ZLPs counted |
| 04h | Control Packets Received | <ul style="list-style-type: none"> Only data stage counted Only successfully received packets are counted Only vendor command control packets are counted |
| 08h | Control Packets Transmitted | <ul style="list-style-type: none"> Only data stage counted Only successfully transmitted packets are counted Only responses to vendor command packets are counted |
| 0Ch | Errorred Super Speed Vendor Setup Packets Received | <ul style="list-style-type: none"> Number of unsuccessfully received vendor setup packets. This doesn't include cases where host retries setup packet because ACK TP was lost Only applies to Super Speed operation |

Table 4.7 Format of Get Statistics Data Stage

| OFFSET | FIELD | DESCRIPTION |
|--------|--|--|
| 10h | Errored Super Speed Control Data Packets Received | <ul style="list-style-type: none"> Number of unsuccessfully received control data packets. This doesn't include cases where host retries data packet because ACK TP was lost Only applies to Super Speed operation |
| 14h | Errored Super Speed Control Data Packets Transmitted | <ul style="list-style-type: none"> Number of unsuccessfully transmitted control data packets. Counts cases where same sequence number is requested again Only applies to Super Speed operation |
| 18h | Errored Super Speed Bulk Out Data Packets Received | <ul style="list-style-type: none"> Number of unsuccessfully received Bulk OUT packets Only errored packets made visible to the application interface by the device controller are counted Only applies to Super Speed operation |
| 1Ch | Number of 8b/10b Decode or Disparity Errors | <ul style="list-style-type: none"> Count number of pipe_rxstatus = 100 or 111 Only applies to Super Speed operation |
| 20h | Elastic Buffer Overflow | <ul style="list-style-type: none"> Count pipe_rxstatus = 101 Only applies to Super Speed operation |
| 24h | Elastic Buffer Underflow | <ul style="list-style-type: none"> Count pipe_rxstatus = 110 Only applies to Super Speed operation |
| 28h | U0->Recovery Transitions | <ul style="list-style-type: none"> Count mac_ltssm_state transitions from 'hC to 'h11 Only applies to Super Speed operation |

Note: Zeroes will be returned in USB2 mode for counters that only apply to USB3.

4.3 USB Bulk Out Endpoint

The Bulk-out endpoint is responsible for receiving graphics commands and passing them to the Graphics Engine via an RX FIFO. An 18 KB FIFO allows for efficient utilization of USB3 bandwidth.

4.4 USB Interrupt Endpoint

The Interrupt endpoint is responsible for indicating the device status at each polling interval. When the endpoint is accessed, the following fields are presented to the host, when an enabled interrupt is asserted:

Table 4.8 Interrupt Packet Format

| OFFSET | BITS | DESCRIPTION |
|--------|-------|---------------------|
| 0 | 31:20 | RESERVED |
| 0 | 19 | DISPLAY_BLANK_INT |
| 0 | 18 | H_BLANK_START_INT |
| 0 | 17 | V_BLANK_START_INT |
| 0 | 16 | FRAME_BASE_ADDR_INT |
| 0 | 15:3 | RESERVED |
| 0 | 2 | I2C_INT |
| 0 | 1 | HDMI_INT |
| 0 | 0 | INTPIN |

If there is no interrupt status to report, the device responds with a NAK.

Note: The polling interval is static and set through the EEPROM. The polling interval can be changed by the host updating the contents of the EEPROM and resetting the part.

The interrupt status can be cleared by clearing the interrupt at its source. The means specific to each interrupt is indicated in [Section 4.6.2, "USB Interrupt Endpoint Status Register \(USB_INT_STS\)," on page 70](#).

Note: If the interrupt packet is not sent successfully, the same interrupt packet, not an updated one, is resent.

4.5 USB Descriptors

4.5.1 USB Descriptors - UFX7000

The following tables illustrate the USB descriptor values for UFX7000.

4.5.1.1 Device Descriptor

The Device Descriptors are initialized based on values stored in EEPROM. [Table 4.9](#) shows the default Device Descriptor values. These values are used for Full-Speed, High-Speed, and Super-Speed operation.

Table 4.9 Device Descriptor

| OFFSET | FIELD | SIZE (BYTES) | DEFAULT VALUE | LOADED FROM EEPROM | DESCRIPTION |
|--------|--------------------|--------------|--------------------------|--------------------------|--|
| 00h | bLength | 1 | 12h | Note 4.3 | Size of the Descriptor in Bytes (18 bytes) |
| 01h | bDescriptorType | 1 | 01h | Note 4.3 | Device Descriptor (0x01) |
| 02h | bcdUSB | 2 | Note 4.4 | No | USB Specification Number which device complies to. |
| 04h | bDeviceClass | 1 | FFh | Yes | Class Code |
| 05h | bDeviceSubClass | 1 | 00h | Yes | Subclass Code |
| 06h | bDeviceProtocol | 1 | FFh | Yes | Protocol Code |
| 07h | bMaxPacketSize | 1 | 09h | Note 4.6 | Maximum Packet Size for Endpoint 0 |
| 08h | IdVendor | 2 | 0424h | Yes | Vendor ID |
| 0Ah | IdProduct | 2 | 9D00h | Yes | Product ID |
| 0Ch | bcdDevice | 2 | Note 4.5 | Yes | Device Release Number |
| 0Eh | iManufacturer | 1 | 00h | Yes | Index of Manufacturer String Descriptor |
| 0Fh | iProduct | 1 | 00h | Yes | Index of Product String Descriptor |
| 10h | iSerialNumber | 1 | 00h | Yes | Index of Serial Number String Descriptor |
| 11h | bNumConfigurations | 1 | 01h | Note 4.4 | Number of Possible Configurations |

Note 4.3 The descriptor length and descriptor type for Device Descriptors specified in EEPROM are a “don’t cares” and are always overwritten by hardware as 0x12 and 0x01, respectively.

Note 4.4 When operating in USB 2.0 mode the default value is 0210h (USB 2.10). When operating in USB 3.0 mode the default value is 0300h (USB 3.0)

Note 4.5 Default value is dependent on device release. MSB matches the device release and LSB hardcoded to 00h. The initial release value is 01h for the MPW. The value for A0 is 02h. Subsequent versions after A0 will increment the value.

Note 4.6 The bMaxPacketSize must be set to 09h when operating in super-speed mode. When operating in full-speed or high-speed mode it should be set to 40h. When no EEPROM is present the aforementioned values are returned.

4.5.1.2 Configuration Descriptor

The Configuration Descriptor is initialized based on values stored in EEPROM. [Table 4.10](#) shows the default Configuration Descriptor values. These values are used for Full-Speed, High-Speed, and Super-Speed operation.

Table 4.10 Configuration Descriptor

| OFFSET | FIELD | SIZE (BYTES) | DEFAULT VALUE | LOADED FROM EEPROM | DESCRIPTION |
|--------|---------------------|--------------|---------------------------|--------------------------|--|
| 00h | bLength | 1 | 09h | Note 4.7 | Size of the Configuration Descriptor in bytes (9 bytes) |
| 01h | bDescriptorType | 1 | 02h | Note 4.8 | Configuration Descriptor (0x02) |
| 02h | wTotalLength | 2 | Note 4.9 | Note 4.7 | Total length in bytes of data returned |
| 04h | bNumInterfaces | 1 | 01h | Note 4.7 | Number of Interfaces |
| 05h | bConfigurationValue | 1 | 01h | Note 4.7 | Value to use as an argument to select this configuration |
| 06h | iConfiguration | 1 | 00h | Yes | Index of String Descriptor describing this configuration |
| 07h | bmAttributes | 1 | 80h | Yes | Bus powered and remote wakeup disabled. |
| 08h | bMaxPower | 1 | Note 4.10 | Yes | Maximum Power Consumption |

Note 4.7 Value is loaded from EEPROM, but must be equal to the Default Value in order to comply with the USB 3.0 Specification and provide for normal device operation. Specification of any other value will result in unwanted behavior and untoward operation.

Note 4.8 The descriptor type for Configuration Descriptors specified in EEPROM is a “don’t care” and is always overwritten by hardware as 0x02.

Note 4.9 Default value is 0020h (32 bytes) when operating in USB 2.0 mode and 002Ch (44 bytes) when operating in USB 3.0 mode.

Note 4.10 Default value is 01h in Self Powered mode. In Bus Powered mode, default value is FAh (500mA) when operating in USB 2.0 mode and 70h (900mA) when operating in USB 3.0 mode.

Note: The Power Method bit of the Configuration Flags of the EEPROM may affect the default value of bmAttributes.

4.5.1.3 Interface Descriptor 0 Default

Table 4.11 shows the default value for Interface Descriptor 0. This descriptor is initialized based on values stored in EEPROM.

Table 4.11 Interface Descriptor 0

| OFFSET | FIELD | SIZE (BYTES) | DEFAULT VALUE | LOADED FROM EEPROM | DESCRIPTION |
|--------|--------------------|--------------|---------------|--------------------|---|
| 00h | bLength | 1 | 09h | Note 4.11 | Size of Descriptor in Bytes (9 Bytes) |
| 01h | bDescriptorType | 1 | 04h | Note 4.11 | Interface Descriptor (0x04) |
| 02h | bInterfaceNumber | 1 | 00h | Note 4.11 | Number identifying this Interface |
| 03h | bAlternateSetting | 1 | 00h | Note 4.11 | Value used to select alternative setting |
| 04h | bNumEndpoints | 1 | 02h | Note 4.11 | Number of Endpoints used for this interface (Less endpoint 0) |
| 05h | bInterfaceClass | 1 | FFh | Yes | Class Code |
| 06h | bInterfaceSubClass | 1 | 00h | Yes | Subclass Code |
| 07h | bInterfaceProtocol | 1 | FFh | Yes | Protocol Code |
| 08h | iInterface | 1 | 00h | Yes | Index of String Descriptor Describing this interface |

Note 4.11 Value is loaded from EEPROM, but must be equal to the Default Value in order to comply with the USB 3.0 Specification and provide for normal device operation. Specification of any other value will result in unwanted behavior and untoward operation.

4.5.1.4 Endpoint 1 Descriptor (Bulk-Out)

Table 4.12 shows the default value for Endpoint Descriptor 1. This descriptor is not initialized from values stored in EEPROM.

Table 4.12 Endpoint 1 Descriptor

| OFFSET | FIELD | SIZE (BYTES) | DEFAULT VALUE | LOADED FROM EEPROM | DESCRIPTION |
|--------|------------------|--------------|---------------|--------------------|---|
| 00h | bLength | 1 | 07h | No | Size of Descriptor in bytes |
| 01h | bDescriptorType | 1 | 05h | No | Endpoint Descriptor |
| 02h | bEndpointAddress | 1 | 01h | No | Endpoint Address |
| 03h | bmAttributes | 1 | 02h | No | Bulk Transfer Type |
| 04h | wMaxPacketSize | 2 | Note 4.12 | No | Maximum Packet Size this endpoint is capable of sending. |
| 06h | bInterval | 1 | 00h | No | Interval for polling endpoint data transfers. Ignored for bulk endpoints. |

Note 4.12 64 bytes for full-speed mode. 512 bytes for high-speed mode, 1024 bytes for super-speed mode.

4.5.1.5 Super-Speed Endpoint Companion Descriptor (Bulk Out Endpoint)

The Super-Speed Endpoint Companion Descriptor is initialized based on values stored in EEPROM. [Table 4.13](#) shows the default Super-Speed Endpoint Companion Descriptor (Bulk Out Endpoint) values.

Table 4.13 Super-Speed Endpoint Companion Descriptor (Bulk Out Endpoint)

| OFFSET | FIELD | SIZE (BYTES) | DEFAULT VALUE | LOADED FROM EEPROM | DESCRIPTION |
|--------|-----------------|--------------|----------------------------------|---------------------------|--|
| 00h | bLength | 1 | 06h | Note 4.13 | Size of Descriptor in bytes (6 bytes) |
| 01h | bDescriptorType | 1 | 30h | Note 4.13 | Super-Speed Endpoint Companion Descriptor (0x30) |
| 02h | bMaxBurst | 1 | 07h Note 4.14 | Yes | <p>Maximum number of packets -1 the endpoint can send or receive as part of a burst.</p> <p>Note: Only values from 0 to 15, inclusive, are valid. I.e., 0 = 1 packet, 15 = 16 packets.</p> <p>Note: For endpoints of type control, this shall be set to 0.</p> |

Table 4.13 Super-Speed Endpoint Companion Descriptor (Bulk Out Endpoint)

| OFFSET | FIELD | SIZE (BYTES) | DEFAULT VALUE | LOADED FROM EEPROM | DESCRIPTION | | | | | | |
|---|---|--------------|---------------|--------------------|--|---|----------|-----|--------------|-----|--|
| 03h | bmAttributes | 1 | 00h | Note 4.15 | If this is a Bulk Endpoint: | | | | | | |
| | | | | | <table><tr><th>BIT</th><th>ENCODING</th></tr><tr><td>7:5</td><td>RESERVED (0)</td></tr><tr><td>4:0</td><td>MaxStreams. The maximum number of streams this endpoint supports. Valid values are from 0 to 16, where 0 indicates that the endpoint does not define streams. For the values 1 to 16, the number of streams support is equal to 2^{MaxStreams}</td></tr></table> | BIT | ENCODING | 7:5 | RESERVED (0) | 4:0 | MaxStreams. The maximum number of streams this endpoint supports. Valid values are from 0 to 16, where 0 indicates that the endpoint does not define streams. For the values 1 to 16, the number of streams support is equal to 2 ^{MaxStreams} |
| | | | | | BIT | ENCODING | | | | | |
| | | | | | 7:5 | RESERVED (0) | | | | | |
| | | | | | 4:0 | MaxStreams. The maximum number of streams this endpoint supports. Valid values are from 0 to 16, where 0 indicates that the endpoint does not define streams. For the values 1 to 16, the number of streams support is equal to 2 ^{MaxStreams} | | | | | |
| If this is an isochronous Endpoint: | | | | | | | | | | | |
| <table><tr><th>BIT</th><th>ENCODING</th></tr><tr><td>7:2</td><td>RESERVED (0)</td></tr><tr><td>1:0</td><td>Mult. A zero based value that determines the maximum number of packets within a service interval that this endpoint supports. Maximum number of packets = bMaxBurst * (Mult + 1). Note: The maximum value that can be set in this field is 2.</td></tr></table> | BIT | ENCODING | 7:2 | RESERVED (0) | 1:0 | Mult. A zero based value that determines the maximum number of packets within a service interval that this endpoint supports. Maximum number of packets = bMaxBurst * (Mult + 1). Note: The maximum value that can be set in this field is 2. | | | | | |
| BIT | ENCODING | | | | | | | | | | |
| 7:2 | RESERVED (0) | | | | | | | | | | |
| 1:0 | Mult. A zero based value that determines the maximum number of packets within a service interval that this endpoint supports. Maximum number of packets = bMaxBurst * (Mult + 1). Note: The maximum value that can be set in this field is 2. | | | | | | | | | | |
| If this is a control or interrupt endpoint: Bits 7:0 are RESERVED (0) | | | | | | | | | | | |
| 04h | wBytesPerInterval | 2 | 0000h | Yes | <p>The total number of bytes this endpoint will transfer every service interval.This field is only valid for periodic endpoints.</p> <p>For isochronous endpoints, this value is used to reserve the bus time, in the schedule, required for the frame data payloads per 125 uS. The pipe may, on an ongoing basis, actually use less bandwidth than reserved. The device reports, if necessary, the actual bandwidth used via its normal, non-USB defined mechanisms.</p> | | | | | | |



Note 4.13 The descriptor length and descriptor type for super-speed Endpoint Companion Descriptors specified in EEPROM are "don't cares" and are always overwritten by hardware as 0x06 and 0x30, respectively.

Note 4.14 This value can be overridden by the [Super-speed Maximum Burst Size \(MAX_BURST\)](#) field of the [USB Configuration Register \(USB_CFG\)](#).

Note 4.15 Streams are not supported in this device. This value must always be written to 0h.

4.5.1.6 Endpoint 2 Descriptor (Interrupt)

Table 4.14 shows the default value for Endpoint Descriptor 2. Only the bInterval field of this descriptor is initialized from EEPROM.

Table 4.14 Endpoint 2 Descriptor

| OFFSET | FIELD | SIZE (BYTES) | DEFAULT VALUE | LOADED FROM EEPROM | DESCRIPTION |
|--------|------------------|--------------|---------------|--------------------|--|
| 00h | bLength | 1 | 07h | No | Size of Descriptor in bytes |
| 01h | bDescriptorType | 1 | 05h | No | Endpoint Descriptor |
| 02h | bEndpointAddress | 1 | 82h | No | Endpoint Address |
| 03h | bmAttributes | 1 | 03h | No | Interrupt Transfer Type |
| 04h | wMaxPacketSize | 2 | 10h | No | Maximum Packet Size this endpoint is capable of sending. |
| 06h | bInterval | 1 | Note 4.16 | Yes | Interval for polling endpoint data transfers. |

Note 4.16 This value is loaded from the EEPROM. A full-speed and high-speed polling interval exists. If no EEPROM exists then this value defaults to 04h for HS, 01h for FS, and 06h for super-speed.

4.5.1.7 Super-Speed Endpoint Companion Descriptor (Interrupt Endpoint)

The Super-Speed Endpoint Companion Descriptor is initialized based on values stored in EEPROM. [Table 4.15](#) shows the default Super-Speed Endpoint Companion Descriptor (Interrupt Endpoint) values.

Table 4.15 Super-Speed Endpoint Companion Descriptor (Interrupt Endpoint)

| OFFSET | FIELD | SIZE (BYTES) | DEFAULT VALUE | LOADED FROM EEPROM | DESCRIPTION |
|--------|-----------------|--------------|---------------|---------------------------|--|
| 00h | bLength | 1 | 06h | Note 4.17 | Size of Descriptor in bytes (6 bytes) |
| 01h | bDescriptorType | 1 | 30h | Note 4.17 | Super-Speed Endpoint Companion Descriptor (0x30) |
| 02h | bMaxBurst | 1 | 00h | Yes | <p>Maximum number of packets -1 the endpoint can send or receive as part of a burst.</p> <p>Note: Only values from 0 to 15, inclusive, are valid. I.e., 0 = 1 packet, 15 = 16 packets.</p> <p>Note: For endpoints of type control, this shall be set to 0.</p> |

Table 4.15 Super-Speed Endpoint Companion Descriptor (Interrupt Endpoint)

| OFFSET | FIELD | SIZE (BYTES) | DEFAULT VALUE | LOADED FROM EEPROM | DESCRIPTION | | | | | | |
|---|---|--------------|---------------|--------------------|--|---|----------|-----|--------------|-----|--|
| 03h | bmAttributes | 1 | 00h | Yes | If this is a Bulk Endpoint: | | | | | | |
| | | | | | <table><tr><th>BIT</th><th>ENCODING</th></tr><tr><td>7:5</td><td>RESERVED (0)</td></tr><tr><td>4:0</td><td>MaxStreams. The maximum number of streams this endpoint supports. Valid values are from 0 to 16, where 0 indicates that the endpoint does not define streams. For the values 1 to 16, the number of streams support is equal to 2^{MaxStreams}</td></tr></table> | BIT | ENCODING | 7:5 | RESERVED (0) | 4:0 | MaxStreams. The maximum number of streams this endpoint supports. Valid values are from 0 to 16, where 0 indicates that the endpoint does not define streams. For the values 1 to 16, the number of streams support is equal to 2 ^{MaxStreams} |
| | | | | | BIT | ENCODING | | | | | |
| | | | | | 7:5 | RESERVED (0) | | | | | |
| | | | | | 4:0 | MaxStreams. The maximum number of streams this endpoint supports. Valid values are from 0 to 16, where 0 indicates that the endpoint does not define streams. For the values 1 to 16, the number of streams support is equal to 2 ^{MaxStreams} | | | | | |
| If this is an isochronous Endpoint: | | | | | | | | | | | |
| <table><tr><th>BIT</th><th>ENCODING</th></tr><tr><td>7:2</td><td>RESERVED (0)</td></tr><tr><td>1:0</td><td>Mult. A zero based value that determines the maximum number of packets within a service interval that this endpoint supports. Maximum number of packets = bMaxBurst * (Mult + 1). Note: The maximum value that can be set in this field is 2.</td></tr></table> | BIT | ENCODING | 7:2 | RESERVED (0) | 1:0 | Mult. A zero based value that determines the maximum number of packets within a service interval that this endpoint supports. Maximum number of packets = bMaxBurst * (Mult + 1). Note: The maximum value that can be set in this field is 2. | | | | | |
| BIT | ENCODING | | | | | | | | | | |
| 7:2 | RESERVED (0) | | | | | | | | | | |
| 1:0 | Mult. A zero based value that determines the maximum number of packets within a service interval that this endpoint supports. Maximum number of packets = bMaxBurst * (Mult + 1). Note: The maximum value that can be set in this field is 2. | | | | | | | | | | |
| If this is a control or interrupt endpoint: Bits 7:0 are RESERVED (0) | | | | | | | | | | | |
| 04h | wBytesPerInterval | 2 | 0000h | Yes | <p>The total number of bytes this endpoint will transfer every service interval.This field is only valid for periodic endpoints.</p> <p>For isochronous endpoints, this value is used to reserve the bus time, in the schedule, required for the frame data payloads per 125 uS. The pipe may, on an ongoing basis, actually use less bandwidth than reserved. The device reports, if necessary, the actual bandwidth used via its normal, non-USB defined mechanisms.</p> | | | | | | |



Note 4.17 The descriptor length and descriptor type for Super-Speed Endpoint Companion Descriptors specified in EEPROM are "don't cares" and are always overwritten by hardware as 0x06 and 0x30, respectively.

4.5.1.8 Other Speed Configuration Descriptor

The fields in this descriptor are derived from Configuration Descriptor information that is stored in the EEPROM.

Note: This descriptor is only available when the UFX7000 is operating in USB 2.0 mode.

Table 4.16 Other Speed Configuration Descriptor

| OFFSET | FIELD | SIZE (BYTES) | DEFAULT VALUE | LOADED FROM EEPROM | DESCRIPTION |
|--------|---------------------|--------------|---------------------------|---------------------------|--|
| 00h | bLength | 1 | 09h | Note 4.18 | Size of Descriptor in bytes (9 bytes) |
| 01h | bDescriptorType | 1 | 07h | No | Other Speed Configuration Descriptor (0x07) |
| 02h | wTotalLength | 2 | 0020h | Note 4.18 | Total length in bytes of data returned (32 bytes) |
| 04h | bNumInterfaces | 1 | 01h | Note 4.18 | Number of Interfaces |
| 05h | bConfigurationValue | 1 | 01h | Note 4.18 | Value to use as an argument to select this configuration |
| 06h | iConfiguration | 1 | 00h | Yes | Index of String Descriptor describing this configuration |
| 07h | bmAttributes | 1 | 80h | Yes | Bus powered and remote wakeup disabled. |
| 08h | bMaxPower | 1 | Note 4.19 | Yes | Maximum Power Consumption |

Note: EEPROM values are obtained for the Configuration Descriptor at the other USB speed. I.e., if the current operating speed is FS, then the HS Configuration Descriptor values are used, and vice-versa.

Note 4.18 Value is loaded from EEPROM, but must be equal to the Default Value in order to comply with the USB 3.0 Specification and provide for normal device operation. Specification of any other value will result in unwanted behavior and untoward operation.

Note 4.19 Default value is 01h in Self Powered mode and FAh (500 mA) in Bus Powered mode.

Note: The Power Method bit of the Configuration Flags of the EEPROM may affect the default value of bmAttributes.

4.5.1.9 Device Qualifier Descriptor

The fields in this descriptor are derived from Device Descriptor information that is stored in the EEPROM.

Note: This descriptor is only available when the UFX7000 is operating in USB 2.0 mode.

Table 4.17 Device Qualifier Descriptor

| OFFSET | FIELD | SIZE (BYTES) | DEFAULT VALUE | LOADED FROM EEPROM | DESCRIPTION |
|--------|--------------------|--------------|---------------|---------------------------|--|
| 00h | bLength | 1 | 0Ah | No | Size of Descriptor in bytes (10 bytes) |
| 01h | bDescriptorType | 1 | 06h | No | Device Qualifier Descriptor (0x06) |
| 02h | bcdUSB | 2 | 0210h | Note 4.20 | USB Specification Number which device complies to. |
| 04h | bDeviceClass | 1 | FFh | Yes | Class Code |
| 05h | bDeviceSubClass | 1 | 00h | Yes | Subclass Code |
| 06h | bDeviceProtocol | 1 | FFh | Yes | Protocol Code |
| 07h | bMaxPacketSize0 | 1 | 40h | Note 4.20 | Maximum Packet Size |
| 08h | bNumConfigurations | 1 | 01h | Note 4.20 | Number of Other-Speed Configurations |
| 09h | Reserved | 1 | 00h | No | Must be zero |

Note: EEPROM values are from the Device Descriptor (including any EEPROM override) at the opposite HS/FS operating speed. I.e., if the current operating speed is HS, then Device Qualifier data is based on the FS Device Descriptor, and vice-versa.

Note 4.20 .Value is loaded from EEPROM, but must be equal to the Default Value in order to comply with the USB 3.0 Specification and provide for normal device operation.

4.5.1.10 Binary Device Object Store Descriptor

The Binary Device Object Store Descriptor is initialized based on values stored in EEPROM. [Table 4.18](#) shows the default Binary Device Object Store Descriptor values.

Table 4.18 Binary Device Object Store Descriptor

| OFFSET | FIELD | SIZE (BYTES) | DEFAULT VALUE | LOADED FROM EEPROM | DESCRIPTION |
|--------|-----------------|--------------|---------------|---------------------------|--|
| 00h | bLength | 1 | 05h | Note 4.21 | Size of Descriptor in bytes (5 bytes) |
| 01h | bDescriptorType | 1 | 0Fh | Note 4.21 | BOS Descriptor (0x0F) |
| 02h | wTotalLength | 2 | 0016h | Yes | Total length of this descriptor and its subdescriptors. (22 bytes) |
| 04h | bNumDeviceCaps | 1 | 02h | Yes | Number of Device Capability Descriptors in this BOS. |

Note 4.21 The descriptor length and descriptor type for Binary Device Object Store Descriptors specified in EEPROM are “don’t cares” and are always overwritten by hardware as 0x05 and 0x0F, respectively.

4.5.1.11 USB 2.0 Extension Descriptor

The USB 2.0 Extension Descriptor is initialized based on values stored in EEPROM. [Table 4.19](#) shows the default USB 2.0 Extension Descriptor values.

Table 4.19 USB 2.0 Extension Descriptor

| OFFSET | FIELD | SIZE (BYTES) | DEFAULT VALUE | LOADED FROM EEPROM | DESCRIPTION | | | | | | | | |
|--------|---|-----------------|------------------|---------------------------|---|-----|----------|------|--------------|---|---|---|--------------|
| 00h | bLength | 1 | 07h | Note 4.22 | Size of Descriptor in bytes (7 bytes) | | | | | | | | |
| 01h | bDescriptorType | 1 | 10h | Note 4.22 | Device Capability Descriptor (0x10) | | | | | | | | |
| 02h | bDevCapabilityType | 1 | 02h | Note 4.22 | USB 2.0 Extension Capability (0x02) | | | | | | | | |
| 03h | bmAttributes | 4 | 0002h | Yes | <div>Bitmap encoding of number of supported device level features. A value of 1 in a bit location indicates a feature is supported. A value of 0 indicates it is not supported. Encodings are:</div> <table><tr><th>BIT</th><th>ENCODING</th></tr><tr><td>31:2</td><td>RESERVED (0)</td></tr><tr><td>1</td><td>1 (LPM) Note 4.23 A value of 1 in this bit position indicates that this device supports the Link Power Management protocol. Super Speed devices shall set this bit to 1.</td></tr><tr><td>0</td><td>RESERVED (0)</td></tr></table> | BIT | ENCODING | 31:2 | RESERVED (0) | 1 | 1 (LPM) Note 4.23 A value of 1 in this bit position indicates that this device supports the Link Power Management protocol. Super Speed devices shall set this bit to 1. | 0 | RESERVED (0) |
| BIT | ENCODING | | | | | | | | | | | | |
| 31:2 | RESERVED (0) | | | | | | | | | | | | |
| 1 | 1 (LPM) Note 4.23 A value of 1 in this bit position indicates that this device supports the Link Power Management protocol. Super Speed devices shall set this bit to 1. | | | | | | | | | | | | |
| 0 | RESERVED (0) | | | | | | | | | | | | |

Note 4.22 The descriptor length, descriptor type, and device capability type for USB 2.0 Extension Descriptors specified in EEPROM are “don’t cares” and are always overwritten by hardware as 0x07, 0x10, and 0x02, respectively.

Note 4.23 The value of this bit must match that of the LPM_ENABLE flag contained in the Configuration Flags byte of the EEPROM, if present. If the bit values disagree, unexpected results and untoward operation may result.

4.5.1.12 Super-Speed USB Device Capabilities Descriptor

The Super Speed USB Device Capabilities Descriptor is initialized based on values stored in EEPROM. [Table 4.20](#) shows the default Super Speed USB Device Capabilities Descriptor values.

Table 4.20 Super-Speed USB Device Capabilities Descriptor

| OFFSET | FIELD | SIZE (BYTES) | DEFAULT VALUE | LOADED FROM EEPROM | DESCRIPTION | | | | | | | | | | | | |
|--------|---|--------------|---------------|---------------------------|--|-----|----------|------|--------------|---|---|---|-------------------------------------|---|-------------------------------------|---|------------------------------------|
| 00h | bLength | 1 | 0Ah | Note 4.24 | Size of Descriptor in bytes (10 bytes) | | | | | | | | | | | | |
| 01h | bDescriptorType | 1 | 10h | Note 4.24 | Device Capability Descriptor (0x10) | | | | | | | | | | | | |
| 02h | bDevCapabilityType | 1 | 03h | Note 4.24 | USB 2.0 Extension Capability (0x03) | | | | | | | | | | | | |
| 03h | bmAttributes | 1 | 00h | Yes | <div>Bitmap encoding of number of supported device level features. A value of 1 in a bit location indicates a feature is supported. A value of 0 indicates it is not supported. Encodings are:</div> <table><tr><th>BIT</th><th>ENCODING</th></tr><tr><td>7:2</td><td>RESERVED (0)</td></tr><tr><td>1</td><td>LTM Capable. A value of 1 in this bit position indicates that this device is capable of generating Latency Tolerance Messages.</td></tr><tr><td>0</td><td>RESERVED (0)</td></tr></table> | BIT | ENCODING | 7:2 | RESERVED (0) | 1 | LTM Capable. A value of 1 in this bit position indicates that this device is capable of generating Latency Tolerance Messages. | 0 | RESERVED (0) | | | | |
| BIT | ENCODING | | | | | | | | | | | | | | | | |
| 7:2 | RESERVED (0) | | | | | | | | | | | | | | | | |
| 1 | LTM Capable. A value of 1 in this bit position indicates that this device is capable of generating Latency Tolerance Messages. | | | | | | | | | | | | | | | | |
| 0 | RESERVED (0) | | | | | | | | | | | | | | | | |
| 04h | wSpeedsSupported | 2 | 000Eh | Yes | <div>Bitmap encoding of the speed supported by this device when operating in Super-Speed mode.:</div> <table><tr><th>BIT</th><th>ENCODING</th></tr><tr><td>15:4</td><td>RESERVED (0)</td></tr><tr><td>3</td><td>1 = Device Supports operation at 5 Gbps.</td></tr><tr><td>2</td><td>1 = Device supports High Speed USB.</td></tr><tr><td>1</td><td>1 = Device supports Full Speed USB.</td></tr><tr><td>0</td><td>1 = Device supports Low Speed USB.</td></tr></table> | BIT | ENCODING | 15:4 | RESERVED (0) | 3 | 1 = Device Supports operation at 5 Gbps. | 2 | 1 = Device supports High Speed USB. | 1 | 1 = Device supports Full Speed USB. | 0 | 1 = Device supports Low Speed USB. |
| BIT | ENCODING | | | | | | | | | | | | | | | | |
| 15:4 | RESERVED (0) | | | | | | | | | | | | | | | | |
| 3 | 1 = Device Supports operation at 5 Gbps. | | | | | | | | | | | | | | | | |
| 2 | 1 = Device supports High Speed USB. | | | | | | | | | | | | | | | | |
| 1 | 1 = Device supports Full Speed USB. | | | | | | | | | | | | | | | | |
| 0 | 1 = Device supports Low Speed USB. | | | | | | | | | | | | | | | | |

Table 4.20 Super-Speed USB Device Capabilities Descriptor

| OFFSET | FIELD | SIZE (BYTES) | DEFAULT VALUE | LOADED FROM EEPROM | DESCRIPTION |
|--------|-----------------------|--------------|---------------|--------------------|---|
| 06h | bFunctionalitySupport | 1 | 02h | Yes | <p>The lowest speed at which all the functionality supported by the device is available to the user.</p> <p>0 = Low Speed 1 = Full Speed 2 = High Speed 3 = 5 Gbps 4-255 = RESERVED</p> |
| 07h | bU1DevExitLat | 1 | 0Ah | Yes | <p>U1 Device Exit Latency Worst case latency to transition from U1 to U0, assuming the latency is limited only by the device and not the device's link partner. This field applies only to the exit latency associated with an individual port and does not apply to the total latency through a hub (i.e., from downstream port to upstream port).</p> <p>00h = 0 01h = Less than 1 uS 02h = Less than 2 uS 03h = Less than 3 uS = 09h = Less than 9 uS 0Ah = Less than 10 uS 0Bh - FFh = RESERVED</p> |
| 08h | wU2DevExitLat | 2 | 05DCh | Yes | <p>U2 Device Exit Latency Worst case latency to transition from U2 to U0, assuming the latency is limited only by the device and not the device's link partner. Applies to all ports on a device.</p> <p>0000h = 0 0001h = Less than 1 uS 0002h = Less than 2 uS 0003h = Less than 3 uS = 07FFh = Less than 2047 uS 0800h - FFFFh = RESERVED</p> |

Note 4.24 The descriptor length, descriptor type, and device capability type for Super-Speed USB Device Capabilities Descriptors specified in EEPROM are “don't cares” and are always overwritten by hardware as 0x0A, 0x10, and 0x03, respectively.

4.5.1.13 String Descriptors

4.5.1.13.1 STRING INDEX = 0 (LANGID)

Table 4.21 LANGID String Descriptor

| OFFSET | FIELD | SIZE (BYTES) | DEFAULT VALUE | LOADED FROM EEPROM | DESCRIPTION |
|--------|-----------------|--------------|---------------|--------------------|--|
| 00h | bLength | 1 | 04h | No | Size of LANGID Descriptor in bytes (4 bytes) |
| 01h | bDescriptorType | 1 | 03h | No | String Descriptor (0x03) |
| 02h | LANGID | 2 | None | Yes | Must be set to 0x0409 (US English). |

4.5.1.13.2 STRING INDICES 1-5

Table 4.22 String Descriptor (Indices 1-5)

| OFFSET | FIELD | SIZE (BYTES) | DEFAULT VALUE | LOADED FROM EEPROM | DESCRIPTION |
|--------|-----------------|--------------|---------------|--------------------|--|
| 00h | bLength | 1 | none | Yes | Size of the String Descriptor in bytes (4 bytes) |
| 01h | bDescriptorType | 1 | none | Yes | String Descriptor (0x03) |
| 02h | Unicode String | 2*N | none | Yes | 2 bytes per unicode character, no trailing NULL. |

Note: UFX7000 returns whatever bytes are in the designated EEPROM area for each of these strings it is the responsibility of the EEPROM programmer to correctly set the bLength and bDescriptorType fields in the descriptor consistent with the byte length specified in the corresponding EEPROM locations.

4.6 Control and Status Registers

Table 4.23 USB Control and Status Register Map

| ADDRESS OFFSET | SYMBOL | REGISTER NAME |
|-------------------|------------------|---|
| 5000h | USB_CFG | USB Configuration Register |
| 5004h | USB_INT_STS | USB Interrupt Endpoint Status Register |
| 5008h | USB_INT_CTL | USB Interrupt Endpoint Control Register |
| 500Ch – 50FCh | RESERVED | Reserved for future expansion |
| 5100h | USB_PHY_PARAMS0 | USB PHY Parameters 0 Register |
| 5104h | USB_PHY_PARAMS1 | USB PHY Parameters 1 Register |
| 5108h | USB_PHY_MPLL_CTL | USB PHY MPLL Control Register |
| 510Ch – 5FFFh | RESERVED | Reserved for future expansion |

4.6.1 USB Configuration Register (USB_CFG)

Address: 0000h Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|--|------|---------------------------|
| 31:14 | RESERVED | RO | - |
| 13 | Port Swap (PORT_SWAP) Swaps the mapping of the USB DP and USB DM pins. 0 = USB DP maps to the USB D+ line and USB DM maps to the USB D- line. 1 = USB DP maps to the USB D- line. USB DM maps to the USB D+ line. Note: Only for USB2.0 operation. Does not affect USB3.0 operation. USB3.0 pins can not be swapped. | RO | Note 4.25 |
| 12 | Super-speed Termination Resistance Calibration Request (RTUNE_REQ) Setting this bit initiates a resistor calibration on the Super-speed link. Calibration can not be performed while the link is operational. At the completion of the request, the bit is cleared by hardware. | SC | 0b |
| 11 | USB3 PHY Loopback (LANE0_TX2RX_LOOP) Setting this bit puts the USB3 PHY in loopback mode. | R/W | 0b |
| 10:7 | Super-speed Maximum Burst Size (MAX_BURST) This field determines the maximum/ burst size for the Super-Speed bulk out endpoint. Note: This field is protected by Reset Protection (RST_PROTECT) . | R/W | Note 4.26 |
| 6:5 | Squelch Threshold (SQU_THR) Varies reference voltage levels for squelch and HS Disconnect. 00 = Default 01 = -25mV Change 10 = +25mV Change 11 = RESERVED Note: This field is protected by Reset Protection (RST_PROTECT) . | R/W | Note 4.27 |
| 4:3 | PHY Boost (PHY_BOOST) This field provides the ability to boost the electrical drive strength of the HS output current to the upstream port. 00 = Normal electrical drive strength 01 = Elevated electrical drive strength (+4% boost) 10 = Elevated electrical drive strength (+8% boost) 11 = Elevated electrical drive strength (+12% boost) Note: This field is protected by Reset Protection (RST_PROTECT) . | R/W | Note 4.28 |
| 2 | LPM Enable (LPM_ENABLE) This bit enables the support of the Link Power Management (LPM) protocol. 0 = LPM not supported 1 = LPM supported. Note: This field is protected by Reset Protection (RST_PROTECT) . | R/W | Note 4.29 |

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|---|------|---------------------------|
| 1 | Power Method (PWR_SEL) This bit controls the device's USB power mode. 0 = The device is bus powered. 1 = The device is self powered. Note: This field is protected by Reset Protection (RST_PROTECT) . | R/W | Note 4.30 |
| 0 | Stall Bulk Out Pipe Disable (SBP) This bit controls the operation of the Bulk Out pipe when the FCT detects the loss of sync condition. Please refer to the Graphics Engine Error Status (GE_ERROR_STAT) field of the Graphic Engine Command Status Register (GE_CMD_STATUS) for error status details. 0 = Stall the Bulk Out pipe when loss of sync detected. 1 = Do not stall the Bulk Out pipe when loss of sync detected. | R/W | 0b |

Note 4.25 The default value of this bit is determined by the value of the [Port Swap \(CFG0_PORT_SWAP\)](#) bit of the [Configuration Flags 0](#) contained within the EEPROM, if present. If no EEPROM is present, 0b is the default.

Note 4.26 The default value of this field is determined by the value of the bMaxBurst field of the Bulk Out Endpoint Companion Descriptor contained within the EEPROM, if present. If no EEPROM is present, 7h is the default. A USB Reset or Lite Reset (LRST) will cause this field to be restored to the image value last loaded from EEPROM, or to be set to 7h if no EEPROM is present.

Note 4.27 The default value of this field is determined by the value of the [Squelch Threshold \(CFG0_SQU_THR\)](#) field of the [Configuration Flags 0](#) contained within the EEPROM, if present. If no EEPROM is present, 00b is the default. A USB Reset or Lite Reset (LRST) will cause this field to be restored to the image value last loaded from EEPROM, or to be set to 00b if no EEPROM is present.

Note 4.28 The default value of this field is determined by the value of the [PHY Boost \(CFG0_PHY_BOOST\)](#) field of the [Configuration Flags 0](#) contained within the EEPROM, if present. If no EEPROM is present, 00b is the default. A USB Reset or Lite Reset (LRST) will cause this field to be restored to the image value last loaded from EEPROM, or to be set to 00b if no EEPROM is present.

Note 4.29 The default value of this field is determined by the value of the [LPM Enable \(CFG0_LPM_ENABLE\)](#) field of the [Configuration Flags 0](#) contained within the EEPROM, if present. If no EEPROM is present, 1b is the default. A USB Reset or Lite Reset (LRST) will cause this field to be restored to the image value last loaded from EEPROM, or to be set to 1b if no EEPROM is present.

Note 4.30 The default value of this field is determined by the value of the [Power Method \(CFG0_PWR_SEL\)](#) field of the [Configuration Flags 0](#) contained within the EEPROM, if present. If no EEPROM is present, 0b is the default. A USB Reset or Lite Reset (LRST) will cause this field to be restored to the image value last loaded from EEPROM, or to be set to 0b if no EEPROM is present.

4.6.2 USB Interrupt Endpoint Status Register (USB_INT_STS)

Offset: 004h Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|---|------|---------------------------|
| 31:20 | RESERVED | RO | - |
| 19 | Display Blank Interrupt (DISPLAY_BLANK_INT) Cleared by writing a 1 to the Display Blank Interrupt (DISPLAY_BLANK_INT) bit of the Display Status Register (DISPLAY_STATUS) . | RO | 0b |
| 18 | Horizontal Blank Start Interrupt (H_BLANK_START_INT) Cleared by writing a 1 to the Horizontal Blank Start Interrupt (H_BLANK_START_INT) bit of the Display Status Register (DISPLAY_STATUS) . | RO | 0b |
| 17 | Vertical Blank Start Interrupt (V_BLANK_START_INT) Cleared by writing a 1 to the Vertical Blank Start Interrupt (V_BLANK_START_INT) bit of the Display Status Register (DISPLAY_STATUS) . | RO | 0b |
| 16 | Frame Base Address Interrupt (FRAME_BASE_ADDR_INT) Cleared by writing a 1 to the Frame Base Address Interrupt (FRAME_BASE_ADDR_INT) bit of the Display Status Register (DISPLAY_STATUS) . | RO | 0b |
| 15:3 | RESERVED | RO | - |
| 2 | I²C Interrupt (I2C_INT) If the interrupt is software clearable, it can be cleared by reading it's corresponding interrupt clear register, or the I²C Clear Interrupt Register (I2C_CLR_INTR) which clears all I ² C software interrupts. If the interrupt is clearable by hardware only, the interrupt will be cleared automatically when the interrupt conditions have changed. | RO | 0b |
| 1 | HDMI Interrupt (HDMI_INT) Cleared when all asserted interrupt bits in the HDMI block are cleared. | RO | 0b |
| 0 | Interrupt Pin Assertion (INTPIN) This interrupt is asserted when the INT pin is asserted. The polarity of the INT pin is determined by the Interrupt Pin Polarity (INT_POL) bit of the Hardware Configuration Register (HW_CFG) . | R/WC | Note 4.31 |

Note 4.31 Default value depends on the state of the INT pin.

4.6.3 USB Interrupt Endpoint Control Register (USB_INT_CTL)

Offset: 008h Size: 32 bits

This register determines which events cause status to be reported by the interrupt endpoint. See [Section 4.4, "USB Interrupt Endpoint"](#) for more details.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|--|------|---------|
| 31:20 | RESERVED | RO | - |
| 19 | Display Blank Enable (DISPLAY_BLANK_EN) | R/W | 0b |
| 18 | Horizontal Blank Start Enable (H_BLANK_START_EN) | R/W | 0b |
| 17 | Vertical Blank Start Enable (V_BLANK_START_EN) | R/W | 0b |
| 16 | Frame Base Address Enable (FRAME_BASE_ADDR_EN) | R/W | 0b |
| 15:3 | RESERVED | RO | - |
| 2 | I²C Interrupt Enable (I2C_EN) | R/W | 0b |
| 1 | HDMI Interrupt Enable (HDMI_EN) | R/W | 0b |
| 0 | Interrupt Pin Assertion Enable (INTPIN_EN) This interrupt is asserted when the INT pin is asserted. The polarity of the INT pin is determined by the Interrupt Pin Polarity (INT_POL) bit of the Hardware Configuration Register (HW_CFG) . | R/W | 0b |

4.6.4 USB PHY Parameters 0 Register (USB_PHY_PARAMS0)

Offset: 0100h Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|--|------|---------------------------|
| 31:30 | RESERVED | RO | - |
| 29:24 | TX De-Emphasis At 3.5 dB (PCS_TX_DEEMPH_3P5DB) This field sets the TX driver de-emphasis value for the case where pipe_tx_deemph is set to 1 (the default setting for USB3). This field may be used to tune at the board level for RX eye compliance, in order to account for different device or host channel loss in the PCB traces. Note: This field is protected by Reset Protection (RST_PROTECT) . | R/W | Note 4.32 |
| 23:22 | RESERVED | RO | - |
| 21:16 | TX De-Emphasis at 6 dB (PCS_TX_DEEMPH_6DB) This field sets the TX driver de-emphasis value for the case where pipe_tx_deemph is set to 0 (this should never happen for USB3). This field is provided for completeness and as a 2nd potential launch amplitude. Note: This field is protected by Reset Protection (RST_PROTECT) . | R/W | Note 4.33 |
| 15 | RESERVED | RO | - |
| 14:8 | TX Amplitude For Full Swing Mode (PCS_TX_SWING_FULL) This field sets the launch amplitude of the transmitter when pipe_tx_swing is set to 0 (the default setting for USB3 for the required 1.0V launch amplitude). This field may be used to tune at the board level for RX eye compliance, in order to account for different device or host channel loss in the PCB traces. Note: This field is protected by Reset Protection (RST_PROTECT) . | R/W | Note 4.34 |
| 7 | RESERVED | RO | - |
| 6:0 | TX Amplitude For Low Swing Mode (PCS_TX_SWING_LOW) This field sets the launch amplitude of the transmitter when pipe_tx_swing is set to 1 (this should never happen for USB3). This field is provided for completeness and can be used to set an alternate launch amplitude, if desired. Note: This field is protected by Reset Protection (RST_PROTECT) . | R/W | Note 4.35 |

Note 4.32 The default value of this field is determined by the value of the [TX De-Emphasis At 3.5 dB \(CFG1_TX_DEEMPH_3P5DB\)](#) field of [Configuration Flags 1](#) contained within the EEPROM, if present. If no EEPROM is present, 16h is the default. A USB Reset or Lite Reset (LRST) will cause this field to be restored to the image value last loaded from EEPROM, or to be set to 16h if no EEPROM is present.

Note 4.33 The default value of this field is determined by the value of the [TX De-Emphasis at 6 dB \(CFG1_TX_DEEMPH_6DB\)](#) field of [Configuration Flags 1](#) contained within the EEPROM, if present. If no EEPROM is present, 20h is the default. A USB Reset or Lite Reset (LRST) will cause this field to be restored to the image value last loaded from EEPROM, or to be set to 20h if no EEPROM is present.

Note 4.34 The default value of this field is determined by the value of the [TX Amplitude For Full Swing Mode \(CFG1_TX_SWING_FULL\)](#) field of [Configuration Flags 1](#) contained within the EEPROM, if present. If no EEPROM is present, 69h is the default. A USB Reset or Lite Reset (LRST) will cause this field to be restored to the image value last loaded from EEPROM, or to be set to 69h if no EEPROM is present.



Note 4.35 The default value of this field is determined by the value of the [TX Amplitude For Low Swing Mode \(CFG1_TX_SWING_LOW\)](#) field of [Configuration Flags 1](#) contained within the EEPROM, if present. If no EEPROM is present, 35h is the default. A USB Reset or Lite Reset (LRST) will cause this field to be restored to the image value last loaded from EEPROM, or to be set to 35h if no EEPROM is present.

4.6.5 USB PHY Parameters 1 Register (USB_PHY_PARAMS1)

Offset: 0104h Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|--|------|---------------------------|
| 31:13 | RESERVED | RO | - |
| 12:8 | Loss Of Signal Detection Threshold Level (LOS_LEVEL) This field sets the signal level for the detection of loss of signal. Note: This field is protected by Reset Protection (RST_PROTECT) . | R/W | Note 4.36 |
| 7:5 | RESERVED | RO | - |
| 4:0 | TX Termination Offset (TX_TERM_OFFSET) This field allows the termination impedance of the transmitter to be shifted off from the nominal value of 50 ohms after calibration. This allows the user to potentially optimize the signal integrity of the link. Use of this signal is optional. When not used it should be set to 00h. Note: This field is protected by Reset Protection (RST_PROTECT) . | R/W | Note 4.37 |

Note 4.36 The default value of this field is determined by the value of the [Loss Of Signal Detection Threshold Level \(CFG2_LOS_LEVEL\)](#) field of [Configuration Flags 2](#) contained within the EEPROM, if present. If no EEPROM is present, 09h is the default. A USB Reset or Lite Reset (LRST) will cause this field to be restored to the image value last loaded from EEPROM, or to be set to 09h if no EEPROM is present.

Note 4.37 The default value of this field is determined by the value of the [TX Termination Offset \(CFG2_TX_TERM_OFFSET\)](#) field of [Configuration Flags 2](#) contained within the EEPROM, if present. If no EEPROM is present, 00h is the default. A USB Reset or Lite Reset (LRST) will cause this field to be restored to the image value last loaded from EEPROM, or to be set to 00h if no EEPROM is present.

4.6.6 USB PHY MPLL Control Register (USB_PHY_MPLL_CTL)

Offset: 0108h Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|---|------|---------------------------|
| 31:3 | RESERVED | RO | - |
| 2:1 | Spread Spectrum Clock Range (SSC_RANGE) This field selects the range of modulation to insert. It specifies the amount of clock spreading that will be added and applies a fixed offset to the phase accumulator. The following values select the indicated PPM downspread of the clock: 00 = 5000 01 = 4500 10 = 4000 11 = 3025 Note: This field is protected by Reset Protection (RST_PROTECT) . | R/W | Note 4.38 |
| 0 | Spread Spectrum Enable (SSC_EN) When set, this bit enables spread spectrum clock production in the USB3 SuperSpeed PHY, required for transmitting 5Gb/sec Super Speed data. When this bit is de-asserted, Spread Spectrum Clock Range (SSC_RANGE) is ignored. Note: This field is protected by Reset Protection (RST_PROTECT) . | R/W | Note 4.39 |

Note 4.38 The default value of this field is determined by the value of the [Spread Spectrum Clock Range\(CFG3_SSC_RANGE\)](#) field of [Configuration Flags 3](#) contained within the EEPROM, if present. If no EEPROM is present, 10b is the default. A USB Reset or Lite Reset (LRST) will cause this field to be restored to the image value last loaded from EEPROM, or to be set to 10b if no EEPROM is present.

Note 4.39 The default value of this bit is determined by the value of the [Spread Spectrum Enable \(CFG3_SSC_EN\)](#) bit of [Configuration Flags 3](#) contained within the EEPROM, if present. If no EEPROM is present, 1b is the default. A USB Reset or Lite Reset (LRST) will cause this field to be restored to the image value last loaded from EEPROM, or to be set to 1b if no EEPROM is present.

Chapter 5 Clocks and Power Management

5.1 Overview

The Clock and Power Management module (CPM) is responsible for generating the clocks and controlling the power management logic for the device.

CPM automatically power downs various functional blocks in order to minimize power consumption with the assistance of system software. Specifically the CPM.

- Powers down/up the USB 3.0 PHY depending on the power state.
- Powers down/up the USB 2.0 PHY depending on the power state.
- Enables/Disables the crystal oscillator depending on the power state.

5.2 Power States

5.2.1 Power State Definitions

The following power states are supported.

Table 5.1 Power States

| Power State | Description |
|--------------|--|
| RESET | The device enters, and stays in, this state when the nReset pin asserted. This state is optimized for low power consumption. |
| UNPOWERED | This state is only supported during self powered operation. It is entered when VBUS_DET is not asserted. Minimal circuitry is enabled to detect, and deglitch, VBUS assertion. |
| UNCONFIGURED | This is the state of the device after reset de-assertion but before being configured. When bus powered, and operating in FS/HS mode, the current consumed in this state is limited to 100 mA. In SS mode the limit is 150 mA. |
| CONFIGURED | This is the state of the device after being configured. Only in this state is the device fully functional. When bus powered, and operating in FS/HS mode, the current consumed in this state is limited to 500 mA per USB specification. In SS mode the limit is 900 mA. |
| SUSPEND | The USB Host places the device in this state to conserve power. When bus-powered the current consumed in this state is limited to 2.5 mA per USB specification. Remote wakeups are not supported by this device. The SUSPEND state can only be exited via a USB Resume or a system reset event. <ul style="list-style-type: none"> ■ In USB 2.0 mode the CPM is able to detect USB Resume. ■ In USB 2.0 mode the CPM is able to support LPM low power states ■ In USB 3.0 mode the CPM is able to support Ux states |

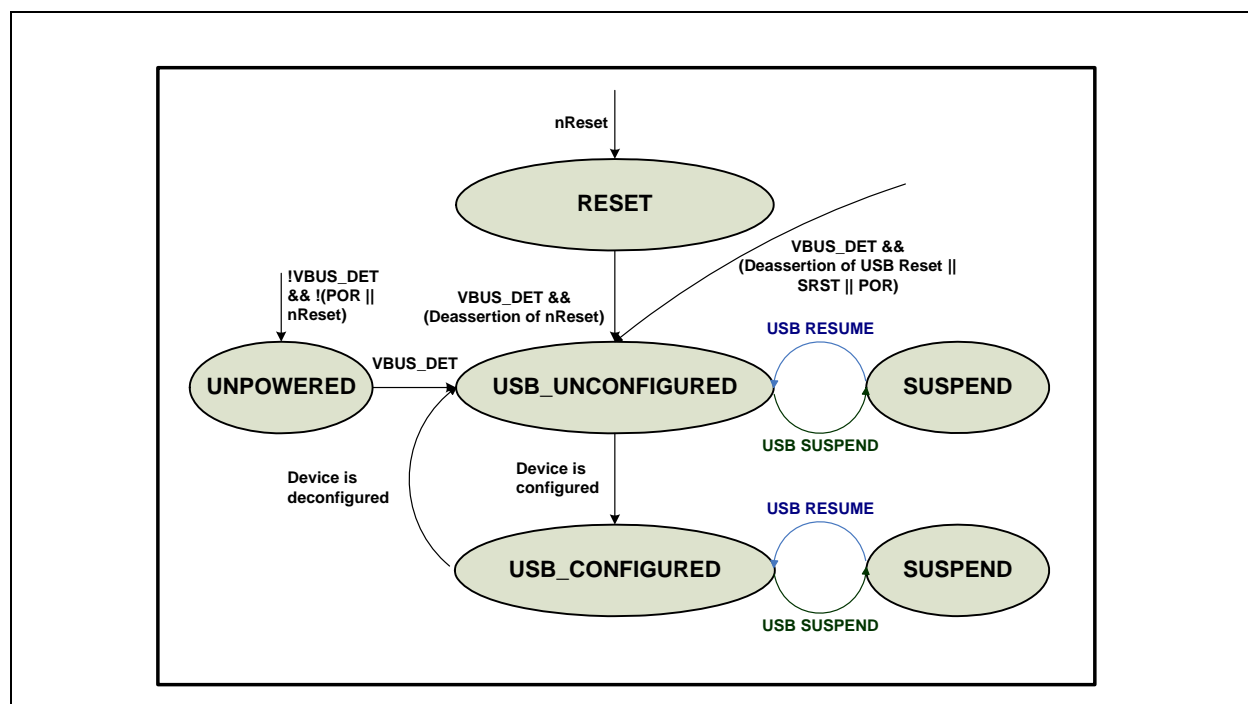


Figure 5.1 Power States

5.2.2 VBUS Detection

VBUS_DET is deglitched for a period of 10 ms.

Note: The deglitch interval may exceed 10 ms due to startup time of the crystal oscillator and the lock time for the common block.

The logic doesn't propagate any transitions on VBUS for the duration of the deglitch timer, after VBUS_DET changes state. The deglitcher logic must operate in the clk_xtal domain. This is to ensure that VBUS_DET assertion detection can occur while in the UNPOWERED state. Figure 5.2 illustrates the derivation of VBUS_DET and its use as an enable for the crystal oscillator.

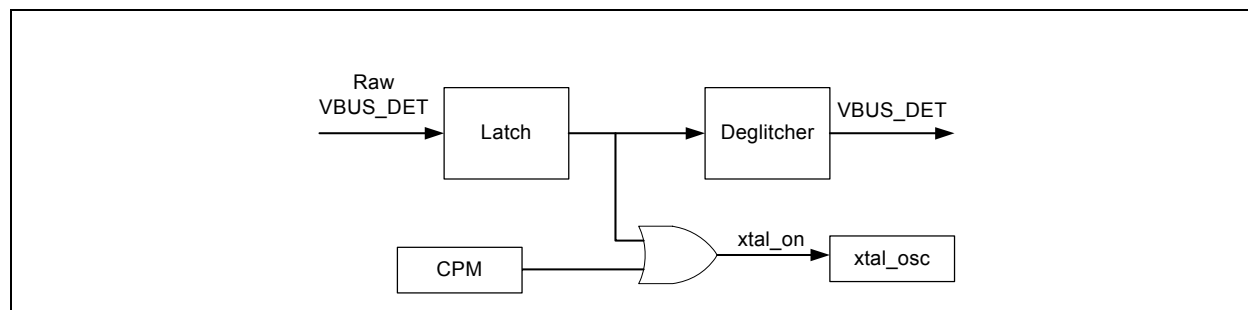


Figure 5.2 VBUS Detection

5.2.3 Power State Mapping Tables

| | | | | | |
|----------------------|-----|-----|-----|-----|-----|
| USB 3.0 PHY | Off | Off | Off | Off | Off |
| UDC 3.0 | Off | Off | Off | Off | Off |
| USB 3.0 Common Block | Off | Off | Off | Off | Off |

Table 5.2 Functionality to Power State Mapping - UFX7000

| MODULE | RESET | NORMAL (UNCONFIGURED) | NORMAL (CONFIGURED) | SUSPEND | UNPOWERED |
|----------------------|---------------------------------|--------------------------|--------------------------|---------------------------------|---------------------------------|
| Crystal Oscillator | Off Note 5.1 | On | On | Off Note 5.1 | Off Note 5.1 |
| USB 2.0 PHY | Off | Note 5.4 | Note 5.4 | On | Off |
| USB 3.0 PHY | Off | Note 5.3 | Note 5.3 | Off | Off |
| UDC 2.0 | Off | Note 5.4 | Note 5.4 | On | Off |
| UDC 3.0 | Off | Note 5.3 | Note 5.3 | Off | Off |
| USB 2.0 Common Block | Off | On | Note 5.4 | Off | Off |
| USB 3.0 Common Block | Off | Note 5.3 | Note 5.3 | Off | Off |
| System PLL | Off | Off | On | Off | Off |
| Pixel PLL | Off | Off | On | Off | Off |
| URX | Off | Off | On | Off | Off |
| CTL | Off | On | On | Off | Off |
| System CSRs | Off | On | On | Off | Off |
| FCT | Off | Off | On | Off | Off |
| GPH | Off | Off | On | Off | Off |
| DDR2 CTL | Off | Off | On | Off | Off |
| DDR2 PHY | Off | Off | On | Off | Off |
| VDAC | Off | Off | Note 5.5 | Off | Off |
| HDMI CTL | Off | Off | Note 5.5 | Off | Off |
| HDMI PHY | Off | Off | Note 5.5 | Off | Off |
| Digital RGB I/F | Off | Off | Note 5.5 | Off | Off |
| EPC | Off | On | On | Off | Off |
| I ² C | Off | Off | On | Off | Off |
| LED Pin | Off | On | Note 5.2 | Off | Off |
| nSW_MODE Pin | On | Off | Off | On | On |

Note 5.1 When the [Clock Crystal Keep Alive Enable \(CLK_XTAL_EN\)](#) bit of the [CPM Control Register \(CPM_CTL\)](#) is set, the crystal oscillator will never be turned off by the CPM.

Note 5.2 The LED pin behavior is a function of its configuration and the amount of received traffic.

Note 5.3 Enabled when connected on super-speed. These elements are also enabled as needed when attempting a super-speed connection and operating in USB 2.0 mode.

Note 5.4 Enabled when connected on USB 2.0. These elements also become enabled when operating in super-speed mode and the super-speed connection is lost.

Note 5.5 Only activated when the respective video interface is enabled and connected.

Note: The DDR2 controller should be placed in self-refresh by system software before entering SUSPEND.

Note: The HDMI PHY and VDAC default to being in the power down state.

5.3 Clock Generation

The below table lists the clocks generated by the CPM and their respective frequency range.

Table 5.3 Clocks

| CLOCK NAME | SOURCE | FREQUENCY RANGE | COMMENT |
|-----------------|----------------------|---|---|
| clk_xtal | USB 2.0 Common Block | 25 MHz | This clock is generated by the crystal oscillator which is incorporated in the USB 2.0 common block. |
| clk_sys_pll | System PLL output | Supported Operating Range: 250 MHz to 666 MHz Maximum Range of PLL: 5 MHz to 700 MHz | This clock is the output of the system PLL. The frequency is programmed via CSRs. |
| clk_sys_pll_180 | System PLL output | Supported Operating Range: 250 MHz to 666 MHz Maximum Range of PLL: 5 MHz to 700 MHz | This clock is a 180 degree phase shifted version of the clk_sys_pll. |
| clk_ddr2 | CPM | 125 MHz to 333 MHz | This clock generated from the output of the System PLL. It is obtained by digitally dividing the output of the System PLL by 2. |
| clk_sys | CPM | 62.5 MHz to 167 MHz | This clock generated by digitally dividing the output of the System PLL by 4. |
| clk_pixel_pll | Pixel PLL output | Supported Operating Range: 50 MHz to 400 MHz Maximum Range of PLL: 5 MHz to 700 MHz | This clock is the output of the Pixel PLL. The frequency is programmed via CSRs. |

Table 5.3 Clocks (continued)

| CLOCK NAME | SOURCE | FREQUENCY RANGE | COMMENT |
|-------------------|------------------|--|--|
| clk_pixel_pll_180 | Pixel PLL output | Supported Operating Range: 50 MHz to 400 MHz Maximum Range of PLL: 5 MHz to 700 MHz | This clock is a 180 degree phase shifted version of the clk_pixel_pll. This clock is used to assist in the generation of clk_pixel_90. |
| clk_pixel | CPM | 25 MHz to 200 MHz | Source clock for the display controller and video interface modules. Derived by dividing clk_pixel_pll in half. Note: The pixel clock does not exceed 165 MHz in HDMI mode. 200MHz is the limit for digital RGB and Video DAC. |
| clk_jpg | CPM | 240 MHz USB2.0 mode 250 MHz USB3.0 mode | Source clock for the JPEG Decoder. |
| clk_pixel_90 | CPM | 25 MHz to 200 MHz | 90 degree shifted version of clk_pixel. This clock is used to build the digital RGB interface when operating in 12-bit DDR mode. |
| clk_125m | USB 3.0 PHY | 125 MHz | This is derived from the PIPE clock |
| clk_60m | USB 2.0 PHY | 60 MHz | This is derived from the UTMI clock |
| clk_eep | CPM | 25 MHz | Based on clk_xtal |
| clk_pmt | CPM | 25 MHz | Based on clk_xtal |
| tck | TCK input pin | <= 10 MHz | Used by JTAG and test controller |
| clk_i2c | CPM | 2.08 MHz | Generated by digitally dividing clk_xtal. |

The CPM may generate additional intermediary clocks.

5.4 Pixel and System PLLs

5.4.1 Overview

The Pixel and System PLLs create the source clocks for the majority of the datapath. Both PLLs are instances of the same IP module procured from Analog Bits. They have been optimized for size, power, and accuracy. The last point is critical for the Pixel PLL, as the VESA DMT specification requires a 0.5% accuracy for supported pixel clocks.

5.4.2 PLL Programming

Initially both PLLs are disabled. The respective bypass and reset bits asserted. Only after the device is configured and the PLLs are programmed by software will they be operational.

The following steps are required for programming the PLLs into their normal operating mode.

1. All operation should commence by setting PLL_GATE_x for the PLL. This glitchlessly disables all clocks derived from the PLL.
2. Both stages of the PLL must be placed in the reset or bypass state (RESET_x=1 or BYPASS_x=1, x = 0 and 1) with the reference running stably and the clock feedback path intact.
3. Set normal operating mode for both PLL stages (RESET₀ and RESET₁ = 0 and BYPASS₀ and BYPASS₁ = 0).
4. Wait the specified lock time (100 us).
5. It is normal for the LOCK_x signal to come high before the specified lock time, and may glitch as jitter is acquired. This shows that the PLL has achieved a lock, but it will continue adjusting itself to a more perfect operating point. In a jittery environment, it is possible that LOCK_x will not assert, but this does not mean that the PLL stage is not working.
6. Clear PLL_GATE_x for the PLL. This glitchlessly enables all clocks derived from the PLL.
7. To leave normal operating mode, assert PLL_GATE_x and RESET_x or BYPASS_x for both PLL stages.

The DDR2 Controller must be placed in self-refresh before changing the frequency of the System PLL. Refer to [Section 7.4, "Operating Modes," on page 111](#) for a description of how to place the controller in self refresh.

Note: All traffic to the controller must be disabled, otherwise it will not enter self-refresh.

5.4.3 PLL Programming Examples

Table 5.4 illustrates how some typical operating frequencies are generated.

Table 5.4 PLL Frequency Configurations

| PLL STAGE 0 | | | | | | PLL STAGE 1 | | | | | | TARGET (MHZ) |
|--|----------------|--|------------------|----------------|------------------|--|----------------|--|------------------|----------------|------------------|-----------------|
| 1-32 | 5-200 (MHZ) | 1-256 | 350-700 (MHZ) | 1-64 (2^N) | 5-700 (MHZ) | 1-32 | 5-200 (MHZ) | 1-256 | 350-700 (MHZ) | 1-64 (2^N) | 5-700 (MHZ) | |
| DIVR0 [4:0] Note 5.6 | REF0 (MHZ) | DIVF0 [7:0] Note 5.6 | VCO0 (MHZ) | DIVQ0 [2:0] | PLLOUT0 (MHZ) | DIVR1 [4:0] Note 5.6 | REF1 (MHZ) | DIVF1 [7:0] Note 5.6 | VCO1 (MHZ) | DIVQ1 [2:0] | PLLOUT1 (MHZ) | |
| Note 5.7 | | | | | 25 | 1 | 25 | 16 | 400 | 8 | 50 | 50.000 |
| 1 | 25 | 16 | 400 | 1 | 400 | 5 | 80 | 6 | 480 | 2 | 240 | 240.000 |
| Note 5.7 | | | | | 25 | 1 | 25 | 20 | 500 | 2 | 250 | 250.000 |
| Note 5.7 | | | | | 25 | 5 | 5 | 132 | 660 | 2 | 330 | 330.000 |
| 1 | 25 | 16 | 400 | 1 | 400 | 3 | 133.333 | 5 | 666.667 | 2 | 333.333 | 333.333 |
| Note 5.7 | | | | | 25 | 1 | 25 | 16 | 400 | 1 | 400 | 400.000 |
| 1 | 25 | 16 | 400 | 1 | 400 | 3 | 133.333 | 4 | 533.333 | 1 | 533.333 | 533.333 |
| 1 | 25 | 16 | 400 | 1 | 400 | 3 | 133.333 | 5 | 666.667 | 1 | 333.333 | 666.667 |

Note 5.6 The value to be written to the corresponding CSR field is the value indicated plus 1.

Note 5.7 Single stage configurations with PLL Stage 0 set to BYPASS mode.

Note: REF0, VCO0, REF1, and VCO1 are internal nodes, not primary inputs or outputs of the PLL.

5.4.4 DDR2 Clock Frequency Considerations

DDR2 controller operations required when changing clock frequencies are described herein:

1. Request the controller to enter self refresh by asserting the [Self Refresh Enable \(REG_DDRC_SELFREF_EN\)](#) bit of the [DDR2 Dynamic Register \(DDR_DYN_REG\)](#).
2. Wait until [DDR2 Controller Operating Mode \(DDRC_REG_OPERATING_MODE\)](#) field of [DDR2 Control Register 10 \(DDR_CONTROL_10\)](#) == 3h, indicating that the DRAM is in self refresh.
3. Glitchlessly change the clock frequency to the controller.
4. Update any registers which may be required to change for the new frequency.
5. Take the controller out of self refresh by de-asserting the [Self Refresh Enable \(REG_DDRC_SELFREF_EN\)](#) bit of the [DDR2 Dynamic Register \(DDR_DYN_REG\)](#).

5.5 USB Suspend

The [Figure 5.3](#) illustrates the steps taken by system software and the CPM to place the device into the SUSPEND state from the CONFIGURED state.

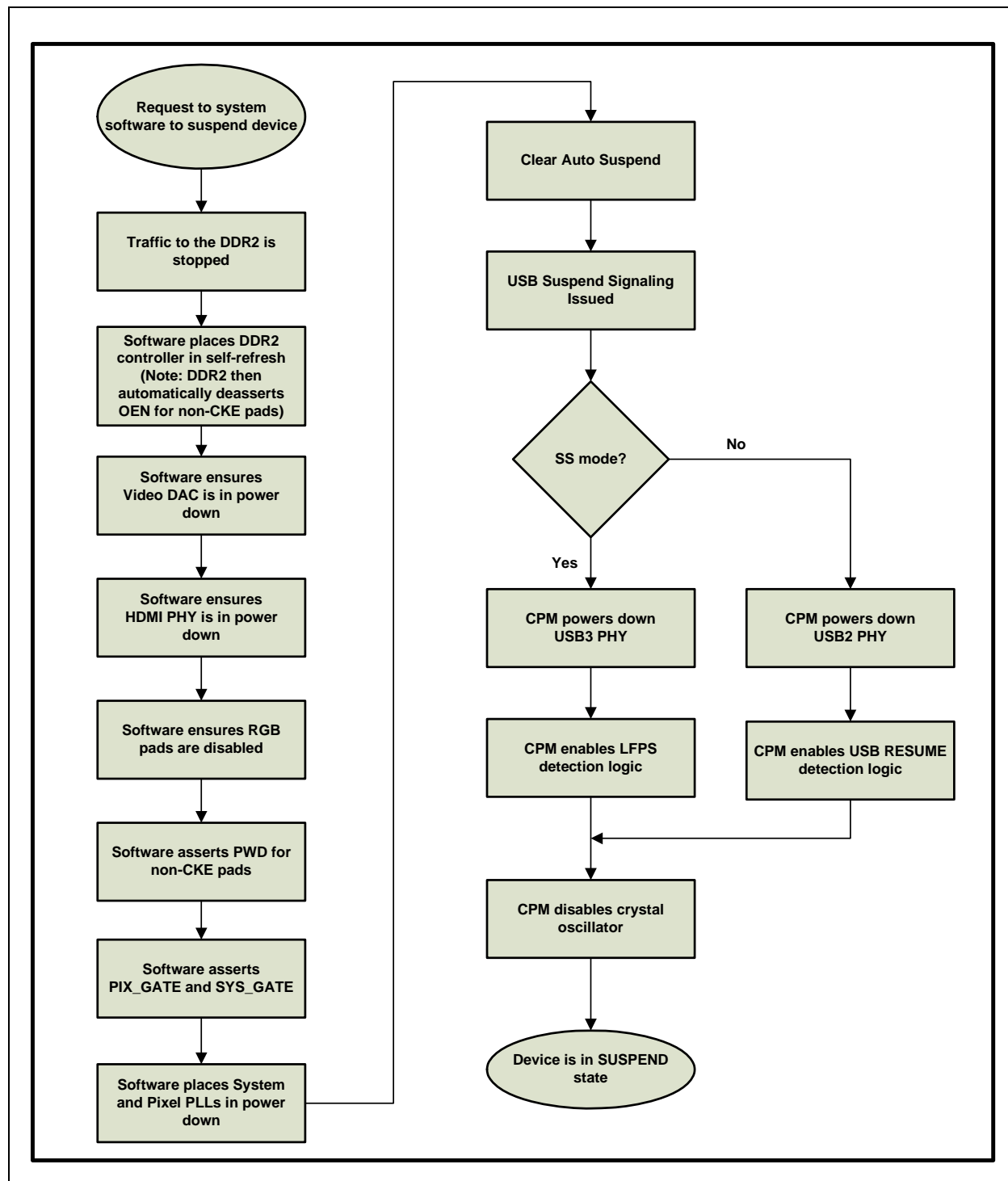


Figure 5.3 USB Suspend Sequence

DDR2 self-refresh, HDMI PHY power down, Video DAC power down, and Digital RGB pads are controlled via CSRs. The [Self Refresh Enable \(REG_DDRC_SELFREF_EN\)](#) bit of the [DDR2 Dynamic Register \(DDR_DYN_REG\)](#) controls DDR2 self-refresh. HDMI PHY power down is controlled by the [Enable HDMI PHY \(EN_HDMI\)](#) bit of the [HDMI Configuration Register \(HDMI_CFG\)](#). Video DAC power down is controlled by the [Enable DAC Channel 0 \(EN_DAC0\)](#), [Enable DAC Channel 1 \(EN_DAC1\)](#), and [Enable DAC Channel 2 \(EN_DAC2\)](#) bits of the [Video DAC Configuration Register \(VDAC_CFG\)](#). The Digital RGB pads are controlled by the [RGB Configuration Register \(RGB_CFG\)](#).

The PWD and OEN for DDR2 non-CKE pads are controlled by the [DDR2 Pads Power Down Control \(REG_DDR2_PWD\)](#) and the [DDR2 Pads Output Enable Disable \(REG_DDR2_OE_DISABLE\)](#) bits of the [DDR2 PHY Control Register 0 \(DDR_PHY_CTL_0\)](#).

Note: Setting the [Self Refresh Enable \(REG_DDRC_SELFREF_EN\)](#) bit of the [DDR2 Dynamic Register \(DDR_DYN_REG\)](#) automatically disables OEN for non-CKE pads. The [DDR2 Pads Output Enable Disable \(REG_DDR2_OE_DISABLE\)](#) bit does not need to be programmed to do this.

The DCTL interface select CSRs disable the respective video interface. The clock to the interface is shut down when the interface is disabled. Refer to the [Interface 0 Select \(INTF_SEL_0\)](#), [Interface 1 Select \(INTF_SEL_1\)](#), and [Interface 2 Select \(INTF_SEL_2\)](#) bits of the [Display Interface Control 0 Register \(DISPLAY_INTERFACE_CTRL_0\)](#), [Display Interface Control 1 Register \(DISPLAY_INTERFACE_CTRL_1\)](#), and the [Display Interface Control 2 Register \(DISPLAY_INTERFACE_CTRL_2\)](#), respectively, for additional information.

The actions illustrated in [Figure 5.4](#) are performed when the [Auto Suspend \(AUTO_SUSPEND\)](#) bit of the [CPM Control Register \(CPM_CTL\)](#) is set. This feature is used when the driver is unable to manually place the part in suspend.

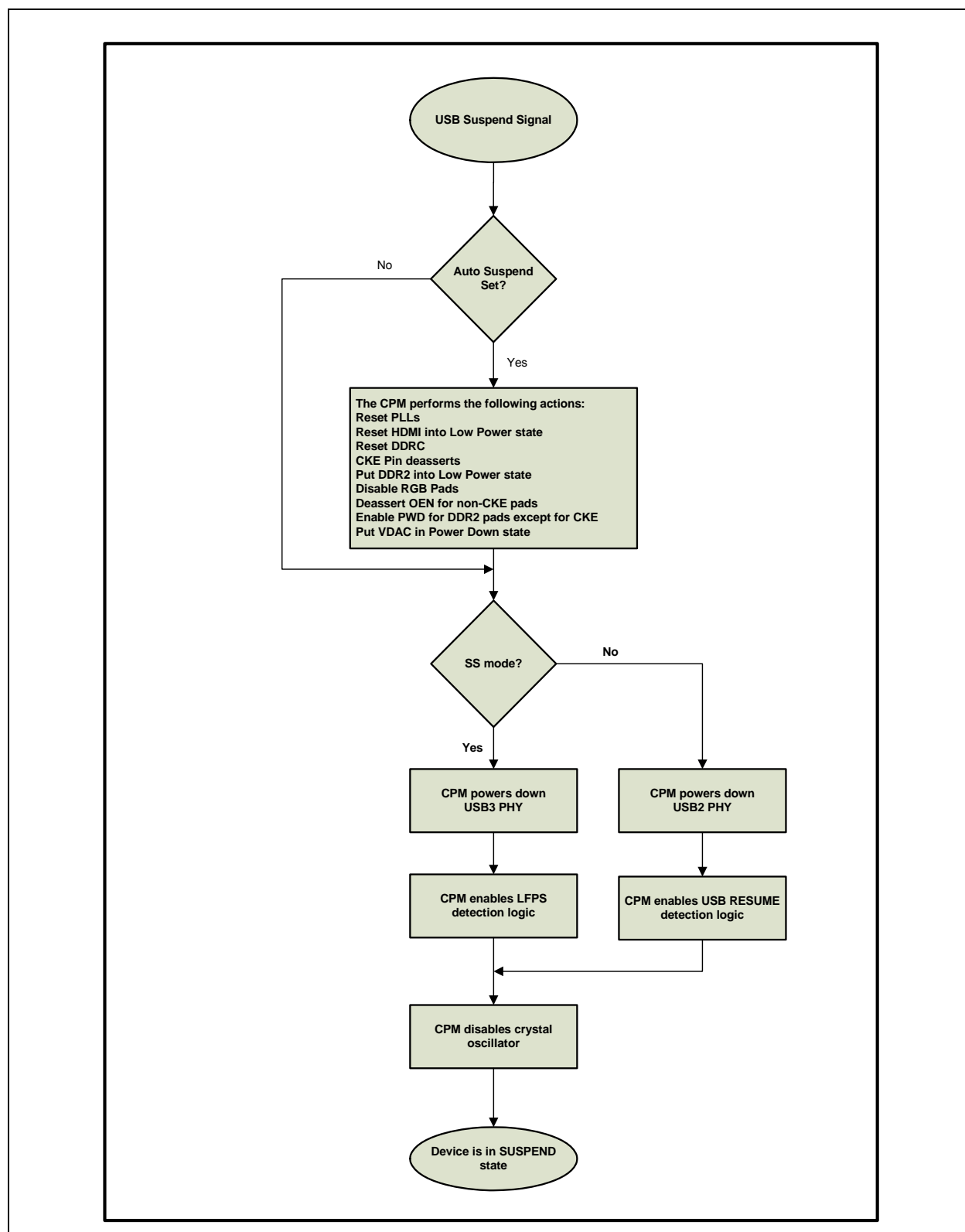


Figure 5.4 Auto Suspend Actions

Note: Software should issue a [Lite Reset \(LRST\)](#) when resuming, as portions of the chip are reset during the Auto Suspend sequence.

5.6 USB Resume

The [Figure 5.5](#) illustrates the steps taken by system software and the CPM to place the device into the SUSPEND state from the CONFIGURED state.

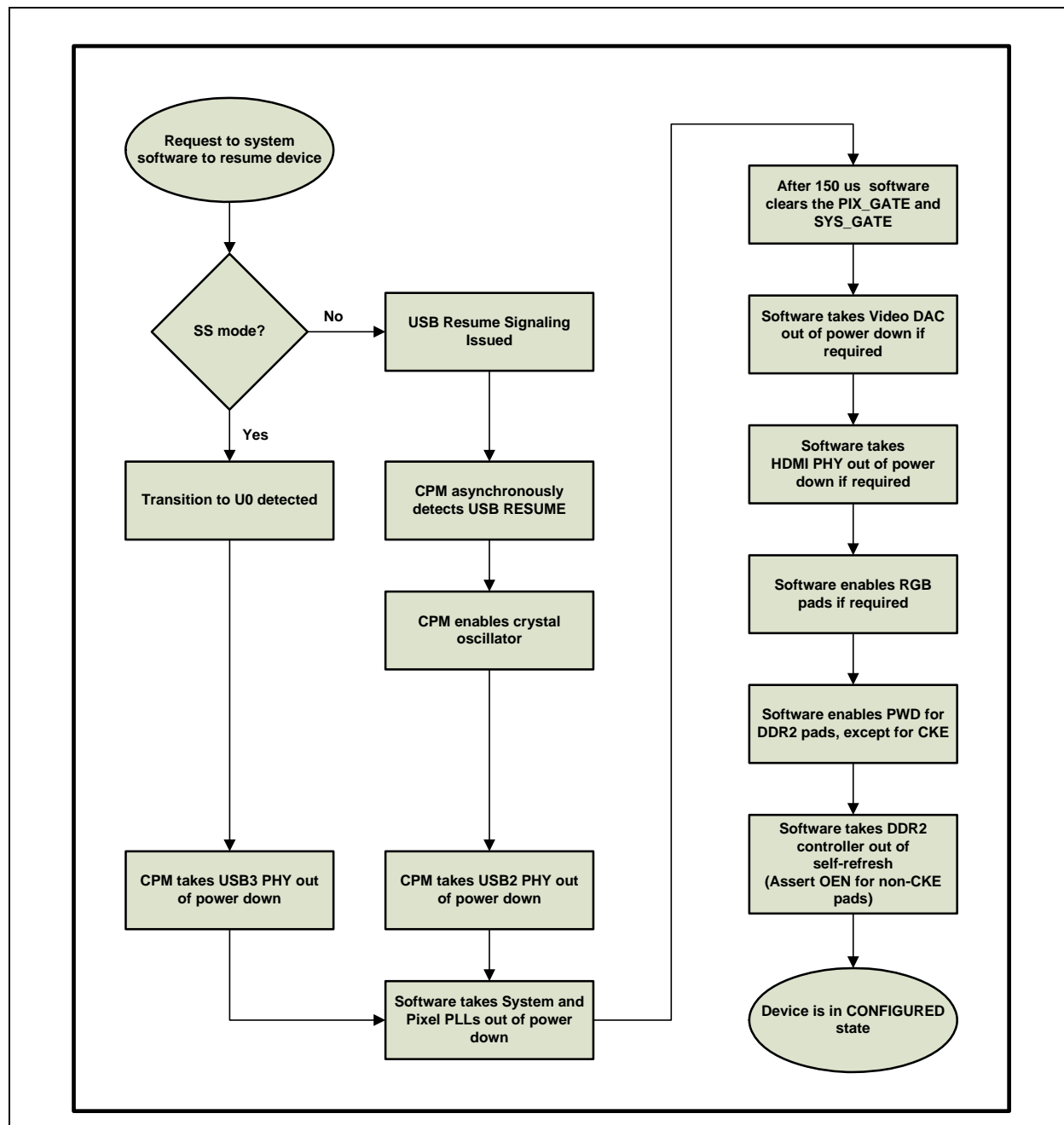


Figure 5.5 USB Resume Sequence

5.7 LPM Support

The device supports LPM.

Table 5.5 LPM States

| LPM STATE | DESCRIPTION |
|--------------|--|
| L0 (On) | This is the full operational state of the device. |
| L1 (Sleep) | This state is meant to provide finer granularity than the L2 state. Because of the short resume time required per LPM spec the CPM will not power down any modules. Note: There are no specific power draw requirements on VBUS in this state. |
| L2 (Suspend) | This state is equivalent to USB suspend state. |
| L3 (Off) | In this state, the device is not capable of performing any data signaling. It corresponds to the USB powered-off, disconnected, and disabled states. |

5.8 U0, U1, U2, U3 States

Table 5.6 LFPS States

| LFPS STATE | DESCRIPTION |
|------------|---|
| U0 | This is the CONFIGURED state after PLLs and datapath have been configured to receive/transmit data. |
| U1 | CPM does not take any actions to suspend power. |
| U2 | CPM does not take any actions to suspend power. |
| U3 | Equivalent to the SUSPEND state. |

5.9 Control and Status Registers

The CPM CSRs are implemented in the clk_xtal domain. Therefore, they can be accessed via the control endpoint before enabling the System or Pixel PLLs.

Table 5.7 Clocks and Power Management Control and Status Register Map

| ADDRESS OFFSET | SYMBOL | REGISTER NAME |
|-------------------|--------------|---|
| 7000h | PIX_PLL_CTL | Pixel Clock PLL Control Register |
| 7004h | PIX_PLL0_CFG | Pixel Clock PLL Stage 0 Configuration Register |
| 7008h | PIX_PLL1_CFG | Pixel Clock PLL Stage 1 Configuration Register |
| 700Ch | SYS_PLL_CTL | System Clock PLL Control Register |
| 7010h | SYS_PLL0_CFG | System Clock PLL Stage 0 Configuration Register |
| 7014h | SYS_PLL1_CFG | System Clock PLL Stage 1 Configuration Register |
| 7018h – 701Ch | RESERVED | Reserved for future expansion |
| 7020h | CPM_CTL | CPM Control Register |
| 7024h-7FFFh | RESERVED | Reserved for future expansion |

5.9.1 Pixel Clock PLL Control Register (PIX_PLL_CTL)

Offset: 0000h Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|---|------|---------|
| 31 | PIX PLL Gate (PIX_GATE) When set, will glitchlessly stop all clocks derived from the PIX PLL. When cleared, will glitchlessly enable all clocks derived from the PIX PLL. Note: This bit should only be cleared when the PIX PLL is in a stable state. | R/W | 1b |
| 30:6 | RESERVED | RO | - |
| 5 | PIX PLL Stage 1 Lock (PIX_LOCK1) When high, this bit indicates that PIX PLL Stage 1 has locked on to the incoming signal. It indicates that the PLL has achieved frequency lock with a good phase lock. PIX_LOCK0 and PIX_LOCK1 are asynchronous. Note: In the case of jittery source clock or feedback tree, it is possible that this bit does not assert because the phases do not match perfectly. Note: It is recommended that PIX_LOCK0 and PIX_LOCK1 are only used for test and system status information and not for critical system functions without thorough characterization in the host system. | RO | 0b |
| 4 | PIX PLL Stage 0 Lock (PIX_LOCK0) When high, this bit indicates that PIX PLL Stage 0 has locked on to the incoming signal. It indicates that the PLL has achieved frequency lock with a good phase lock. PIX_LOCK0 and PIX_LOCK1 are asynchronous. Note: In the case of jittery source clock or feedback tree, it is possible that this bit does not assert because the phases do not match perfectly. Note: It is recommended that PIX_LOCK0 and PIX_LOCK1 are only used for test and system status information and not for critical system functions without thorough characterization in the host system. | RO | 0b |
| 3 | PIX PLL Stage 1 Bypass (PIX_BYPASS1) A BYPASS signal is provided for each PLL stage, which both powers-down the PLL stage and bypasses it such that PIX PLLOUT/ PIX PLLOUTB tracks PIX REF. When this bit is asserted, the PIX PLLOUT/PIX PLLOUTB output tracks the PIX PLLOUT0 node (input to the Stage 1 PLL). Note: In order to completely bypass both PLL Stages (from PIX REF to PIX PLLOUT/PIX PLLOUTB), both PIX_BYPASS0 and PIX_BYPASS1 will need to be asserted simultaneously. | R/W | 1b |
| 2 | PIX PLL Stage 0 Bypass (PIX_BYPASS0) A BYPASS signal is provided for each PLL stage, which both powers-down the PLL stage and bypasses it such that PIX PLLOUT/ PIX PLLOUTB tracks PIX REF. When this bit is asserted, the node PIX PLLOUT0 tracks the PIX REF input. Note: In order to completely bypass both PLL Stages (from PIX REF to PIX PLLOUT/PIX PLLOUTB), both PIX_BYPASS0 and PIX_BYPASS1 will need to be asserted simultaneously. | R/W | 1b |
| 1 | PIX PLL Stage 1 Reset (PIX_RESET1) A RESET signal is provided for each PLL stage to power down the PLL and reset it to a known state (low). When this bit is asserted, PIX PLL stage 1 will be powered down and PIX PLLOUT/PIX PLLOUTB will be held low. | R/W | 1b |

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|--|------|---------|
| 0 | PIX PLL Stage 0 Reset (PIX_RESET0) A RESET signal is provided for each PLL stage to power down the PLL and reset it to a known state (low). When this bit is asserted, PIX PLL stage 0 will be powered down and node PIX PLLOUT0 will be held low. | R/W | 1b |

5.9.2 Pixel Clock PLL Stage 0 Configuration Register (PIX_PLL0_CFG)

Offset: 0004h Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|---|------|---------|
| 31:23 | RESERVED | RO | - |
| 22:20 | PIX PLL Stage 0 Filter Range (PIX_RANGE0) This field sets the PLL loop filter for PIX PLL Stage 0 to work with the post-reference divider frequency (PIX REF0). The highest valid range should be specified for best jitter performance. 000 = BYPASS 001 = 5 - 10 MHz 010 = 8 - 16 MHz 011 = 13 - 26 MHz 100 = 21 - 42 MHz 101 = 34 - 68 MHz 110 = 54 - 108 MHz 111 = 88 - 200 MHz | R/W | 000b |
| 19 | RESERVED | RO | - |
| 18:16 | PIX PLL Stage 0 Output Divider Value (PIX_DIVQ0) The Output Divider value is 2 raised to the power of this field, i.e., $2^0 = 1$. PIX PLLOUT0 should be within the range of 350 MHz to 700 MHz. (When either of the PLL Stages are in BYPASS, PIX PLLOUT0 should be within the range of 5MHz to 200 MHz.) Note: 111 is not a valid setting for this field. Only divide by 1-64 ($2^0 - 2^6$) are allowed. | R/W | 000b |
| 15:13 | RESERVED | RO | - |
| 12:8 | PIX PLL Stage 0 Reference Divider Value (PIX_DIVR0) The Reference Divider value is the value of this field + 1. Note: PIX REF and divided PIX REF (PIX REF0) must be within the range of 5MHz to 200MHz | R/W | 00000b |
| 7:0 | PIX PLL Stage 0 Feedback Divider Value (PIX_DIVF0) The Feedback Divider value is the value of this field + 1. Note: VCO0 must be with the range of 350 MHz to 700 MHz. | R/W | 00h |

5.9.3 Pixel Clock PLL Stage 1 Configuration Register (PIX_PLL1_CFG)

Offset: 0008h Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|---|------|---------|
| 31:23 | RESERVED | RO | - |
| 22:20 | PIX PLL Stage 1 Filter Range (PIX_RANGE1) This field sets the PLL loop filter for PIX PLL Stage 1 to work with the post-reference divider frequency (PIX REF1). The highest valid range should be specified for best jitter performance. 000 = BYPASS 001 = 5 - 10 MHz 010 = 8 - 16 MHz 011 = 13 - 26 MHz 100 = 21 - 42 MHz 101 = 34 - 68 MHz 110 = 54 - 108 MHz 111 = 88 - 200 MHz | R/W | 000b |
| 19 | RESERVED | RO | - |
| 18:16 | PIX PLL Stage 1 Output Divider Value (PIX_DIVQ1) The Output Divider value is 2 raised to the power of this field, i.e., $2^0 = 1$. Note: 111 is not a valid setting for this field. Only divide by 1-64 ($2^0 - 2^6$) are allowed. | R/W | 000b |
| 15:13 | RESERVED | RO | - |
| 12:8 | PIX PLL Stage 1 Reference Divider Value (PIX_DIVR1) The Reference Divider value is the value of this field + 1. Note: The PIX REF1 node must be within the range of 5MHz to 200MHz | R/W | 00000b |
| 7:0 | PIX PLL Stage 1 Feedback Divider Value (PIX_DIVF1) The Feedback Divider value is the value of this field + 1. Note: VCO1 must be with the range of 350 MHz to 700 MHz. | R/W | 00h |

5.9.4 System Clock PLL Control Register (SYS_PLL_CTL)

Offset: 000Ch Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|---|------|---------|
| 31 | SYS PLL Gate (SYS_GATE) When set, will glitchlessly stop all clocks derived from the SYS PLL. When cleared, will glitchlessly enable all clocks derived from the SYS PLL. Note: This bit should only be cleared when the SYS PLL is in a stable state. | R/W | 1b |
| 30:6 | RESERVED | RO | - |
| 5 | SYS PLL Stage 1 Lock (SYS_LOCK1) When high, this bit indicates that SYS PLL Stage 1 has locked on to the incoming signal. It indicates that the PLL has achieved frequency lock with a good phase lock. SYS_LOCK0 and SYS_LOCK1 are asynchronous. Note: In the case of jittery source clock or feedback tree, it is possible that this bit does not assert because the phases do not match perfectly. Note: It is recommended that SYS_LOCK0 and SYS_LOCK1 are only used for test and system status information and not for critical system functions without thorough characterization in the host system. | RO | 0b |
| 4 | SYS PLL Stage 0 Lock (SYS_LOCK0) When high, this bit indicates that SYS PLL Stage 0 has locked on to the incoming signal. It indicates that the PLL has achieved frequency lock with a good phase lock. SYS_LOCK0 and SYS_LOCK1 are asynchronous. Note: In the case of jittery source clock or feedback tree, it is possible that this bit does not assert because the phases do not match perfectly. Note: It is recommended that SYS_LOCK0 and SYS_LOCK1 are only used for test and system status information and not for critical system functions without thorough characterization in the host system. | RO | 0b |
| 3 | SYS PLL Stage 1 Bypass (SYS_BYPASS1) A BYPASS signal is provided for each PLL stage, which both powers-down the PLL stage and bypasses it such that SYS PLLOUT/ SYS PLLOUTB tracks SYS REF. When this bit is asserted, the SYS PLLOUT/SYS PLLOUTB output tracks the SYS PLLOUT0 node (input to the Stage 1 PLL). Note: In order to completely bypass both PLL Stages (from SYS REF to SYS PLLOUT/SYS PLLOUTB), both SYS_BYPASS0 and SYS_BYPASS1 will need to be asserted simultaneously. | R/W | 1b |
| 2 | SYS PLL Stage 0 Bypass (SYS_BYPASS0) A BYPASS signal is provided for each PLL stage, which both powers-down the PLL stage and bypasses it such that SYS PLLOUT/ SYS PLLOUTB tracks SYS REF. When this bit is asserted, the node SYS PLLOUT0 tracks the SYS REF input. Note: In order to completely bypass both PLL Stages (from SYS REF to SYS PLLOUT/SYS PLLOUTB), both SYS_BYPASS0 and SYS_BYPASS1 will need to be asserted simultaneously. | R/W | 1b |

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|--|------|---------|
| 1 | SYS PLL Stage 1 Reset (SYS_RESET1) A RESET signal is provided for each PLL stage to power down the PLL and reset it to a known state (low). When this bit is asserted, SYS PLL stage 1 will be powered down and SYS PLLOUT/SYS PLLOUTB will be held low. | R/W | 1b |
| 0 | SYS PLL Stage 0 Reset (SYS_RESET0) A RESET signal is provided for each PLL stage to power down the PLL and reset it to a known state (low). When this bit is asserted, SYS PLL stage 0 will be powered down and node SYS PLLOUT0 will be held low. | R/W | 1b |

5.9.5 System Clock PLL Stage 0 Configuration Register (SYS_PLL0_CFG)

Offset: 0010h Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|---|------|---------|
| 31:23 | RESERVED | RO | - |
| 22:20 | SYS PLL Stage 0 Filter Range (SYS_RANGE0) This field sets the PLL loop filter for SYS PLL Stage 0 to work with the post-reference divider frequency (SYS_REF0). The highest valid range should be specified for best jitter performance. 000 = BYPASS 001 = 5 - 10 MHz 010 = 8 - 16 MHz 011 = 13 - 26 MHz 100 = 21 - 42 MHz 101 = 34 - 68 MHz 110 = 54 - 108 MHz 111 = 88 - 200 MHz | R/W | 000b |
| 19 | RESERVED | RO | - |
| 18:16 | SYS PLL Stage 0 Output Divider Value (SYS_DIVQ0) The Output Divider value is 2 raised to the power of this field, i.e., $2^0 = 1$. SYS_PLLOUT0 should be within the range of 350 MHz to 700 MHz. (When either of the PLL Stages are in BYPASS, SYS_PLLOUT0 should be within the range of 5MHz to 200 MHz.) Note: 111 is not a valid setting for this field. Only divide by 1-64 ($2^0 - 2^6$) are allowed. | R/W | 000b |
| 15:13 | RESERVED | RO | - |
| 12:8 | SYS PLL Stage 0 Reference Divider Value (SYS_DIVR0) The Reference Divider value is the value of this field + 1. Note: SYS_REF and divided SYS_REF (SYS_REF0) must be within the range of 5MHz to 200MHz | R/W | 00000b |
| 7:0 | SYS PLL Stage 0 Feedback Divider Value (SYS_DIVF0) The Feedback Divider value is the value of this field + 1. Note: VCO0 must be with the range of 350 MHz to 700 MHz. | R/W | 00h |

5.9.6 System Clock PLL Stage 1 Configuration Register (SYS_PLL1_CFG)

Offset: 0014h Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|---|------|---------|
| 31:23 | RESERVED | RO | - |
| 22:20 | SYS PLL Stage 1 Filter Range (SYS_RANGE1) This field sets the PLL loop filter for SYS PLL Stage 1 to work with the post-reference divider frequency (SYS REF1). The highest valid range should be specified for best jitter performance. 000 = BYPASS 001 = 5 - 10 MHz 010 = 8 - 16 MHz 011 = 13 - 26 MHz 100 = 21 - 42 MHz 101 = 34 - 68 MHz 110 = 54 - 108 MHz 111 = 88 - 200 MHz | R/W | 000b |
| 19 | RESERVED | RO | - |
| 18:16 | SYS PLL Stage 1 Output Divider Value (SYS_DIVQ1) The Output Divider value is 2 raised to the power of this field, i.e., $2^0 = 1$. Note: 111 is not a valid setting for this field. Only divide by 1-64 ($2^0 - 2^6$) are allowed. | R/W | 000b |
| 15:13 | RESERVED | RO | - |
| 12:8 | SYS PLL Stage 1 Reference Divider Value (SYS_DIVR1) The Reference Divider value is the value of this field + 1. Note: The SYS REF1 node must be within the range of 5MHz to 200MHz | R/W | 00000b |
| 7:0 | SYS PLL Stage 1 Feedback Divider Value (SYS_DIVF1) The Feedback Divider value is the value of this field + 1. Note: VCO1 must be with the range of 350 MHz to 700 MHz. | R/W | 00h |

5.9.7 CPM Control Register (CPM_CTL)

Offset: 0020h Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|---|------|--------------------------|
| 31:3 | RESERVED | RO | - |
| 2 | Clock JPEG Source Select (CLK_JPG_SRC_SEL) Selects the source for clk_jpg. This clock is only used by the JPEG decoder. This bit does not affect the operation of the JPEG-LS decoder. 0b: clk_jpg operates at 240 MHz when in USB 2.0 mode or 250 MHz in USB3 mode. 1b: clk_jpg uses clk_sys as its source. Note: Using clk_sys as a source will lower the performance of the JPEG decoder, however it will result in lower power consumption. When operating in lower resolutions it may be advisable to use clk_sys as the source. | R/W | 0b |
| 1 | Clock Crystal Keep Alive Enable (CLK_XTAL_EN) When set, the xtal oscillator will never be turned off by the CPM. Note: This field is protected by Reset Protection (RST_PROTECT) | R/W | Note 5.8 |
| 0 | Auto Suspend (AUTO_SUSPEND) When set, the CPM puts the chip in low power state automatically when SUSPENDED. Refer to Figure 5.4 Auto Suspend Actions on page 85 for further details. | R/W | 1b |

Note 5.8 The default value of this bit is determined by the value of the [Clock Crystal Keep Alive Enable \(CFG0_CLK_XTAL_EN\)](#) bit of [Configuration Flags 0](#) contained within the EEPROM, if present. If no EEPROM is present, 0b is the default. A USB Reset or Lite Reset (LRST) will cause this field to be restored to the image value last loaded from EEPROM, or to be set to 0b if no EEPROM is present.

Chapter 6 Reset Generator

6.1 Description

The reset generator module (RST) drives all of the resets required by the device. This module asserts resets as a result of hardware or software events as described below. The reset generator has various clock inputs. The resets are deasserted synchronously to the corresponding clock.

[Table 6.1](#) describes the chip level reset events.

Table 6.1 Chip Level Resets

| RESET EVENT | DESCRIPTION |
|-------------------------------------|---|
| External Chip Reset (nRESET) | This active-low signal is an input from the external nRESET pin. Note: The EEPROM contents are loaded by this reset. |
| Power on Reset (POR) | POR is based on the 1.8V (POR from DDR2 pad cells), 3.3V, and 1.2 V (both POR from the USB PHY) being stable. Note: The EEPROM contents are loaded by this reset. |
| Lite Reset (LRST) | This reset is initiated via the Soft Lite Reset (LRST) bit of the Hardware Configuration Register (HW_CFG) . Resets the entire chip with the exception of the enabled USB device controller, control endpoint and USB PHY. Note: This reset does not cause the USB contents from the EEPROM to be reloaded. Note: This reset does not place the device into the Unconfigured state. Note: After the LRST, the USB pipes corresponding to the Bulk Out and Interrupt endpoints must be reset. This process entails clearing the device's ENDPOINT_HALT feature and resetting the data toggle on the host side. Note: The System and Pixel PLLs are reset. |

Table 6.1 Chip Level Resets (continued)

| RESET EVENT | DESCRIPTION |
|---|---|
| Software Reset (SRST) | <p>This reset is initiated via the Soft Reset (SRST) bit in the Hardware Configuration Register (HW_CFG). It will cause the entire chip to reset and the USB PHY to disconnect.</p> <p>Note: The EEPROM contents are reloaded by this reset.</p> <p>Note: Writing SRST=1 will cause the device to disconnect from the USB shortly after the first good OUT Data packet during the Data Phase.</p> <p>In SS/HS mode, a brief delay will allow enough time for the device to send the ACK for the Data Stage, but the device will be disconnected (causing a 3-strikes timeout failure) for any next transaction (e.g., the Status Stage, or a repeated Data Stage, if there were any bus errors).</p> <p>In FS mode, the brief delay will be short enough that the device will disconnect during the ACK packet, causing CRC, bit-stuff, etc. errors on USB.</p> <p>To the USB Host, the aforementioned behaviors are the same as what happens during any Surprise Removal of a USB Device. This behavior is completely normal, and a compliant Host must be tolerant of it.</p> |
| Auto Suspend | <p>This reset is asserted whenever the Auto Suspend (AUTO_SUSPEND) bit in the CPM Control Register (CPM_CTL) is set and a USB SUSPEND signal is received. Refer to Figure 5.4 Auto Suspend Actions on page 85 for details on CPM actions and portions of the chip that are reset when this event occurs. It is recommended that software should issue a Lite Reset (LRST) after resuming.</p> <p>Note: This reset does not cause the contents from the EEPROM to be reloaded.</p> |
| USB Reset | <p>Resets the entire chip with the exception of the enabled USB device controller, control endpoint and USB PHY.</p> <p>Note: This reset does not cause the contents from the EEPROM to be reloaded.</p> |
| TAP Reset (nTRST) | This active-low reset is used by the TAP controller. |
| HDMI Reset | This reset is initiated via the Reset HDMI (RST_HDMI) bit in the HDMI Configuration Register (HDMI_CFG) . It will cause the HDMI controller to be reset. |
| External Transmitter Reset (nEXTRST) | <p>This reset is used for an external transmitter. This reset is asserted under the following conditions.</p> <ul style="list-style-type: none"> ■ When the device is in the USB UNCONFIGURED state. After the device is configured the reset is deasserted. ■ When programmed to be asserted via the External Reset Configuration Register (EXT_RST_CFG). |
| VBUS Detection (VBUS_DET) | <p>The removal of USB power causes the device to transition to the UNPOWERED state. The chip is held in reset while in the UNPOWERED state.</p> <p>Note: After VBUS is applied, the contents of the EEPROM are reloaded.</p> |

Table 6.1 Chip Level Resets (continued)

| RESET EVENT | DESCRIPTION |
|-------------------|--|
| Warm Reset | Resets the entire chip with the exception of the enabled USB 3.0 device controller, control endpoint and USB 3.0 PHY. Note: This reset does not cause the contents from the EEPROM to be reloaded. |
| Hot Reset | Resets the entire chip with the exception of the enabled USB 3.0 device controller, control endpoint and USB 3.0 PHY. Note: This reset does not cause the contents from the EEPROM to be reloaded. |

Chapter 7 Memory Controller

7.1 Overview

The device contains a DDR2 controller and PHY. The DDR2 controller operates at a speed up to 333 MHz, while the DDR2 PHY operates at maximum of DDR2-666. While the DDR2 PHY has a 13-bit address bus capable of supporting up to 512 Mbit DDR2 RAM, only 12 address bits are used, as the device only utilizes a 256 Mbit RAM. The DDR2 PHY has 2 bank pins, thus supports up to four memory banks. Its data bus is 16-bits wide. The device can read and write in burst mode with a burst length of 16-bits.

The DDR2 controller's flexible address mapper logic allows for application specific mapping of system addresses to DRAM addresses (rank, bank, row, and column). The controller performs dynamic scheduling to optimize bandwidth and latency. It prioritizes requests to minimize the latency of reads (especially high priority reads) and maximize page hits. It also ensures that the DRAM is properly initialized, that all requests are legal (depending on DRAM constraints), that refreshes are inserted as required, and that the DRAM enters and exits power-saving modes in an appropriate manner. Delayed writes are also used for optimum performance on DRAM data bus.

The device's advanced power-saving design includes no unnecessary toggling of command, address, and data pins (CA/RAS/CAS/WE/BA/A hold last state after each command; DQ does not transition on writes when bytes are disabled).

DRAM operational parameters are programmable via Control and Status Registers (CSR). Support is provided for explicit software-controlled mode register updates, as well as for 2T timing, where the clock edge that the chip select signal is asserted (the address and command is registered into memory) is statically selectable via a programming register.

There are four modes of operations: Uninitialized, Normal, Powerdown, and Self Refresh. Autonomous DRAM power down with entry and exit based on programmable idle periods is supported. Support for Self Refresh entry on software command and automatic exit on command arrival is provided. The clock frequency may be dynamically changed while in Self Refresh.

When in USB SUSPEND mode, DDR2 pads will be disabled, with the exception of CKE. Pads are disabled only after the controller has been being successfully placed in Self Refresh mode. ODT is supported in pads, however, ODT tuning is not available. ODT tolerance across PVT is +/- 20%.

Access to DDR2 RAM is provided via a register based Data Port. The [Data Port Select Register \(DP_SEL\)](#) is used to select the DDR2 RAM, while the [Data Port Address Register \(DP_ADDR\)](#) is used to specify the address of the DDR2 RAM element to be accessed. Read/Write selection is controlled by the [Data Port Command Register \(DP_CMD\)](#). Data to Read/Written is accessed via the [Data Port Data 0 Register \(DP_DATA0\)](#). Refer to the definition of these registers for details on their use.

7.2 Description of Controller Functions

7.2.1 Address Mapper

Read and write requests are provided to the controller with a system address. The controller is responsible for mapping this system address to rank, bank, row, and column address to DRAM.

7.2.1.1 DDRC Register Fields Related to the Address Mapper

The following CSR register fields are related to DRAM Initialization and must be set appropriately:

[DDR2 Bank/CS Address Mapping Register \(DDR_AD_MAP_BA_CS\)](#):

[Address Map Bank Address Bit 0 \(REG_DDRC_ADDRMAP_BANK_B0\)](#)

[Address Map Bank Address Bit 1 \(REG_DDRC_ADDRMAP_BANK_B1\)](#)

[Address Map Bank Address Bit 2 \(REG_DDRC_ADDRMAP_BANK_B2\)](#)

Address Map Rank Address Bit 0 (REG_DDRC_ADDRMAP_CS_BIT0)

DDR2 Column Address Mapping Register (DDR_AD_MAP_COL):

Address Map Column Address Bit 2 (REG_DDRC_ADDRMAP_COL_B2)

Address Map Column Address Bit 3 (REG_DDRC_ADDRMAP_COL_B3)

Address Map Column Address Bits 4 To 6 (REG_DDRC_ADDRMAP_COL_B4_6)

Address Map Column Address Bit 7 (REG_DDRC_ADDRMAP_COL_B7)

Address Map Column Address Bit 8 (REG_DDRC_ADDRMAP_COL_B8)

Address Map Column Address Bit 9 (REG_DDRC_ADDRMAP_COL_B9)

Address Map Column Address Bit 10 (REG_DDRC_ADDRMAP_COL_B10)

Address Map Column Address Bit 11 (REG_DDRC_ADDRMAP_COL_B11)

DDR2 Row Address Mapping Register (DDR_AD_MAP_ROW):

Address Map Row Address Bit 0 (REG_DDRC_ADDRMAP_ROW_B0)

Address Map Row Address Bit 1 (REG_DDRC_ADDRMAP_ROW_B1)

Address Map Row Address Bits 2 To 11 (REG_DDRC_ADDRMAP_ROW_B2_11)

Address Map Row Address Bit 12 (REG_DDRC_ADDRMAP_ROW_B12)

Address Map Row Address Bit 13 (REG_DDRC_ADDRMAP_ROW_B13)

Address Map Row Address Bit 14 (REG_DDRC_ADDRMAP_ROW_B14)

Address Map Row Address Bit 15 (REG_DDRC_ADDRMAP_ROW_B15)

7.2.1.2 Address Mapper Description

The address mapper maps linear request addresses to DRAM addresses by selecting the source bit that will map to each and every applicable DRAM address bit. While it's possible to map source address bits to DRAM address in any manner the user desires, the full available address space is only accessible to the user when no two DRAM address bits are determined by the same source address bit; therefore the address map must always be programmed to ensure that this is the case. Each DRAM address bit has an associated register vector to determine its source. The source address bit number is determined by adding the "internal base" of a given register to the programmed value for that register, as described in the following equation:

$$[\text{internal base}] + [\text{register value}] = [\text{source address bit number}]$$

For one example, reading the description for [Address Map Column Address Bit 7 \(REG_DDRC_ADDRMAP_COL_B7\)](#), we see that the internal base is 7; so when the full data bus is in use, column bit 7 is determined by the following: $7 + [\text{register value}]$. If this register were programmed to "2", then the source address bit would be: $7 + 2 = 9$.

In other words, the column address bit 7 sent to DRAM would always be equal to bit 9 of the corresponding source address.

Note that all of the column bits shift up 1 bit when only half of the data bus is in use. In this case, we would need to look at [Address Map Column Address Bits 4 To 6 \(REG_DDRC_ADDRMAP_COL_B4_6\)](#) instead to determine the value of column address bit 7.

Also, note that some registers map multiple source address bits ([Address Map Column Address Bits 4 To 6 \(REG_DDRC_ADDRMAP_COL_B4_6\)](#)).

Finally, note that for any address bits which may not be in use in all cases, the associated address map register must be set to all ones when the associated DRAM address bit it is not in use.

7.2.2 ODT Controls

Register inputs are provided for controlling:

- The value desired for local ODT following a read command (DDR2 PHY Read Local ODT (REG_PHY_RD_LOCAL_ODT), typically programmed to 1 if used).
- The value desired for local ODT following a write command (DDR2 PHY Write ODT Control (REG_PHY_WR_LOCAL_ODT), typically programmed to 0 if used).
- The number of cycles to delay following a write command before driving the programmed values for write ODT (Write ODT Delay (REG_DDRC_WR_ODT_DELAY), which will depend primarily on CAS latency).
- The number of cycles to hold the programmed write value after it is first driven (Write ODT Hold (REG_DDRC_WR_ODT_HOLD)).

Note: The number of cycles to delay following a read command before driving the programmed values for read ODT (Read ODT Delay) is determined by the controller.

Note: The number of cycles to hold the programmed read value after it is first driven (Read ODT Hold) is determined by the controller.

These quantities must all be set before taking the controller out of soft reset; they will then be applied to every read or write issued by the controller.

7.2.2.1 ODT Write Timing

Figure 7.1 illustrates ODT Write Timing. The following defaults are assumed for the indicated fields of DDR2 Control Register 3 (DDR_CONTROL_3):

- Rank 0 Write ODT (REG_DDRC_RANK0_WR_ODT) = 10b (DRAM ODT on, local ODT off during writes).

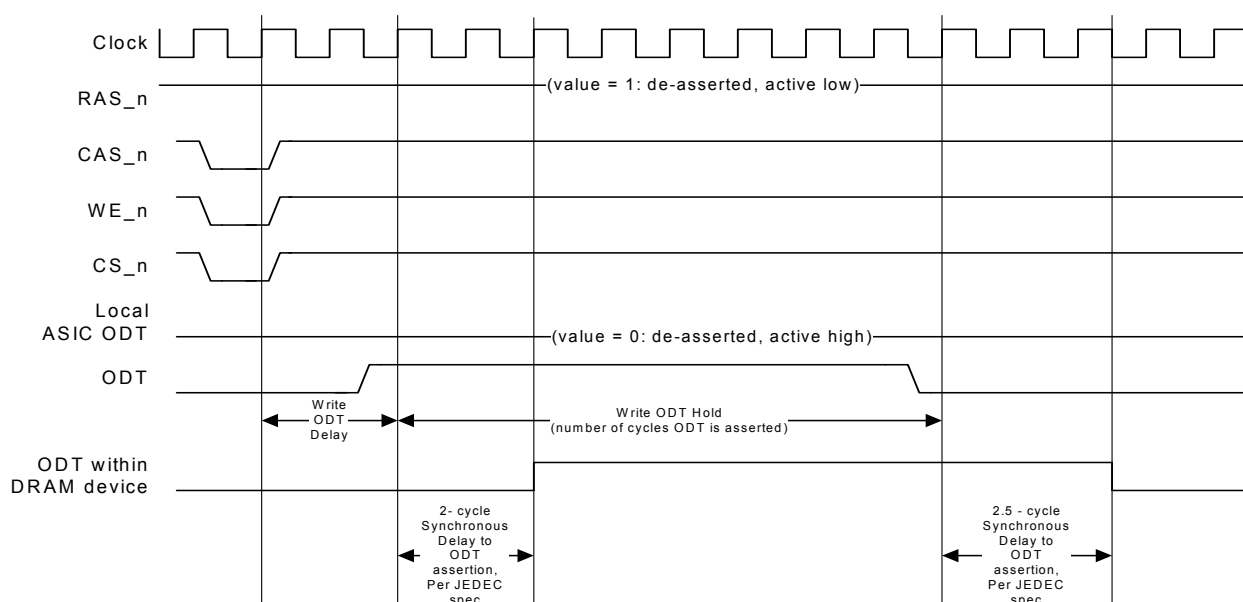


Figure 7.1 ODT Write Timing Diagram

7.2.2.2 ODT Read Timing

Figure 7.2 illustrates ODT Read Timing. The following defaults are assumed for the indicated fields of DDR2 Control Register 3 (DDR_CONTROL_3):

- Rank 0 Read ODT (REG_DDRC_RANK0_RD_ODT) = 01b (DRAM ODT off, local ODT on during reads).

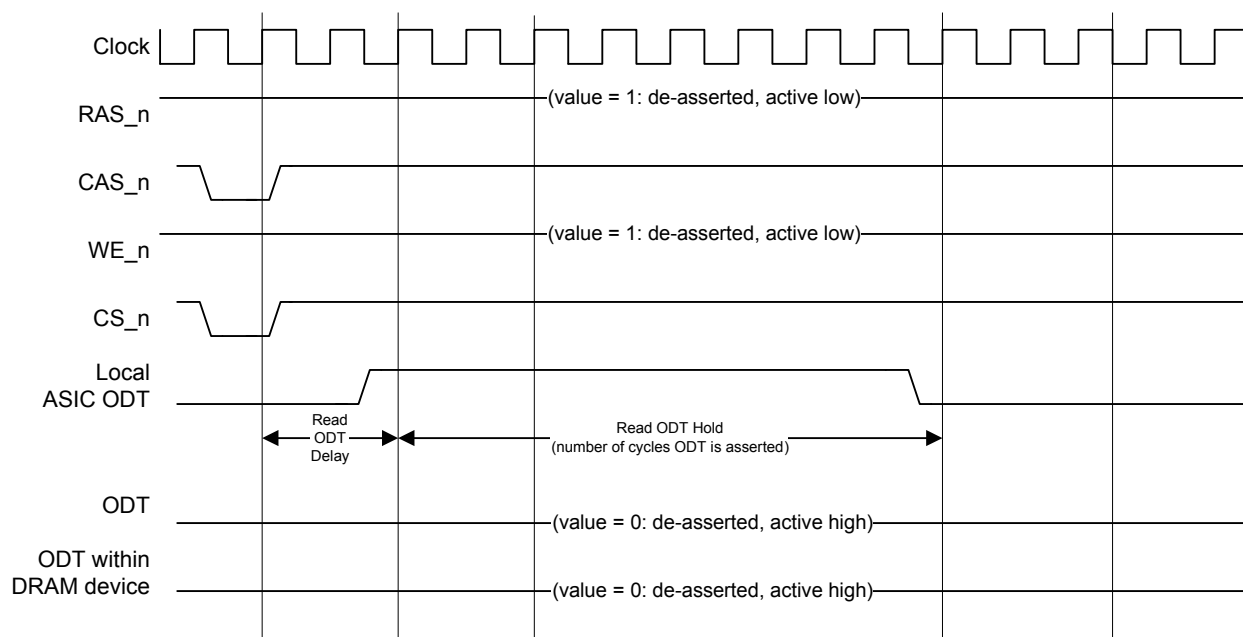


Figure 7.2 ODT Read Timing Diagram

7.2.3 Refresh Controls

The controller provides advanced refresh controls that support fully-configurable refresh constraints ($t_{RFC(min)}$ and t_{REFI}).

7.2.3.1 DDRC Register Fields Related to Refresh Control

The following CSR register fields are related to refresh control and must be set appropriately:

DDR2 Refresh Register (DDR_REFRESH):

Refresh Burst (REG_DDRC_REFRESH_BURST)

DDR2 Refresh Register (DDR_REFRESH):

Refresh Idle Timeout (REG_DDRC_REFRESH_TO_X32)

7.2.3.2 Refresh Controls Description

The goals of the refresh controls are:

- Reduce the bandwidth impact of refresh cycles.

- Increase the likelihood of refreshes being serviced during idle periods.
- Fine-grain control of the trading off the above benefits (for gathering refreshes) versus the increased worst case latencies associated with gathering refreshes, and

The controller can be programmed to issue single refreshes at a time (`refresh_burst=0`) to minimize the worst-case impact of a forced refresh cycle. It can be programmed to burst the maximum number of refreshes allowed to be gathered for DDR2 (`refresh_burst=7`, for performing 8 refreshes at a time) to minimize the bandwidth lost to closing pages for refresh and to increase the likelihood that refreshes may be serviced during idle periods. It may also be programmed to any number in between to trade-off the benefits of each.

7.2.3.2.1 SINGLE REFRESH

When using single refresh (`refresh_burst=0`), the controller will issue refreshes every time the refresh timer (t_{REFI}) expires. This is the optimal mode of operation for systems that must minimize the maximum latency associated with refresh cycles.

7.2.3.2.2 BURST REFRESH

When `refresh_burst>0`, the controller will issue refreshes in bursts of (`refresh_burst+1`) refreshes at one time. Bursting refreshes reducing the total latency associated with those refreshes by reducing the number of precharges and activates required for refresh (because banks must be precharged only once to perform the entire group of refreshes, instead of once for each refresh).

7.2.3.2.3 SPECULATIVE REFRESH

When burst refresh is enabled (`refresh_burst>0`), the user may also utilize speculative refresh.

Burst refresh is implemented by counting the number of times t_{REFI} expires, and issuing a group of refreshes when that number reaches the refresh burst number. Once t_{REFI} has expired at least, the controller may also perform speculative refreshes. This is done by automatically inserting refreshes when the controller is idle.

The [Refresh Idle Timeout \(REG_DDRC_REFRESH_TO_X32\)](#) determines how long the controller must be idle before considering inserting these speculative refreshes. Each time a speculative refresh is performed, the count of t_{REFI} expirations will be decremented, and thus increasing the time before a burst of refreshes will be required. This also ensures that speculative refreshes never occur any more often than is required to keep the DRAM properly refreshed.

If a new read or write transaction is accepted by the controller during speculative refresh, the controller will service it as soon as legally possible. Most often that will entail waiting for the required NOP cycles after a refresh before performing an activate and then servicing the read or write. If the controller has begun closing pages for a speculative refresh but has not yet issued the refresh when the new transaction arrives, the speculative refresh will be cancelled.

7.2.4 DLL Calibration

7.2.4.1 DLL Calibration Overview

DLL calibration is a signal to the PHY indicating that the PHY may update delay line values from the master DLL. The `co_gs_dll_calib` signal serves this function; it is a synchronous signal that may be pulsed or held asserted longer. This must be asserted periodically to ensure that PVT (process/voltage/temperature) variations are accounted for in the delay line over time. It is important to control this carefully to ensure that the delay line is never updated during a read or a write, as this could degrade the data eye. There are 2 methods for doing the DLL Calibration:

1. Automatic DLL Calibration by the controller.

2. Forced DLL Calibration using a command from the user on co_gs_dll_calib input.

7.2.4.2 Automatic DLL Calibration Policy

This method is used when [Disable DLL Calibration \(REG_DDRC_DIS_DLL_CALIB\)](#) is set to 0.

As noted in the overview, the controller must carefully control the assertion of co_gs_dll_calib. The controller uses the following policy to do this:

1. Assert co_gs_dll_calib any time the controller is idle.
2. Use a timeout mechanism to force DLL calibrations if controller is not idle for a long period of time. When a DLL calibration timeout occurs, the controller ceases to schedule reads and writes until calibration is performed.

7.2.4.3 Forced DLL Calibration

This method is used when [Disable DLL Calibration \(REG_DDRC_DIS_DLL_CALIB\)](#) is set to 1.

In this case, the user decides when to do the DLL Calibration. This is useful in cases where the bandwidth utilization of the DRAM bus is extremely crucial and the core doesn't want any break in transactions. The user utilizes [Request DLL Calibration \(CO_GS_DLL_CALIB\)](#) to invoke the operation.

When the controller is running with auto_refresh disabled, co_gs_dll_calib must be disabled as well ([Disable DLL Calibration \(REG_DDRC_DIS_DLL_CALIB\)](#) =1). The user may assert co_gs_dll_calib at the same time as refresh assertion. For both the refresh and the co_gs_dll_calib, the controller will take care to ensure that these happen in a functionally-correct manner.

7.2.5 Power Saving Features

The controller supports two power-saving modes: precharge power down and self refresh. When enabled, the controller automatically enters and exits precharge power down mode based on a programmable idle timeout period. Self refresh entry and exit is explicitly controlled by software. This section describes the signals that control both of those activities as well as the controls for the constraints related to exiting each mode properly.

7.2.5.1 DDRC Register Fields Related to Power Saving Features

[DDR2 Timing Register 3 \(DDR_TIMING_3\):](#)

[CKE Minimum Pulse Width \(REG_DDRC_T_CKE\)](#)

[DDR2 Dynamic Register \(DDR_DYN_REG\):](#)

[Power Down Enable \(REG_DDRC_POWERDOWN_EN\)](#)

[Self Refresh Enable \(REG_DDRC_SELFREF_EN\)](#)

[DDR2 Control Register 8 \(DDR_CONTROL_8\):](#)

[Precharge Power Down Timeout \(REG_DDRC_POWERDOWN_TO_X32\)](#)

[DDR2 Timing Register 2 \(DDR_TIMING_2\):](#)

[Minimum Idle Precharge Power Down To Command \(REG_DDRC_T_XP\)](#)

[DDR2 Timing Register 0 \(DDR_TIMING_0\):](#)

[Self Refresh Wait \(REG_DDRC_POST_SELFREF_GAP_X32\)](#)

7.2.5.2 Power Saving Modes

The memory controller supports two power saving modes: Precharge Power Down and Self Refresh mode.

Note: These power saving modes must never be enabled simultaneously.

7.2.5.2.1 ENTERING PRECHARGE POWER DOWN

When **Power Down Enable (REG_DDRC_POWERDOWN_EN)** =1, the controller automatically enters precharge power down when the period specified by **Precharge Power Down Timeout (REG_DDRC_POWERDOWN_TO_X32)** has passed while the controller is idle (except for issuing refreshes).

Entering precharge power down involves the following steps:

1. Precharging (closing) all open pages. Pages are closed one-at-a-time in no specified order.
2. Waiting for t_{RP} (row precharge) idle period.

Issuing the command to enter precharge power down (NOP/Deselect with CKE=0).

If the controller receives a read or write request from the core logic during steps 1. and 2. above, the power down entry will be immediately aborted. The same is true if **Power Down Enable (REG_DDRC_POWERDOWN_EN)** is driven to '0' during steps 1. or 2. Once the power down entry command has been issued, and proper power down exit is required, as described in the following section.

7.2.5.2.2 EXITING PRECHARGE POWER DOWN

Once the controller has put the DDR DRAM device(s) in precharge power down mode, the controller automatically performs the precharge power down exit sequence for any of the following reasons:

- Refresh cycle is required to any rank in the system,
- The controller receives a new request from the core logic, or
- **Power Down Enable (REG_DDRC_POWERDOWN_EN)** is set to '0'.

To exit precharge power down, the controller does the following:

1. Insert any NOP/Deselect commands required to satisfy the t_{CKE} requirement after entering precharge power down.
2. Issuing the power down exit command (NOP/Deselect with CKE=1).
3. Issuing NOP/Deselect for the period defined by t_{RP} .

Note: Fast Exit vs. Slow Exit Active Power Down

The DDR specs describe 2 different variations on active power down exit, depending on the programmed value of MR bit 12. Since the controller uses precharge power down rather than active power down, this programming has no effect on the controller or the DRAM devices.

7.2.5.2.3 ENTERING SELF REFRESH

The controller puts the DDR DRAM device(s) in self refresh mode whenever the **Self Refresh Enable (REG_DDRC_SELFREF_EN)** bit is set and either (a) no reads or writes are pending in the controller or (b) **Disable Dequeue (REG_DDRC_DIS_DQ)** is set.

Entering self refresh mode involves the following steps:

1. Precharging (closing) all open pages. Pages are closed one-at-a-time in no specified order.
2. Waiting for t_{RP} (row precharge) idle period.
3. Issuing the command to enter self refresh mode (asserting RAS and CAS with CKE=0).

7.2.5.2.4 EXITING SELF REFRESH

The controller takes the DDR DRAM out of self refresh mode whenever the **Self Refresh Enable (REG_DDRC_SELFREF_EN)** input is deasserted or new commands are received by the controller.

To exit self refresh, the controller typically does the following:

1. Insert any NOP/Deselect commands required to satisfy the t_{CKE} requirement after entering precharge power down.

2. Issuing the self refresh exit command (refresh with CKE=1).
3. Issuing NOP/Deselect for the period defined by t_{xp}.

7.2.5.2.5 RESTARTING TO EXIT SELF REFRESH

In self-refresh mode, the DRAM will maintain itself as long it is properly powered and CKE is held low. This may allow the core logic to be completely powered down. If this is done, the controller will perform the standard initialization sequence when reset is de-asserted to the controller. The standard initialization sequence will properly bring the DRAM out of self refresh mode.

Refer to [Section 7.2.6](#) for details on the initialization sequence.

7.2.6 DRAM Initialization Sequence

A specific command sequence must be followed to properly initialize DRAM modules. Much of the sequence is standard for all parts, but parts of it must be controlled differently according to system configuration, clock speeds, and the specific DRAM parts in use. For this reason, the controller provides a great deal of flexibility in how the initialization sequence will be performed.

This section describes the initialization sequence and programmability provided by the controller.

7.2.6.1 DDRC Register Fields Related to the DRAM Initialization Sequence

The following CSR register fields are related to DRAM Initialization and must be set appropriately:

DDR2 Control Register 4 (DDR_CONTROL_4):

Minimum Idle Time After Reset Before First CKE Assertion (REG_DDRC_PRE_CKE_X1024)

Minimum Idle Time After First CKE Assertion (REG_DDRC_POST_CKE_X1024)

DDR2 Timing Register 4 (DDR_TIMING_4):

Mode Register Delay (REG_DDRC_T_MRD)

DDR2 Control Register 8 (DDR_CONTROL_8):

Wait Period Before OCD Complete (REG_DDRC_PRE_OCD_X32)

Final Wait (REG_DDRC_FINAL_WAIT_X32)

DDR2 Control Register 2 (DDR_CONTROL_2):

Mode Register Value (REG_DDRC_MR)

Extended Mode Register 3 Value (REG_DDRC_EMR3)

DDR2 Control Register 1 (DDR_CONTROL_1):

Extended Mode Register Value (REG_DDRC_EMR)

Extended Mode Register 2 Value (REG_DDRC_EMR2)

7.2.6.2 DRAM Initialization Sequence Description

When the CSR fields listed in [Section 7.2.6.1](#) are set appropriately, the controller automatically performs the sequence necessary to properly initialize the DDR DRAM devices.

Note: In addition to initializing the DRAM after a hard reset, the initialization sequence may also be used to bring the DRAM out of self-refresh mode.

The initialization state machine will execute the following initialization sequence:

1. Power up.

2. Issue NOP/Deselect for the duration specified by [Minimum Idle Time After Reset Before First CKE Assertion \(REG_DDRC_PRE_CKE_X1024\)](#) (spec requires at least 200 us with stable power and clock).
3. Assert CKE and issue NOP/Deselect for [Minimum Idle Time After First CKE Assertion \(REG_DDRC_POST_CKE_X1024\)](#) (spec requires at least 400 ns).
4. Issue PRECHARGE ALL followed by NOP/Deselect for [Precharge Command Period \(REG_DDRC_T_RP\)](#) cycles.
5. Program EMR2 to the [Extended Mode Register 2 Value \(REG_DDRC_EMR2\)](#) value followed by NOP/Deselect for [Mode Register Delay \(REG_DDRC_T_MRD\)](#) cycles.
6. Program EMR3 to the [Extended Mode Register 3 Value \(REG_DDRC_EMR3\)](#) value followed by NOP/Deselect for [Mode Register Delay \(REG_DDRC_T_MRD\)](#) cycles.
7. Enable DLL by programming EMR to the [Extended Mode Register Value \(REG_DDRC_EMR\)](#) value followed by NOP/deselect for [Mode Register Delay \(REG_DDRC_T_MRD\)](#) cycles.
8. Issue DLL reset by programming MR to the [Mode Register Value \(REG_DDRC_MR\)](#) value followed by NOP/Deselect for [Mode Register Delay \(REG_DDRC_T_MRD\)](#) cycles.
9. Issue Precharge All followed by NOP/Deselect for [Precharge Command Period \(REG_DDRC_T_RP\)](#) +1 cycles.
10. Issue Refresh followed by NOP/Deselect for [Minimum Refresh Cycle Time \(REG_DDRC_T_RFC_MIN\)](#) cycles. Repeat 9 times.
11. Program MR without resetting the DLL by setting MR to [Mode Register Value \(REG_DDRC_MR\)](#) value with bit 8 set to '1'.
12. Issue NOP/Deselect for duration specified by [Wait Period Before OCD Complete \(REG_DDRC_PRE_OCD_X32\)](#) (no spec requirement).
13. Issue "OCD complete" command, indicating that no on-chip driver calibration will be performed.
14. Issue NOP/Deselect for [Final Wait \(REG_DDRC_FINAL_WAIT_X32\)](#) cycles (no spec requirement).
15. Begin normal operation.

7.3 DDR2 Controller Configuration

7.3.1 Auto Initialization

After a hard reset, the DDR2 controller will be in soft reset. Power and clocks must be brought up in accordance with the requirements of the full ASIC/SoC design requirements and the DRAM power and clocks must be brought up in accordance with the appropriate DRAM spec prior to removing the controller from soft reset. (See JEDEC document JESD79-2C section 2.3.1 steps a and b.) A state machine inside the DDR2 controller automatically programs the DDR2 in accordance with the CSR values defined in [Table 7.1](#).

When in soft reset, initialize all registers. The behavior of the controller is unpredictable if registers contain illegal values at the time that the controller is brought out of soft reset, or any time thereafter.

After all registers have been programmed, set the [Soft Reset \(REG_DDRC_SOFT_RSTB\)](#) bit of the [DDR2 Dynamic Register \(DDR_DYN_REG\)](#) to take the controller out of soft reset. The DDR controller will initialize and bring up the DRAM. Software can poll the status bits within the CSRs ([DDR Controller Initialization Done \(DDRC_INIT_DONE\)](#) and [PHY Initialization Done \(PHY_INIT_DONE\)](#) bits of the [DDR2 Dynamic Register \(DDR_DYN_REG\)](#) and the [DDR2 Controller Operating Mode \(DDRC_REG_OPERATING_MODE\)](#) bits of [DDR2 Control Register 10 \(DDR_CONTROL_10\)](#)) to determine when initialization is complete.

7.3.2 Configuration Example

Table 7.1 illustrates the programming values for SDRAM under DDR2 operation. When supporting lower display resolutions, it is possible to run the DDR2 at a lower speed.

Table 7.1 Sample Programming Values for DDR2

| PARAMETER | DDR2-400C 4-4-4 | DDR2-533C 5-5-5 | DDR2-667D 5-5-5 |
|--|--------------------|--------------------|--------------------|
| Burst Length (REG_DDRC_BURST8_RDWR) | | | 0b |
| Nominal Refresh Cycle Time (REG_DDRC_T_RFC_NOM_X32) | | | 50h |
| Self Refresh Wait (REG_DDRC_POST_SELFREF_GAP_X32) | | | 8h |
| Minimum Refresh Cycle Time (REG_DDRC_T_RFC_MIN) | | | 19h |
| Active to Active Command Period (REG_DDRC_T_RC) | | | 14h |
| Minimum Bank Active Time (REG_DDRC_T_RAS_MIN) | | | Ch |
| Maximum Bank Active Time (REG_DDRC_T_RAS_MAX_X1024) | | | 11h |
| Precharge Command Period (REG_DDRC_T_RP) | | | 5h |
| RAS To RAS Delay (REG_DDRC_T_RRD) | | | 3h |
| Minimum Time Between Two Reads Or Two Writes (REG_DDRC_T_CCD) | | | 2h |
| RAS TO CAS Delay (REG_DDRC_T_RCD) | | | 5h |
| Mode Register Delay (REG_DDRC_T_MRD) | | | 2h |
| Final Wait (REG_DDRC_FINAL_WAIT_X32) | | | 4h |
| Minimum Idle Time After Reset Before First CKE Assertion (REG_DDRC_PRE_CKE_X1024) | | | 43h |
| Mode Register Value (REG_DDRC_MR) | | | 952h |
| Extended Mode Register Value (REG_DDRC_EMR) | | | 40h |
| Extended Mode Register 2 Value (REG_DDRC_EMR2) | | | 0h |
| Extended Mode Register 3 Value (REG_DDRC_EMR3) | | | 0h |
| Write To Precharge Turnaround Time (REG_DDRC_WR2PRE) | | | Bh |
| Read To Precharge Turnaround Time (REG_DDRC_RD2PRE) | | | 3h |
| Write To Read Turnaround Time (REG_DDRC_WR2RD) | | | 9h |
| Read To Write Turnaround Time (REG_DDRC_RD2WR) | | | 5h |
| Write Latency (REG_DDRC_WRITE_LATENCY) | | | 3h |
| Four Active Window (REG_DDRC_T_FAW) | | | Ah |
| Disable DLL Calibration (REG_DDRC_DIS_DLL_CALIB) | | | 0h |
| Skip On-Chip_drive Calibration (REG_DDRC_SKIP_OCD) | | | 1h |
| Write ODT Hold (REG_DDRC_WR_ODT_HOLD) | | | 2h |

Table 7.1 Sample Programming Values for DDR2

| PARAMETER | DDR2-400C 4-4-4 | DDR2-533C 5-5-5 | DDR2-667D 5-5-5 |
|--|--------------------|--------------------|--------------------|
| Rank 0 Write ODT (REG_DDRC_RANK0_WR_ODT) | | | 1h |
| Rank 0 Read ODT (REG_DDRC_RANK0_RD_ODT) | | | 2h |
| Write ODT Delay (REG_DDRC_WR_ODT_DELAY) | | | 1h |

7.4 Operating Modes

The [DDR2 Controller Operating Mode \(DDRC_REG_OPERATING_MODE\)](#) field of [DDR2 Control Register 10 \(DDR_CONTROL_10\)](#) can be polled to determine the current mode of operation of the controller. The modes are:

- 00 – un-initialized. The controller may be in soft reset, or it may be out of soft reset, but the DRAM initialization sequence has not yet completed.
- 01 – “normal” operating mode. The controller is ready to accept read and write requests and the controller may issue reads and writes to DRAM.
- 10 – DRAM is in power down.
- 11 – DRAM is in self refresh.

7.4.1 Moving To Power Down

Enable power down mode in the Master Control register. Once enabled, DDRC will automatically put the DRAM into precharge all power down after the programmed number of idle cycles.

A refresh request will bring the DRAM out of power down. It will go back into power down after the idle period.

Any transaction will bring the DRAM out of power down automatically.

Clearing the power down enable bit will also bring the DRAM out of power down.

Note: The [DDR2 CKE Pads Output Enable Disable \(REG_DDR2_OE_CKE_DISABLE\)](#), [DDR2 CKE Pads Power Down Control \(REG_DDR2_PWD_CKE\)](#), [DDR2 Pads Output Enable Disable \(REG_DDR2_OE_DISABLE\)](#), and [DDR2 Pads Power Down Control \(REG_DDR2_PWD\)](#) fields of the [DDR2 PHY Control Register 0 \(DDR_PHY_CTL_0\)](#) facilitate configuring the DDR2 pads for minimal power consumption.

7.4.2 Moving To Self Refresh

To place the DRAM in self refresh, Set the [Self Refresh Enable \(REG_DDRC_SELFREF_EN\)](#) bit in [DDR2 Dynamic Register \(DDR_DYN_REG\)](#). The DDRC will put the DRAM into self refresh as soon as the transaction buffers are empty. Software must ensure that no transactions arrive. If transactions keep arriving the DDRC will never put the DRAM into self refresh. The first valid transaction will bring the DRAM out of self refresh.

Note: Clock frequency can not be changed if the DRAM is not in self refresh mode.

Placing the DRAM in self refresh is part of the USB Suspend sequence and is illustrated in [Figure 5.3 USB Suspend Sequence on page 83](#).

7.4.3 List of Dynamic Registers

Most registers are meant to be programmed while the controller is in soft reset. The exceptions are referred to as “dynamic registers”. In the event that the register control is not in the same clock domain as the controller, these signals will need to be synchronized to the controller core clock domain. This synchronization must be done ‘outside’ of the DRAM Interface hierarchy. The following is a list of such signals.

1. [Soft Reset \(REG_DDRC_SOFT_RSTB\)](#)
2. [Self Refresh Enable \(REG_DDRC_SELFREF_EN\)](#)
3. [Power Down Enable \(REG_DDRC_POWERDOWN_EN\)](#)
4. [Disable Dequeue \(REG_DDRC_DIS_DQ\)](#)
5. [Refresh Update \(REG_DDRC_REFRESH_UPDATE_LEVEL\)](#)
6. [Disable Auto Refresh \(REG_DDRC_DIS_AUTO_REFRESH\)](#)

7.4.4 Refresh Related Signals

Any time a refresh-related signal is updated, the following must be done:

1. Change the refresh-associated register as desired.
2. After the changed register is known stable, toggle the [Refresh Update \(REG_DDRC_REFRESH_UPDATE_LEVEL\)](#) signal.

The DRAM Controller will note the refresh_update_level signal change and update all refresh-related register values accordingly. This mechanism is needed to avoid sampling errors in the target clock domain as well as to allow the controller to provide special handling (such as issuing an additional refresh and resetting the refresh timer if needed) when a refresh-related timing register has changed.

7.4.5 Changing Clock Frequencies

Changing clock frequencies is a simple process, described here:

1. Request the controller to enter self refresh by asserting [Self Refresh Enable \(REG_DDRC_SELFREF_EN\)](#).
2. Wait until [DDR2 Controller Operating Mode \(DDRC_REG_OPERATING_MODE\)](#) is equal to 03h, indicating that the controller is in self refresh.
3. Glitchlessly change the clock frequency to the controller.
4. Update any registers which may be required to change for the new frequency.
5. Take the controller out of self refresh by de-asserting [Self Refresh Enable \(REG_DDRC_SELFREF_EN\)](#).

7.5 Control and Status Registers

Table 7.2 Memory Controller Control and Status Register Map

| ADDRESS OFFSET | SYMBOL | REGISTER NAME |
|----------------|------------------|---------------------------------------|
| 0000h | DDR_DYN_REG | DDR2 Dynamic Register |
| 0004h | DDR_AD_MAP_BA_CS | DDR2 Bank/CS Address Mapping Register |
| 0008h | DDR_AD_MAP_COL | DDR2 Column Address Mapping Register |
| 000Ch | DDR_AD_MAP_ROW | DDR2 Row Address Mapping Register |
| 0010h | DDR_TIMING_0 | DDR2 Timing Register 0 |
| 0014h | DDR_TIMING_1 | DDR2 Timing Register 1 |
| 0018h | DDR_TIMING_2 | DDR2 Timing Register 2 |
| 001Ch | DDR_TIMING_3 | DDR2 Timing Register 3 |
| 0020h | DDR_TIMING_4 | DDR2 Timing Register 4 |
| 0024h | DDR_REFRESH | DDR2 Refresh Register |
| 0028h | DDR_CONTROL_0 | DDR2 Control Register 0 |
| 002Ch | DDR_CONTROL_1 | DDR2 Control Register 1 |
| 0030h | DDR_CONTROL_2 | DDR2 Control Register 2 |
| 0034h | DDR_CONTROL_3 | DDR2 Control Register 3 |
| 0038h | DDR_CONTROL_4 | DDR2 Control Register 4 |
| 003Ch | DDR_CONTROL_5 | DDR2 Control Register 5 |
| 0040h | DDR_CONTROL_6 | DDR2 Control Register 6 |
| 0044h | DDR_CONTROL_7 | DDR2 Control Register 7 |
| 0048h | DDR_CONTROL_8 | DDR2 Control Register 8 |
| 004Ch | DDR_CONTROL_9 | DDR2 Control Register 9 |
| 0050h | DDR_CONTROL_10 | DDR2 Control Register 10 |
| 0054h | DDR_CONTROL_11 | DDR2 Control Register 11 |
| 0058h | DDR_CONTROL_12 | DDR2 Control Register 12 |
| 005Ch | DDR_CONTROL_13 | DDR2 Control Register 13 |
| 0060h – 00FFh | RESERVED | Reserved for future expansion |
| 0100h | DDR_PHY_CTL_0 | DDR2 PHY Control Register 0 |
| 0104h | DDR_PHY_CTL_1 | DDR2 PHY Control Register 1 |
| 0108h – 010Fh | RESERVED | Reserved for future expansion |
| 0110h | DDR_PHY_DEBUG_0 | DDR2 PHY Debug Register 0 |
| 0114h | DDR_PHY_DEBUG_1 | DDR2 PHY Debug Register 1 |
| 0118h | DDR_PHY_DEBUG_2 | DDR2 PHY Debug Register 2 |
| 011Ch | DDR_PHY_DEBUG_3 | DDR2 PHY Debug Register 3 |
| 0120h | DDR_PHY_DEBUG_4 | DDR2 PHY Debug Register 4 |
| 0124h | DDR_PHY_DEBUG_5 | DDR2 PHY Debug Register 5 |
| 0128h | DDR_PHY_DEBUG_6 | DDR2 PHY Debug Register 6 |
| 012Ch | DDR_PHY_DEBUG_7 | DDR2 PHY Debug Register 7 |
| 0130h | DDR_PHY_DEBUG_8 | DDR2 PHY Debug Register 8 |
| 0134h – 0FFFh | RESERVED | Reserved for future expansion |

7.5.1 DDR2 Dynamic Register (DDR_DYN_REG)

Offset: 0000h Size: 32 bits

This register is used to configure and control the DDR2 controller.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|---|------|---------|
| 31 | PHY Initialization Done (PHY_INIT_DONE) Indicates that the PHY initialization is complete. After reset has been released, this bit should be polled for a 0x1 value before the DDR2 controller is used. | RO | 0b |
| 30 | DDR Controller Initialization Done (DDRC_INIT_DONE) Indicates that the DDRC initialization is complete. PHY initialization does not begin until DDRC initialization completes. | RO | 0b |
| 29:6 | RESERVED | RO | - |
| 5 | Loopback Mode (REG_DDRC_LOOPBACK) When this bit is set, the controller is put in a "loopback" mode for test purposes. This signal should only be toggled when the controller is idle (i.e., no commands have been issued to the controller which are yet to be serviced.) | R/W | 0b |
| 4 | Disable Dequeue (REG_DDRC_DIS_DQ) When set, DDRC will not de-queue any transactions from the CAM. Bypass will also be disabled. All transactions will be queued in the CAM. This is for debug only; no reads or writes are issued to DRAM as long as this is asserted. Note: This bit is intended to be switched on-the-fly. Note: FOR DEBUG ONLY | R/W | 0b |
| 3 | Refresh Update (REG_DDRC_REFRESH_UPDATE_LEVEL) This bit must be toggled to indicate that refresh register(s) have been updated. Note: The value will be automatically updated when exiting soft reset. Therefore, it does not need to be toggled initially. | R/W | 0b |
| 2 | Self Refresh Enable (REG_DDRC_SELFREF_EN) When set, the controller will put the DRAM into self refresh when the transaction store is empty. | R/W | 0b |
| 1 | Power Down Enable (REG_DDRC_POWERDOWN_EN) When set, the controller will go into power down after a programmable number of cycles "Max idle clocks before power down" Precharge Power Down Timeout (REG_DDRC_POWERDOWN_TO_X32) This register bit may be reprogrammed during the course of normal operation. | R/W | 0b |
| 0 | Soft Reset (REG_DDRC_SOFT_RSTB) This bit is used to reset the controller. 0 = Resets the controller 1 = Takes the controller out of reset Note: The controller should be taken out of reset only after all other registers have been programmed. | R/W | 0b |

7.5.2 DDR2 Bank/CS Address Mapping Register (DDR_AD_MAP_BA_CS)

Offset: 0004h Size: 32 bits

This register is used to configure and control the DDR2 controller.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|---|------|---------|
| 31:21 | RESERVED | RO | - |
| 20:16 | Address Map Rank Address Bit 0 (REG_DDRC_ADDRMAP_CS_BIT0) Selects the address bit used as rank address bit 0. The selected address bit is determined by adding the Internal Base to the value of this field. Valid Range: 0 to 22, and 31 Internal Base: 9 Note: If set to 31, rank address bit 0 is set to 0. | R/W | 00h |
| 15:12 | RESERVED | RO | - |
| 11:8 | Address Map Bank Address Bit 2 (REG_DDRC_ADDRMAP_BANK_B2) Selects the address bit used as bank address bit 2. The selected address bit for each of the bank address bits is determined by adding the Internal Base to the value of this field. Valid Range: 0 to 14, and 15 Internal Base: 4 Note: If set to 15, bank address bit 2 is set to 0. Note: The device only supports two bank pins (four banks). This register must always be set to Fh. | R/W | 0h |
| 7:4 | Address Map Bank Address Bit 1 (REG_DDRC_ADDRMAP_BANK_B1) Selects the address bits used as bank address bit 1. The selected address bit for each of the bank address bits is determined by adding the Internal Base to the value of this field. Valid Range: 0 to 14 Internal Base: 3 | R/W | 0h |
| 3:0 | Address Map Bank Address Bit 0 (REG_DDRC_ADDRMAP_BANK_B0) Selects the address bits used as bank address bit 0. The selected address bit for each of the bank address bits is determined by adding the Internal Base to the value of this field. Valid Range: 0 to 14 Internal Base: 2 | R/W | 0h |

7.5.3 DDR2 Column Address Mapping Register (DDR_AD_MAP_COL)

Offset: 0008h Size: 32 bits

This register is used to configure and control the DDR2 controller.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|--|------|---------|
| 31:28 | <p>Address Map Column Address Bit 11 (REG_DDRC_ADDRMAP_COL_B11)</p> <p>Full Bus Width Mode: Selects the address bit used as column address bit 12.</p> <p>Half Bus Width Mode: Unused. Should be tied to 4'hF.</p> <p>Quarter Bus Width Mode: Unused. To make it unused, this must be tied to 4'hF.</p> <p>The selected address bit is determined by adding the Internal Base to the value of this field.</p> <p>Valid Range: 0 to 7, and 15 Internal Base: 11</p> <p>Note: If set to 15, this column address bit is set to 0.</p> <p>Note: Per JEDEC DDR2 specification, column address bit 10 is reserved for indicating auto-precharge, therefore no source address bit can be mapped to column address bit 10.</p> | R/W | 0h |
| 27:24 | <p>Address Map Column Address Bit 10 (REG_DDRC_ADDRMAP_COL_B10)</p> <p>Full Bus Width Mode: Selects the address bit used as column address bit 11.</p> <p>Half Bus Width Mode: Selects the address bit used as column address bit 12.</p> <p>Quarter Bus Width Mode: Unused. To make it unused, this must be tied to 4'hF.</p> <p>The selected address bit is determined by adding the Internal Base to the value of this field.</p> <p>Valid Range: 0 to 7, and 15 Internal Base: 10</p> <p>Note: If set to 15, this column address bit is set to 0.</p> <p>Note: Per JEDEC DDR2 specification, column address bit 10 is reserved for indicating auto-precharge, therefore no source address bit can be mapped to column address bit 10.</p> | R/W | 0h |

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|---|------|---------|
| 23:20 | <p>Address Map Column Address Bit 9 (REG_DDRC_ADDRMAP_COL_B9) Full Bus Width Mode: Selects the address bit used as column address bit 9.</p> <p>Half Bus Width Mode: Selects the address bit used as column address bit 11.</p> <p>Quarter Bus Width Mode: Selects the address bit used as column address bit 12.</p> <p>The selected address bit is determined by adding the Internal Base to the value of this field.</p> <p>Valid Range: 0 to 7, and 15 Internal Base: 9</p> <p>Note: If set to 15, this column address bit is set to 0.</p> <p>Note: Per JEDEC DDR2 specification, column address bit 10 is reserved for indicating auto-precharge, therefore no source address bit can be mapped to column address bit 10.</p> | R/W | 0h |
| 19:16 | <p>Address Map Column Address Bit 8 (REG_DDRC_ADDRMAP_COL_B8) Full Bus Width Mode: Selects the address bit used as column address bit 8.</p> <p>Half Bus Width Mode: Selects the address bit used as column address bit 9.</p> <p>Quarter Bus Width Mode: Selects the address bit used as column address bit 11.</p> <p>The selected address bit is determined by adding the Internal Base to the value of this field.</p> <p>Valid Range: 0 to 7, and 15 Internal Base: 8</p> <p>Note: If set to 15, this column address bit is set to 0.</p> <p>Note: Note: Per JEDEC DDR2 spec, column address bit 10 is reserved for indicating auto-precharge, and hence no source address bit can be mapped to column address bit 10.</p> | R/W | 0h |
| 15:12 | <p>Address Map Column Address Bit 7 (REG_DDRC_ADDRMAP_COL_B7) Full Bus Width Mode: Selects the address bit used as column address bit 7.</p> <p>Half Bus Width Mode: Selects the address bit used as column address bit 8.</p> <p>Quarter Bus Width Mode: Selects the address bit used as column address bit 9.</p> <p>The selected address bit is determined by adding the Internal Base to the value of this field.</p> <p>Valid Range: 0 to 7, and 15 Internal Base: 7</p> <p>Note: If set to 15, this column address bit is set to 0.</p> | R/W | 0h |

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|--|------|---------|
| 11:8 | Address Map Column Address Bits 4 To 6 (REG_DDRC_ADDRMAP_COL_B4_6) Full Bus Width Mode: Selects the address bits used as column address bits 4 through 6. Half Bus Width Mode: Selects the address bit used as column address bits 5 through 7. Quarter Bus Width Mode: Selects the address bit used as column address bits 6 through 8. The selected address bit for each of the column address bits is determined by adding the Internal Base to the value of this field. Valid Range: 0 to 7 Internal Base: 4 (for column address bit 4) to 6 (for column address bit 6). | R/W | 0h |
| 7:4 | Address Map Column Address Bit 3 (REG_DDRC_ADDRMAP_COL_B3) Full Bus Width Mode: Selects the address bit used as column address bit 3. Half Bus Width Mode: Selects the address bit used as column address bit 4. Quarter Bus Width Mode: Selects the address bit used as column address bit 5. The selected address bit is determined by adding the Internal Base to the value of this field. Valid Range: 0 to 7 Internal Base: 3 | R/W | 0h |
| 3:0 | Address Map Column Address Bit 2 (REG_DDRC_ADDRMAP_COL_B2) Full Bus Width Mode: Selects the address bit used as column address bit 2. Half Bus Width Mode: Selects the address bit used as column address bit 3. Quarter Bus Width Mode: Selects the address bit used as column address bit 4. The selected address bit is determined by adding the Internal Base to the value of this field. Valid Range: 0 to 7 Internal Base: 2 | R/W | 0h |

7.5.4 DDR2 Row Address Mapping Register (DDR_AD_MAP_ROW)

Offset: 000Ch Size: 32 bits

This register is used to configure and control the DDR2 controller.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|---|------|---------|
| 31:28 | RESERVED | RO | - |
| 27:24 | Address Map Row Address Bit 15 (REG_DDRC_ADDRMAP_ROW_B15) Selects the address bit used as row address bit 15. The selected address bit is determined by adding the Internal Base to the value of this field. Valid Range: 0 to 8, and 15 Internal Base: 24 Note: If set to 15, row address bit 15 is set to 0. | R/W | 0h |
| 23:20 | Address Map Row Address Bit 14 (REG_DDRC_ADDRMAP_ROW_B14) Selects the address bit used as row address bit 14. The selected address bit is determined by adding the Internal Base to the value of this field. Valid Range: 0 to 8, and 15 Internal Base: 23 Note: If set to 15, row address bit 14 is set to 0. | R/W | 0h |
| 19:16 | Address Map Row Address Bit 13 (REG_DDRC_ADDRMAP_ROW_B13) Selects the address bit used as row address bit 13. The selected address bit is determined by adding the Internal Base to the value of this field. Valid Range: 0 to 8, and 15 Internal Base: 22 Note: If set to 15, row address bit 13 is set to 0. | R/W | 0h |
| 15:12 | Address Map Row Address Bit 12 (REG_DDRC_ADDRMAP_ROW_B12) Selects the address bit used as row address bit 12. The selected address bit is determined by adding the Internal Base to the value of this field. Valid Range: 0 to 8, and 15 Internal Base: 21 Note: If set to 15, row address bit 12 is set to 0. | R/W | 0h |
| 11:8 | Address Map Row Address Bits 2 To 11 (REG_DDRC_ADDRMAP_ROW_B2_11) Selects the address bits used as row address bits 2 through 11. The selected address bit for each of the row address bits is determined by adding the Internal Base to the value of this field. Valid Range: 0 to 8 Internal Base: 11 (for row address bit 2) to 20 (for row address bit 11). | R/W | 0h |
| 7:4 | Address Map Row Address Bit 1 (REG_DDRC_ADDRMAP_ROW_B1) Selects the address bits used as row address bit 1. The selected address bit for each of the row address bits is determined by adding the Internal Base to the value of this field. Valid Range: 0 to 8 Internal Base: 10 | R/W | 0h |

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|--|------|---------|
| 3:0 | Address Map Row Address Bit 0 (REG_DDRC_ADDRMAP_ROW_B0) Selects the address bits used as row address bit 0. The selected address bit for each of the row address bits is determined by adding the Internal Base to the value of this field. Valid Range: 0 to 8 Internal Base: 9 | R/W | 0h |

7.5.5 DDR2 Timing Register 0 (DDR_TIMING_0)

Offset: 0010h Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|---|------|---------|
| 31:21 | RESERVED | RO | - |
| 20:16 | Write Latency (REG_DDRC_WRITE_LATENCY) Time from write command to write data on DDRC to PHY interface. $AL + CL - 2$ (PHY adds an extra flop delay on the write data, hence the "-2" here). Note: Each unit is a clock cycle. | R/W | 0h |
| 15 | RESERVED | RO | - |
| 14:8 | Self Refresh Wait (REG_DDRC_POST_SELFREF_GAP_X32) Minimum time to wait after coming out of self refresh before doing anything. This must be bigger than all the constraints that exist. (spec: maximum of tXSNR and tXSRD and tXSDLL which is 512 clocks) Note: Each unit is a multiple of 32 clocks. | R/W | 0h |
| 7:6 | RESERVED | RO | - |
| 5:0 | Active to Active Command Period (REG_DDRC_T_RC) Minimum time between activates to the same bank (t_{RC}). (spec: 65 ns for DDR2-400 and smaller for faster parts) Note: Each unit is a clock cycle. | R/W | 0h |

7.5.6 DDR2 Timing Register 1 (DDR_TIMING_1)

Offset: 0014h Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|--|------|---------|
| 31:28 | RESERVED | RO | - |
| 27:16 | Nominal Refresh Cycle Time (REG_DDRC_T_RFC_NOM_X32) Average time between refreshes (t_{REFI}). (spec: 7.8 μ S) Note: Each unit is a multiple of 32 clocks. | RW | 0h |
| 15:8 | RESERVED | RO | - |
| 7:0 | Minimum Refresh Cycle Time (REG_DDRC_T_RFC_MIN) Minimum time from refresh to refresh or activate t_{RFC} (min). (spec: 75 ns to 195 ns) Note: Each unit is a clock cycle. | RW | 0h |

7.5.7 DDR2 Timing Register 2 (DDR_TIMING_2)

Offset: 0018h Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|--|------|---------|
| 31:29 | RESERVED | RO | - |
| 28:24 | Minimum Idle Precharge Power Down To Command (REG_DDRC_T_XP) Minimum time after exiting pre-charge power down to any operation (t_{XP}). Note: Each unit is a clock cycle. | R/W | 0h |
| 23:22 | RESERVED | RO | - |
| 21:16 | Four Active Window (REG_DDRC_T_FAW) Sliding time window in which a maximum of 4 bank activates are allowed (t_{FAW}). Valid only in burst of 8 mode. Four banks at most may be activated in a rolling window of t_{FAW} cycles. Note: Each unit is a clock cycle. | R/W | 0h |
| 15:13 | RESERVED | RO | - |
| 12:8 | Minimum Bank Active Time (REG_DDRC_T_RAS_MIN) Minimum time between an activate command and a precharge command to the same bank ($t_{RAS(MIN)}$). (spec: 45 ns) Note: Each unit is a clock cycle. | R/W | 0h |
| 7:6 | RESERVED | RO | - |
| 5:0 | Maximum Bank Active Time (REG_DDRC_T_RAS_MAX_X1024) Maximum time between an activate and a precharge command to the same bank ($t_{RAS(MAX)}$). Maximum time that a page can be kept open. Note: Each unit is a multiple of 1024 clock cycles. Note: If this is zero, the page is closed after each transaction. | R/W | 0h |

7.5.8 DDR2 Timing Register 3 (DDR_TIMING_3)

Offset: 001Ch Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|--|------|---------|
| 31:27 | RESERVED | RO | - |
| 26:24 | Mode Register Delay (REG_DDRC_T_MRD) Cycles between Load Mode commands (t_{MRD}). Required period of NOP/Deselect after programming a mode register. | R/W | 0h |
| 23:20 | Precharge Command Period (REG_DDRC_T_RP) Minimum time from precharge to activate of the same bank (t_{RP}). Note: Each unit is a clock cycle. | R/W | 0h |
| 19 | RESERVED | RO | - |
| 18:16 | RAS To RAS Delay (REG_DDRC_T_RRD) Minimum time between activate commands from bank a to bank b (t_{RRD}). (spec: 10 ns or less) Note: Each unit is a clock cycle. | R/W | 0h |
| 15 | RESERVED | RO | - |
| 14:12 | Minimum Time Between Two Reads Or Two Writes (REG_DDRC_T_CCD) The value of this field plus one is the minimum time between two reads or two writes (from bank a to bank b) (t_{CCD}). (spec: 2 cycles) Note: Each unit is a clock cycle. | R/W | 0h |
| 11:10 | RESERVED | RO | - |
| 9:8 | CKE Minimum Pulse Width (REG_DDRC_T_CKE) Minimum number of cycles of CKE HIGH/LOW during power down and self refresh (t_{CKE}). Note: Each unit is a clock cycle. | R/W | 0h |
| 7:4 | RESERVED | RO | - |
| 3:0 | RAS TO CAS Delay (REG_DDRC_T_RCD) Minimum time from an activate command to a read or write command in the same bank. (spec: 15ns for DDR2-400 and lower for faster devices) Note: Each unit is a clock cycle. | R/W | 0h |

7.5.9 DDR2 Timing Register 4 (DDR_TIMING_4)

Offset: 0020h Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|---|------|---------|
| 31:29 | RESERVED | RO | - |
| 28:24 | Write To Precharge Turnaround Time (REG_DDRC_WR2PRE) Minimum time between write and precharge to same bank. (spec: $WL + BL/2 + t_{WR} = \text{approx } 8 \text{ cycles} + 15 \text{ ns} = 14 \text{ clocks @ } 400\text{MHz}$ and less for lower frequencies) Where: WL = write latency BL = burst length. This must match the value programmed in the BL bit of the mode register to the DRAM. t_{WR} = write recovery time This comes directly from the DRAM specs. Note: Each unit is a clock cycle. | R/W | 0h |
| 23:21 | RESERVED | RO | - |
| 20:16 | Read To Precharge Turnaround Time (REG_DDRC_RD2PRE) Minimum time from read to precharge of same bank (t_{RTP}). (spec: t_{RTP} for $BL=4$ and $t_{RTP}+2$ for $BL=8$. $t_{RTP} = 7.5 \text{ ns}$) Note: Each unit is a clock cycle. | R/W | 0h |
| 15:13 | RESERVED | RO | - |
| 12:8 | Write To Read Turnaround Time (REG_DDRC_WR2RD) Minimum time from write command to read command. Includes time for bus turnaround and recovery times and all per-bank, per-rank, and global constraints. $WL + t_{WTR} + BL/2$ Where: WL = write latency. BL = burst length. This must match the value programmed in the BL bit of the mode register to the DRAM. t_{WTR} = internal write to read command delay. This comes directly from the DRAM specs. Note: Each unit is a clock cycle. | R/W | 0h |
| 7:5 | RESERVED | RO | - |
| 4:0 | Read To Write Turnaround Time (REG_DDRC_RD2WR) Minimum time from read command to write command. Include time for bus turnaround and all per-bank, per-rank, and global constraints. $RL + BL/2 + 2 - WL$ Where: WL = write latency. BL = burst length. This must match the value programmed in the BL bit of the mode register to the DRAM. RL = read latency = CAS latency Each unit is a clock cycle. | R/W | 0h |

7.5.10 DDR2 Refresh Register (DDR_REFRESH)

Offset: 0024h Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|---|------|---------|
| 31:13 | RESERVED | RO | - |
| 12:8 | Refresh Idle Timeout (REG_DDRC_REFRESH_TO_X32) If the refresh timer (t_{RFC_nom} , as known as t_{REFI}) has expired at least once, but has not expired $burst_of_N_refresh$ times yet, then a "speculative refresh" may be performed. This refresh is performed at a time when refresh would be useful, but before it is absolutely required. When the DRAM bus is idle for a period of time determined by this refresh idle timeout and the refresh timer has expired at least once since the last refresh, then a speculative refresh will be performed. Speculative refreshes will continue successively until there are no refreshes pending or until new reads or writes are issued to the controller. Note: FOR PERFORMANCE ONLY Note: Each unit is a multiple of 32 clocks. | R/W | 0h |
| 7:4 | Refresh Margin (REG_DDRC_REFRESH_MARGIN) Issue critical refresh or page close this many cycles before the critical refresh or page timer expires. Note: It is recommended that values other than the default value NOT be used. Note: Each unit is a multiple of 32 clocks. | R/W | 0h |
| 3 | RESERVED | RO | - |
| 2:0 | Refresh Burst (REG_DDRC_REFRESH_BURST) The programmed value plus one will be the number of refresh timeouts that will be allowed to accumulate before traffic is blocked and the refreshes are forced to execute. Closing pages to perform a refresh is a one-time penalty that must be paid for each group of refreshes; therefore, performing refreshes in a burst reduces the per-refresh penalty of these page closings. Higher numbers for $burst_of_N_refresh$ will slightly increase utilization; lower numbers will decrease the worst-case latency associated with refreshes. 0 = single refresh 1 = burst-of-2 7 = burst-of-8 refresh. | R/W | 0h |

7.5.11 DDR2 Control Register 0 (DDR_CONTROL_0)

Offset: 0028h Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|--|------|---------|
| 31:11 | RESERVED | RO | - |
| 10 | Disable Active Bypass Path (REG_DDRC_DIS_ACT_BYPASS) Only present in designs supporting activate bypass. When 1, disable bypass path for high priority read activates. Note: FOR DEBUG ONLY | R/W | 0b |
| 9 | Disable Read Bypass Path (REG_DDRC_DIS_RD_BYPASS) Only present in designs supporting read bypass. When 1, disable bypass path for high priority read page hits. Note: FOR DEBUG ONLY | R/W | 0b |
| 8 | Collision Page Option (REG_DDRC_DIS_COLLISION_PAGE_OPT) When clear, auto-precharge will be disabled for the flushed command in a collision case. Collision cases are write followed by read to the same address, read followed by write to the same address, or write followed by write to the same address with Disable Write Combine (REG_DDRC_DIS_WC) bit = 1 (where "same address" comparisons exclude the two address bits representing critical word). Note: FOR DEBUG ONLY. | R/W | 0b |
| 7 | Disable Write Combine (REG_DDRC_DIS_WC) When set, disables Write Combine. Note: FOR DEBUG ONLY. | R/W | 0b |
| 6 | Prefer Writes (REG_DDRC_PREFER_WRITE) When set, the bank selector prefers writes over reads. Note: FOR DEBUG ONLY. | R/W | 0b |
| 5 | Force To Low Priority (REG_DDRC_FORCE_LOW_PRI_N) This is an active low signal. When clear (asserted), all incoming transactions will be forced to low priority. Forcing the incoming transactions to low priority implicitly turns off Bypass. Note: FOR PERFORMANCE ONLY. | R/W | 0b |
| 4 | Auto Precharge Enable (REG_DDRC_AUTO_PRE_EN) When set, most reads and writes will be issued with auto-precharge. Exceptions can be made for collision cases. Note: FOR PERFORMANCE ONLY. | R/W | 0b |
| 3 | Keep Bank Closed (REG_DDRC_PAGECLOSE) When set, bank will be closed and kept closed if no transactions are available for it. This is different from auto-precharge for the following reasons: <ul style="list-style-type: none"> ■ Explicit precharge commands are used, and not read/write with auto-precharge ■ Page is not closed after a read/write if there is another read/write pending to the same page. When clear, bank will remain open until there is a need to close it (i.e., to open a different page, or for page timeout or refresh timeout). This is not applicable when auto-refresh is used. Note: FOR PERFORMANCE ONLY. | R/W | 0b |

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|---|------|---------|
| 2 | Enable 2T Timing (REG_DDRC_EN_2T_TIMING_MODE) If set, then DDRC will use 2T timing. Otherwise 1T timing is used. | R/W | 0b |
| 1 | Data Bus Width (REG_DDRC_DATA_BUS_WIDTH) 0 = Full DQ bus width to DRAM 1 = Half DQ bus width to DRAM | R/W | 0b |
| 0 | Burst Length (REG_DDRC_BURST8_RDWR) This bit controls the burst size used to access the DRAM. This must match the BL mode register setting in the DRAM. 0 = Burst length of 4 1 = Burst length of 8 | R/W | 0b |

7.5.12 DDR2 Control Register 1 (DDR_CONTROL_1)

Offset: 002Ch Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|---|------|---------|
| 31:16 | Extended Mode Register 2 Value (REG_DDRC_EMR2) Value to be loaded into DRAM EMR2 registers. May be referred to as "MR2" in some DDR devices. | R/W | 0000h |
| 15:0 | Extended Mode Register Value (REG_DDRC_EMR) Value to be loaded into DRAM EMR registers. May be referred to as "MR1" in some DDR devices. Bits [9:7] are for OCD and the setting in this field is ignored. The controller will set those bits appropriately. | R/W | 0000h |

7.5.13 DDR2 Control Register 2 (DDR_CONTROL_2)

Offset: 0030h Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|---|------|---------|
| 31:16 | Mode Register Value (REG_DDRC_MR) Value to be loaded into the DRAM Mode register. May be referred to as “MR0” in some DDR devices. Bit 8 is for DLL and the setting in this field is ignored. The controller will set it appropriately. | R/W | 0000h |
| 15:0 | Extended Mode Register 3 Value (REG_DDRC_EMR3) Value to be loaded into DRAM EMR3 registers. May be referred to as “MR3” in some DDR devices. | R/W | 0000h |

7.5.14 DDR2 Control Register 3 (DDR_CONTROL_3)

Offset: 0034h Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|---|------|---------|
| 31:26 | RESERVED | RO | - |
| 25 | Skip On-Chip_driver Calibration (REG_DDRC_SKIP_OCD) Skip on-chip-driver calibration in init sequence | R/W | 0b |
| 24 | RESERVED | RO | - |
| 23:20 | Write ODT Hold (REG_DDRC_WR_ODT_HOLD) The value of this field + 1 is the number of cycles to hold ODT for a Write command. I.e., when this field is 0h, the ODT signal is on for 1 cycle, etc. The values to program in different modes are as follows: DRAM Burst of 4 – 2 DRAM Burst of 8 – 4 | R/W | 0h |
| 19:14 | RESERVED | RO | - |
| 13:12 | Rank 0 Write ODT (REG_DDRC_RANK0_WR_ODT) Bit [0]: When set, enables remote ODT during a write to Rank 0. Bit [1]: If set, then local ODT is enabled during writes to Rank 0. | R/W | 00b |
| 11:10 | RESERVED | RO | - |
| 9:8 | Rank 0 Read ODT (REG_DDRC_RANK0_RD_ODT) Bit [0]: When set, enables remote ODT during a read to Rank 0. Bit [1]: If set, then local ODT is enabled during reads to Rank 0. | R/W | 00b |
| 7:4 | Write ODT Delay (REG_DDRC_WR_ODT_DELAY) The delay, in clock cycles, from issuing a write command to setting ODT values associated with that command. ODT setting should remain constant for the entire time that DQS is driven by the controller. The suggested value for DDR2 is $WL - 3$, where WL is Write latency ($WL = CL - 1$). DDR2 ODT has a 2-cycle on-time delay and a 2.5-cycle off-time delay. The extra 1 cycle is to turn ON ODT during Write Preamble. | R/W | 0h |
| 3:0 | RESERVED | RO | - |

7.5.15 DDR2 Control Register 4 (DDR_CONTROL_4)

Offset: 0038h Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|--|------|---------|
| 31:16 | RESERVED | RO | - |
| 25:16 | Minimum Idle Time After First CKE Assertion (REG_DDRC_POST_CKE_X1024) Cycles to wait after driving CKE high to start the DRAM initialization sequence. DDR2 specifications require a delay of ≥ 200 μ S. DDR2 typically require a 400 ns delay, requiring this value to be programmed to 2 at all clock speeds. Note: Each unit is a multiple of 1024 clock cycles. | R/W | 0h |
| 15:10 | RESERVED | RO | - |
| 9:0 | Minimum Idle Time After Reset Before First CKE Assertion (REG_DDRC_PRE_CKE_X1024) Cycles to wait after reset before driving CKE high to start the DRAM initialization sequence. DDR2 specifications require a delay of ≥ 200 μ S. Note: Each unit is a multiple of 1024 clock cycles. | R/W | 0h |

7.5.16 DDR2 Control Register 5 (DDR_CONTROL_5)

Offset: 003Ch Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|---|------|---------|
| 31:28 | High Priority Transaction Run Length (REG_DDRC_HPR_XACT_RUN_LENGTH) Number of transactions that will be serviced once the High Priority Read queue goes critical. The value is the minimum of this field or the number of transactions available. Note: FOR PERFORMANCE ONLY. Note: Each unit is a transaction. | R/W | 0h |
| 27 | RESERVED | RO | - |
| 26:16 | HPR Max Starve (REG_DDRC_HPR_MAX_STARVE_X32) Number of clocks that the High Priority Read queue can be starved before it goes critical. Note: FOR PERFORMANCE ONLY. Note: Each unit is a multiple of 32 clocks. | R/W | 0h |
| 15:11 | RESERVED | RO | - |
| 10:0 | HPR Min Non Critical (REG_DDRC_HPR_MIN_NON_CRITICAL_X32) Number of clocks that the High Priority Read queue is guaranteed to be non-critical. Note: FOR PERFORMANCE ONLY. Note: Each unit is a multiple of 32 clocks. | R/W | 0h |

7.5.17 DDR2 Control Register 6 (DDR_CONTROL_6)

Offset: 0040h Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|--|------|---------|
| 31:28 | Low Priority Transaction Run Length (REG_DDRC_LPR_XACT_RUN_LENGTH) Number of transactions that will be serviced once the Low Priority Read queue goes critical. The value is the minimum of this field or the number of transactions available. Note: FOR PERFORMANCE ONLY. Note: Each unit is a transaction. | R/W | 0h |
| 27:23 | Low Priority Number of Entries (REG_DDRC_LPR_NUM_ENTRIES) Number of entries in the low priority transaction store is this value plus 1. Value must be 11 or less and 3 or more. Sum of low and high priority read transaction stores must be 16 or less. Note: FOR PERFORMANCE ONLY. Note: Each unit is a transaction. | R/W | 0h |
| 22:12 | Low Priority Max Starve (REG_DDRC_LPR_MAX_STARVE_X32) Number of clocks that the Low Priority Read queue can be starved before it goes critical. Note: FOR PERFORMANCE ONLY. Note: Each unit is a multiple of 32 clocks. | R/W | 0h |
| 11 | RESERVED | RO | - |
| 10:0 | Low Priority Min Non-Critical (REG_DDRC_LPR_MIN_NON_CRITICAL_X32) Number of clocks that the Low Priority Read queue is guaranteed to be non-critical. Note: FOR PERFORMANCE ONLY. Note: Each unit is a multiple of 32 clocks. | R/W | 0h |

7.5.18 DDR2 Control Register 7 (DDR_CONTROL_7)

Offset: 0044h Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|--|------|---------|
| 31:28 | Write Transaction Run Length (REG_DDRC_W_XACT_RUN_LENGTH) Number of transactions that will be serviced once the WR queue goes critical. The value is the minimum of this field or the number of transactions available. Note: FOR PERFORMANCE ONLY. Note: Each unit is a transaction. | R/W | 0h |
| 27 | RESERVED | RO | - |
| 26:16 | Write Min Non-Critical (REG_DDRC_W_MIN_NON_CRITICAL_X32) Number of clocks that the Write queue is guaranteed to be non-critical. Note: FOR PERFORMANCE ONLY. Note: Each unit is a multiple of 32 clocks. | R/W | 0h |
| 15 | RESERVED | RO | - |
| 14:8 | Read Write Idle Gap (REG_DDRC_RDWR_IDLE_GAP) When the preferred transaction store is empty for the number of clock cycles specified in this field, switch to the alternate transaction store, if it is non-empty. The read transaction store (both hi and low priority) is the default preferred transaction store, while the write transaction store is the alternate store. When Prefer Writes (REG_DDRC_PREFER_WRITE) is set, this is reversed. Note: FOR PERFORMANCE ONLY. | R/W | 0h |
| 7:0 | DLL Calibration Timeout (REG_DDRC_DLL_CALIB_TO_X1024) This field specifies the time between DLL calibrations, which will be executed when the controller is idle. | R/W | 0h |

7.5.19 DDR2 Control Register 8 (DDR_CONTROL_8)

Offset: 0048h Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|--|------|---------|
| 31:24 | DRAM Reset to Low Time (REG_DDRC_DRAM_RSTN_X1024) Cycle for the dram reset to stay low. Note: Each unit is a multiple of 1024 clocks. | R/W | 0h |
| 23:20 | RESERVED | RO | - |
| 19:16 | Wait Period Before OCD Complete (REG_DDRC_PRE_OCD_X32) This field defines the wait period before driving the "OCD Complete" command to DRAM. Note: Each unit is a multiple of 32 clocks. A global timer is assumed to pulse every 32 clock cycles. | R/W | 0h |
| 15 | RESERVED | RO | - |
| 14:8 | Final Wait (REG_DDRC_FINAL_WAIT_X32) This field defines the number of cycles to wait after completing the DRAM initialization sequence before starting the dynamic scheduler. Note: Each unit is a multiple of 32 clocks. A global timer is assumed to pulse every 32 clock cycles. | R/W | 0h |
| 7:5 | RESERVED | RO | - |
| 4:0 | Precharge Power Down Timeout (REG_DDRC_POWERDOWN_TO_X32) Idle period required before entering precharge power down. After the number of clocks specified by this field of NOP or DESELECT, the controller will put the DRAM into power down. Only applicable when the Power Down Enable (REG_DDRC_POWERDOWN_EN) bit of the DDR2 Dynamic Register (DDR_DYN_REG) is set. Note: Each unit is a multiple of 32 clocks. | R/W | 0h |

7.5.20 DDR2 Control Register 9 (DDR_CONTROL_9)

Offset: 004Ch Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|---|------|---------|
| 31:13 | RESERVED | RO | - |
| 12 | Pad Powerdown Disable (REG_DDRC_DIS_PAD_PD) 0 = Enable the pad power down feature. 1 = Disable the pad power down feature. Note: This field is protected by Reset Protection (RST_PROTECT) . | R/W | 0b |
| 11 | RESERVED | RO | - |
| 10:8 | Pad Powerdown Time (REG_DDRC_PAD_PD) If pads have a power-saving mode, this is the maximum of the time for the pads to enter powerdown or the time for the pads to exit powerdown. Note: Units here are clock cycles. Note: This field is protected by Reset Protection (RST_PROTECT) . | R/W | 7h |
| 7:0 | GO_2_CRITICAL_HYSTERESIS (REG_DDRC_GO2CRITICAL_HYSTERESIS) The number of cycles that co_gs_go2critical_rd or co_gs_go2critical_wr must be asserted before the corresponding queue moves to the critical state in the DDRC. Note: FOR PERFORMANCE ONLY. | R/W | 0h |

7.5.21 DDR2 Control Register 10 (DDR_CONTROL_10)

Offset: 0050h Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|--|------|---------|
| 31:30 | RESERVED | RO | - |
| 29:28 | DDR2 Controller Operating Mode (DDRC_REG_OPERATING_MODE) Gives the status of the controller. 00 = Uninitialized. The controller may be in soft reset, or it may be out of soft reset, but DRAM initialization sequence has not yet completed. 01 = "Normal" operating mode. The controller is ready to accept read and write requests and the controller may issue reads and writes to DRAM. 10 = DRAM is in powerdown. 11 = DRAM is in self refresh | RO | 0h |
| 27:24 | RESERVED | RO | - |
| 23:20 | Number of Entries in the High Priority CAM (DDRC_REG_DBG_HPR_Q_DEPTH) Indicates the number of entries currently in the HPR CAM | RO | 0h |
| 19:16 | RESERVED | RO | - |
| 15:12 | Number of Entries in the Low Priority CAM (DDRC_REG_DBG_LPR_Q_DEPTH) Indicates the number of entries currently in the LPR CAM | RO | 0h |
| 11:8 | RESERVED | RO | - |
| 7:4 | Number of Entries in the Write CAM (DDRC_REG_DBG_WR_Q_DEPTH) Indicates the number of entries currently in the WR CAM | RO | 0h |
| 3:1 | RESERVED | RO | - |
| 0 | DDR2 Controller Stalled (DDRC_REG_DBG_STALL) 0 = Indicates that commands are being accepted 1 = Indicates that no commands are accepted by the controller | RO | 0h |

7.5.22 DDR2 Control Register 11 (DDR_CONTROL_11)

Offset: 0054h Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|--|------|---------|
| 31 | Mode Register Busy (DDRC_REG_MR_WR_BUSY) 0 = Indicates that the core can initiate a mode register read or write operation. Core must initiate a MR read or write operation only if this signal is low. This signal goes high in the clock after the controller accepts the read or write request. During MR writes, it goes low when the MR write command has been issued to the DRAM. During MR Reads, it goes low when MR Read data has been delivered to the core interface. Any MR read or write command that is received when Mode Register Busy (DDRC_REG_MR_WR_BUSY) is high, is not accepted. 1 = Indicates that mode register read or write operation is in progress. | RO | 0b |
| 30:22 | RESERVED | RO | - |
| 21:20 | Mode Register Address (REG_DDRC_MR_ADDR) Address of the Mode register that is to be written to. 00 = MR0 01 = MR1 10 = MR2 11 = MR3 | R/W | 00b |
| 19:17 | RESERVED | RO | - |
| 16 | Mode Register Write (REG_DDRC_MR_WR) Pulse this signal for 1 clock to do a mode register write. Controller will accept this command, if this signal is detected high and Mode Register Busy (DDRC_REG_MR_WR_BUSY) is detected | R/W | 0b |
| 15:0 | Mode Register Write Data (REG_DDRC_MR_DATA) Write data to the mode register. | R/W | 0000h |

7.5.23 DDR2 Control Register 12 (DDR_CONTROL_12)

Offset: 0058h Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|---|------|---------|
| 31:6 | RESERVED | RO | - |
| 5 | Disable DLL Calibration (REG_DDRC_DIS_DLL_CALIB) When '0', the controller will issue dll_calib periodically. When '1', disable dll_calib generated by the controller. The user must issue the dll_calib signal using Request DLL Calibration (CO_GS_DLL_CALIB) input. This input is changeable on the fly. | R/W | 0b |
| 4 | Request DLL Calibration (CO_GS_DLL_CALIB) Command that indicates to the controller to issue a dll_calib to the DRAM. | R/SC | 0b |
| 3:2 | RESERVED | RO | - |
| 1 | Rank Refresh (CO_GS_RANK_REFRESH) Command that indicates to the controller to issue a refresh to the DRAM. | R/SC | 0b |
| 0 | Disable Auto Refresh (REG_DDRC_DIS_AUTO_REFRESH) When '1', disable auto-refresh generated by the controller. This input is changeable on the fly. When this transitions from 0 to 1, any pending refreshes will be immediately scheduled by the controller | R/W | 0b |

7.5.24 DDR2 Control Register 13 (DDR_CONTROL_13)

Offset: 005Ch Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|---|------|---------|
| 31:8 | RESERVED | RO | - |
| 7:0 | Minimum Time Between DLL Calibrations (REG_DDRC_DLL_CALIB_TO_MIN_X1024) This is the minimum amount of time between DLL calibrations (which will be executed whenever the controller is idle). Set this number higher to reduce the frequency of DLL calibrations, which can have a small impact on the latency of the first read request when the controller is idle. | R/W | 00h |

7.5.25 DDR2 PHY Control Register 0 (DDR_PHY_CTL_0)

Offset: 0100h Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|---|------|---------|
| 31:28 | First Read (REG_FIRST_RD) This value defines the delay from when a read command is issued by the DDR Controller to when the DRAM device sends out the first piece of read data. Set this value to AL+CL (Additive Latency + CAS Latency) | R/W | 0h |
| 27:24 | First Write (REG_FIRST_WR) This value defines the delay from when a write command is issued by the DDR controller to when the first piece of write data is sent by the controller. Set this value to AL+CL -1 (Additive Latency + CAS Latency -1) | R/W | 0h |
| 23 | S1 Strength for Clock Signal (REG_DDR2_S1_CLK) Pad strength control for the DDR2 clock signal | R/W | 0b |
| 22 | S1 Strength for Control Signals (REG_DDR2_S1_CONTROL) Pad strength control for the DDR2 control signals except the clock | R/W | 0b |
| 21 | S1 Strength for Byte Lane 1 (REG_DDR2_S1_BYTE_LANE1) Pad strength control for the DDR2 data signals - Byte Lane 1. | R/W | 0b |
| 20 | S1 Strength for Byte Lane 0 (REG_DDR2_S1_BYTE_LANE0) Pad strength control for the DDR2 data signals - Byte Lane 0. | R/W | 0b |
| 19:18 | RESERVED | RO | - |
| 17:16 | DDR2 PHY Global ODT Control (REG_DDR2_ODT) ODT pad control for all DDR2 signals (except DQ & DQS which come from the DDR2 PHY) | R/W | 0b |
| 15 | DDR2 CKE Pads Output Enable Disable (REG_DDR2_OE_CKE_DISABLE) This bit controls output enable (OEN) for the CKE pads. Its default state forces the outputs off. Note: This bit must be programmed to 0b for normal PHY control of the CKE pad OE inputs. This field is protected by Reset Protection (RST_PROTECT) . | R/W | 1b |
| 14 | DDR2 Pads Output Enable Disable (REG_DDR2_OE_DISABLE) This bit controls output enable (OEN) for all DDR2 pads except for CKE pads. Its default state forces the outputs off. Note: This bit must be programmed to 0b for normal PHY control of the DDR2 pad OE inputs. This field is protected by Reset Protection (RST_PROTECT) . | R/W | 1b |
| 13 | DDR2 CKE Pads Power Down Control (REG_DDR2_PWD_CKE) This bit controls power down (PWD) for the CKE pads. It is used to power down the pad's receiving circuits. Note: This bit must be programmed to 0b for proper operation. This field is protected by Reset Protection (RST_PROTECT) . | R/W | 1b |

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|--|------|---------|
| 12 | DDR2 Pads Power Down Control (REG_DDR2_PWD) This bit controls power down (PWD) for all pads except for CKE pads. It is used to power down the pad's receiving circuits. Note: This bit must be programmed to 0b for proper operation. This field is protected by Reset Protection (RST_PROTECT) . | R/W | 1b |
| 11:10 | RESERVED | RO | - |
| 9:8 | DDR2 PHY Idle ODT Control (REG_PHY_IDLE_LOCAL_ODT) Value to drive on the 2-bit local_odt PHY outputs when not in write and read operation. | R/W | 0b |
| 7:6 | RESERVED | RO | - |
| 5:4 | DDR2 PHY Write ODT Control (REG_PHY_WR_LOCAL_ODT) Value to drive on the 2-bit local_odt PHY outputs when output is enabled on DQ and DQS (write operation). | R/W | 0b |
| 3:2 | RESERVED | RO | - |
| 1:0 | DDR2 PHY Read Local ODT (REG_PHY_RD_LOCAL_ODT) Value to drive on the 2-bit local_odt PHY outputs when output is not enabled on DQ and DQS (read operation). | R/W | 0b |

7.5.26 DDR2 PHY Control Register 1 (DDR_PHY_CTL_1)

Offset: 0104h Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|---|------|---------|
| 31 | Disable DRAM Clock (STOP_CLK) 0 = dram_clk is running 1 = dram_clk is stopped | R/W | 0b |
| 30:21 | RESERVED | RO | - |
| 20 | Clear Read Pointer Reset Counter (RDC_FIFO_RST_ERR_CNT_CLR) Clear/reset for counter Read Pointer Reset Count (RDC_FIFO_RST_ERR_CNT)[3:0] . 0 = Not cleared 1 = Clear | R/W | 0b |
| 19:16 | Read Pointer Reset Count (RDC_FIFO_RST_ERR_CNT) Counter for counting how many times the pointers of read data capture FIFO are reset when the FIFO is not empty (an error). It gets saturated after reaching 0xF. Cleared by Clear Read Pointer Reset Counter (RDC_FIFO_RST_ERR_CNT_CLR) . It is recommended that connecting this output to a register so that it can be checked when needed. | RO | 0h |
| 15:12 | Maximum Lock Delay Difference (REG_MAX_LOCK_DIFF) Master DLL lock signal is asserted when the difference between averaged delay sr_out and incoming delay sr_in in output filter module is less than (or equal to) the value of this field. | R/W | 4h |
| 11:7 | RESERVED | RO | - |
| 6 | Disable Resetting of the Read Capture FIFO Pointers (DIS_CALIB_RST) Disable the resetting of the Read Capture FIFO pointers. The pointers are reset to ensure that the PHY can recover if the appropriate number of DQS edges is not observed after a read command (which can happen when the DQS squelch timing is manually overridden via the debug registers). 0 = Enable 1 = Disable. | R/W | 0b |
| 5 | RESERVED | RO | - |
| 4 | Use Fixed Delay for FIFO Read Enable (USE_FIXED_RE) When 0, PHY uses the not_empty method to generate FIFO read enable. When 1, PHY generates FIFO read enable after fixed number of clock cycles. Note: THIS FIELD SHOULD ALWAYS BE INITIALIZED TO 1 BY THE SOFTWARE. | R/W | 0b |
| 3:2 | RESERVED | RO | - |
| 1:0 | Read Capture FIFO Read Enable Delay (RDC_WE_TO_RE_DELAY) Read Capture FIFO read is enabled after (Write Latency (REG_DDRC_WRITE_LATENCY) + RDC_WE_TO_RE_DELAY + 2) clock cycles. Valid if Use Fixed Delay for FIFO Read Enable (USE_FIXED_RE) is high. Note: THIS FIELD SHOULD ALWAYS BE INITIALIZED TO 2 BY THE SOFTWARE. | R/W | 0h |

7.5.27 DDR2 PHY Debug Register 0 (DDR_PHY_DEBUG_0)

Offset: 0110h Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|---|------|---------|
| 31 | Debug WL DLL Force 1 (DBG_WL_DLL_FORCE1) 0 = Use the value from the Master DLL. 1 = Overwrite the coarse and fine values calculated by the Master DLL going to the write slave, with the value in Debug WL DLL Value 1 (DBG_WL_DLL_VALUE1) . | R/W | 0b |
| 30:25 | RESERVED | RO | - |
| 24:16 | Debug WL DLL Value 1 (DBG_WL_DLL_VALUE1) If Debug WL DLL Force 1 (DBG_WL_DLL_FORCE1) is one, replace: <ul style="list-style-type: none"> ■ Fine value from the Master DLL going to write slave 1 with DBG_WL_DLL_VALUE1[1:0]. ■ Coarse Value from the Master DLL going to write slave 1 with DBG_WL_DLL_VALUE1[8:2]. | R/W | 000h |
| 15 | Debug WL DLL Force 0 (DBG_WL_DLL_FORCE0) 0 = Use the value from the Master DLL. 1 = Overwrite the coarse and fine values calculated by the Master DLL going to the write slave, with the value in Debug WL DLL Value 0 (DBG_WL_DLL_VALUE0) . | R/W | 0b |
| 14:9 | RESERVED | RO | - |
| 8:0 | Debug WL DLL Value 0 (DBG_WL_DLL_VALUE0) If Debug WL DLL Force 0 (DBG_WL_DLL_FORCE0) is one, replace: <ul style="list-style-type: none"> ■ Fine value from the Master DLL going to write slave 0 with DBG_WL_DLL_VALUE0[1:0]. ■ Coarse Value from the Master DLL going to write slave 0 with DBG_WL_DLL_VALUE0[8:2]. | R/W | 000h |

7.5.28 DDR2 PHY Debug Register 1 (DDR_PHY_DEBUG_1)

Offset: 0114h Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|--|------|---------|
| 31 | Debug RC DLL Force 1 (DBG_RC_DLL_FORCE1) 0 = Use the value from the Master DLL. 1 = Overwrite the coarse and fine values calculated by the Master DLLs going to data slice "1" with the value in Debug RC DLL Value 1 (DBG_RC_DLL_VALUE1) . | R/W | 0b |
| 30:25 | RESERVED | RO | - |
| 24:16 | Debug RC DLL Value 1 (DBG_RC_DLL_VALUE1) If Debug RC DLL Force 1 (DBG_RC_DLL_FORCE1) is one, replace: ■ Fine value from the Master DLL with DBG_RC_DLL_VALUE1[1:0]. ■ Coarse Value from the Master DLL with DBG_RC_DLL_VALUE1[8:2]. | R/W | 000h |
| 15 | Debug RC DLL Force 0 (DBG_RC_DLL_FORCE0) 0 = Use the value from the Master DLL. 1 = Overwrite the coarse and fine values calculated by the Master DLLs going to data slice "0" with the value in Debug RC DLL Value 0 (DBG_RC_DLL_VALUE0) . | R/W | 0b |
| 14:9 | RESERVED | RO | - |
| 8:0 | Debug RC DLL Value 0 (DBG_RC_DLL_VALUE0) If Debug RC DLL Force 0 (DBG_RC_DLL_FORCE0) is one, replace: ■ Fine value from the Master DLL with DBG_RC_DLL_VALUE0[1:0]. ■ Coarse Value from the Master DLL with DBG_RC_DLL_VALUE0[8:2]. | R/W | 000h |

7.5.29 DDR2 PHY Debug Register 2 (DDR_PHY_DEBUG_2)

Offset: 0118h Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|--|------|---------|
| 31:24 | Read DQS Slave Ratio 1 (RD_DQS_SLAVE_RATIO1) Ratio value for read DQS slave DLL. This is the fraction of a clock cycle represented by the shift to be applied to the read DQS in units of 256ths. In other words, the full-cycle tap value from the master DLL will be scaled by this number over 256 to get the delay value for the slave delay line. | R/W | 00h |
| 23:16 | Read DQS Slave Ratio 0 (RD_DQS_SLAVE_RATIO0) Ratio value for read DQS slave DLL. This is the fraction of a clock cycle represented by the shift to be applied to the read DQS in units of 256ths. In other words, the full-cycle tap value from the master DLL will be scaled by this number over 256 to get the delay value for the slave delay line. | R/W | 00h |
| 15:8 | Write DQS Slave Ratio 1 (WR_DQS_SLAVE_RATIO1) Ratio value for write DQS slave DLL. This is the fraction of a clock cycle represented by the shift to be applied to the read DQS in units of 256ths. In other words, the full-cycle tap value from the master DLL will be scaled by this number over 256 to get the delay value for the slave delay line. | R/W | 00h |
| 7:0 | Write DQS Slave Ratio 0 (WR_DQS_SLAVE_RATIO0) Ratio value for write DQS slave DLL. This is the fraction of a clock cycle represented by the shift to be applied to the read DQS in units of 256ths. In other words, the full-cycle tap value from the master DLL will be scaled by this number over 256 to get the delay value for the slave delay line. | R/W | 00h |

7.5.30 DDR2 PHY Debug Register 3 (DDR_PHY_DEBUG_3)

Offset: 011Ch Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|--|------|---------|
| 31:25 | RESERVED | RO | - |
| 24:16 | Debug Status of In Delay Value 0 (DGB_STATUS_IN_DELAY_VALUE0) The Coarse and Fine values going into the Filter in Master DLL 0. DGB_STATUS_IN_DELAY_VALUE0[1:0] = Fine value DGB_STATUS_IN_DELAY_VALUE0[8:2] = Coarse value | RO | X |
| 15:9 | RESERVED | RO | - |
| 8:0 | Debug Status Master DLL Slave Value 0 (DGB_STATUS_MASTER_DLL_SLAVE_VALUE0) Shows the current Coarse and Fine values going to all the Write Slave DLLs. DGB_STATUS_MASTER_DLL_SLAVE_VALUE0[1:0] = Fine value DGB_STATUS_MASTER_DLL_SLAVE_VALUE0[8:2] = Coarse value | RO | X |

7.5.31 DDR2 PHY Debug Register 4 (DDR_PHY_DEBUG_4)

Offset: 0120 Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|--|------|---------|
| 31:21 | RESERVED | RO | - |
| 20 | Master DLL 0 Locked (DBG_STATUS_MASTER_DLL_LOCK0) Status signal: 0 = Master DLL is not locked 1 = Master DLL is locked | RO | X |
| 19:18 | RESERVED | RO | - |
| 17:16 | Debug Status of In Lock State 0 (DBG_STATUS_IN_LOCK_STATE0) Lock status from the Output Filter module inside the Master DLL 0. DBG_STATUS_IN_LOCK_STATE0[0] = Fine lock DBG_STATUS_IN_LOCK_STATE0[1] = Coarse lock | RO | X |
| 15:0 | RESERVED | RO | - |

7.5.32 DDR2 PHY Debug Register 5 (DDR_PHY_DEBUG_5)

Offset: 0124h Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|--|------|---------|
| 31:25 | RESERVED | RO | - |
| 24:16 | Debug Status of In Delay Value 1 (DBG_STATUS_IN_DELAY_VALUE_1) The Coarse and Fine values going into the Filter in Master DLL 1. DBG_STATUS_IN_DELAY_VALUE_1[1:0] = Fine value DBG_STATUS_IN_DELAY_VALUE_1[6:0] = Coarse value | RO | X |
| 15:9 | RESERVED | RO | |
| 8:0 | Debug Status Master DLL Slave Value 1 (DGB_STATUS_MASTER_DLL_SLAVE_VALUE1) Shows the current Coarse and Fine values going to all the Write Slave DLLs from master DLL 1. DGB_STATUS_MASTER_DLL_SLAVE_VALUE1[1:0] = Fine value DGB_STATUS_MASTER_DLL_SLAVE_VALUE1[8:2] = Coarse value | RO | X |

7.5.33 DDR2 PHY Debug Register 6 (DDR_PHY_DEBUG_6)

Offset: 0128h Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|---|------|---------|
| 31:21 | RESERVED | RO | - |
| 20 | Master DLL 1 Locked (DBG_STATUS_MASTER_DLL_LOCK1) Status signal: 0 = Master DLL 1 is not locked 1 = Master DLL 1 is locked | RO | X |
| 19:18 | RESERVED | RO | - |
| 17:16 | Debug Status of In Lock State 1 (DBG_STATUS_IN_LOC_STATE1) Lock status from the Output Filter module inside the Master DLL 1. DBG_STATUS_IN_LOC_STATE1[0] = Fine lock DBG_STATUS_IN_LOC_STATE1[1] = Coarse lock | RO | X |
| 15:0 | RESERVED | RO | - |

7.5.34 DDR2 PHY Debug Register 7 (DDR_PHY_DEBUG_7)

Offset: 012Ch Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|---|------|---------|
| 31:25 | RESERVED | RO | - |
| 24:16 | Debug Status RC DLL Slave Value 1 (DBG_STATUS_RC_DLL_SLAVE_VALUE1) Shows the current Coarse and Fine values going to the Read Slave DLLs in data slice 1. DBG_STATUS_RC_DLL_SLAVE_VALUE1[1:0] = Fine value DBG_STATUS_RC_DLL_SLAVE_VALUE1[8:2] = Coarse value | RO | X |
| 15:9 | RESERVED | RO | - |
| 8:0 | Debug Status RC DLL Slave Value 0 (DBG_STATUS_RC_DLL_SLAVE_VALUE0) Shows the current Coarse and Fine values going to the Read Slave DLLs in data slice 0. DBG_STATUS_RC_DLL_SLAVE_VALUE0[1:0] = Fine value DBG_STATUS_RC_DLL_SLAVE_VALUE0[8:2] = Coarse value | RO | X |

7.5.35 DDR2 PHY Debug Register 8 (DDR_PHY_DEBUG_8)

Offset: 0130h Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|---|------|---------|
| 31:25 | RESERVED | RO | - |
| 24:16 | Debug Write Slave DLL 1 Value (DBG_WR_SLAVE_DLL_1_VALUE) Shows the current coarse and fine values going to all the write slave delay lines in the PHY top module from master DLL 1: DBG_WR_SLAVE_DLL_1_VALUE[1:0] = Fine value. DBG_WR_SLAVE_DLL_1_VALUE[8:2] = Coarse value. | RO | X |
| 15:9 | RESERVED | RO | - |
| 8:0 | Debug Write Slave DLL 0 Value (DBG_WR_SLAVE_DLL_0_VALUE) Shows the current coarse and fine values going to all the write slave delay lines in the PHY top module from master DLL 1: DBG_WR_SLAVE_DLL_0_VALUE[1:0] = Fine value. DBG_WR_SLAVE_DLL_0_VALUE[8:2] = Coarse value. | RO | X |

Chapter 8 EEPROM

8.1 Description

The device uses a EEPROM to store the default values for the USB descriptors. It supports most Atmel 93C46x type of EEPROMs.

Note: A 3-wire style 4K EEPROM that is organized for 256/512 x 8-bit operation must be used.

Various system level resets, as specified in [Table 6.1, “Chip Level Resets,” on page 98](#), cause the EEPROM contents to be loaded into the device. After a reset, the EEPROM controller attempts to read the first byte of data from the EEPROM. If the value 0xA5 is read from the first address, then the EEPROM controller will assume that an external Serial EEPROM is present.

The EEPROM Controller will then load the entire contents of the EEPROM into an internal 512 byte SRAM. The contents of the SRAM are accessed by the CTL block as needed (I.E. to fill Get Descriptor commands).

The device may not respond to the USB host until the EEPROM loading sequence has completed. Therefore, after reset the USB PHY is kept in the disconnect state until the EEPROM load has completed.

The host can initiate a reload of the EEPROM by issuing the RELOAD command via the [EEPROM Command Register \(E2P_CMD\)](#). If the first byte read from the EEPROM is not 0xA5, it is assumed that the EEPROM is not present, or not programmed, and the reload will fail. The [EPC Data Loaded \(EPC_LOADED\)](#) bit indicates a successful reload of the EEPROM data.

Note: It is not recommended that the reload be used as part of normal operation as race conditions can occur with USB Commands that access descriptor data. It is best for the host to issue a SRST to reload the EEPROM data.

The EEPROM Controller also allows the Host system to read, write and erase the contents of the Serial EEPROM using the [EEPROM Command Register \(E2P_CMD\)](#) and [EEPROM Data Register \(E2P_DATA\)](#). The Command and Data registers are accessed through the SCSR.

Note: The EEPROM device powers-up in the erase/write disabled state. To modify the contents of the EEPROM the host must first issue the EWEN command.

To write to the EEPROM the host must first write the desired data into the Data register. It must then initiate the cycle by performing a single write to the E2P_CMD register in which it sets the [EPC Busy \(EPC_BSY\)](#) bit, writes the appropriate address, and sets the [EPC Command \(EPC_CMD\)](#) field to WRITE. The completion of the write cycle is indicated when the [EPC Busy \(EPC_BSY\)](#) bit is cleared. Other commands that modify the EEPROM contents are executed using the same mechanism.

If an operation is attempted, and an EEPROM device does not respond within 30mS, the EPC must time-out, and the [EPC Time-out \(EPC_TIMEOUT\)](#) bit in the [EEPROM Command Register \(E2P_CMD\)](#) must be set.

To read from the EEPROM, the host must first initiate the cycle by performing a single write to the Command register in which it sets the [EPC Busy \(EPC_BSY\)](#) bit, writes the appropriate address and sets the [EPC Command \(EPC_CMD\)](#) field to READ. Valid data is available when the [EPC Busy \(EPC_BSY\)](#) bit is cleared, indicating to the host that it can read valid data from the Data register.

Figure 8.1 illustrates the host accesses required to perform an EEPROM Read or Write.

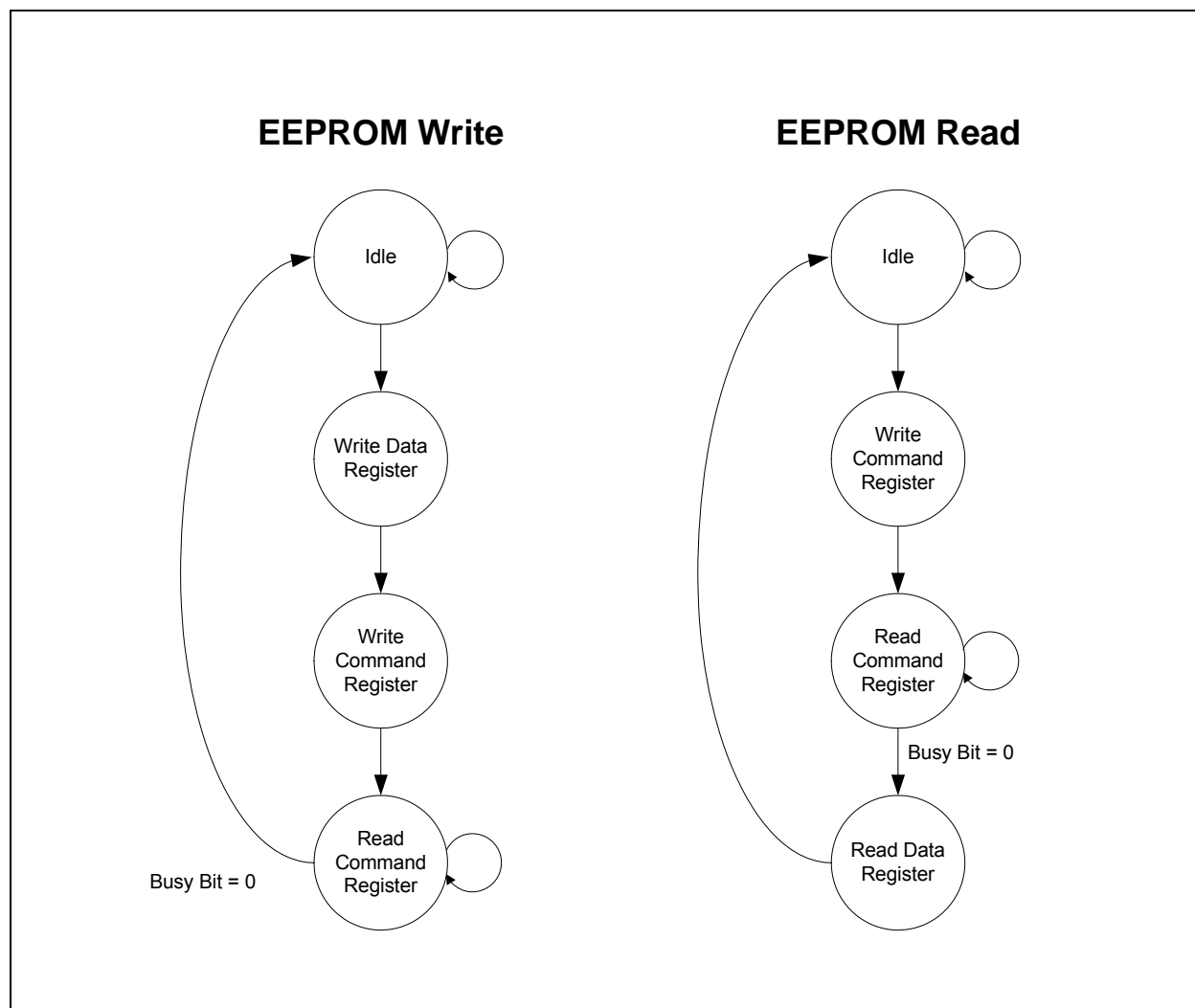


Figure 8.1 EEPROM Access Flow Diagram

8.1.1 Command Register Format

Refer to [Section 8.5.1, "EEPROM Command Register \(E2P_CMD\)"](#) for a detailed description of this register.

8.1.2 Data Register Format

Refer to [Section 8.5.2, "EEPROM Data Register \(E2P_DATA\)"](#) for a detailed description of this register.

8.2 EEPROM Format

[Table 8.1](#) illustrates the format in which data is stored inside of the EEPROM.

Note the EEPROM offsets are given in units of 16-bit word offsets. A length field with a value of zero indicates that the field does not exist in the EEPROM. The device will use the field's HW default value in this case.

Note: For the device descriptor the only valid values for the length are 0 and 18.

Note: For the configuration and interface descriptor the only valid values for the length are 0 and 18.

Note: For the BOS Block, the length varies and is dependent on block components.

Note: For SS Configuration Block, only valid values for the length are 0 and 1Eh

Note: The EEPROM programmer must ensure that if a string descriptor does not exist in the EEPROM, the referencing descriptor must contain 00h for the respective string index field.

Note: If all string descriptor lengths are zero, then a Language ID will not be supported.

Note: All reserved EEPROM bits must be set to 0.

Table 8.1 EEPROM Format

| EEPROM ADDRESS | EEPROM CONTENTS |
|----------------|--|
| 00h | 0xA5 |
| 01h | Full-Speed Polling Interval for Interrupt Endpoint |
| 02h | High-Speed Polling Interval for Interrupt Endpoint |
| 03h | Super-Speed Polling Interval for Interrupt Endpoint |
| 04h | Configuration Flags 0 [7:0] |
| 05h | Configuration Flags 0 [15:8] |
| 06h | Configuration Flags 0 [23:16] |
| 07h | Configuration Flags 0 [31:24] |
| 08h | Configuration Flags 1 [7:0] |
| 09h | Configuration Flags 1 [15:8] |
| 0Ah | Configuration Flags 1 [23:16] |
| 0Bh | Configuration Flags 1 [31:24] |
| 0Ch | Configuration Flags 2 [7:0] |
| 0Dh | Configuration Flags 2 [15:8] |
| 0Eh | Configuration Flags 2 [23:16] |
| 0Fh | Configuration Flags 2 [31:24] |
| 10h | Configuration Flags 3 [7:0] |
| 11h | Configuration Flags 3 [15:8] |
| 12h | Configuration Flags 3 [23:16] |
| 13h | Configuration Flags 3 [31:24] |
| 14h | Software Configuration Data Structure Length (bytes) Note 8.1 |
| 15h | Software Configuration Data Structure Word Offset Note 8.1 |
| 16h - 1Fh | RESERVED |

Table 8.1 EEPROM Format

| | |
|-----|---|
| 20h | GPIO Enable Config Byte 0 (GPIOEN_CFG0) GPIO Enable Register (GPIO_ENABLE) [7:0] initialization value |
| 21h | GPIO Enable Config Byte 1 (GPIOEN_CFG1) GPIO Enable Register (GPIO_ENABLE) [15:8] initialization value |
| 22h | GPIO Enable Config Byte 2 (GPIOEN_CFG2) GPIO Enable Register (GPIO_ENABLE) [23:16] initialization value |
| 23h | GPIO Enable Config Byte 3 (GPIOEN_CFG3) GPIO Enable Register (GPIO_ENABLE) [31:24] initialization value |
| 24h | GPIO Buffer Type Config Byte 0 (GPIOBUF_CFG0) GPIO Buffer Type Register (GPIO_BUF) [7:0] initialization value |
| 25h | GPIO Buffer Type Config Byte 1 (GPIOBUF_CFG1) GPIO Buffer Type Register (GPIO_BUF) [15:8] initialization value |
| 26h | GPIO Buffer Type Config Byte 2 (GPIOBUF_CFG2) GPIO Buffer Type Register (GPIO_BUF) [23:16] initialization value |
| 27h | GPIO Buffer Type Config Byte 3 (GPIOBUF_CFG3) GPIO Buffer Type Register (GPIO_BUF) [31:24] initialization value |
| 28h | GPIO Direction Config Byte 0 (GPIODIR_CFG0) GPIO Direction Register (GPIO_DIR) [7:0] initialization value |
| 29h | GPIO Direction Config Byte 1 (GPIODIR_CFG1) GPIO Direction Register (GPIO_DIR) [15:8] initialization value |
| 2Ah | GPIO Direction Config Byte 2 (GPIODIR_CFG2) GPIO Direction Register (GPIO_DIR) [23:16] initialization value |
| 2Bh | GPIO Direction Config Byte 3 (GPIODIR_CFG3) GPIO Direction Register (GPIO_DIR) [31:24] initialization value |
| 2Ch | GPIO Data Config Byte 0 (GPIOD_CFG0) GPIO Data Register (GPIO_DATA) [7:0] initialization value |
| 2Dh | GPIO Data Config Byte 1 (GPIOD_CFG1) GPIO Data Register (GPIO_DATA) [15:8] initialization value |
| 2Eh | GPIO Data Config Byte 2 (GPIOD_CFG2) GPIO Data Register (GPIO_DATA) [23:16] initialization value |
| 2Fh | GPIO Data Config Byte 3 (GPIOD_CFG3) GPIO Data Register (GPIO_DATA) [31:24] initialization value |
| 30h | Language ID [7:0] |
| 31h | Language ID [15:8] |
| 32h | Manufacturer ID String Descriptor Length (bytes) |
| 33h | Manufacturer ID String Descriptor EEPROM Word Offset |
| 34h | Product Name String Descriptor Length (bytes) |
| 35h | Product Name String Descriptor EEPROM Word Offset |
| 36h | Serial Number String Descriptor Length (bytes) |
| 37h | Serial Number String Descriptor EEPROM Word Offset |
| 38h | Configuration String Descriptor Length (bytes) |
| 39h | Configuration String Descriptor Word Offset |

Table 8.1 EEPROM Format

| | |
|-----|--|
| 3Ah | Interface String Descriptor Length (bytes) |
| 3Bh | Interface String Descriptor Word Offset |
| 3Ch | Binary Object Store (BOS) Block Length (Bytes) Note 8.2 |
| 3Dh | Binary Object Store (BOS) Block Word Offset |
| 3Eh | Super-Speed Device Descriptor Length (bytes) |
| 3Fh | Super-Speed Device Descriptor Word Offset |
| 40h | Super-Speed Configuration Block Length (bytes) Note 8.3 |
| 41h | Super-Speed Configuration Block Word Offset Note 8.3 |
| 42h | High-Speed Device Descriptor Length (bytes) |
| 43h | High-Speed Device Descriptor Word Offset |
| 44h | High-Speed Configuration and Interface Descriptor Length (bytes) |
| 45h | High-Speed Configuration and Interface Descriptor Word Offset |
| 46h | Full-Speed Device Descriptor Length (bytes) |
| 47h | Full-Speed Device Descriptor Word Offset |
| 48h | Full-Speed Configuration and Interface Descriptor Length (bytes) |
| 49h | Full-Speed Configuration and Interface Descriptor Word Offset |

Note: Locations 4Ah and above may be used for any purpose.

Note 8.1 Refer to the software programming manual for information concerning this data structure.

Note 8.2 This block may include Binary Object Store (BOS) Descriptor, USB 2.0 Extension Descriptor, Super-Speed Device Capabilities Descriptor, and Container ID Descriptor.

Note 8.3 This block **must** include the following descriptors in the following order:
SS Configuration descriptor
SS Interface descriptor
Bulk-Out Endpoint Companion descriptor
Interrupt Endpoint Companion descriptor

Table 8.2 describes Configuration Flags 0. If a configuration descriptor exists in the EEPROM, its values must agree with analogous values contained in the Configuration Flags 0. If they do not, unexpected results and untoward operation may occur.

Table 8.2 Configuration Flags 0

| BITS | DESCRIPTION |
|-------------|--|
| 31:14 | RESERVED |
| 13 | Clock Crystal Keep Alive Enable (CFG0_CLK_XTAL_EN) Refer to the Clock Crystal Keep Alive Enable (CLK_XTAL_EN) bit of the CPM Control Register (CPM_CTL) for permissible values. |
| 12 | Port Swap (CFG0_PORT_SWAP) Refer to the Port Swap (PORT_SWAP) bit of the USB Configuration Register (USB_CFG) for permissible values. |
| 11 | SW_MODE Polarity (CFG0_SW_MODE_POL) Refer to the SW_MODE Polarity (SW_MODE_POL) bit of the Hardware Configuration Register (HW_CFG) for permissible values. |
| 10 | LED Buffer Type (CFG0_LED_TYPE) Refer to the LED Buffer Type (LED_TYPE) bit of the LED Configuration Register 0 (LED_CFG0) for permissible values. |
| 9 | LED Polarity (CFG0_LED_POL) Refer to the LED Polarity (LED_POL) bit of the LED Configuration Register 0 (LED_CFG0) for permissible values. |
| 8 | LED Enable (CFG0_LED_ENABLE) Refer to the LED Enable (LED_EN) bit of the LED Configuration Register 0 (LED_CFG0) for permissible values. |
| 7 | Interrupt Pin Polarity (CFG0_INT_PIN_POL) Refer to the Interrupt Pin Polarity (INT_POL) bit of the Hardware Configuration Register (HW_CFG) for permissible values. |
| 6 | External Reset Polarity (CFG0_EXT_RST_POL) Refer to the External Reset Polarity (EXT_RST_POL) bit of the External Reset Configuration Register (EXT_RST_CFG) for permissible values. |
| 5:4 | Squelch Threshold (CFG0_SQU_THR) Refer to the Squelch Threshold (SQU_THR) field of the USB Configuration Register (USB_CFG) on page 68 for permissible values. |
| 3 | LPM Enable (CFG0_LPM_ENABLE) Refer to the LPM Enable (LPM_ENABLE) bit of the USB Configuration Register (USB_CFG) on page 68 for permissible values. |
| 2:1 | PHY Boost (CFG0_PHY_BOOST) Refer to the PHY Boost (PHY_BOOST) field of the USB Configuration Register (USB_CFG) on page 68 for permissible values. |
| 0 | Power Method (CFG0_PWR_SEL) Refer to the Power Method (PWR_SEL) bit of the USB Configuration Register (USB_CFG) on page 68 for permissible values. |

Note: LPM Enable and Power method specified in Configuration Flags 0 must agree with analogous quantities specified in descriptors. If they do not, unexpected results and untoward operation may occur.

Table 8.3 describes Configuration Flags 1.

Table 8.3 Configuration Flags 1

| BITS | DESCRIPTION |
|-------------|---|
| 31:30 | RESERVED |
| 29:24 | TX De-Emphasis At 3.5 dB (CFG1_TX_DEEMPH_3P5DB) Refer to the TX De-Emphasis At 3.5 dB (PCS_TX_DEEMPH_3P5DB) field of the USB PHY Parameters 0 Register (USB_PHY_PARAMS0) on page 72 for permissible values. |
| 23:22 | RESERVED |
| 21:16 | TX De-Emphasis at 6 dB (CFG1_TX_DEEMPH_6DB) Refer to the TX De-Emphasis at 6 dB (PCS_TX_DEEMPH_6DB) field of the USB PHY Parameters 0 Register (USB_PHY_PARAMS0) on page 72 for permissible values. |
| 15 | RESERVED |
| 14:8 | TX Amplitude For Full Swing Mode (CFG1_TX_SWING_FULL) Refer to the TX Amplitude For Full Swing Mode (PCS_TX_SWING_FULL) field of the USB PHY Parameters 0 Register (USB_PHY_PARAMS0) on page 72 for permissible values. |
| 7 | RESERVED |
| 6:0 | TX Amplitude For Low Swing Mode (CFG1_TX_SWING_LOW) Refer to the TX Amplitude For Low Swing Mode (PCS_TX_SWING_LOW) field of the USB PHY Parameters 0 Register (USB_PHY_PARAMS0) on page 72 for permissible values. |

Table 8.4 describes [Configuration Flags 2](#).

Table 8.4 Configuration Flags 2

| BITS | DESCRIPTION |
|-------------|---|
| 31:13 | RESERVED |
| 12:8 | Loss Of Signal Detection Threshold Level (CFG2_LOS_LEVEL) Refer to the Loss Of Signal Detection Threshold Level (LOS_LEVEL) field of the USB PHY Parameters 1 Register (USB_PHY_PARAMS1) on page 74 for permissible values. |
| 7:5 | RESERVED |
| 4:0 | TX Termination Offset (CFG2_TX_TERM_OFFSET) Refer to the TX Termination Offset (TX_TERM_OFFSET) field of the USB PHY Parameters 1 Register (USB_PHY_PARAMS1) on page 74 for permissible values. |

Table 8.5 describes [Configuration Flags 3](#).

Table 8.5 Configuration Flags 3

| BITS | DESCRIPTION |
|------|---|
| 31:3 | RESERVED |
| 2:1 | Spread Spectrum Clock Range(CFG3_SSC_RANGE) Refer to the Spread Spectrum Clock Range (SSC_RANGE) field of the USB PHY MPLL Control Register (USB_PHY_MPLL_CTL) on page 75 for permissible values. |
| 0 | Spread Spectrum Enable (CFG3_SSC_EN) Refer to the Spread Spectrum Enable (SSC_EN) bit of the USB PHY MPLL Control Register (USB_PHY_MPLL_CTL) on page 75 for permissible values. |

8.3 EEPROM Defaults

The signature value of 0xA5 is stored at address 0. A different signature value indicates to the EEPROM controller that no EEPROM is attached to the device. In this case, the hardware default values used are as shown in [Table 8.6](#). See [Section 4.5, "USB Descriptors,"](#) on [page 49](#) for more information about the default USB values.

Table 8.6 EEPROM Defaults

| FIELD | DEFAULT VALUE |
|--|--------------------------|
| Full-Speed Polling Interval | 01h |
| High-Speed Polling Interval | 04h |
| Super-Speed Polling Interval | 06h |
| Maximum Burst Size for Bulk-Out Endpoint | 07h |
| Configuration Flags 0 | 00000008h |
| Configuration Flags 1 | 16206935h |
| Configuration Flags 2 | 00000900h |
| Configuration Flags 3 | 00000005h |
| Maximum Power | Note 8.4 |
| Vendor ID | 0424h |
| Product ID | 9D00h |

Note 8.4 Default value is FAh (500mA) when operating in USB 2.0 mode and 70h (900mA) when operating in USB 3.0 mode.

8.4 Customized Operation Without EEPROM

The device provides the capability to customize operation without the use of an EEPROM. Descriptor information and initialization quantities normally fetched from EEPROM and used to initialize descriptors and elements of the Control and Status Registers may be specified via an alternate mechanism. This alternate mechanism involves the use of the Descriptor RAM in conjunction with the Attribute Registers and select elements of the System Control and Status Registers. The software device driver orchestrates the process by performing the following actions in the order indicated:

- Initialization of SCSR Elements in Lieu of EEPROM Load
- Attribute Register Initialization
- Descriptor RAM Initialization
- Enable Descriptor RAM and Attribute Registers as Source
- Inhibit Reset of Select SCSR Elements

The following subsections explain these actions. The attribute registers must be written prior to initializing the Descriptor RAM. Failure to do this will prevent the PWR_SEL and RMT_WKUP flags from being overwritten by the bmAttributes of the Configuration Descriptor.

8.4.1 Initialization of SCSR Elements in Lieu of EEPROM Load

During EEPROM operation, the following register fields are initialized by the hardware using the values contained in the EEPROM. In the absence of an EEPROM, the software device driver must initialize these quantities:

- LED Enable (LED_EN) bit of the LED Configuration Register 0 (LED_CFG0)
- LED Polarity (LED_POL) bit of the LED Configuration Register 0 (LED_CFG0)
- LED Buffer Type (LED_TYPE) bit of the LED Configuration Register 0 (LED_CFG0)
- GPIO Enable Register (GPIO_ENABLE)
- GPIO Buffer Type Register (GPIO_BUF)
- GPIO Direction Register (GPIO_DIR)
- GPIO Data Register (GPIO_DATA)
- SW_MODE Polarity (SW_MODE_POL) bit of the Hardware Configuration Register (HW_CFG)
- Interrupt Pin Polarity (INT_POL) bit of the Hardware Configuration Register (HW_CFG)
- External Reset Polarity (EXT_RST_POL) bit of the External Reset Configuration Register (EXT_RST_CFG)
- Clock Crystal Keep Alive Enable (CFG0_CLK_XTAL_EN) bit of the CPM Control Register (CPM_CTL)
- Super-speed Maximum Burst Size (MAX_BURST) field of the USB Configuration Register (USB_CFG)
- Squelch Threshold (SQU_THR) field of the USB Configuration Register (USB_CFG)
- PHY Boost (PHY_BOOST) field of the USB Configuration Register (USB_CFG)
- LPM Enable (LPM_ENABLE) field of the USB Configuration Register (USB_CFG)
- Power Method (PWR_SEL) field of the USB Configuration Register (USB_CFG)
- TX De-Emphasis At 3.5 dB (PCS_TX_DEEMPH_3P5DB) field of the USB PHY Parameters 0 Register (USB_PHY_PARAMS0)
- TX De-Emphasis at 6 dB (PCS_TX_DEEMPH_6DB) field of the USB PHY Parameters 0 Register (USB_PHY_PARAMS0)
- TX Amplitude For Full Swing Mode (PCS_TX_SWING_FULL) field of the USB PHY Parameters 0 Register (USB_PHY_PARAMS0)

- TX Amplitude For Low Swing Mode (PCS_TX_SWING_LOW) field of the USB PHY Parameters 0 Register (USB_PHY_PARAMS0)
- Loss Of Signal Detection Threshold Level (LOS_LEVEL) field of the USB PHY Parameters 1 Register (USB_PHY_PARAMS1)
- TX Termination Offset (TX_TERM_OFFSET) field of the USB PHY Parameters 1 Register (USB_PHY_PARAMS1)
- Spread Spectrum Clock Range (SSC_RANGE) field of the USB PHY MPLL Control Register (USB_PHY_MPLL_CTL)
- Spread Spectrum Enable (SSC_EN) bit of the USB PHY MPLL Control Register (USB_PHY_MPLL_CTL)

8.4.2 Attribute Register Initialization

The Attribute Registers are as follows:

- BOS Attributes Register (BOS_ATTR)
- SS Attributes Register (SS_ATTR)
- HS Attributes Register (HS_ATTR)
- FS Attributes Register (FS_ATTR)
- String Attributes Register 0 (STRNG_ATTR0)
- String Attributes Register 1 (STRNG_ATTR1)

All of these registers contain fields defining the lengths of the descriptors or block contents written into the Descriptor RAM. If an item is not written into the Descriptor RAM, the associated entry in the Attributes Register must be written as 0. Writing an erroneous or illegal length will result in untoward operation and unexpected results.

Note: The software device driver must initialize these registers prior to initializing the Descriptor RAM.

Note: The bmAttributes field of the SS, HS, and FS descriptors in descriptor RAM (if present) must be consistent with the contents of the [Power Method \(PWR_SEL\)](#) field of the [USB Configuration Register \(USB_CFG\)](#).

8.4.3 Descriptor RAM Initialization

The Descriptor RAM contents are initialized using the Data Port registers. The Data Port registers are used to select the Descriptor RAM and write the descriptor elements into it. The Descriptor RAM is 512 bytes in length. Every descriptor/block written into the Descriptor RAM must be DWORD aligned. The Attribute Registers discussed in [Section 8.4.2](#) must be written with the length of the descriptors written into the Descriptor RAM. If a descriptor/block is not used, hence not written into Descriptor RAM, its length must be written as 0 into the associated Attribute Register.

Note: The Attribute Registers must be initialized before the Descriptor RAM.

Note: Address 0 of the Descriptor RAM is always reserved for the Language ID, even if it will not be supported.

The descriptors/blocks must be written in the following order, starting at address 0 of the RAM and observing the DWORD alignment rule:

- Language ID (2 bytes)
- Manufacturing String Descriptor (String Index 1)
- Product Name String Descriptor (String Index 2)
- Serial Number String Descriptor (String Index 3)
- Configuration String Descriptor (String Index 4)

- Interface String Descriptor (String Index 5)
- BOS Block
- SS Device Descriptor
- SS Configuration Block
- HS Device Descriptor
- HS Configuration Descriptor
- FS Device Descriptor
- FS Configuration Descriptor

An example of Descriptor RAM use is illustrated in [Figure 8.2](#). In it, the BOS Block contains the BOS Descriptor (5 bytes), a USB 2.0 Extension Descriptor (7 bytes), and a Super-Speed USB Device Capability Descriptor (10 bytes) for a total length of 22 bytes. The SS Configuration Block always contains a Super-Speed Configuration Descriptor (9 bytes), a Interface Descriptor (9 bytes), a Bulk-Out Endpoint Companion Descriptor (6 bytes), and a Interrupt Endpoint Companion Descriptor (6 bytes) for a total length of 30 bytes.

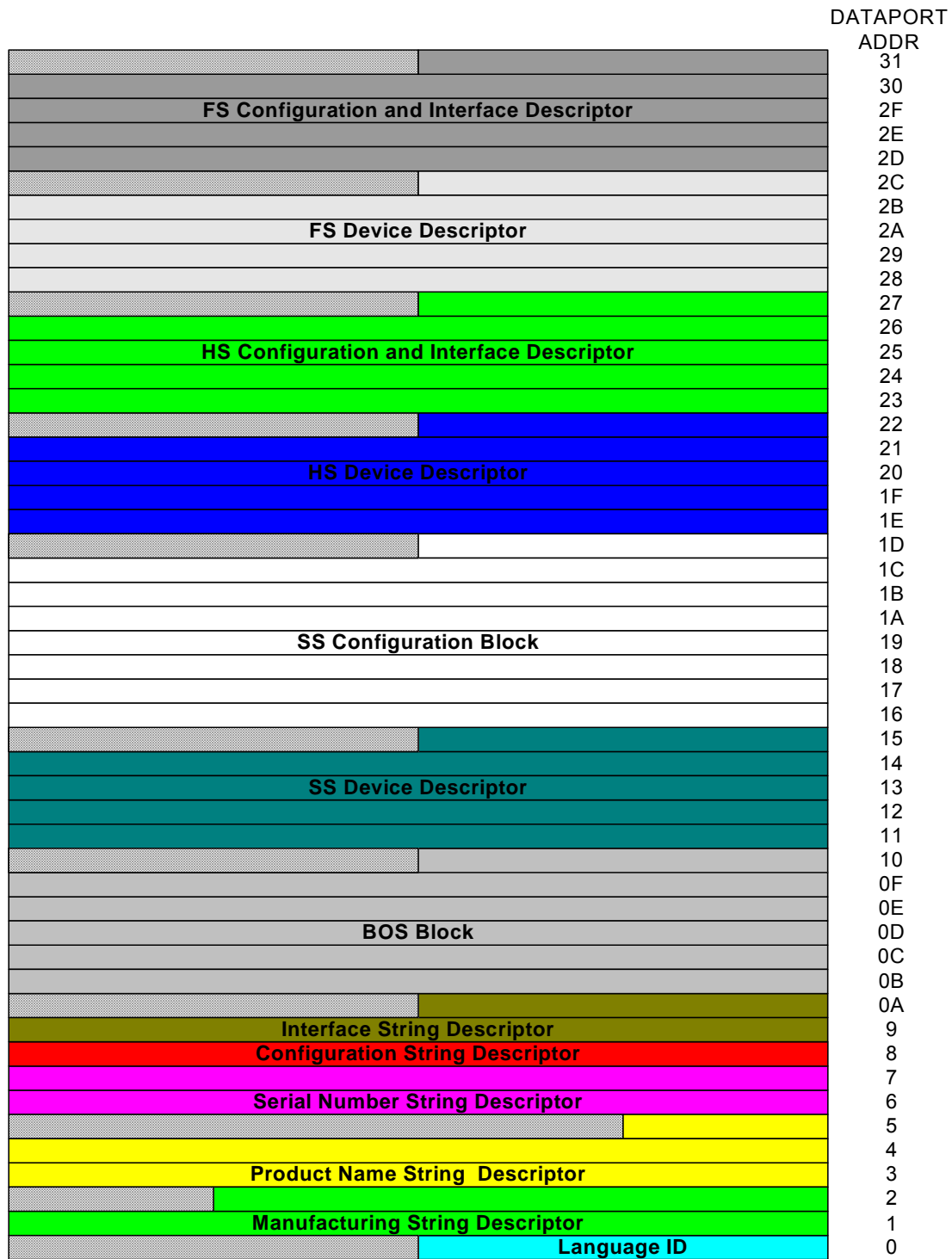
As in the case of descriptors specified in EEPROM, the following restrictions apply to descriptors written into Descriptor RAM:

1. For Device Descriptors, the only valid values for the length are 0 and 18. The descriptor size for the Device Descriptors specified in the Descriptor RAM is a don't care and always overwritten by HW to 0x12 when transmitting the descriptor to the host.
2. The descriptor type for Device Descriptors specified in the Descriptor RAM is a don't care and is always overwritten by HW to 0x1 when transmitting the descriptor to the host.
3. For the Configuration and Interface descriptor, the only valid values for the length are 0 and 18. The descriptor size for the Device Descriptors specified in the Descriptor RAM is a don't care and always overwritten by HW to 0x12 when transmitting the descriptor to the host.
4. The descriptor type for the configuration descriptors specified in the Descriptor RAM is a don't care and always overwritten by HW to 0x2 when transmitting the descriptor to the host.
5. If a string descriptor does not exist in the Descriptor RAM, the referencing descriptor must contain 00h for the respective string index field.
6. If all string descriptor lengths are zero, then a Language ID will not be supported.

Note: The first entry in the Descriptor RAM is always reserved for the Language ID, even if it will not be supported.

Note: Descriptors having bcdUSB value other than 0300h will result in unwanted behavior and untoward results. Descriptor having bMaxPacketSize other than 09h when operating in Super-Speed mode and other than 40h when operating in Full-Speed or High-Speed mode will result in unwanted behavior and untoward results. Descriptors having bNumConfigurations with values other than 1 will result in unwanted behavior and untoward results.

The **RAM Test Mode Enable (DP_SEL_TESTEN)** bit must be deasserted after programming the descriptor RAM.



 = Unused Space Required For Alignment Purposes

Figure 8.2 Descriptor RAM Example

8.4.4 Enable Descriptor RAM and Attribute Registers as Source

The [EEPROM Emulation Enable \(EEM\)](#) bit of the [Hardware Configuration Register \(HW_CFG\)](#) must be configured by the software device driver to use the Descriptor RAM and the Attribute Registers for custom operation. Upon assertion of [EEPROM Emulation Enable \(EEM\)](#), the hardware will utilize the Descriptor information contained in the Descriptor RAM, the Attributes Registers, and the values of the items listed in [Section 8.4.1](#) to facilitate custom operation.

8.4.5 Inhibit Reset of Select SCSR Elements

The software device driver must take care to ensure that the contents of the Descriptor RAM and SCSR register content critical to custom operation using Descriptor RAM are preserved across reset operations other than POR. The driver must configure the [Reset Protection \(RST_PROTECT\)](#) bit of the [Hardware Configuration Register \(HW_CFG\)](#) in order to accomplish this.

The following registers have contents that can be preserved across all resets other than POR. Consult the register's description for additional details.

- Descriptor RAM
- Attribute Registers
- [Hardware Configuration Register \(HW_CFG\)](#)
- [External Reset Configuration Register \(EXT_RST_CFG\)](#)
- [LED Configuration Register 0 \(LED_CFG0\)](#)
- [GPIO Enable Register \(GPIO_ENABLE\)](#)
- [GPIO Buffer Type Register \(GPIO_BUF\)](#)
- [GPIO Direction Register \(GPIO_DIR\)](#)
- [GPIO Data Register \(GPIO_DATA\)](#)
- [USB Configuration Register \(USB_CFG\)](#)
- [USB PHY Parameters 0 Register \(USB_PHY_PARAMS0\)](#)
- [USB PHY Parameters 1 Register \(USB_PHY_PARAMS1\)](#)
- [USB PHY MPLL Control Register \(USB_PHY_MPLL_CTL\)](#)
- [CPM Control Register \(CPM_CTL\)](#)
- [DDR2 Control Register 9 \(DDR_CONTROL_9\)](#)
- [DDR2 PHY Control Register 0 \(DDR_PHY_CTL_0\)](#)

8.5 Control and Status Registers

These CSRs are available when sys_clk is disabled. This permits software to access system level configuration information such as audio characteristics and DDR2 speed.

Table 8.7 USB Control and Status Register Map

| ADDRESS OFFSET | SYMBOL | REGISTER NAME |
|-------------------|----------|-------------------------------|
| 6000h | E2P_CMD | EEPROM Command Register |
| 6004h | E2P_DATA | EEPROM Data Register |
| 6008h – 6FFFh | RESERVED | Reserved for future expansion |

8.5.1 EEPROM Command Register (E2P_CMD)

Offset: 0000h Size: 32 bits

This register is used to control the read and write operations on the Serial EEPROM.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|--|------|---------|
| 31 | <p>EPC Busy (EPC_BSY) When a “1” is written into this bit, the operation specified in the EPC Command field is performed at the specified EEPROM address. This bit will remain set until the operation is complete. In the case of a read, this means that the Host can read valid data from the E2P Data register. The E2P_CMD and E2P_DATA registers should not be modified until this bit is cleared. In the case where a write is attempted and an EEPROM is not present, the EPC Busy remains busy until the EPC Time-out occurs. At that time, the busy bit is cleared.</p> <p>Note: EPC busy will be high immediately following power-up, chip-level, or USB reset. After the EEPROM controller has finished reading (or attempting to read) the USB Descriptors and Ethernet default register values, the EPC Busy bit is cleared.</p> | SC | 0b |

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|--|------|---------|
| 30:28 | <p>EPC Command (EPC_CMD) This field is used to issue commands to the EEPROM controller. The EPC will execute commands when the EPC Busy bit is set. A new command must not be issued until the previous command completes. This field is encoded as follows:</p> <p>000 = READ 001 = EWDS 010 = EWEN 011 = WRITE 100 = WRAL 101 = ERASE 110 = ERAL 111 = RELOAD</p> <p>READ (Read Location): This command will cause a read of the EEPROM location pointed to by EPC Address. The result of the read is available in the E2P_DATA register.</p> <p>EWDS (Erase/Write Disable): After issued, the EEPROM will ignore erase and write commands. To re-enable erase/write operations, issue the EWEN command.</p> <p>EWEN (Erase/Write Enable): Enables the EEPROM for erase and write operations. The EEPROM will allow erase and write operations until the Erase/Write Disable command is sent, or until power is cycled.</p> <p>Note: The EEPROM device will power-up in the erase/write-disabled state. Any erase or write operations will fail until an Erase/Write Enable command is issued.</p> <p>WRITE (Write Location): If erase/write operations are enabled in the EEPROM, this command will cause the contents of the E2P_DATA register to be written to the EEPROM location selected by the EPC Address field.</p> <p>WRAL (Write All): If erase/write operations are enabled in the EEPROM, this command will cause the contents of the E2P_DATA register to be written to every EEPROM memory location.</p> <p>ERASE (Erase Location): If erase/write operations are enabled in the EEPROM, this command will erase the location selected by the EPC Address field.</p> <p>ERAL (Erase All): If erase/write operations are enabled in the EEPROM, this command will initiate a bulk erase of the entire EEPROM.</p> <p>RELOAD (Data Reload): Instructs the EEPROM controller to reload the data from the EEPROM. If a value of A5h is not found in the first address of the EEPROM, the EEPROM is assumed to be un-programmed and the Reload operation will fail. The "Data Loaded" bit indicates a successful load of the data.</p> <p>Note: A failed reload operation will result in no change to descriptor information or register contents. These items will not be set to default values as a result of the reload failure.</p> | R/W | 000b |
| 27:11 | RESERVED | RO | - |

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|--|------|---------|
| 10 | <p>EPC Time-out (EPC_TIMEOUT) If an EEPROM operation is performed, and there is no response from the EEPROM within 30mS, the EEPROM controller will time-out and return to its idle state. This bit is set when a time-out occurs, indicating that the last operation was unsuccessful.</p> <p>Note: If the EEDI pin is pulled-high (default if left unconnected), EPC commands will not time out if the EEPROM device is missing. In this case, the EPC Busy bit will be cleared as soon as the command sequence is complete. It should also be noted that the ERASE, ERAL, WRITE and WRAL commands are the only EPC commands that will time-out if an EEPROM device is not present and the EEDI signal is pulled low.</p> | R/WC | 0 |
| 9 | <p>EPC Data Loaded (EPC_LOADED) When set, this bit indicates that a valid EEPROM was found, and that the USB and Data programming has completed normally. This bit is set after a successful load of the data after power-up, or after a RELOAD command has completed.</p> | R/WC | 0 |
| 8:0 | <p>EPC Address (EPC_ADDR) The 9-bit value in this field is used by the EEPROM Controller to address a specific memory location in the Serial EEPROM. This is a BYTE aligned address.</p> | R/W | 00h |

8.5.2 EEPROM Data Register (E2P_DATA)

Offset: 0004h Size: 32 bits

This register is used in conjunction with the E2P_CMD register to perform read and write operations to the Serial EEPROM.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|--|------|---------|
| 31:8 | RESERVED | RO | - |
| 7:0 | EEPROM Data (EPC_DATA) Value read from or written to the EEPROM. | R/W | 00h |

Chapter 9 HDMI

9.1 Overview

The device integrates an HDMI/DVI Controller that is compliant with revision 1.3 of the HDMI specification. It consists of both a transmitter and PHY. DVI output, compliant with revision 1.0 of the DVI specification, is also supported. Deep color and High-bandwidth Digital Content Protection (HDCP) are not supported. RGB video pixel encoding is supported, while YCbCr 4:4:4 and YCbCr 4:2:2 are not.

S/PDIF and single channel I2S audio are supported. A separate master clock input (MCLK), coherent with the input, is required for time-stamping purposes. Due to bandwidth requirements MLP (18 Mbps) and DTS-HD are not supported. The controller supports 2-channel uncompressed PCM data (IEC 60958) and compressed bit stream data for multi-channel (IEC 61937) formats. Audio is transmitted with HDMI data and audio sampling rates range from 32 to 192 KHz. Typical usage would be two channel uncompressed audio, with a target sampling rate of 44.1 or 48 KHz.

Master I2C interface for Display Data Channel (DDC) connection is available. Pins are shared with the internal I2C controller.

Monitor detection is supported via Hot Plug and Receiver detection. The Hot Plug Detect (HPD) pin is 5V tolerant.

Note: Detailed HDMI information is confidential. Contact your Microchip sales representative for additional information.

Chapter 10 Display Controller

10.1 Introduction

The UFX7000/UFX6000 Display Controller Subsystem (DCTL) is a basic CRT/LCD display controller that interfaces to a frame buffer on one side and the display on the other.

Features include:

- Programmable video resolution of up to 2048 x 1152 or 1920 x 1200 (WUXGA).
- 8, 16 and 24 bit color.
- 8 to 30 bit color table lookup.
- Gamma correction (3 x 8 bit to 10 bit color table lookup).
- Deep pixel FIFO buffer to tolerate frame buffer read access latency and to minimize instantaneous frame buffer RAM bandwidth requirements.
- vsync, hsync, composite sync, composite blank.
- Programmable sync and blank polarities.

10.2 Block Diagram

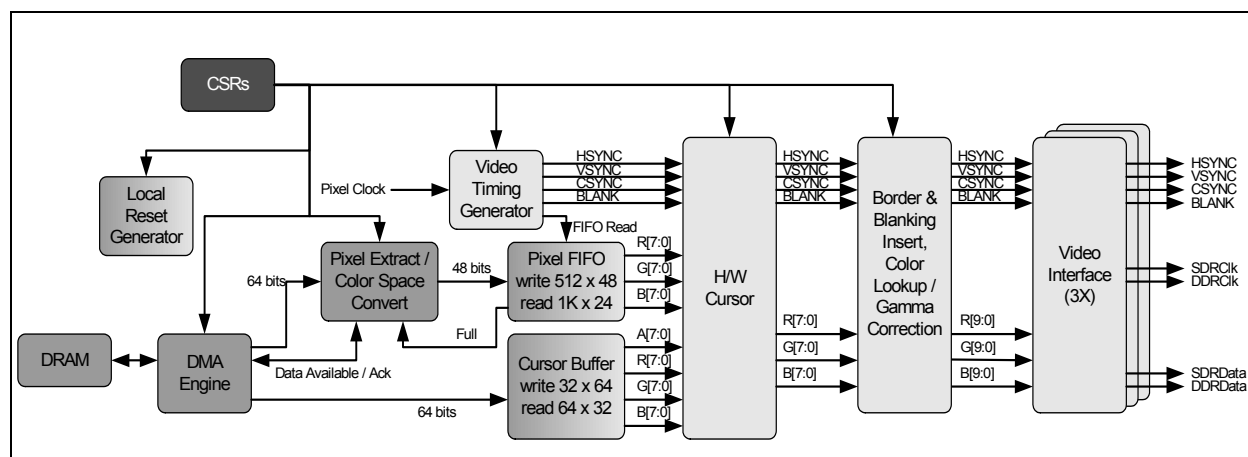


Figure 10.1 Display Controller Block Diagram

10.3 Configuration

10.3.1 Controller Reset / Enable / Status

The DCTL is reset by setting the [Reset \(RESET\)](#) bit of the [Display Control Register \(DISPLAY_CTRL\)](#). Setting this self-clearing bit causes the controller to be reset to its initial state. All registers are set to their default values.

The [Video Sub-system Enable \(ENABLE\)](#) bit of the [Display Control Register \(DISPLAY_CTRL\)](#) enables/disables the video sub-system. When this bit is set, frame buffers are read from system memory and the color and timing outputs are generated. When the bit is cleared, the video sub-system is reset. The controller finishes the current frame prior to stopping. After setting this bit, the [Display Controller Status \(DC_STS\)](#) bit of the [Display Status Register \(DISPLAY_STATUS\)](#) must be read to determine when the Display Controller has actually stopped.

Note: Configuration changes should only be attempted when the Display Controller is stopped, as determined by the [Display Controller Status \(DC_STS\)](#) bit.

In addition to containing the [Display Controller Status \(DC_STS\)](#) bit, the [Display Status Register \(DISPLAY_STATUS\)](#) contains interrupt and status bits that determine the operational state of the DCTL. Refer to [Display Status Register \(DISPLAY_STATUS\)](#) on page 192 for a complete description of these bits.

10.3.2 Horizontal and Vertical Timing Configuration

[Figure 10.2](#) and [Figure 10.3](#) illustrate the Horizontal and Vertical Timing of the DCTL. The units of horizontal and vertical timing are pixel clocks and lines, respectively. The pixel clock frequency must be set to support the desired screen resolution, based on industry standards. The labels within these figures correspond to the range values defined in the [Display Horizontal Size Register \(DISPLAY_H_SIZE\)](#), [Display Horizontal Blank Register \(DISPLAY_H_BLANK\)](#), [Display Horizontal Sync Register \(DISPLAY_H_SYNC\)](#), [Display Vertical Size Register \(DISPLAY_V_SIZE\)](#), [Display Vertical Blank Register \(DISPLAY_V_BLANK\)](#), and the [Display Vertical Sync Register \(DISPLAY_V_SYNC\)](#). All fields within these registers are programmed to the desired value -1.

Note: The aforementioned registers should only be changed when the DCTL is stopped, as indicated by the [Display Controller Status \(DC_STS\)](#) bit of the [Display Status Register \(DISPLAY_STATUS\)](#).

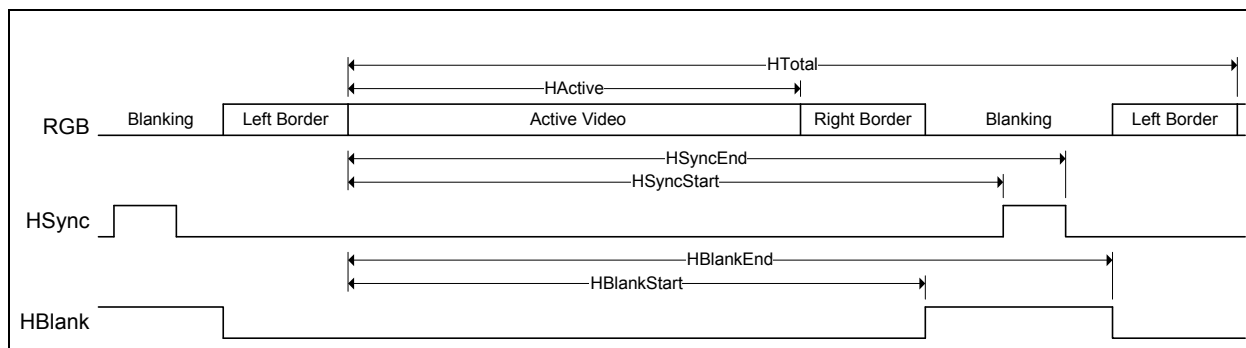


Figure 10.2 Horizontal Timing

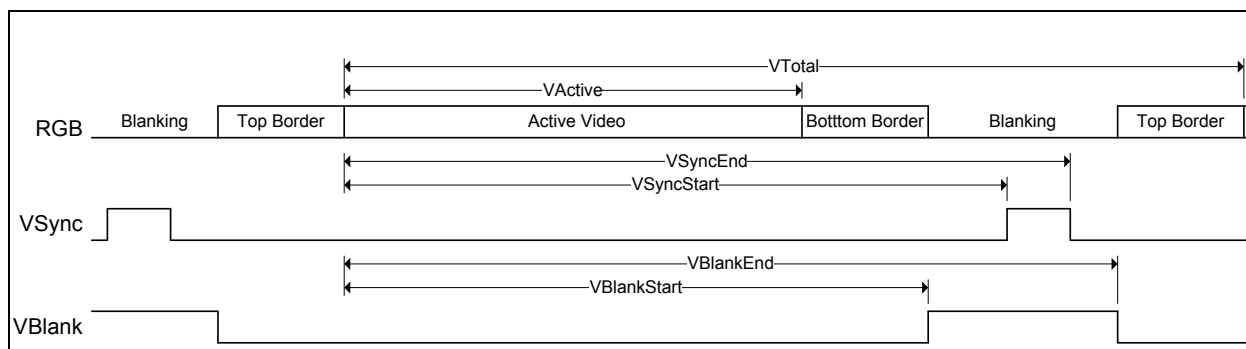


Figure 10.3 Vertical Timing

The [Display Horizontal Size Register \(DISPLAY_H_SIZE\)](#) contains the values for the [Horizontal Total \(H_TOTAL\)](#) and [Horizontal Active \(H_ACTIVE\)](#) fields illustrated in [Figure 10.2](#). [Horizontal Total \(H_TOTAL\)](#) is the total horizontal time in pixels that encompasses the active display, left and right borders, front and back porches, and horizontal sync pulse time. [Horizontal Active \(H_ACTIVE\)](#) is the

horizontal active (addressable) resolution. The first active display pixel is considered to be pixel number 0. This field should be programmed to less than **Horizontal Total (H_TOTAL)**.

The **Display Horizontal Blank Register (DISPLAY_H_BLANK)** contains the values for the **Horizontal Blank Start (H_BLANK_START)** and **Horizontal Blank End (H_BLANK_END)** fields illustrated in Figure 10.2. **Horizontal Blank Start (H_BLANK_START)** is the horizontal blanking start time relative to the start of the active display. The first active pixel is considered to be pixel number 0. This field is programmed to the desired right border size plus the horizontal active resolution minus 1. If this field is programmed to the same pixel as **Horizontal Active (H_ACTIVE)**, then there is no right border. This field should be programmed in the range \geq **Horizontal Active (H_ACTIVE)** and $<$ **Horizontal Total (H_TOTAL)**. **Horizontal Blank End (H_BLANK_END)** is the horizontal blanking end time relative to the start of the active display. This field is programmed to **Horizontal Total (H_TOTAL)** minus the desired left border size minus 1. If this field is programmed to the same value as **Horizontal Total (H_TOTAL)**, then there is no left border. This field should be programmed in the range $>$ **Horizontal Blank Start (H_BLANK_START)** and \leq **Horizontal Total (H_TOTAL)**.

The **Display Horizontal Sync Register (DISPLAY_H_SYNC)** contains the values for the **Horizontal Sync Start (H_SYNC_START)** and **Horizontal Sync End (H_SYNC_END)** fields illustrated in Figure 10.2. **Horizontal Sync Start (H_SYNC_START)** is the horizontal sync start time relative to the start of the active display. This field is programmed to the desired VESA value minus 1. **Horizontal Sync End (H_SYNC_END)** is the horizontal sync end time relative to the start of the active display. This field is programmed to the desired VESA value minus 1.

The **Display Vertical Size Register (DISPLAY_V_SIZE)** contains the values for the **Vertical Total (V_TOTAL)** and **Vertical Active (V_ACTIVE)** fields illustrated in Figure 10.2. **Vertical Total (V_TOTAL)** is the total vertical time in lines that encompasses the active display, top and bottom borders, front and back porches, and vertical sync pulse time. **Vertical Active (V_ACTIVE)** is the vertical active resolution in lines. This field should be programmed to the number of lines desired minus 1. The first vertical active display line is considered to be line number 0.

The **Display Vertical Blank Register (DISPLAY_V_BLANK)** contains the values for the **Vertical Blank Start (V_BLANK_START)** and **Vertical Blank End (V_BLANK_END)** fields illustrated in Figure 10.2. **Vertical Blank Start (V_BLANK_START)** is the start time in lines relative to the start of the active display. The first active line is considered to be line number 0. This field is programmed to the desired bottom border size plus the vertical active resolution minus 1. If this field is programmed to the same line as **Vertical Active (V_ACTIVE)**, then there is no bottom border. This field should be programmed in the range \geq **Vertical Active (V_ACTIVE)** and $<$ **Vertical Total (V_TOTAL)**. **Vertical Blank End (V_BLANK_END)** is the vertical blanking end time in lines relative to the start of the active display. This field is programmed to **Vertical Total (V_TOTAL)** minus the desired top border size minus 1. If this field is programmed to the same value as **Vertical Total (V_TOTAL)**, then there is no top border. This field should be programmed in the range $>$ **Vertical Blank Start (V_BLANK_START)** and \leq **Vertical Total (V_TOTAL)**.

The **Display Vertical Sync Register (DISPLAY_V_SYNC)** contains the values for the **Vertical Sync Start (V_SYNC_START)** and **Vertical Sync End (V_SYNC_END)** fields illustrated in Figure 10.2. **Vertical Sync Start (V_SYNC_START)** is the vertical sync start time in lines relative to the start of the active display. The first active line is considered to be line number 0. This field is programmed to the desired VESA value minus 1. **Vertical Sync End (V_SYNC_END)** is the vertical sync end time in lines relative to the start of the active display. This field is programmed to the desired VESA value minus 1.

An example of the combined horizontal and vertical timing is shown in Figure 10.4. The vertical sync pulse and vertical blank change on the same clock as the leading edge of the horizontal sync pulse. Note that the top/bottom/left/right borders are not shown.

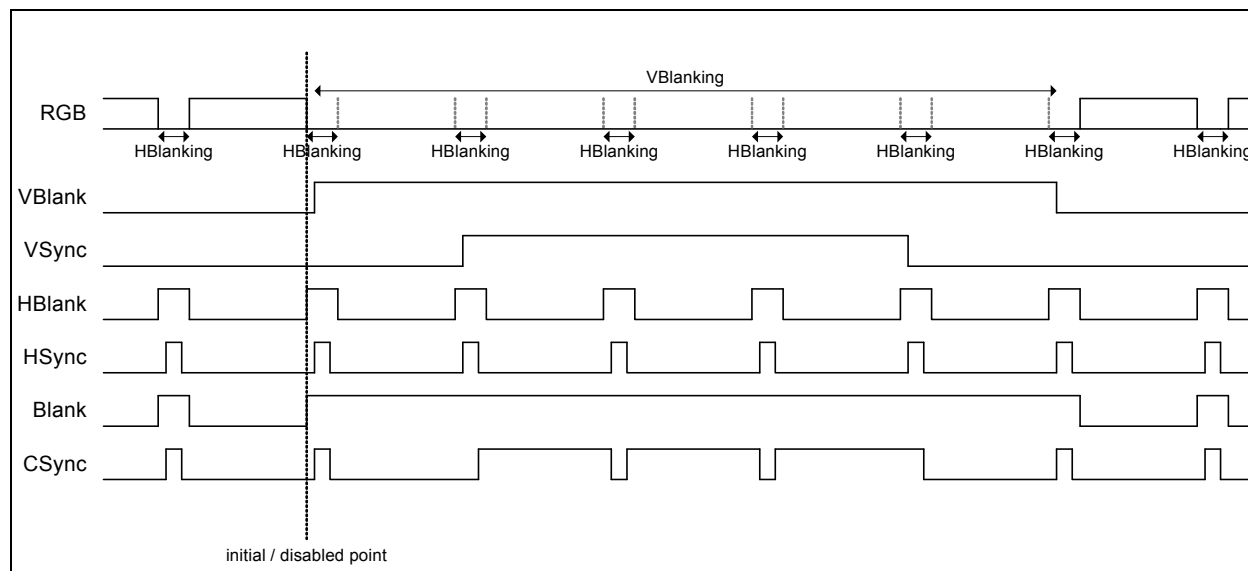


Figure 10.4 Combined Vertical and Horizontal Timing

10.3.2.1 Horizontal and Vertical Blanking Interrupts

During system operation, the [Horizontal Blank Start Interrupt \(H_BLANK_START_INT\)](#) and the [Vertical Blank Start Interrupt \(V_BLANK_START_INT\)](#) are propagated from the Display Controller to the [USB Interrupt Endpoint Status Register \(USB_INT_STS\)](#). These interrupts occur at the start of the Horizontal and Vertical Blank times, respectively. They are cleared by writing a 1 to the [Horizontal Blank Start Interrupt \(H_BLANK_START_INT\)](#) and [Vertical Blank Start Interrupt \(V_BLANK_START_INT\)](#) bits of the [Display Status Register \(DISPLAY_STATUS\)](#).

10.3.3 Display Blanking

The display may be forced into blanking by setting the [Force Display Blanking \(DISPLAY_BLANK\)](#) bit of the [Display Control Register \(DISPLAY_CTRL\)](#). When this bit is set, the RGB output is forced to zero and blanking is asserted. This bit may be changed at any time, however, it takes effect when either the current frame is finished or if the DCTL has been disabled via the [Video Sub-system Enable \(ENABLE\)](#).

Note: if the DCTL has not been disabled, then the HSYNC and VSYNC signals are still active and color output is turned off.

The [Blank Status \(BLANK_STS\)](#) and [Display Blank Pending \(DISPLAY_BLANK_PENDING\)](#) bits of the [Display Status Register \(DISPLAY_STATUS\)](#) are used to determine when a [Force Display Blanking \(DISPLAY_BLANK\)](#) request has been accomplished. [Display Blank Pending \(DISPLAY_BLANK_PENDING\)](#) is set when [Force Display Blanking \(DISPLAY_BLANK\)](#) is changed. It is cleared when [Blank Status \(BLANK_STS\)](#) has been updated to reflect the new setting of [Force Display Blanking \(DISPLAY_BLANK\)](#).

The [Display Blank Interrupt \(DISPLAY_BLANK_INT\)](#) is asserted whenever [Blank Status \(BLANK_STS\)](#) bit is updated to reflect a change in [Force Display Blanking \(DISPLAY_BLANK\)](#) and the [Display Blank Pending \(DISPLAY_BLANK_PENDING\)](#) bit was set. This interrupt is propagated to the [USB Interrupt Endpoint Status Register \(USB_INT_STS\)](#). It is cleared by writing a 1 to [Display Blank Interrupt \(DISPLAY_BLANK_INT\)](#).

Note: [Display Blank Pending \(DISPLAY_BLANK_PENDING\)](#) is also cleared whenever the [Display Controller Status \(DC_STS\)](#) bit is cleared.

Note: If the display is forced into blanking, scanning resumes on line 0, pixel 0 when blanking is deasserted.

10.3.4 Display Frame Buffer Processing

Display frames are resident in DRAM and the information contained therein is translated into video signals for display, under DMA control. The base addresses of the frame buffers are written into the [Display Frame Buffer Base Address Register \(DISPLAY_FRAME_BASE_ADDR\)](#) for use by the DMA controller. Up to four base addresses may be resident in the Display Frame Buffer Base Address Queue. As the DMA Engine starts each new frame, an address is removed from the queue. When the queue becomes empty, the last address is reused repeatedly. The maximum depth of the queue is specified by the [Frame Base Address Maximum Count \(FRAME_BASE_ADDR_MAX_COUNT\)](#) field of the [Display Control Register \(DISPLAY_CTRL\)](#) (maximum is 4). The current entry count can be read from the [Frame Base Address Pending Count \(FRAME_BASE_ADDR_PENDING_CNT\)](#) field of the [Display Status Register \(DISPLAY_STATUS\)](#). Once the queue is full, subsequent writes are ignored. The [Frame Base Address Queue Full \(FRAME_BASE_ADDR_QUEUE_FULL\)](#) flag of the [Display Status Register \(DISPLAY_STATUS\)](#) is set whenever the queue is full. The number of times the active Frame Base Address has changes is indicated by the [Display Frame Base Address Changed Count Register \(DISPLAY_FRAME_BASE_ADDR_CHANGED_CNT\)](#).

When the [Frame Base Address Queue Full \(FRAME_BASE_ADDR_QUEUE_FULL\)](#) flag goes low due to the DMA Controller utilizing the Display Frame Buffer Address at the head of the Display Frame Buffer Base Address Queue, the [Frame Base Address Interrupt \(FRAME_BASE_ADDR_INT\)](#) is asserted. This interrupt is propagated to the [USB Interrupt Endpoint Status Register \(USB_INT_STS\)](#). It is cleared by writing a 1 to the [Frame Base Address Interrupt \(FRAME_BASE_ADDR_INT\)](#) bit.

Note: Thus two mechanisms exist for determining whether the Display Frame Buffer Base Address queue may be replenished by writing to the [Display Frame Buffer Base Address Register \(DISPLAY_FRAME_BASE_ADDR\)](#). The [Frame Base Address Interrupt \(FRAME_BASE_ADDR_INT\)](#) can be used in conjunction with the [Frame Base Address Pending Count \(FRAME_BASE_ADDR_PENDING_CNT\)](#) to determine how many buffers may be written. Alternatively, the [Frame Base Address Pending Count \(FRAME_BASE_ADDR_PENDING_CNT\)](#) may be periodically polled.

The [Display Frame Buffer Length Register \(DISPLAY_FRAME_LEN\)](#) must be configured to the length of the display frame buffer in bytes. The value is set to the total pixel count multiplied by the bytes per pixel (bpp) value for the current display mode (1, 2, or 3 for 8-bit, 16-bit, and 24-bit modes, respectively, as determined by the [Color Depth \(COLOR_DEPTH\)](#) field of the [Display Control Register \(DISPLAY_CTRL\)](#). Total pixel count is $(\text{Horizontal Active (H_ACTIVE)} + 1) * (\text{Vertical Active (V_ACTIVE)} + 1)$.

Normally the memory slave to the DMA controller returns an OKAY bus response. In the event that a ERROR response is returned, the DMA controller will stop transfers, enter an error state and set the [Bus Error \(BUS_ERROR\)](#) bit in the [Display Status Register \(DISPLAY_STATUS\)](#). The error state is left only when the DCTL is disabled.

10.3.5 Display Data Interpretation

10.3.5.1 Border Generation

Border dimensions are determined by the horizontal and vertical timing registers. Right border size is [Horizontal Blank Start \(H_BLANK_START\)](#) - [Horizontal Active \(H_ACTIVE\)](#), left border size is [Horizontal Total \(H_TOTAL\)](#) - [Horizontal Blank End \(H_BLANK_END\)](#), top border size is [Vertical Total \(V_TOTAL\)](#) - [Vertical Blank End \(V_BLANK_END\)](#), and bottom border size is [Vertical Blank Start \(V_BLANK_START\)](#) - [Vertical Active \(V_ACTIVE\)](#).

The [Display Border Color Register \(DISPLAY_BORDER_COLOR\)](#) is used to set the desired Border Red, Green, and Blue values for all pixels within the border area.

The Border color is handled as specified in [Section 10.3.9, "Border and Blanking Insert, Color Lookup/Gamma Correction," on page 185](#). Refer to that section for details. The term "designated

register” used in that section refers to the [Display Border Color Register \(DISPLAY_BORDER_COLOR\)](#).

10.3.5.2 Display of Frame Buffer Contents

The DMA Controller moves Frame Buffer Data from DRAM to an internal FIFO whenever the DCTL is enabled and the [Display Frame Buffer Base Address Register \(DISPLAY_FRAME_BASE_ADDR\)](#) has been written with at least one Frame Buffer Base Address. Whenever the Frame Buffer Base Address Queue goes empty, the last Frame Buffer Base Address read will be used repeatedly.

The Pixel Extractor interfaces to the output of the DMA FIFO. It fetches Quad words from the DMA FIFO and extracts pixels from them. The pixel information is in RGB24 format (8-bits each of Red, Green, and Blue). Two pixels are generated on each system clock. Pixel FIFO underrun/overflow should never occur. However, the [Pixel FIFO Overflow \(PIXEL_FIFO_OVERFLOW\)](#) and [Pixel FIFO Underrun \(PIXEL_FIFO_UNDERRUN\)](#) flags in the [Display Status Register \(DISPLAY_STATUS\)](#) permit the software to poll for these untoward conditions.

Depending upon the [Color Depth \(COLOR_DEPTH\)](#) setting, a pixel either occupies 8, 16 or 24 bits. Conversely, a 64 memory QWORD either contains 2 2/3, 4 or 8 pixels. Each system clock (assuming data is available), two pixels are extracted, converted (if necessary) and written into the Pixel FIFO (assuming the FIFO is not full). The quad word to pixel mapping is shown in [Figure 10.5](#), [Figure 10.6](#), and [Figure 10.7](#) for each color depth setting.

As shown, in 24 bit mode, three quad words are mapped to eight pixels. Any unused bytes from a quad word are used to form the next pixel before the next quad word is processed.

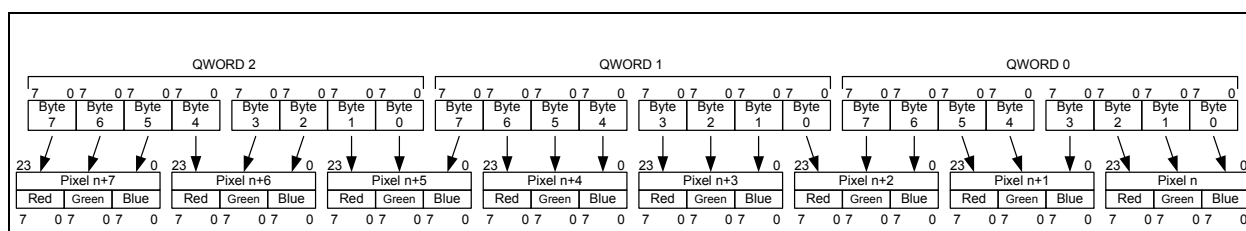


Figure 10.5 Quad Word to Pixel Mapping 24 bpb

16 bit mode splits each 16 bits into a 5 bit Red, a 6 bit Green and a 5 bit Blue value, each with leading 0's.

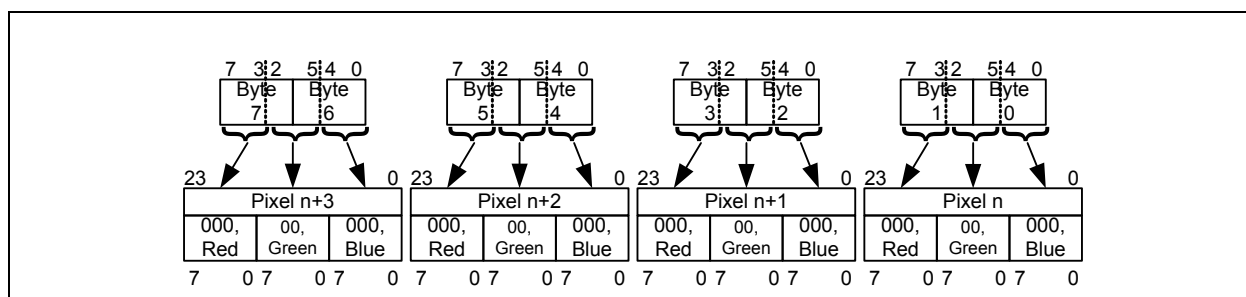


Figure 10.6 Quad Word to Pixel Mapping 16 bpb

8 bit mode replicates a byte onto each color of the pixel. Mapping / replicating at this point makes the subsequent color lookup function a uniform operation.

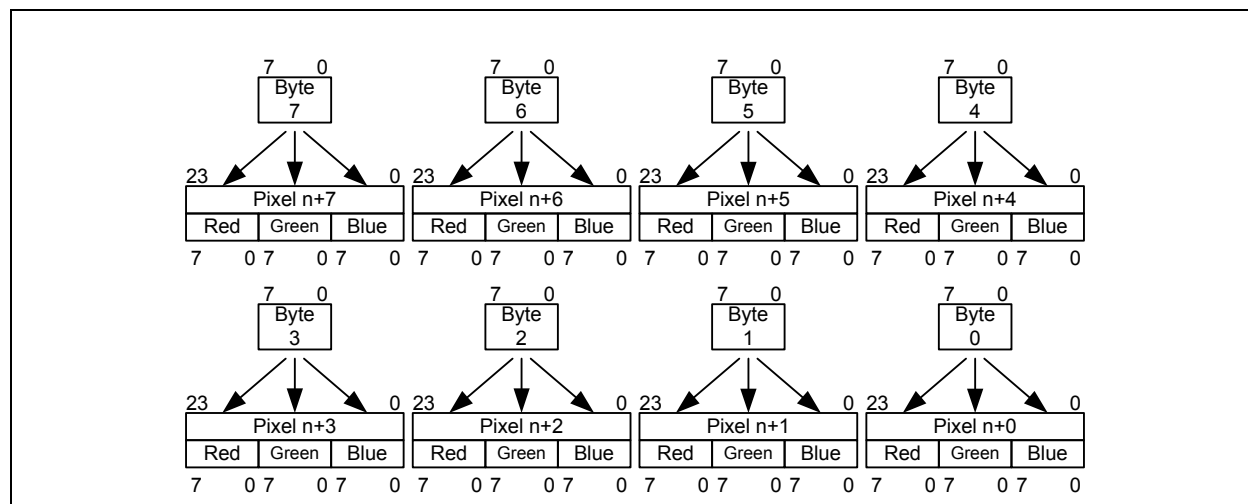


Figure 10.7 Quad Word to Pixel Mapping 8 bbp

Once a complete quad word is used, an acknowledge is sent to the DMA controller so that it may present the next quad word. A quad word is consumed every other system clock in 16 bit mode and every four clocks in 8 bit mode. In 24 bit mode, 3 quad words are used every four clocks. (All of the above assume that the Pixel FIFO is not full).

The mapped pixel data in RGB24 format contained within the Pixel FIFO is then presented to the HW cursor module, where it is combined with any active cursor information. Refer to [Section 10.3.7, "Combining Cursor and Display Frame Data,"](#) on page 184 for details.

10.3.6 Display Cursor

10.3.6.1 Double Buffered Display Cursor Registers Description

Pairs of registers are used to configure and determine the configuration status of the Display Cursor. One register, the Control Register of the pair, is R/W and is written to set the configuration parameter. The other register of the pair, the Status Register, is RO and provides the configuration parameter value currently in effect. The registers that are written to set the configuration parameters are double buffered. Writes to this type of register are performed to a holding register, not the actual hardware register controlling the cursor parameter. The contents of the holding register are transferred into the actual hardware register controlling the cursor following the vertical blank signal that occurs after a write to the [Display Cursor Position Register \(DISPLAY_CURSOR_POSITION\)](#). Writing to [DISPLAY_CURSOR_POSITION](#) also sets the [Cursor Trigger Armed \(CURSOR_TRIG\)](#) bit in the [Display Cursor Status Register \(DISPLAY_CURSOR_STATUS\)](#). Whenever [CURSOR_TRIG](#) is set, the contents of the control register should not be changed. Note also that double buffering is in effect only when the DCTL is running. If it is stopped, as indicated by the [Display Controller Status \(DC_STS\)](#) bit in the [Display Status Register \(DISPLAY_STATUS\)](#), the writes have immediate effect. [Table 10.1](#) enumerates the Display Cursor register pairs having the aforementioned properties.

Table 10.1 Display Cursor Status/Control Register Pairs

| Control Register | Status Register |
|---|--|
| Display Cursor Control Register (DISPLAY_CURSOR_CTRL) | Display Cursor Status Register (DISPLAY_CURSOR_STATUS) |

Table 10.1 Display Cursor Status/Control Register Pairs

| Control Register | Status Register |
|---|---|
| Display Cursor Base Address Register (DISPLAY_CURSOR_BASE_ADDR) | Display Cursor Current Base Address Register (DISPLAY_CURSOR_CUR_BASE_ADDR) |
| Display Cursor Position Register (DISPLAY_CURSOR_POSITION) | Display Cursor Current Position Register (DISPLAY_CURSOR_CUR_POSITION) |
| Display Cursor Color 0 Register (DISPLAY_CURSOR_COLOR_0) | Display Cursor Current Color 0 Register (DISPLAY_CURSOR_CUR_COLOR_0) |
| Display Cursor Color 1 Register (DISPLAY_CURSOR_COLOR_1) | Display Cursor Current Color 1 Register (DISPLAY_CURSOR_CUR_COLOR_1) |
| Display Cursor Color 2 Register (DISPLAY_CURSOR_COLOR_2) | Display Cursor Current Color 2 Register (DISPLAY_CURSOR_CUR_COLOR_2) |
| Display Cursor Color 3 Register (DISPLAY_CURSOR_COLOR_3) | Display Cursor Current Color 3 Register (DISPLAY_CURSOR_CUR_COLOR_3) |

The phrase “is determined by” in the following discussion refers to the control register of a pair. It should be construed as meaning that the register must be written to configure the parameter, regardless of when the parameter takes effect.

10.3.6.2 General Cursor Control/Status Parameters

Status information concerning the cursor is obtained by reading the contents of the [Display Cursor Status Register \(DISPLAY_CURSOR_STATUS\)](#), while the operational characteristics of the cursor are controlled by fields within the [Display Cursor Control Register \(DISPLAY_CURSOR_CTRL\)](#).

The operational mode of the cursor is determined by the [Cursor Mode \(CURSOR_MODE\)](#) field of the [Display Cursor Control Register \(DISPLAY_CURSOR_CTRL\)](#). [Table 10.3, “Cursor Modes,” on page 209](#) describes each mode. Color 0, Color 1, Color 2, and Color 3 referred to in this table are the [Display Cursor Color 0 Register \(DISPLAY_CURSOR_COLOR_0\)](#), [Display Cursor Color 1 Register \(DISPLAY_CURSOR_COLOR_1\)](#), [Display Cursor Color 2 Register \(DISPLAY_CURSOR_COLOR_2\)](#), and [Display Cursor Color 3 Register \(DISPLAY_CURSOR_COLOR_3\)](#), respectively, which are discussed in [Section 10.3.6.5](#).

The cursor size is determined by the [Cursor Size \(CURSOR_SIZE\)](#) field of the [Display Cursor Control Register \(DISPLAY_CURSOR_CTRL\)](#).

Enabling/Disabling the cursor is accomplished by setting/clearing the [Cursor Enable \(CURSOR_ENABLE\)](#) bit of the [Display Cursor Control Register \(DISPLAY_CURSOR_CTRL\)](#). Setting the bit causes cursor information to be read from system memory and generation of the cursor image. Clearing the bit terminates the reading of cursor information from system memory and stops display of the cursor. Clearing the bit does not result in the immediate cessation of cursor display.

Note: The [Display Cursor Control Register \(DISPLAY_CURSOR_CTRL\)](#) is double buffered and has the properties listed in [Section 10.3.6.1](#).

The current cursor mode, size and enable status in effect maybe determined by examining the [Cursor Current Mode \(CURSOR_CUR_MODE\)](#), [Cursor Current Size \(CURSOR_CUR_SIZE\)](#), and [Cursor Status \(CURSOR_STAT\)](#) fields of the [Display Cursor Status Register \(DISPLAY_CURSOR_STATUS\)](#).

Note, as previously discussed in [Section 10.3.6.1](#), the [Cursor Trigger Armed \(CURSOR_TRIG\)](#) bit of the [Display Cursor Status Register \(DISPLAY_CURSOR_STATUS\)](#) returns status indicating whether or not there is a pending cursor change. If set, a change is pending, if clear, no change is pending. A pending change may be cancelled by writing a one to this bit, however, there is no guarantee that the attempt at cancellation will be successful, as the time window formed by the CURSOR_TRIG being

set to the time when it is cleared as a result of transfer of data from the holding register to the hardware is quite narrow.

10.3.6.3 Setting Source of Cursor Image Data

Multiple cursor images may be stored in system memory. The [Display Cursor Base Address Register \(DISPLAY_CURSOR_BASE_ADDR\)](#) is used to set the 128 byte aligned starting address of the cursor image buffer to be used.

Note: The [Display Cursor Base Address Register \(DISPLAY_CURSOR_BASE_ADDR\)](#) is double buffered and has the properties listed in [Section 10.3.6.1](#).

The current cursor base address in effect can be obtained by reading the [Display Cursor Current Base Address Register \(DISPLAY_CURSOR_CUR_BASE_ADDR\)](#).

The DMA Engine uses the current base address to move cursor data from the DRAM into the Cursor Buffer. The system memory byte address for the cursor data is generated by a Y offset (line number within the cursor - provided by the Cursor Buffer block), along with [Cursor Current Base Address Upper Bits \(CURSOR_CUR_BASE_ADDR_HI\)](#), [Cursor Current Base Address Lower Bits \(CURSOR_CUR_BASE_ADDR_LO\)](#), [Cursor Current Size \(CURSOR_CUR_SIZE\)](#) (32x32, 48x48 or 64x64) and [Cursor Current Mode \(CURSOR_CUR_MODE\)](#) (2 bpp or 32 bpp).

The number of bytes/cursor line image is based on [Cursor Current Size \(CURSOR_CUR_SIZE\)](#) (32x32, 48x48 or 64x64) and [Cursor Current Mode \(CURSOR_CUR_MODE\)](#) (2 bpp or 32 bpp) and is as illustrated in the following table:.

| CURSOR MODE | CURSOR WIDTH | BYTES PER LINE OF CURSOR IMAGE |
|-------------|--------------|--------------------------------|
| 2 bpp | 32 | 8 |
| 2 bpp | 48 | 12 |
| 2 bpp | 64 | 16 |
| 32 bpp | 32 | 128 |
| 32 bpp | 38 | 192 |
| 32 bpp | 64 | 256 |

10.3.6.4 Display Cursor Positioning

Cursor Position on the screen is specified by the [Display Cursor Position Register \(DISPLAY_CURSOR_POSITION\)](#). It specifies the signed X, Y coordinates of the cursor, corresponding to the cursor's top left corner. Cursor pixels outside the active screen are clipped. When using negative values, at least one pixel must be positioned on the active screen. Writing this register sets [Cursor Trigger Armed \(CURSOR_TRIG\)](#).

Note: This register is also double buffered and operates with the constraints outlined in [Section 10.3.6.1](#), with the exception that it can be changed when [Cursor Trigger Armed \(CURSOR_TRIG\)](#) is set, however, the new value will be used only if the write occurs before the vertical blank signal.

The X, Y coordinates of the current cursor position in effect can be obtained by reading the [Display Cursor Current Position Register \(DISPLAY_CURSOR_CUR_POSITION\)](#).

10.3.6.5 Display Cursor Color Registers

The Display Cursor Color x Registers (DISPLAY_CURSOR_x), 0 ≤ x ≤ 3, are used to specify the Red, Green, and Blue portions of color x when operating in the 2 Color, 3 Color, or 4 Color cursor mode. These operating modes are all 2 bits per pixel modes and their use is determined by the setting of the [Cursor Current Mode \(CURSOR_CUR_MODE\)](#) field of the [Display Cursor Status Register \(DISPLAY_CURSOR_STATUS\)](#). These registers are also double buffered and operate with the constraints outlined in [Section 10.3.6.1](#). Like all the other double buffered registers associated with the cursor, these registers have analogs that indicate the current values in effect for the particular color they correspond to, i.e., [Display Cursor Current Color 0 Register \(DISPLAY_CURSOR_CUR_COLOR_0\)](#) indicates the [Cursor Current Red \(CURSOR_CUR_RED_0\)](#), [Cursor Current Green \(CURSOR_CUR_GREEN_0\)](#), and [Cursor Current Blue \(CURSOR_CUR_BLUE_0\)](#) values for color 0 that are in effect. In general, DISPLAY_CURSOR_CUR_COLOR_x can be accessed to determine CURSOR_CUR_RED_x, CURSOR_CUR_GREEN_x, and CURSOR_CUR_BLUE_x values.

10.3.7 Combining Cursor and Display Frame Data

The HW Cursor module is responsible for determining whether to display frame data from the Pixel FIFO or cursor data from the Cursor Buffer. It reads both data sources in parallel and utilizes the data read according to the values of the [Cursor Current Mode \(CURSOR_CUR_MODE\)](#), and the internal hardware pixel and line counters. Either 2-bits or 32-bits are read by the HW Cursor Module from the Cursor Buffer, depending on whether the [Cursor Current Mode \(CURSOR_CUR_MODE\)](#) indicates 2-bits/pixel or 32-bits/pixel operation.

The cursor timing is based on a comparison of the internal hardware horizontal pixel and vertical line counters with the left, right, top and bottom cursor positions. If the cursor is partially off screen, it is clipped. If it is fully off screen, or not enabled, it is not displayed.

Whenever the internal counters determine that a cursor element is not currently being processed, the pixel data read from the Pixel FIFO is passed on to the Border and Blanking Insert, Color Lookup/Gamma Correction Module without modification and the cursor data is ignored.

When the internal counters indicate cursor data is being processed, the HW Cursor Module performs the following processing, based on the [Cursor Current Mode \(CURSOR_CUR_MODE\)](#):

- [Cursor Current Mode \(CURSOR_CUR_MODE\) = 000 = 2 Color \(Monochrome\)](#)
- [Cursor Current Mode \(CURSOR_CUR_MODE\) = 001 = 3 Color](#)
- [Cursor Current Mode \(CURSOR_CUR_MODE\) = 010 = 4 Color](#)

The 2-bits read from the Cursor Buffer indicate the data source to be used, as indicated in [Table 10.3](#). If the data source is a color register, then the current value of that color register is fetched by the HW Cursor module and output to the Border and Blanking Insert, Color Lookup/Gamma Correction Module. If the data source is Frame data, then the Pixel FIFO element that was read is transformed as indicated in [Table 10.3](#) and output to the Border and Blanking Insert, Color Lookup/Gamma Correction Module. Refer to [Section 10.3.9](#) for details.

- [Cursor Current Mode \(CURSOR_CUR_MODE\) = 011 = RGB Color](#)

The 32-bit cursor data and the Frame Pixel data read by the HW Cursor Module are transformed by the algorithm indicated in [Table 10.3](#). The result is output to the Border and Blanking Insert, Color Lookup/Gamma Correction Module. Refer to [Section 10.3.9](#) for details.

Note: When in 8-bits per pixel display color mode, only the Blue Pixel data and Cursor Data byte are used by the algorithm. The Red and Green outputs of the HW Cursor Module are “don’t cares”.

Note: When in 16-bits per pixel display color mode, only the lower 5, 6, and 5 bits of Red, Blue, and Green are used. are used by the algorithm. The outputs of the HW Cursor Module associated with the remaining bits are “don’t cares”.

- [Cursor Current Mode \(CURSOR_CUR_MODE\) = 100 = Masked RGB Color](#)

The 32-bit cursor data and the Frame Pixel data read by the HW Cursor Module are transformed by the algorithm indicated in [Table 10.3](#). The result is output to the Border and Blanking Insert, Color Lookup/Gamma Correction Module. Refer to [Section 10.3.9](#) for details.

Note: When in 8-bits per pixel display color mode, only the Blue Pixel data and Cursor Data byte are used by the algorithm. The Red and Green outputs of the HW Cursor Module are “don’t cares”.

Note: When in 16-bits per pixel display color mode, only the lower 5, 6, and 5 bits of Red, Blue, and Green are used. are used by the algorithm. The outputs of the HW Cursor Module associated with the remaining bits are “don’t cares”.

10.3.8 Color Lookup/Gamma Correction Via The Color Lookup Table

The [Display Color Lookup Table \(DISPLAY_CLUT\)](#) is a 256 entry table that must be initialized by software prior to its use. Each entry in the table contains 10-bit Red, Green, and Blue values that are used to map RGB24 pixel data to a 30-bit color.

10.3.9 Border and Blanking Insert, Color Lookup/Gamma Correction

The Border and Blanking Insert, Color Lookup/Gamma Correction Module is responsible for displaying the border area of the screen, applying Color Lookup/Gamma correction values via use of the CLUT entries, and blanking the screen. Output data from this module is zeroed during the blanking time, which is determined by the setting of the [Horizontal Blank Start \(H_BLANK_START\)](#), [Horizontal Blank End \(H_BLANK_END\)](#), [Vertical Blank Start \(V_BLANK_START\)](#) and [Vertical Blank End \(V_BLANK_END\)](#) fields of the [Display Horizontal Blank Register \(DISPLAY_H_BLANK\)](#) and [Display Vertical Blank Register \(DISPLAY_V_BLANK\)](#), as well as the setting of the [Force Display Blanking \(DISPLAY_BLANK\)](#) bit of the [Display Control Register \(DISPLAY_CTRL\)](#). The polarity of the sync and blanking outputs is selectable via CSR settings.

Whenever the internal timing registers indicate blanking is to occur, all color output data is driven to zero.

Whenever the internal timing registers indicate a border element is to be displayed, the RGB24 value of the [Display Border Color Register \(DISPLAY_BORDER_COLOR\)](#) is read as used as the “designated item”. In all other cases, the RGB24 output of the HW Cursor Module is read and used as the “designated item”.

Use of the Color Lookup Table (CLUT) is enabled whenever [Gamma Correction and Color Lookup Enable \(COLOR_EN\)](#) bit in the [Display Control Register \(DISPLAY_CTRL\)](#) is set.

If COLOR_EN is set, designated item is mapped as follows, depending on the value of [Color Depth \(COLOR_DEPTH\)](#) specified in the [Display Control Register \(DISPLAY_CTRL\)](#):

- COLOR_DEPTH = 24-bits

Each byte of data in the designated item will be mapped separately to form the 30-bit color. The value of the 8-bit red portion of the designated item is used to index into the CLUT and the [CLUT Red \(RED\)](#) value at the indexed location is the 10-bit red value output. The values of 8-bit green and blue portions of the designated item are similarly used as indices into the CLUT and the values associated with the indices, [CLUT Green \(GREEN\)](#) and [CLUT Blue \(BLUE\)](#) are used as the remaining 20-bits of color.
- COLOR_DEPTH = 16-bits

The low order 5-bits of the red portion, 6-bits of the green portion, and 5-bits of the blue portion of the designated item are each packed into a byte and padded with leading 0s. The resulting bytes are mapped separately, using the CLUT, as described in the previous bulleted item, to form the 30-bit color.
- COLOR_DEPTH = 8-bits

The 8-bit blue data portion of the designated item is mapped into a 30-bit color. The 8-bit data is used to index into the CLUT and the [CLUT Red \(RED\)](#), [CLUT Green \(GREEN\)](#), and [CLUT Blue \(BLUE\)](#) entries at that index are used as a 3 color set and form the 30-bit color.

Use of the CLUT is disabled whenever COLOR_EN is cleared. Pixel data is not mapped, but is output as follows, depending on the value of [Color Depth \(COLOR_DEPTH\)](#) specified in the [Display Control Register \(DISPLAY_CTRL\)](#):

- COLOR_DEPTH = 24-bits
Each byte of data in the designated item will be used separately to form the 30-bit color. The value of the 8-bit red portion of the designated item has two bits of zero appended to the low order end to compose the 10-bit red value output. The values of the 8-bit green and 8-bit blue portions of the designated item are similarly appended with two bits of zero at their lower ends to formulate the remaining 20-bits of color output.
- COLOR_DEPTH = 16-bits
The low order 5-bits of the red portion, 6-bits of the green portion, and 5-bits of the blue portion of the designated item are each appended with 5-bits, 4-bits, and 5-bits, respectively of trailing zeros, to formulate the 30-bit color output.
- COLOR_DEPTH = 8 bits
The 8-bit blue color portion of the designated item is used to generate the 30-bit color. Two bits of zero are appended to the low order end of the blue color value of the designated item to form a 10-bit color value. This value is then replicated onto each color output (grey scale).

10.3.10 Interface Control

The Display Controller controls its output interfaces via three interface registers. Only one of these output interfaces should be selected at any time. [Display Interface Control 0 Register \(DISPLAY_INTERFACE_CTRL_0\)](#) controls the internal DAC interface, [Display Interface Control 1 Register \(DISPLAY_INTERFACE_CTRL_1\)](#) controls the internal HDMI interface, and [Display Interface Control 2 Register \(DISPLAY_INTERFACE_CTRL_2\)](#) controls the external RGB interface.

Note: The values in these registers should be changed only when the Display Controller is stopped, as indicated by the [Display Controller Status \(DC_STS\)](#) bit in the [Display Status Register \(DISPLAY_STATUS\)](#).

10.3.10.1 DAC Interface Control

This interface is enabled by setting the [Interface 0 Select \(INTF_SEL_0\)](#) bit of the [Display Interface Control 0 Register \(DISPLAY_INTERFACE_CTRL_0\)](#). When this interface is selected, the following fields should be initialized to the settings required by the externally connected device:

- [Interface 0 Horizontal Sync Polarity \(HSYNC_POL_0\)](#)
- [Interface 0 Vertical Sync Polarity \(VSYNC_POL_0\)](#)

The following field is set to its default value:

- [Interface 0 Vertical and Horizontal Sync Output Delay \(V_H_SYNC_DELAY_0\)](#)

When the DAC Interface is selected, the VDAC pins, shown in [Table 3.4, "VDAC Pins," on page 32 of Chapter 3, Pin Description and Configuration](#), are active.

10.3.10.2 HDMI Interface Control

This interface is enabled by setting the [Interface 1 Select \(INTF_SEL_1\)](#) bit of the [Display Interface Control 1 Register \(DISPLAY_INTERFACE_CTRL_1\)](#). When this interface is selected, the following fields must be set to the values required for proper operation:

- [Interface 1 Horizontal Sync Polarity \(HSYNC_POL_1\)](#)
- [Interface 1 Vertical Sync Polarity \(VSYNC_POL_1\)](#)

The following field is set to its default value:

- [Interface 1 Vertical and Horizontal Sync Output Delay \(V_H_SYNC_DELAY_1\)](#)

When the HDMI Interface is selected, the HDMI pins, shown in [Table 3.6, “HDMI Pins,” on page 34 of Chapter 3, Pin Description and Configuration](#), are active.

10.3.10.3 RGB Interface Control

This interface is enabled by setting the [Interface 2 Select \(INTF_SEL_2\)](#) bit of the [Display Interface Control 2 Register \(DISPLAY_INTERFACE_CTRL_2\)](#). When this interface is selected, the following fields should be initialized to the settings required by the externally connected device:

- [Interface 2 Double Data Rate Select \(DDR_SEL_2\)](#)
- [Interface 2 Double Data Rate Deep Color Enable \(DDR_COLOR_2\)](#)
- [Interface 2 Vertical and Horizontal Sync Output Delay \(V_H_SYNC_DELAY_2\)](#)
- [Interface 2 Output Clock Polarity \(OUT_CLOCK_POL_2\)](#)
- [Interface 2 Horizontal Sync Polarity \(HSYNC_POL_2\)](#)
- [Interface 2 Vertical Sync Polarity \(VSYNC_POL_2\)](#)
- [Interface 2 Blanking Polarity \(BLANK_POL_2\)](#)

When the RGB Interface is selected, the RGB pins, shown in [Chapter 3, Pin Description and Configuration](#), are active.

Note: The [Interface 2 Double Data Rate Select \(DDR_SEL_2\)](#) and [Interface 2 Double Data Rate Deep Color Enable \(DDR_COLOR_2\)](#) settings affect the pins used during RGB operations. Refer to [Table 3.3, “RGB / DDR Mode Mapping Table,” on page 31](#) for details.

Note: 1x data rate RGB data is mapped to the double data rate output whenever [Interface 2 Double Data Rate Select \(DDR_SEL_2\)](#) is set.

10.4 Display Controller Programming Outline

10.4.1 Initialization

If required, reset the Display Controller by writing the 8000h into the [Display Control Register \(DISPLAY_CTRL\)](#). Wait until the bit is clear before continuing, to ensure that the reset has been accomplished.

Configure screen size by programming the [Display Vertical Size Register \(DISPLAY_V_SIZE\)](#) and the [Display Horizontal Size Register \(DISPLAY_H_SIZE\)](#).

Configure the blank timing by programming the [Display Vertical Blank Register \(DISPLAY_V_BLANK\)](#) and the [Display Horizontal Blank Register \(DISPLAY_H_BLANK\)](#).

Configure the sync timing by programming the [Display Vertical Sync Register \(DISPLAY_V_SYNC\)](#) and the [Display Horizontal Sync Register \(DISPLAY_H_SYNC\)](#).

Configure the color used for the frame border by programming the [Display Border Color Register \(DISPLAY_BORDER_COLOR\)](#).

If the [Display Color Lookup Table \(DISPLAY_CLUT\)](#) will be utilized, initialize the 256 words of the table with the desired values.

Initialize the appropriate Interface Control Register, depending on the desired output:

- [Display Interface Control 0 Register \(DISPLAY_INTERFACE_CTRL_0\)](#) for VDAC output.
- [Display Interface Control 1 Register \(DISPLAY_INTERFACE_CTRL_1\)](#) for HDMI output.
- [Display Interface Control 2 Register \(DISPLAY_INTERFACE_CTRL_2\)](#) for RGB output.

If the Cursor is to be used, perform the following initialization:

- Configure the [Cursor Mode \(CURSOR_MODE\)](#), [Cursor Size \(CURSOR_SIZE\)](#) and [Cursor Enable \(CURSOR_ENABLE\)](#) fields of the [Display Cursor Control Register \(DISPLAY_CURSOR_CTRL\)](#).
- Depending on the selected [Cursor Mode \(CURSOR_MODE\)](#) Initialize the Display Cursor Color Registers ([Display Cursor Color 0 Register \(DISPLAY_CURSOR_COLOR_0\)](#) through [Display Cursor Color 3 Register \(DISPLAY_CURSOR_COLOR_3\)](#)) as required, with the desired RGB color characteristics.
- Program the [Display Cursor Base Address Register \(DISPLAY_CURSOR_BASE_ADDR\)](#) with the cursor image base address.
- Write the initial cursor x, y position into the [Display Cursor Position Register \(DISPLAY_CURSOR_POSITION\)](#).

Program the [Display Frame Buffer Length Register \(DISPLAY_FRAME_LEN\)](#) with the desired frame length.

Program the [Display Control Register \(DISPLAY_CTRL\)](#) with the desired values for the following fields. Make sure the [Video Sub-system Enable \(ENABLE\)](#) bit is set, in order to enable the Display Controller:

- [Frame Base Address Maximum Count \(FRAME_BASE_ADDR_MAX_COUNT\)](#)
- [Color Depth \(COLOR_DEPTH\)](#)
- [Gamma Correction and Color Lookup Enable \(COLOR_EN\)](#) (if the lookup table is to be utilized)

10.4.2 System Operation

While the Display Controller is enabled, write Display Frame Buffer Base Addresses into the [Display Frame Buffer Base Address Register \(DISPLAY_FRAME_BASE_ADDR\)](#). Make sure that entries are not written while the queue is full, otherwise they will be ignored.

Note: The [Frame Base Address Interrupt \(FRAME_BASE_ADDR_INT\)](#) may be used to signal when the Frame Base Address Queue has room for queuing the addresses of frames for display. Status of this interrupt appears in the [Display Status Register \(DISPLAY_STATUS\)](#).

If a change in Cursor Mode, Size, or Visibility is desired, update the [Display Cursor Control Register \(DISPLAY_CURSOR_CTRL\)](#) register as required. If a change in the cursor image is desired, update the [Display Cursor Base Address Register \(DISPLAY_CURSOR_BASE_ADDR\)](#) with the address of the buffer containing cursor image information. To invoke updates made to [Display Cursor Control Register \(DISPLAY_CURSOR_CTRL\)](#) or [Display Cursor Base Address Register \(DISPLAY_CURSOR_BASE_ADDR\)](#), or to update the Cursor's position, write the current or updated Cursor position to the [Display Cursor Position Register \(DISPLAY_CURSOR_POSITION\)](#).

During system operation, the following four interrupts are propagated from the Display Controller to the [USB Interrupt Endpoint Status Register \(USB_INT_STS\)](#):

- [Display Blank Interrupt \(DISPLAY_BLANK_INT\)](#)
- [Horizontal Blank Start Interrupt \(H_BLANK_START_INT\)](#)
- [Vertical Blank Start Interrupt \(V_BLANK_START_INT\)](#)
- [Frame Base Address Interrupt \(FRAME_BASE_ADDR_INT\)](#)

All of these interrupts may be cleared by writing a 1 to the analogous bits of the [Display Status Register \(DISPLAY_STATUS\)](#).

Note: The [Display Status Register \(DISPLAY_STATUS\)](#) may be queried at any time to determine the status of the Display Controller.

10.5 Control and Status Registers

Table 10.2 Display Controller Control and Status Register Map

| ADDRESS OFFSET | REGISTER NAME (SYMBOL) |
|------------------------------|--|
| Control and Status Registers | |
| 000h | Display Control Register (DISPLAY_CTRL) |
| 004h | Display Status Register (DISPLAY_STATUS) |
| Video Timing Registers | |
| 008h | Display Horizontal Size Register (DISPLAY_H_SIZE) |
| 00Ch | Display Horizontal Blank Register (DISPLAY_H_BLANK) |
| 010h | Display Horizontal Sync Register (DISPLAY_H_SYNC) |
| 014h | Display Vertical Size Register (DISPLAY_V_SIZE) |
| 018h | Display Vertical Blank Register (DISPLAY_V_BLANK) |
| 01Ch | Display Vertical Sync Register (DISPLAY_V_SYNC) |
| DMA Control Registers | |
| 020h | Display Frame Buffer Base Address Register (DISPLAY_FRAME_BASE_ADDR) |
| 028h | Display Frame Buffer Length Register (DISPLAY_FRAME_LEN) |
| 02Ch | Display Frame Base Address Changed Count Register (DISPLAY_FRAME_BASE_ADDR_CHANGED_CNT) |
| Interface Control Registers | |
| 040h | Display Interface Control 0 Register (DISPLAY_INTERFACE_CTRL_0) |
| 044h | Display Interface Control 1 Register (DISPLAY_INTERFACE_CTRL_1) |
| 048h | Display Interface Control 2 Register (DISPLAY_INTERFACE_CTRL_2) |
| Cursor Registers | |
| 080h | Display Cursor Control Register (DISPLAY_CURSOR_CTRL) |
| 084h | Display Cursor Status Register (DISPLAY_CURSOR_STATUS) |
| 088h | Display Cursor Base Address Register (DISPLAY_CURSOR_BASE_ADDR) |
| 08Ch | Display Cursor Current Base Address Register (DISPLAY_CURSOR_CUR_BASE_ADDR) |
| 090h | Display Cursor Position Register (DISPLAY_CURSOR_POSITION) |
| 094h | Display Cursor Current Position Register (DISPLAY_CURSOR_CUR_POSITION) |
| 098h | Display Cursor Color 0 Register (DISPLAY_CURSOR_COLOR_0) |
| 09Ch | Display Cursor Current Color 0 Register (DISPLAY_CURSOR_CUR_COLOR_0) |
| 0A0h | Display Cursor Color 1 Register (DISPLAY_CURSOR_COLOR_1) |
| 0A4h | Display Cursor Current Color 1 Register (DISPLAY_CURSOR_CUR_COLOR_1) |
| 0A8h | Display Cursor Color 2 Register (DISPLAY_CURSOR_COLOR_2) |
| 0ACh | Display Cursor Current Color 2 Register (DISPLAY_CURSOR_CUR_COLOR_2) |
| 0B0h | Display Cursor Color 3 Register (DISPLAY_CURSOR_COLOR_3) |
| 0B4h | Display Cursor Current Color 3 Register (DISPLAY_CURSOR_CUR_COLOR_3) |
| Color Lookup Registers | |
| 024h | Display Border Color Register (DISPLAY_BORDER_COLOR) |
| 400h-7FCh | Display Color Lookup Table (DISPLAY_CLUT) |
| Reserved | |
| 030h – 03Fh | Reserved for future expansion |
| 04Ch – 07Fh | Reserved for future expansion |
| 0B8h – 3FFh | Reserved for future expansion |
| 800h – FFFh | Reserved for future expansion |

10.5.1 Display Control Register (DISPLAY_CTRL)

Address: 000h Size: 32 bits

This register is used to configure and control the Display Controller.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|--|------|---------|
| 31 | Reset (RESET) When set, the display controller will be reset into its initial condition. This bit takes effect immediately upon being set. | SC | 0b |
| 30:27 | RESERVED | RO | - |
| 26:24 | Frame Base Address Maximum Count (FRAME_BASE_ADDR_MAX_COUNT) This field specifies the maximum number of entries that can be queued into Display Frame Buffer Base Address Register (DISPLAY_FRAME_BASE_ADDR) . Any additional entries will be ignored. The maximum allowable value is 4. Note: This value should only be changed when the display controller is stopped as indicated by the Display Controller Status (DC_STS) bit in the Display Status Register (DISPLAY_STATUS) or when there are no entries queued as indicated by Frame Base Address Pending Count (FRAME_BASE_ADDR_PENDING_CNT) in the Display Status Register (DISPLAY_STATUS) . | R/W | 001b |
| 23:9 | RESERVED | RO | - |
| 8 | Force Display Blanking (DISPLAY_BLANK) When set, the display will be forced into blanking. The RGB output will be forced to zero and the blanking output set active. This bit can be changed at any time but takes affect once the current frame is finished or if the display has been disabled via the ENABLE bit. The actual blanking status can be check via the BLANK_STS bit in the Display Status register. | R/W | 1b |
| 7:4 | RESERVED | RO | - |
| 3:2 | Color Depth (COLOR_DEPTH) 11: reserved 10: 24 bits per pixel 01: 16 bits per pixel 00: 8 bits per pixel Note: This value should only be changed when the display controller is stopped as indicated by the Display Controller Status (DC_STS) bit in the Display Status Register (DISPLAY_STATUS) . | R/W | 10b |

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|---|------|---------|
| 1 | <p>Gamma Correction and Color Lookup Enable (COLOR_EN) When cleared, pixel data is not mapped. 8 bit data is replicated on to each output color (grey scale). 16 bit data is interpreted as 5 bits of red, 6 bits of green and 5 bits of blue. 24 bit data is used directly. Since the RGB output is 10 bits, padded with trailing 0s is used to place the color data on the most significant bits.</p> <p>When set, pixel data is mapped using the color lookup table. 8 bit data is mapped into a 30 bit color. 16 bit data is interpreted as 5 bits of red, 6 bits of green and 5 bits of blue, padded with leading 0s, and each resulting byte is mapped separately to form the 30 bit color. Each byte of 24 bit data is mapped separately to form the 30 bit color.</p> | R/W | 0b |
| 0 | <p>Video Sub-system Enable (ENABLE) When set, the video sub-system will read from system memory and generate color and timing outputs.</p> <p>When cleared, the video sub-system is reset. When set to stop, the display controller will finish the current frame before actually stopping. The stopped status can be check via the DC_STS bit in the Display Status register.</p> <p>The disabled condition corresponds to the starting point of vertical blanking just before the sync pulses. In this condition:</p> <p>HCount is set to HBlankStart VCount is set to VBlankStart Blank output is active HSync, VSync and CSync outputs are inactive DAC and DVO output clocks are stopped</p> | R/W | 0b |

10.5.2 Display Status Register (DISPLAY_STATUS)

Address: 004h Size: 32 bits

This register indicates the current state of the Display Controller.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|---|------|---------|
| 31:27 | RESERVED | RO | - |
| 26:24 | Frame Base Address Pending Count (FRAME_BASE_ADDR_PENDING_CNT) This field indicates the number of pending frame base address changes. It is incremented when Display Frame Buffer Base Address Register (DISPLAY_FRAME_BASE_ADDR) is written and is decremented when the current frame base address in use changes. This field is cleared when the Display Controller is stopped as indicated by the Display Controller Status (DC_STS) bit in this register. | RO | 000b |
| 23:16 | Revision (REV) Indicates the current revision if we remember to update it. Otherwise indicates the revision the last time it was updated. | RO | 03h |
| 15:13 | RESERVED | RO | - |
| 12 | Display Blank Interrupt (DISPLAY_BLANK_INT) Set when the Blank Status (BLANK_STS) bit is updated to reflect the new setting of Force Display Blanking (DISPLAY_BLANK) and the Display Blank Pending (DISPLAY_BLANK_PENDING) bit was set. | R/WC | 0b |
| 11 | Display Blank Pending (DISPLAY_BLANK_PENDING) Set when the Force Display Blanking (DISPLAY_BLANK) bit in the Display Control Register (DISPLAY_CTRL) is changed. Cleared when the Blank Status (BLANK_STS) bit is updated to reflect the new setting of Force Display Blanking (DISPLAY_BLANK) . Also cleared when the Display Controller Status (DC_STS) bit is cleared. | RO | 0b |
| 10:9 | RESERVED | RO | - |
| 8 | Blank Status (BLANK_STS) Indicates that the request to blank the display, via the Force Display Blanking (DISPLAY_BLANK) bit, in the Display Control Register (DISPLAY_CTRL) is in affect. | RO | 1b |
| 7 | Horizontal Blank Start Interrupt (H_BLANK_START_INT) Set at the start of the horizontal blank time. | R/WC | 0b |
| 6 | Vertical Blank Start Interrupt (V_BLANK_START_INT) Set at the start of the vertical blank time. | R/WC | 0b |
| 5 | Frame Base Address Interrupt (FRAME_BASE_ADDR_INT) Set when Frame Base Address Queue Full (FRAME_BASE_ADDR_QUEUE_FULL) goes low due to the DMA Engine reading from the start of a new frame buffer. | R/WC | 0b |
| 4 | Frame Base Address Queue Full (FRAME_BASE_ADDR_QUEUE_FULL) This field indicates that the Frame Base Address Queue is full. It is set when Frame Base Address Pending Count (FRAME_BASE_ADDR_PENDING_CNT) reaches Frame Base Address Maximum Count (FRAME_BASE_ADDR_MAX_COUNT) and is cleared when the DMA Engine starts reading from a new frame buffer. This field is cleared when the Display Controller is stopped as indicated by the Display Controller Status (DC_STS) bit in this register. | RO | 0b |

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|--|------|---------|
| 3 | Bus Error (BUS_ERROR) When set indicates that the bus master experienced a transfer error and has stopped. | R/WC | 0b |
| 2 | Pixel FIFO Overflow (PIXEL_FIFO_OVERFLOW) Indicates that the Pixel FIFO was written while full. Should never happen since Pixel extractor is not supposed to cause over-runs. Basically if this is set, it's not good. | R/WC | 0b |
| 1 | Pixel FIFO Underrun (PIXEL_FIFO_UNDERRUN) Indicates that the Pixel FIFO ran out of data. Should never happen in a properly configured system. Basically if this is set, it's not good. | R/WC | 0b |
| 0 | Display Controller Status (DC_STS) Returns the actual run/stop status of the Display Controller. 1=Running, 0=Stopped. | RO | 0b |

10.5.3 Display Horizontal Size Register (DISPLAY_H_SIZE)

Address: 008h Size: 32 bits

This register controls the horizontal timing.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|---|------|---------|
| 31:28 | RESERVED | RO | - |
| 27:16 | Horizontal Total (H_TOTAL) Total horizontal time in pixels encompassing the active display, left and right borders, front and back porches and sync pulse time. Programmed to the number of clocks desired minus 1. | R/W | 000h |
| 15:12 | RESERVED | RO | - |
| 11:0 | Horizontal Active (H_ACTIVE) Horizontal active (addressable) resolution in pixels. The first horizontal active display pixel is considered pixel number 0. Programmed to the desired number of pixels per line minus 1. Should be programmed to less than the total horizontal time. Note: The horizontal active pixel count must be a) a multiple of 8 in 8 bpp mode, b) a multiple of 4 in 16 bpp mode and c) a multiple of 8 in 24 bpp mode. | R/W | 000h |

Note: This register should only be changed when the display controller is stopped as indicated by the [Display Controller Status \(DC_STS\)](#) bit in the [Display Status Register \(DISPLAY_STATUS\)](#).

10.5.4 Display Horizontal Blank Register (DISPLAY_H_BLANK)

Address: 00Ch Size: 32 bits

This register controls the horizontal blank timing.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|---|------|---------|
| 31:28 | RESERVED | RO | - |
| 27:16 | Horizontal Blank Start (H_BLANK_START) Horizontal blanking start time in pixels relative to the start of the active display. The first active pixel is considered pixel number 0. Programmed to the desired right border size plus the horizontal active resolution minus 1. A setting at the same point as the horizontal active implies no right border. Should be programmed to greater than or equal to the horizontal active time and less than the total horizontal time. | R/W | 000h |
| 15:12 | RESERVED | RO | - |
| 11:0 | Horizontal Blank End (H_BLANK_END) Horizontal blanking end time in pixels relative to the start of the active display. The first active pixel is considered pixel number 0. Programmed to the horizontal total time minus the desired left border size minus 1. A setting at the same point as the horizontal total time implies no left border. Should be programmed to greater than the horizontal blanking start point and less than or equal to the total horizontal time. | R/W | 000h |

Note: This register should only be changed when the display controller is stopped as indicated by the [Display Controller Status \(DC_STS\)](#) bit in the [Display Status Register \(DISPLAY_STATUS\)](#).

10.5.5 Display Horizontal Sync Register (DISPLAY_H_SYNC)

Address: 010h Size: 32 bits

This register controls the horizontal sync timing.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|---|------|---------|
| 31:28 | RESERVED | RO | - |
| 27:16 | Horizontal Sync Start (H_SYNC_START) Horizontal sync start time in pixels relative to the start of the active display. The first active pixel is considered pixel number 0. Programmed to the desired VESA value minus 1. | R/W | 000h |
| 15:12 | RESERVED | RO | - |
| 11:0 | Horizontal Sync End (H_SYNC_END) Horizontal sync end time in pixels relative to the start of the active display. The first active pixel is considered pixel number 0. Programmed to the desired VESA value minus 1. | R/W | 000h |

Note: This register should only be changed when the display controller is stopped as indicated by the [Display Controller Status \(DC_STS\)](#) bit in the [Display Status Register \(DISPLAY_STATUS\)](#).

10.5.6 Display Vertical Size Register (DISPLAY_V_SIZE)

Address: 014h Size: 32 bits

This register controls the vertical timing.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|---|------|---------|
| 31:28 | RESERVED | RO | - |
| 27:16 | Vertical Total (V_TOTAL) Total vertical time in lines encompassing the active display, top and bottom borders, front and back porches and sync pulse time. Programmed to the number of lines desired minus 1. | R/W | 000h |
| 15:12 | RESERVED | RO | - |
| 11:0 | Vertical Active (V_ACTIVE) Vertical active resolution in lines. The first vertical active display line is considered line number 0. Programmed to the desired number of lines minus 1. | R/W | 000h |

Note: This register should only be changed when the display controller is stopped as indicated by the [Display Controller Status \(DC_STS\)](#) bit in the [Display Status Register \(DISPLAY_STATUS\)](#).

10.5.7 Display Vertical Blank Register (DISPLAY_V_BLANK)

Address: 018h Size: 32 bits

This register controls the vertical blank timing.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|--|------|---------|
| 31:28 | RESERVED | RO | - |
| 27:16 | Vertical Blank Start (V_BLANK_START) Vertical blanking start time in lines relative to the start of the active display. The first active line is considered line number 0. Programmed to the desired bottom border size plus the vertical active resolution minus 1. A setting at the same point as the vertical active implies no bottom border. Should be programmed to greater than or equal to the vertical active time and less than the total vertical time. | R/W | 000h |
| 15:12 | RESERVED | RO | - |
| 11:0 | Vertical Blank End (V_BLANK_END) Vertical blanking end time in lines relative to the start of the active display. The first active line is considered line number 0. Programmed to the vertical total time minus the desired top border size minus 1. A setting at the same point as the vertical total time implies no top border. Should be programmed to greater than the vertical blanking start point and less than or equal to the total vertical time. | R/W | 000h |

Note: This register should only be changed when the display controller is stopped as indicated by the [Display Controller Status \(DC_STS\)](#) bit in the [Display Status Register \(DISPLAY_STATUS\)](#).

10.5.8 Display Vertical Sync Register (DISPLAY_V_SYNC)

Address: 01Ch Size: 32 bits

This register controls the vertical sync timing.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|--|------|---------|
| 31:28 | RESERVED | RO | - |
| 27:16 | Vertical Sync Start (V_SYNC_START) Vertical sync start time in lines relative to the start of the active display. The first active line is considered line number 0. Programmed to the desired VESA value minus 1. | R/W | 000h |
| 15:12 | RESERVED | RO | - |
| 11:0 | Vertical Sync End (V_SYNC_END) Vertical sync end time in lines relative to the start of the active display. The first active line is considered line number 0. Programmed to the desired VESA value minus 1. | R/W | 000h |

Note: This register should only be changed when the display controller is stopped as indicated by the [Display Controller Status \(DC_STS\)](#) bit in the [Display Status Register \(DISPLAY_STATUS\)](#).

10.5.9 Display Frame Buffer Base Address Register (DISPLAY_FRAME_BASE_ADDR)

Address: 020h Size: 32 bits

This register specifies the starting address in system memory of the frame buffer.

Writes into this register go into a queue. The maximum depth of the queue is specified by [Frame Base Address Maximum Count \(FRAME_BASE_ADDR_MAX_COUNT\)](#) in [Display Control Register \(DISPLAY_CTRL\)](#). Once the queue is full, further writes are ignored. An entry is used once the DMA Engine finishes the current display buffer. If the queue empties, the last entry is reused for subsequent display refreshes.

When the Display Controller is stopped, as indicated by the Display Controller Status (DC_STS) bit in the [Display Status Register \(DISPLAY_STATUS\)](#), the queue is held reset and this register may be written multiple times with the last write being the only retained value.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|--|------|----------|
| 31:7 | Frame Base Address Upper Bits (FRAME_BASE_ADDR_HI) When combined with the 7 read only bits below, this field specifies the starting address, in system memory, of the display frame buffer, specified in bytes. The byte offset of pixel XY within the frame buffer is given by $Adr = (Y * H_ACTIVE + X) * Bytes_per_pixel$. Where Bytes_per_pixel is 1, 2 or 3 for 8 bit, 16 bit and 24 bit modes respectively. | R/W | 0000000h |
| 6:0 | Frame Base Address Lower Bits (FRAME_BASE_ADDR_LO) These are the lower 7 bits of the frame buffer base address. | RO | 00h |

Note: The base address is aligned on 128 byte boundaries.

10.5.10 Display Border Color Register (DISPLAY_BORDER_COLOR)

Address: 024h Size: 32 bits

This register specifies the color used for the frame border.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|---|------|---------|
| 31:24 | RESERVED | RO | - |
| 23:16 | Border Red (RED) The red portion of color level sent during the border region, the periods between the end of blank and the start of active and the end of active and the start of blank. | R/W | 00h |
| 15:8 | Border Green (GREEN) The green portion of color level sent during the border region, the periods between the end of blank and the start of active and the end of active and the start of blank. | R/W | 00h |
| 7:0 | Border Blue (BLUE) The blue portion of color level sent during the border region, the periods between the end of blank and the start of active and the end of active and the start of blank. | R/W | 00h |

Note: The border color is handled the same as any pixel data. In 16 bit mode, only the lowest 5 bits of Red, 6 bits of Green and 5 bits of blue are used. In 8 bit mode, only the blue field is used as either the index into the CLUT or as an 8 bit grey scale value. If gamma correction / color lookup is enabled, the values are mapped through the CLUT.

10.5.11 Display Frame Buffer Length Register (DISPLAY_FRAME_LEN)

Address: 028h Size: 32 bits

This register specifies the length of the frame buffer.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|--|------|---------|
| 31:24 | RESERVED | RO | - |
| 23:3 | Frame Length Upper Bits (FRAME_LEN_HI) When combined with the 3 read only bits below, this field specifies the length of the display frame buffer, specified in bytes. This value can be changed at any time but does not take affect until the current frame is fully read from system memory. The value set should be the total pixel count ($H_ACTIVE+1 * V_ACTIVE+1$) multiplied by the bytes per pixel (bytes per pixel is 1, 2 or 3 for 8 bit, 16 bit and 24 bit modes respectively). | R/W | 000000h |
| 2:0 | Frame Length Lower Bits (FRAME_LEN_LO) These are the lower 3 bits of the frame buffer length. | RO | 000b |

Note: The frame length is always a multiple of 8 bytes in 8 bpp and 16 bpp modes and 24 bytes in 24 bpp mode due to the requirements on [Horizontal Active \(H_ACTIVE\)](#) listed above.

10.5.12 Display Frame Base Address Changed Count Register (DISPLAY_FRAME_BASE_ADDR_CHANGED_CNT)

Address: 02Ch Size: 32 bits

This register indicates the number of times the active Frame Base Address has been changed.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|---|------|-----------|
| 31:0 | Frame Base Address Changed Count (FRAME_BASE_ADDR_CHANGED_CNT) Indicates the number of times the active Frame Base Address has been changed. This field is cleared when the Display Controller is stopped as indicated by the Display Controller Status (DC_STS) bit in the Display Status Register (DISPLAY_STATUS) . | R/W | 00000000h |

10.5.13 Display Interface Control 0 Register (DISPLAY_INTERFACE_CTRL_0)

Address: 040h Size: 32 bits

This register is used to configure and control interface 0 of the Display Controller.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|--|------|---------|
| 31:10 | RESERVED | RO | - |
| 9 | RESERVED | RO | - |
| 8 | RESERVED | RO | - |
| 7:6 | Interface 0 Vertical and Horizontal Sync Output Delay (V_H_SYNC_DELAY_0) This field is used to specify a 0 to 3 clock delay on interface 0, vertical and horizontal sync outputs. This may be used to match the data delay of the interface's transmitter or DAC. 00 = 0 clocks 01 = 1 clock 10 = 2 clocks 11 = 3 clocks | R/W | 00b |
| 5 | RESERVED | RO | 0 |
| 4 | Interface 0 Horizontal Sync Polarity (HSYNC_POL_0) 1 = active high, 0 = active low | R/W | 0b |
| 3 | Interface 0 Vertical Sync Polarity (VSYNC_POL_0) 1 = active high, 0 = active low | R/W | 0b |
| 2 | RESERVED | RO | - |
| 1 | RESERVED | RO | - |
| 0 | Interface 0 Select (INTF_SEL_0) When set, interface 0 is selected. When not selected, the blank output is active, the sync signals, output data and output clock are inactive. Note: This enable is not synchronized to any internal frame timing and changes are immediate. | R/W | 0b |

Note: The values in this register should only be changed when the display controller is stopped as indicated by the [Display Controller Status \(DC_STS\)](#) bit in the [Display Status Register \(DISPLAY_STATUS\)](#).

10.5.14 Display Interface Control 1 Register (DISPLAY_INTERFACE_CTRL_1)

Address: 044h Size: 32 bits

This register is used to configure and control interface 1 of the Display Controller.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|--|------|---------|
| 31:10 | RESERVED | RO | - |
| 9 | RESERVED | RO | - |
| 8 | RESERVED | RO | - |
| 7:6 | Interface 1 Vertical and Horizontal Sync Output Delay (V_H_SYNC_DELAY_1) This field is used to specify a 0 to 3 clock delay on interface 1, vertical and horizontal sync outputs. This may be used to match the data delay of the interface's transmitter or DAC. 00 = 0 clocks 01 = 1 clock 10 = 2 clocks 11 = 3 clocks | R/W | 00b |
| 5 | RESERVED | RO | - |
| 4 | Interface 1 Horizontal Sync Polarity (HSYNC_POL_1) 1 = active high, 0 = active low | R/W | 0b |
| 3 | Interface 1 Vertical Sync Polarity (VSYNC_POL_1) 1 = active high, 0 = active low | R/W | 0b |
| 2 | RESERVED | RO | - |
| 1 | RESERVED | RO | 0b |
| 0 | Interface 1 Select (INTF_SEL_1) When set, interface 1 is selected. When not selected, the blank output is active, the sync signals, output data and output clock are inactive. Note: This enable is not synchronized to any internal frame timing and changes are immediate. | R/W | 0b |

Note: The values in this register should only be changed when the display controller is stopped as indicated by the [Display Controller Status \(DC_STS\)](#) bit in the [Display Status Register \(DISPLAY_STATUS\)](#).

10.5.15 Display Interface Control 2 Register (DISPLAY_INTERFACE_CTRL_2)

Address: 048h Size: 32 bits

This register is used to configure and control interface 2 of the Display Controller.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|--|------|---------|
| 31:10 | RESERVED | RO | - |
| 9 | Interface 2 Double Data Rate Select (DDR_SEL_2) When set, interface 2 supplies data at twice the clock rate. When cleared, single data rate is selected. | R/W | 0 |
| 8 | Interface 2 Double Data Rate Deep Color Enable (DDR_COLOR_2) When set, interface 2 generates 15 bit DDR data. When cleared, 12 bit DDR data is generated. | R/W | 0 |
| 7:6 | Interface 2 Vertical and Horizontal Sync Output Delay (V_H_SYNC_DELAY_2) This field is used to specify a 0 to 3 clock delay on interface 2, vertical and horizontal sync outputs. This may be used to match the data delay of the interface's transmitter or DAC. 00 = 0 clocks 01 = 1 clock 10 = 2 clocks 11 = 3 clocks | R/W | 00b |
| 5 | Interface 2 Output Clock Polarity (OUT_CLOCK_POL_2) This bit inverts the clock for use with single rate data. It is a don't care for double data rate mode. When cleared, the output pixel clock to interface 2 is edge aligned with the positive edge clocked data. When set, the output pixel clock is inverted so that the rising edge is centered in the data. | R/W | 1b |
| 4 | Interface 2 Horizontal Sync Polarity (HSYNC_POL_2) 1 = active high, 0 = active low | R/W | 0b |
| 3 | Interface 2 Vertical Sync Polarity (VSYNC_POL_2) 1 = active high, 0 = active low | R/W | 0b |
| 2 | RESERVED | RO | - |
| 1 | Interface 2 Blanking Polarity (BLANK_POL_2) 1 = active high, 0 = active low | R/W | 0b |
| 0 | Interface 2 Select (INTF_SEL_2) When set, interface 2 is selected. When not selected, the blank output is active, the sync signals, output data and output clock are inactive. Note: This enable is not synchronized to any internal frame timing and changes are immediate. | R/W | 0b |



Note: The values in this register should only be changed when the display controller is stopped as indicated by the [Display Controller Status \(DC_STS\)](#) bit in the [Display Status Register \(DISPLAY_STATUS\)](#).

10.5.16 Display Cursor Control Register (DISPLAY_CURSOR_CTRL)

Address: 080h Size: 32 bits

This register is used to configure and control the Cursor.

This register is double buffered. Writes to it are performed into a holding register. The register value will be used following the vertical blank signal only after a write cycle to the [Display Cursor Position Register \(DISPLAY_CURSOR_POSITION\)](#) (setting [Cursor Trigger Armed \(CURSOR_TRIG\)](#)). Double buffering is not used if the display controller is stopped as indicated by the [Display Controller Status \(DC_STS\)](#) bit in the [Display Status Register \(DISPLAY_STATUS\)](#), in which case writes to this register have immediate affect.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|--|------|---------|
| 31:7 | RESERVED | RO | - |
| 6:4 | Cursor Mode (CURSOR_MODE) 000 = 2 Color (Monochrome) 001 = 3 Color 010 = 4 Color 011 = RGB Color 100 = Masked RGB Color others = reserved See Table 10.3 for the descriptions of each mode. The current cursor mode in affect can be checked via Display Cursor Status Register (DISPLAY_CURSOR_STATUS) . This value normally should not be changed when Cursor Trigger Armed (CURSOR_TRIG) in Display Cursor Status Register (DISPLAY_CURSOR_STATUS) is set. | R/W | 000b |
| 3:1 | Cursor Size (CURSOR_SIZE) 000 = 32 x 32 001 = 48 x 48 010 = 64 x 64 others = reserved The current cursor size in affect can be checked via Display Cursor Status Register (DISPLAY_CURSOR_STATUS) . This value normally should not be changed when Cursor Trigger Armed (CURSOR_TRIG) in Display Cursor Status Register (DISPLAY_CURSOR_STATUS) is set. | R/W | 000b |
| 0 | Cursor Enable (CURSOR_ENABLE) When set, the video sub-system will read from system memory and generate a cursor image as specified. When cleared, the video sub-system will not read the cursor image from system memory nor display the cursor. When changed, the display controller will finish the current frame before enabling or disabling the cursor. The current cursor state in affect can be checked via Display Cursor Status Register (DISPLAY_CURSOR_STATUS) . This value normally should not be changed when Cursor Trigger Armed (CURSOR_TRIG) in Display Cursor Status Register (DISPLAY_CURSOR_STATUS) is set. | R/W | 0b |

Table 10.3 Cursor Modes

| CURSOR MODE | DESCRIPTION |
|----------------------|---|
| 2 Color (Monochrome) | 2 bits per pixel 00 = Color 0 01 = Color 1 10 = Transparent (use frame data) 11 = Transparent Inverted (use inverted frame data) |
| 3 Color | 2 bits per pixel 00 = Color 0 01 = Color 1 10 = Color 2 11 = Transparent (use frame data) |
| 4 Color | 2 bits per pixel 00 = Color 0 01 = Color 1 10 = Color 2 11 = Color 3 |
| RGB Color | 32 bits per pixel in ARGB format. The Alpha value is used to blend the cursor data and the frame data using the formula $\text{Pixel} = \text{Alpha} * \text{cursor} + (255 - \text{Alpha}) * \text{frame data}$ When in 8 bpp display color mode, only the Blue byte is used. When in 16 bpp display color mode, only the lower 5, 6, and 5 bits of Red, Blue and Green respectively are used. |
| Masked RGB Color | 32 bits per pixel in ARGB format. The Alpha value specifies: 0h = Opaque 1h - FEh = Transparent (use frame data) FFh = XOR of RGB and frame data When in 8 bpp display color mode, only the Blue byte is used. When in 16 bpp display color mode, only the lower 5, 6, and 5 bits of Red, Blue and Green respectively are used. |

Note: All cursor colors are specified before color look / gamma correction.

10.5.17 Display Cursor Status Register (DISPLAY_CURSOR_STATUS)

Address: 084h Size: 32 bits

This register indicates the cursor status.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|--|------|---------|
| 31 | Cursor Trigger Armed (CURSOR_TRIG) This bit returns the status of a pending cursor change. 1=Pending, 0=Not Pending. A pending cursor change can be canceled by writing a one to this bit. | WC | 0b |
| 30:7 | RESERVED | RO | - |
| 6:4 | Cursor Current Mode (CURSOR_CUR_MODE) This field indicates the current cursor mode. | RO | 00b |
| 3:1 | Cursor Current Size (CURSOR_CUR_SIZE) This field indicates the current cursor size. | RO | 000b |
| 0 | Cursor Status (CURSOR_STAT) This field indicates the actual enabled/disabled status of the Cursor. 1=Enabled, 0=Disabled. | RO | 0b |

10.5.18 Display Cursor Base Address Register (DISPLAY_CURSOR_BASE_ADDR)

Address: 088h Size: 32 bits

This register specifies the starting address in system memory of the cursor image.

This register is double buffered. Writes to it are performed into a holding register. The register value will be used following the vertical blank signal only after a write cycle to the [Display Cursor Position Register \(DISPLAY_CURSOR_POSITION\)](#) (setting [Cursor Trigger Armed \(CURSOR_TRIG\)](#)). Double buffering is not used if the display controller is stopped as indicated by the [Display Controller Status \(DC_STS\)](#) bit in the [Display Status Register \(DISPLAY_STATUS\)](#), in which case writes to this register have immediate affect.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|--|------|----------|
| 31:7 | Cursor Base Address Upper Bits (CURSOR_BASE_ADDR_HI) When combined with the 7 read only bits below, this field specifies the starting address, in system memory, of the cursor image, specified in bytes. The current cursor base address in affect can be checked via Display Cursor Current Base Address Register (DISPLAY_CURSOR_CUR_BASE_ADDR) . This value normally should not be changed when Cursor Trigger Armed (CURSOR_TRIG) in Display Cursor Status Register (DISPLAY_CURSOR_STATUS) is set. | R/W | 0000000h |
| 6:0 | Cursor Base Address Lower Bits (CURSOR_BASE_ADDR_LO) These are the lower 7 bits of the cursor image base address. | RO | 00h |

Note: The base address is aligned on 128 byte boundaries.

10.5.19 Display Cursor Current Base Address Register (DISPLAY_CURSOR_CUR_BASE_ADDR)

Address: 08Ch Size: 32 bits

This register indicates the current cursor base address that is in affect.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|---|------|----------|
| 31:7 | Cursor Current Base Address Upper Bits (CURSOR_CUR_BASE_ADDR_HI) This field indicates the current cursor base address. | RO | 0000000h |
| 6:0 | Cursor Current Base Address Lower Bits (CURSOR_CUR_BASE_ADDR_LO) These are the lower 7 bits of the cursor image base address. | RO | 00h |

Note: The base address is aligned on 128 byte boundaries.

10.5.20 Display Cursor Position Register (DISPLAY_CURSOR_POSITION)

Address: 090h Size: 32 bits

This register specifies the cursor position on the screen. This corresponds to the top left corner of the cursor image. The top-left of the screen is position 0,0. Cursor pixels outside the active screen are clipped. However, when using negative values, at least one pixel must be positioned on the active screen.

This register is double buffered. Writes to it are performed into a holding register. The register value will be used following the vertical blank signal. Writing this register sets [Cursor Trigger Armed \(CURSOR_TRIG\)](#). Double buffering is not used if the display controller is stopped as indicated by the [Display Controller Status \(DC_STS\)](#) bit in the [Display Status Register \(DISPLAY_STATUS\)](#), in which case writes to this register have immediate affect.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|--|------|---------|
| 31:16 | Cursor Y Position (CURSOR_Y_POSITION) This field specifies the Y coordinate of the cursor as a signed value. The current cursor position in affect can be checked via Display Cursor Current Position Register (DISPLAY_CURSOR_CUR_POSITION) . This value can be changed when Cursor Trigger Armed (CURSOR_TRIG) in Display Cursor Status Register (DISPLAY_CURSOR_STATUS) is set. The new value will be used if the write happens before the vertical blank signal. | R/W | 0000h |
| 15:0 | Cursor X Position (CURSOR_X_POSITION) This field specifies the X coordinate of the cursor as a signed value. The current cursor position in affect can be checked via Display Cursor Current Position Register (DISPLAY_CURSOR_CUR_POSITION) . This value can be changed when Cursor Trigger Armed (CURSOR_TRIG) in Display Cursor Status Register (DISPLAY_CURSOR_STATUS) is set. The new value will be used if the write happens before the vertical blank signal. | R/W | 0000h |

10.5.21 Display Cursor Current Position Register (DISPLAY_CURSOR_CUR_POSITION)

Address: 094h Size: 32 bits

This register indicates the current cursor position that is in affect.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|---|------|---------|
| 31:16 | Cursor Current Y Position (CURSOR_CUR_Y_POSITION) This field indicates the current cursor position. | RO | 0000h |
| 15:0 | Cursor Current X Position (CURSOR_CUR_X_POSITION) This field indicates the current cursor position. | RO | 0000h |

10.5.22 Display Cursor Color 0 Register (DISPLAY_CURSOR_COLOR_0)

Address: 098h Size: 32 bits

This register specifies cursor color 0 when using a 2 bpp cursor mode.

This register is double buffered. Writes to it are performed into a holding register. The register value will be used following the vertical blank signal only after a write cycle to the [Display Cursor Position Register \(DISPLAY_CURSOR_POSITION\)](#) (setting [Cursor Trigger Armed \(CURSOR_TRIG\)](#)). Double buffering is not used if the display controller is stopped as indicated by the [Display Controller Status \(DC_STS\)](#) bit in the [Display Status Register \(DISPLAY_STATUS\)](#), in which case writes to this register have immediate affect.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|---|------|---------|
| 31:24 | RESERVED | RO | - |
| 23:16 | Cursor Red (CURSOR_RED_0) The red portion of cursor color 0. The current cursor color in affect can be checked via Display Cursor Current Color 0 Register (DISPLAY_CURSOR_CUR_COLOR_0) . This value normally should not be changed when Cursor Trigger Armed (CURSOR_TRIG) in Display Cursor Status Register (DISPLAY_CURSOR_STATUS) is set. | R/W | 00h |
| 15:8 | Cursor Green (CURSOR_GREEN_0) The green portion of cursor color 0. The current cursor color in affect can be checked via Display Cursor Current Color 0 Register (DISPLAY_CURSOR_CUR_COLOR_0) . This value normally should not be changed when Cursor Trigger Armed (CURSOR_TRIG) in Display Cursor Status Register (DISPLAY_CURSOR_STATUS) is set. | R/W | 00h |
| 7:0 | Cursor Blue (CURSOR_BLUE_0) The blue portion of cursor color 0. The current cursor color in affect can be checked via Display Cursor Current Color 0 Register (DISPLAY_CURSOR_CUR_COLOR_0) . This value normally should not be changed when Cursor Trigger Armed (CURSOR_TRIG) in Display Cursor Status Register (DISPLAY_CURSOR_STATUS) is set. | R/W | 00h |

Note: The cursor color is handled the same as any pixel data. In 16 bit mode, only the lowest 5 bits of Red, 6 bits of Green and 5 bits of blue are used. In 8 bit mode, only the blue field is used as either the index into the CLUT or as an 8 bit grey scale value. If gamma correction / color lookup is enabled, the values are mapped through the CLUT.

10.5.23 Display Cursor Current Color 0 Register (DISPLAY_CURSOR_CUR_COLOR_0)

Address: 09Ch Size: 32 bits

This register indicates the current cursor color 0 that is in affect.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|--|------|---------|
| 31:24 | RESERVED | RO | - |
| 23:16 | Cursor Current Red (CURSOR_CUR_RED_0) This field indicates the current cursor color 0. | RO | 00h |
| 15:8 | Cursor Current Green (CURSOR_CUR_GREEN_0) This field indicates the current cursor color 0. | RO | 00h |
| 7:0 | Cursor Current Blue (CURSOR_CUR_BLUE_0) This field indicates the current cursor color 0. | RO | 00h |

10.5.24 Display Cursor Color 1 Register (DISPLAY_CURSOR_COLOR_1)

Address: 0A0h Size: 32 bits

This register specifies cursor color 1 when using a 2 bpp cursor mode.

This register is double buffered. Writes to it are performed into a holding register. The register value will be used following the vertical blank signal only after a write cycle to the [Display Cursor Position Register \(DISPLAY_CURSOR_POSITION\)](#) (setting [Cursor Trigger Armed \(CURSOR_TRIG\)](#)). Double buffering is not used if the display controller is stopped as indicated by the [Display Controller Status \(DC_STS\)](#) bit in the [Display Status Register \(DISPLAY_STATUS\)](#), in which case writes to this register have immediate affect.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|---|------|---------|
| 31:24 | RESERVED | RO | - |
| 23:16 | Cursor Red (CURSOR_RED_1) The red portion of cursor color 1. The current cursor color in affect can be checked via Display Cursor Current Color 1 Register (DISPLAY_CURSOR_CUR_COLOR_1) . This value normally should not be changed when Cursor Trigger Armed (CURSOR_TRIG) in Display Cursor Status Register (DISPLAY_CURSOR_STATUS) is set. | R/W | 00h |
| 15:8 | Cursor Green (CURSOR_GREEN_1) The green portion of cursor color 1. The current cursor color in affect can be checked via Display Cursor Current Color 1 Register (DISPLAY_CURSOR_CUR_COLOR_1) . This value normally should not be changed when Cursor Trigger Armed (CURSOR_TRIG) in Display Cursor Status Register (DISPLAY_CURSOR_STATUS) is set. | R/W | 00h |
| 7:0 | Cursor Blue (CURSOR_BLUE_1) The blue portion of cursor color 1. The current cursor color in affect can be checked via Display Cursor Current Color 1 Register (DISPLAY_CURSOR_CUR_COLOR_1) . This value normally should not be changed when Cursor Trigger Armed (CURSOR_TRIG) in Display Cursor Status Register (DISPLAY_CURSOR_STATUS) is set. | R/W | 00h |

Note: The cursor color is handled the same as any pixel data. In 16 bit mode, only the lowest 5 bits of Red, 6 bits of Green and 5 bits of blue are used. In 8 bit mode, only the blue field is used as either the index into the CLUT or as an 8 bit grey scale value. If gamma correction / color lookup is enabled, the values are mapped through the CLUT.

10.5.25 Display Cursor Current Color 1 Register (DISPLAY_CURSOR_CUR_COLOR_1)

Address: 0A4h Size: 32 bits

This register indicates the current cursor color 1 that is in affect.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|--|------|---------|
| 31:24 | RESERVED | RO | - |
| 23:16 | Cursor Current Red (CURSOR_CUR_RED_1) This field indicates the current cursor color 1. | RO | 00h |
| 15:8 | Cursor Current Green (CURSOR_CUR_GREEN_1) This field indicates the current cursor color 1. | RO | 00h |
| 7:0 | Cursor Current Blue (CURSOR_CUR_BLUE_1) This field indicates the current cursor color 1. | RO | 00h |

10.5.26 Display Cursor Color 2 Register (DISPLAY_CURSOR_COLOR_2)

Address: 0A8h Size: 32 bits

This register specifies cursor color 2 when using a 2 bpp cursor mode.

This register is double buffered. Writes to it are performed into a holding register. The register value will be used following the vertical blank signal only after a write cycle to the [Display Cursor Position Register \(DISPLAY_CURSOR_POSITION\)](#) (setting [Cursor Trigger Armed \(CURSOR_TRIG\)](#)). Double buffering is not used if the display controller is stopped as indicated by the [Display Controller Status \(DC_STS\)](#) bit in the [Display Status Register \(DISPLAY_STATUS\)](#), in which case writes to this register have immediate affect.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|---|------|---------|
| 31:24 | RESERVED | RO | - |
| 23:16 | Cursor Red (CURSOR_RED_2) The red portion of cursor color 2. The current cursor color in affect can be checked via Display Cursor Current Color 2 Register (DISPLAY_CURSOR_CUR_COLOR_2) . This value normally should not be changed when Cursor Trigger Armed (CURSOR_TRIG) in Display Cursor Status Register (DISPLAY_CURSOR_STATUS) is set. | R/W | 00h |
| 15:8 | Cursor Green (CURSOR_GREEN_2) The green portion of cursor color 2. The current cursor color in affect can be checked via Display Cursor Current Color 2 Register (DISPLAY_CURSOR_CUR_COLOR_2) . This value normally should not be changed when Cursor Trigger Armed (CURSOR_TRIG) in Display Cursor Status Register (DISPLAY_CURSOR_STATUS) is set. | R/W | 00h |
| 7:0 | Cursor Blue (CURSOR_BLUE_2) The blue portion of cursor color 2. The current cursor color in affect can be checked via Display Cursor Current Color 2 Register (DISPLAY_CURSOR_CUR_COLOR_2) . This value normally should not be changed when Cursor Trigger Armed (CURSOR_TRIG) in Display Cursor Status Register (DISPLAY_CURSOR_STATUS) is set. | R/W | 00h |

Note: The cursor color is handled the same as any pixel data. In 16 bit mode, only the lowest 5 bits of Red, 6 bits of Green and 5 bits of blue are used. In 8 bit mode, only the blue field is used as either the index into the CLUT or as an 8 bit grey scale value. If gamma correction / color lookup is enabled, the values are mapped through the CLUT.

10.5.27 Display Cursor Current Color 2 Register (DISPLAY_CURSOR_CUR_COLOR_2)

Address: 0ACh Size: 32 bits

This register indicates the current cursor color 2 that is in affect.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|--|------|---------|
| 31:24 | RESERVED | RO | - |
| 23:16 | Cursor Current Red (CURSOR_CUR_RED_2) This field indicates the current cursor color 2. | RO | 00h |
| 15:8 | Cursor Current Green (CURSOR_CUR_GREEN_2) This field indicates the current cursor color 2. | RO | 00h |
| 7:0 | Cursor Current Blue (CURSOR_CUR_BLUE_2) This field indicates the current cursor color 2. | RO | 00h |

10.5.28 Display Cursor Color 3 Register (DISPLAY_CURSOR_COLOR_3)

Address: 0B0h Size: 32 bits

This register specifies cursor color 3 when using a 2 bpp cursor mode.

This register is double buffered. Writes to it are performed into a holding register. The register value will be used following the vertical blank signal only after a write cycle to the [Display Cursor Position Register \(DISPLAY_CURSOR_POSITION\)](#) (setting [Cursor Trigger Armed \(CURSOR_TRIG\)](#)). Double buffering is not used if the display controller is stopped as indicated by the [Display Controller Status \(DC_STS\)](#) bit in the [Display Status Register \(DISPLAY_STATUS\)](#), in which case writes to this register have immediate affect.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|---|------|---------|
| 31:24 | RESERVED | RO | - |
| 23:16 | Cursor Red (CURSOR_RED_3) The red portion of cursor color 3. The current cursor color in affect can be checked via Display Cursor Current Color 3 Register (DISPLAY_CURSOR_CUR_COLOR_3) . This value normally should not be changed when Cursor Trigger Armed (CURSOR_TRIG) in Display Cursor Status Register (DISPLAY_CURSOR_STATUS) is set. | R/W | 00h |
| 15:8 | Cursor Green (CURSOR_GREEN_3) The green portion of cursor color 3. The current cursor color in affect can be checked via Display Cursor Current Color 3 Register (DISPLAY_CURSOR_CUR_COLOR_3) . This value normally should not be changed when Cursor Trigger Armed (CURSOR_TRIG) in Display Cursor Status Register (DISPLAY_CURSOR_STATUS) is set. | R/W | 00h |
| 7:0 | Cursor Blue (CURSOR_BLUE_3) The blue portion of cursor color 3. The current cursor color in affect can be checked via Display Cursor Current Color 3 Register (DISPLAY_CURSOR_CUR_COLOR_3) . This value normally should not be changed when Cursor Trigger Armed (CURSOR_TRIG) in Display Cursor Status Register (DISPLAY_CURSOR_STATUS) is set. | R/W | 00h |

Note: The cursor color is handled the same as any pixel data. In 16 bit mode, only the lowest 5 bits of Red, 6 bits of Green and 5 bits of blue are used. In 8 bit mode, only the blue field is used as either the index into the CLUT or as an 8 bit grey scale value. If gamma correction / color lookup is enabled, the values are mapped through the CLUT.

10.5.29 Display Cursor Current Color 3 Register (DISPLAY_CURSOR_CUR_COLOR_3)

Address: 0B4h Size: 32 bits

This register indicates the current cursor color 3 that is in affect.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|--|------|---------|
| 31:24 | RESERVED | RO | - |
| 23:16 | Cursor Current Red (CURSOR_CUR_RED_3) This field indicates the current cursor color 3. | RO | 00h |
| 15:8 | Cursor Current Green (CURSOR_CUR_GREEN_3) This field indicates the current cursor color 3. | RO | 00h |
| 7:0 | Cursor Current Blue (CURSOR_CUR_BLUE_3) This field indicates the current cursor color 3. | RO | 00h |

10.5.30 Display Color Lookup Table (DISPLAY_CLUT)

Address: 400h-7FCh Size: 32 bits

This is a 256 entry table. Each entry contains 10 bit Red, Green and Blue values and is used to map incoming pixel data to a color. The table must be initialized by S/W if color lookup / gamma correction is enabled. In 8 bit color mode, an entry is selected as a 3 color set. In 16 and 24 bit modes, different entries are selected for each color. The pixel clock must be running in order to access this table.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|--|------|-----------|
| 31:30 | RESERVED | RO | - |
| 29:20 | CLUT Red (RED) The red color level associated with this entry. | R/W | undefined |
| 19:10 | CLUT Green (GREEN) The green color level associated with this entry. | R/W | undefined |
| 9:0 | CLUT Blue (BLUE) The blue color level associated with this entry. | R/W | undefined |

Chapter 11 Graphics Engine

11.1 Overview

The Graphics Engine module receives a stream of commands, along with parameters and data, over the USB bulk out endpoint and executes the commands writing the results into the Display Frame Buffer. The various commands range from simple memory write access to various decompression algorithms. The bulk out endpoint interface is by way of a FIFO. The Graphics Engine module contains Control and Status Registers (CSRs) which are used to configure parameters and to indicate various status, such as error status, current command sequence number, etc. For a summary description of the Graphics Engine and the commands it processes, refer to [Section 2.2, "Graphics Engine \(GPH\)," on page 18 in Chapter 2, Overview.](#)

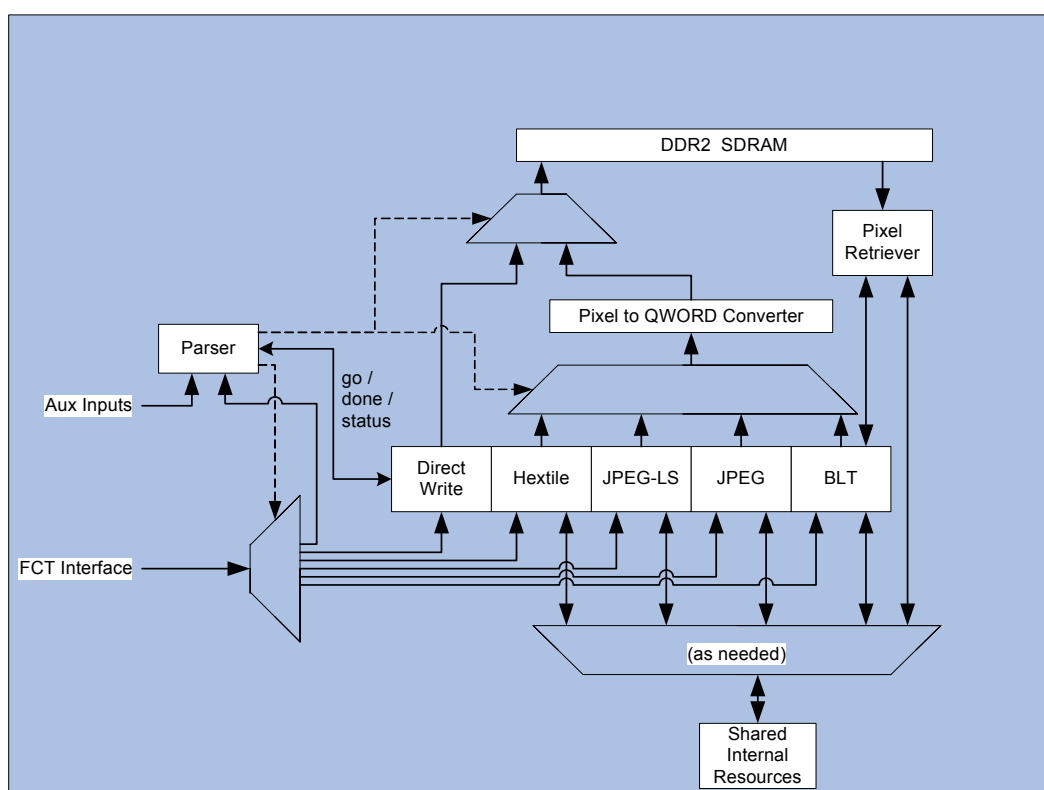


Figure 11.1 Graphics Engine Block Diagram

11.2 FIFO (FCT) Interface

The FIFO (FCT) interface is used to read data from the USB endpoint. The FCT interface is shared amongst the various co-processors under the control of the Command Parser.

Unless otherwise noted, all multiple byte data fields are considered to be little endian (LSB in the lower byte).

11.3 Command Parser

When enabled via the [Graphics Engine Enable \(GE_ENABLE\)](#) bit in [Graphic Engine Control Register \(GE_CTRL\)](#), the Command Parser reads the command and length from the FIFO, checks the command for errors and, based on the command, either handles the command or invokes one of the co-processors.

When a command and its length need to be passed to a co-processor, the Command Parser selects and invokes the co-processor and waits for a completion signal. The Pixel to QWORD Converter will also be selected via the associated multiplexer, if required, to process the command.

The command format is as follows:

- DWORD Command
- DWORD Length

The command format for the PAD command is as follows:

- DWORD Command

The Length field is the length of the following message, in bytes, not counting the Command or Length fields or any end-align padding. All commands are DWORD (4 bytes) aligned and are end padded if needed to maintain alignment of the next command. The Length field is excluded for the PAD command.

The Command field consists of 4 byte fields as follows:

- BYTE - the specific command
- BYTE - the 1's complement of the command
- WORD - sequence number

11.3.1 Error Checking and Handling

As each command arrives, the Command Parser checks for a correct ones complement, for an incrementing sequence number and (as described below) a correct length. The ones complement and incrementing sequence number can each be enabled / disabled via the [Graphics Engine Command Ones Complement Enable \(GE_CMD_ONECOMP_EN\)](#) and the [Graphics Engine Command Sequence Number Enable \(GE_CMD_SEQ_EN\)](#) bits in [Graphic Engine Control Register \(GE_CTRL\)](#). The current incrementing sequence number can be initialized via [Graphics Engine Initial Command Sequence Number \(GE_INIT_CMD_SEQ_NUM\)](#) and read via [Graphics Engine Next Command Sequence Number \(GE_NEXT_CMD_SEQ_NUM\)](#).

There are also command-specific checks that are handled by the various co-processors (e.g. that the drawing area fits within the current resolution).

If a command fails any of the checks or if any of the command co-processors (including the commands processed by the Command Parser) experience an error, the Bulk Out Endpoint is stalled, and the Graphics Engine is stopped. Any and all, co-processors are deselected. The error code is stored in the [Graphics Engine Error Status \(GE_ERROR_STAT\)](#) register. The Graphics Engine is restarted by the host using the [Graphics Engine Restart \(GE_RESTART\)](#) bit in [Graphic Engine Control Register \(GE_CTRL\)](#). The host may need to first flush the endpoint FIFO by setting the [FCT RX FIFO Reset \(FCT_RX_RESET\)](#) bit of the [FIFO Status Register \(FIFO_STATUS\)](#).

11.3.2 Aborting Command

A command (including the commands processed by the Command Parser) maybe aborted by setting the [Graphics Engine Abort \(GE_ABORT\)](#) bit in [Graphic Engine Control Register \(GE_CTRL\)](#). The Bulk Out Endpoint is stalled, and the Graphics Engine is stopped. Any and all, co-processors are deselected. The Graphics Engine is restarted by the host using the [Graphics Engine Restart \(GE_RESTART\)](#) bit in [Graphic Engine Control Register \(GE_CTRL\)](#). The host may need to first flush the endpoint FIFO by setting the [FCT RX FIFO Reset \(FCT_RX_RESET\)](#) bit of the [FIFO Status Register \(FIFO_STATUS\)](#).

11.3.3 Graphics Engine Commands

The following command values are defined (the command specifics are defined with each co-processor):

| | | |
|---|-----------------------|------|
| ■ | DRAW_RAW_RECTANGLE | 0x01 |
| ■ | DRAW_FILLED_RECTANGLE | 0x02 |
| ■ | ROP_RECTANGLE | 0x03 |
| ■ | ROP_RAW_RECTANGLE | 0x04 |
| ■ | DIRECT_RAM_WRITE | 0x08 |
| ■ | WRITE_CSR | 0x10 |
| ■ | NOP | 0x11 |
| ■ | WAIT_FOR_EVENT | 0x12 |
| ■ | PAD | 0x13 |
| ■ | DECOMP_HEXTILE | 0x20 |
| ■ | DECOMP_JPEG | 0x30 |
| ■ | BYPASS_JPEG | 0x31 |
| ■ | DECOMP_JPEG_LS | 0x40 |

11.3.4 Command Parser Implemented Commands

The following command are handled directly by the Command Parser:

NOP

| | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---------|----|----|----|----|----|----|----|----|--------------------------|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| 0 | command | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | length | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 2 | padding | | | | | | | | | number of clocks to wait | | | | | | | | | | | | | | | | | | | | | | |

- Length = 3
- 3 BYTES - the number of clocks to wait (0 = reserved).
- BYTE - padding

As implied, this command does nothing other than to wait a specified amount of time. The amount of time is the number of clocks specified plus whatever minimum time the command takes to execute (i.e. if a setting of 1 takes n+1 clocks, a setting of 2 should take n+2 clocks, etc.)

Before waiting, the length field is checked for a correct value.

WAIT_FOR_EVENT

| | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---------|----|----|----|----|----|----|----|----------|----|----|----|----|----|----|----|-----------------------------|----|----|----|----|----|---|---|----------------|---|---|---|---|---|---|---|
| 0 | command | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | length | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 2 | OR mask | | | | | | | | AND mask | | | | | | | | edge / level select bit map | | | | | | | | invert bit map | | | | | | | |

- Length = 4
- BYTE - invert bit map (1 = invert)
- BYTE - edge / level select bit map (1 = edge, 0 = level)
- BYTE - AND signal mask (1 = enabled into AND function, 0 = signal has no affect on AND)
- BYTE - OR signal mask (1 = enabled into OR function, 0 = signal has no affect on OR)

This command waits for the event specified.

Before waiting, the length field is checked for a correct value.

Each of 8 inputs are optionally inverted, optionally edge detected and then optionally ANDed then optionally ORed.

The AND/OR logic allows any signals to be ANDed and then ORed. Only one AND term and one OR term are supported. For example sig_0 AND sig_1 OR sig_2 is supported but sig_0 AND sig_1 OR sig_2 AND sig_3 is not. The OR mask works by blocking the signal going to the OR term replacing it with a low. The AND mask works by blocking the signal going to the AND term replacing it with a high. However, If no signals are selected to be ANDed the output of the AND term is forced to zero. Also, although supported, it doesn't make logic sense to specify a signal to be both ANDed and ORed. For example sig_0 AND sig_1 OR sig_0 is redundant.

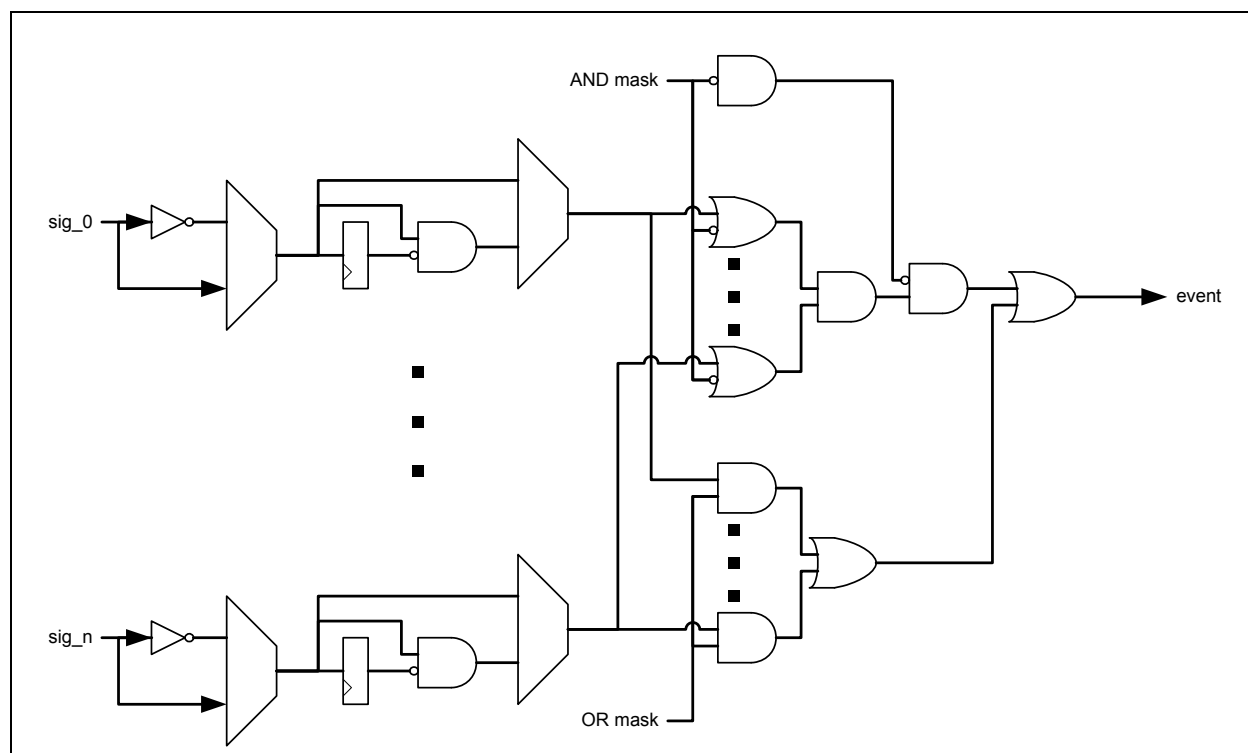


Figure 11.2 Command Parser Wait for Event Logic

The signal inputs are:

- 0 - Display Controller->Display Status Register (DISPLAY_STATUS)->Frame Base Address Pending Count (FRAME_BASE_ADDR_PENDING_CNT)> 0
- 1 - Display Controller->Vertical Blank as specified by the Display Vertical Blank Register (DISPLAY_V_BLANK)
- 2 - Graphics Engine->Graphic Engine Command Status Register (GE_CMD_STATUS)->Graphics Engine Write Master State (GE_WR_MASTER_STATE)
- 3 - Display Controller->Display Status Register (DISPLAY_STATUS)->Frame Base Address Queue Full (FRAME_BASE_ADDR_QUEUE_FULL)
- 4 - Display Controller->Display Cursor Status Register (DISPLAY_CURSOR_STATUS)->Cursor Status (CURSOR_STAT)
- 5 - Display Controller->Display Cursor Status Register (DISPLAY_CURSOR_STATUS)->Cursor Trigger Armed (CURSOR_TRIG)
- 6 - Reserved
- 7 - Reserved

WRITE_CSR

| | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|------------------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| 0 | command | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | length | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 2 | CSR byte address | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 3 | CSR data | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

- Length = 8
- DWORD - CSR byte address
- DWORD - CSR data

This command writes the data to the addressed CSR.

Before writing, the length field is checked for a correct value.

PAD

| | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| 0 | command | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

This command is used to align subsequent commands to any DWORD boundary, if needed by the driver. Other than ones complement and sequence number checking, this command has no other affect.

11.4 Direct RAM Write Co-processor

The Direct RAM Write Co-processor takes a block of data with a starting address and writes the data into the Display Frame Buffer.

DIRECT_RAM_WRITE

| | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-----|------------------------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| 0 | command | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | length | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 2 | starting byte address | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 3 | data array (see below) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 4 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 5 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| etc | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

- Length = 4 + the number of bytes to be written + the number of pre-padding bytes (it does not count the number of post padding bytes)
- DWORD - starting byte address
- DWORD - first DWORD or pre-padding
- DWORD - second DWORD or pre-padding
- DWORD - third DWORD, etc.

All data is assumed QWORD byte lane aligned when it is read from the FCT. Pre-padding is performed by the host. Therefore no byte shifting is required. Up to 7 pre-pad bytes may exist. The command length will include these but the starting address prevents them from being written. The maximum number of post padding bytes is 3 and is used in order to maintain DWORD alignment of the next command. If the data ends on (or within) the first DWORD of a QWORD, there will not be a matching upper DWORD of 4 pad bytes.

11.4.1 Error Checking and Handling

Before writing into the Display Frame Buffer, the length field is checked for a correct value. It should be at least five plus the 3 lsb's of the starting address. For example, if the starting address is 0x12345673, there are 3 pre-pad bytes. Along with the 4 address bytes and at least one real; data byte, there should be at least 8 bytes.

In the event of an command error or an internal bus error, the command is aborted and an error result is returned to the Command Parser. The Direct RAM Write Co-processor then waits until it is deselected by the Command Parser.

11.4.1.1 Error Codes

The following command specific error codes are indicated in the [Graphics Engine Error Type \(GE_ERROR_TYPE\)](#) field in the [Graphics Engine Command State \(GE_CMD_STATE\)](#) register.

- 1 - Length field error

11.4.2 Aborting Command

In the event that the Direct RAM Write Co-processor is deselected by the Command Parser before the command is completed, the Co-processor will abort the current command and return to idle.

11.5 BLT Co-processor

The Block Transfer (BLT) Co-processor operates on un-compressed rectangle data and supports commands such as rectangle drawing and block copying. The BLT block takes source data (either a constant value or data from the USB endpoint or pixels from the display frame buffer) and uses it to

create pixel data which is written into the display frame buffer. Logical operations maybe performed on pixels read from the display frame buffer. The resulting rectangle is either filled solid (DRAW_FILLED_RECTANGLE), colored with the USB data (DRAW_RAW_RECTANGLE), a logic operation of the source rectangle and the USB constant (ROP_RECTANGLE) or a logic operation of the source rectangle and the USB data (ROP_RAW_RECTANGLE). Basic rectangle copy is supported via ROP_RECTANGLE. All commands support 8, 16 and 24 bpp modes, specified within the command. The destination and source frame base addresses as well as the horizontal and vertical resolutions are also specified within the command as required.

Note: The horizontal resolution must be a multiple of 8 while using 24bpp or 16bpp with up-converting modes.

Note: ROP_RECTANGLE and ROP_RAW_RECTANGLE are limited to a maximum width of 2048 pixels.

DRAW_FILLED_RECTANGLE

| | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|--------------------|----|---------------------|----|----|----|----|----|----|----|----|----|----|----|------------|---------|-----------------------|----|----|----|----|----|---|---|--------|---|---|---|---|---|---|---|
| 0 | command | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | length | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 2 | unused | | upper left Y | | | | | | | | | | | | unused | | upper left X | | | | | | | | | | | | | | | |
| 3 | unused | | height | | | | | | | | | | | | unused | | width | | | | | | | | | | | | | | | |
| 4 | frame base address | | | | | | | | | | | | | | | | | | | | | | | | unused | | | | | | | |
| 5 | unused | | vertical resolution | | | | | | | | | | | | color mode | un-used | horizontal resolution | | | | | | | | | | | | | | | |
| 6 | color (see below) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

color mode = 24bpp

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|--------|--|--|-----|--|--|--|--|--|--|--|--|--|--|-------|--|--|--|--|--|--|--|--|--|--|------|--|--|--|--|--|--|--|
| 6 | unused | | | Red | | | | | | | | | | | Green | | | | | | | | | | | Blue | | | | | | | |
|---|--------|--|--|-----|--|--|--|--|--|--|--|--|--|--|-------|--|--|--|--|--|--|--|--|--|--|------|--|--|--|--|--|--|--|

color mode = 16bpp

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|--------|--|--|--------|--|--|--|--|--|--|--|--|--|--|-----|--|--|--|--|--|--|--|--|--|--|-------|--|--|--|--|--|--|--|
| 6 | unused | | | unused | | | | | | | | | | | Red | | | | | | | | | | | Green | | | | | | | |
|---|--------|--|--|--------|--|--|--|--|--|--|--|--|--|--|-----|--|--|--|--|--|--|--|--|--|--|-------|--|--|--|--|--|--|--|

color mode = 8bpp

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|--------|--|--|--------|--|--|--|--|--|--|--|--|--|--|--------|--|--|--|--|--|--|--|--|--|--|-------|--|--|--|--|--|--|--|
| 6 | unused | | | unused | | | | | | | | | | | unused | | | | | | | | | | | color | | | | | | | |
|---|--------|--|--|--------|--|--|--|--|--|--|--|--|--|--|--------|--|--|--|--|--|--|--|--|--|--|-------|--|--|--|--|--|--|--|

- Length = 20
- WORD - bits 15:12 unused / bits 11:0 upper left X (top left of frame is 0)
- WORD - bits 15:12 unused / bits 11:0 upper left Y (top left of frame is 0)
- WORD - bits 15:12 unused / bits 11:0 width
- WORD - bits 15:12 unused / bits 11:0 height
- DWORD - bits 31:7 frame base address in 128 byte increments / bits 6:0 unused
- WORD - bits 15:14 color mode (11 = 16bpp with up-converting, 10 = 24bpp, 01 = 16bpp, 00 = 8bpp) / bits 13:12 unused / bits 11:0 horizontal resolution

Note:The horizontal resolution must be a multiple of 8 while using 24bpp or 16bpp with up-converting modes.

- WORD - bits 15:12 unused / bits 11:0 vertical resolution
- The color DWORD is formatted depending on the color mode as follows:
 24bpp: DWORD - 1 byte unused / 1 byte = Red / 1 byte = Green / 1 byte = Blue
 16bpp: DWORD - 2 bytes unused / 2 bytes = RGB in 565 format
 8bpp: DWORD - 3 bytes unused / 1 byte = color

DRAW_RAW_RECTANGLE

| | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-----|--------------------------------|--------|---------------------|--------|----|----|----|----|----|----|----|----|------------|------------|-----------------------|-------|----|----|----|----|----|----|---|--------|---|---|---|---|---|---|---|---|
| 0 | command | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | length | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 2 | unused | | upper left Y | | | | | | | | | | unused | | upper left X | | | | | | | | | | | | | | | | | |
| 3 | dir | unused | | height | | | | | | | | | | logic oper | | width | | | | | | | | | | | | | | | | |
| 4 | frame base address | | | | | | | | | | | | | | | | | | | | | | | unused | | | | | | | | |
| 5 | unused | | vertical resolution | | | | | | | | | | color mode | un-used | horizontal resolution | | | | | | | | | | | | | | | | | |
| 6 | packed pixel array (see below) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 7 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 8 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| etc | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

- Length = 16 plus length of pixel color data (including padding) (width x BPP + row padding) x height)
- WORD - bits 15:12 unused / bits 11:0 upper left X (top left of frame is 0)
- WORD - bits 15:12 unused / bits 11:0 upper left Y (top left of frame is 0)
- WORD - bits 15:12 logic operation / bits 11:0 width
- WORD - bit 15 drawing direction (0=top down, 1=bottom up) / bits 14:12 unused / bits 11:0 height
- DWORD - bits 31:7 frame base address in 128 byte increments / bits 6:0 unused
- WORD - bits 15:14 color mode (11 = 16bpp with up-converting, 10 = 24bpp, 01 = 16bpp, 00 = 8bpp) / bits 13:12 unused / bits 11:0 horizontal resolution

Note:The horizontal resolution must be a multiple of 8 while using 24bpp or 16bpp with up-converting modes.

- WORD - bits 15:12 unused / bits 11:0 vertical resolution
- The packed pixel array consists of one or more rows of pixels and is formatted depending on the color mode as follows:
 24bpp: 3 BYTES per pixel padded to DWORD at end of each row
 16bpp: 2 BYTES per pixel padded to DWORD at end of each row
 8bpp: 1 BYTE per pixel padded to DWORD at end of each row

ROP_RECTANGLE

| | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|--------------------------------|----|--------------------------|----|----|----|----|----|----|----|----|----|------------|---------|--------------------------|----|----|----|----|----|----|----|---|--------|---|---|---|---|---|---|---|---|
| 0 | command | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | length | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 2 | unused | | destination upper left Y | | | | | | | | | | unused | | destination upper left X | | | | | | | | | | | | | | | | | |
| 3 | unused | | height | | | | | | | | | | logic oper | | width | | | | | | | | | | | | | | | | | |
| 4 | destination frame base address | | | | | | | | | | | | | | | | | | | | | | | unused | | | | | | | | |
| 5 | unused | | vertical resolution | | | | | | | | | | color mode | un-used | horizontal resolution | | | | | | | | | | | | | | | | | |
| 6 | unused | | source upper left Y | | | | | | | | | | unused | | source upper left X | | | | | | | | | | | | | | | | | |
| 7 | source frame base address | | | | | | | | | | | | | | | | | | | | | | | unused | | | | | | | | |
| 8 | color (see below) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

color mode = 24bpp

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|--------|--|--|--|-----|--|--|--|--|--|--|--|-------|--|--|--|--|--|--|--|------|--|--|--|--|--|--|--|--|--|
| 8 | unused | | | | Red | | | | | | | | Green | | | | | | | | Blue | | | | | | | | | |
|---|--------|--|--|--|-----|--|--|--|--|--|--|--|-------|--|--|--|--|--|--|--|------|--|--|--|--|--|--|--|--|--|

color mode = 16bpp

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|--------|--|--|--|--------|--|--|--|--|--|--|--|-----|--|--|--|--|--|--|--|-------|--|--|--|--|--|--|--|------|--|--|--|--|
| 8 | unused | | | | unused | | | | | | | | Red | | | | | | | | Green | | | | | | | | Blue | | | | |
|---|--------|--|--|--|--------|--|--|--|--|--|--|--|-----|--|--|--|--|--|--|--|-------|--|--|--|--|--|--|--|------|--|--|--|--|

color mode = 8bpp

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|--------|--|--|--|--------|--|--|--|--|--|--|--|--------|--|--|--|--|--|--|--|-------|--|--|--|--|--|--|--|--|--|
| 8 | unused | | | | unused | | | | | | | | unused | | | | | | | | color | | | | | | | | | |
|---|--------|--|--|--|--------|--|--|--|--|--|--|--|--------|--|--|--|--|--|--|--|-------|--|--|--|--|--|--|--|--|--|

- Length = 28
- WORD - bits 15:12 unused / bits 11:0 destination upper left X (top left of frame is 0)
- WORD - bits 15:12 unused / bits 11:0 destination upper left Y (top left of frame is 0)
- WORD - bits 15:12 logic operation / bits 11:0 width (the maximum width supported is 2048 pixels)
- WORD - bits 15:12 unused / bits 11:0 height
- DWORD - bits 31:7 destination frame base address in 128 byte increments / bits 6:0 unused
- WORD - bits 15:14 color mode (11 = 16bpp with up-converting, 10 = 24bpp, 01 = 16bpp, 00 = 8bpp) / bits 13:12 unused / bits 11:0 horizontal resolution

Note: The horizontal resolution must be a multiple of 8 while using 24bpp or 16bpp with up-converting modes.

- WORD - bits 15:12 unused / bits 11:0 vertical resolution
- WORD - bits 15:12 unused / bits 11:0 source upper left X (top left of frame is 0)
- WORD - bits 15:12 unused / bits 11:0 source upper left Y (top left of frame is 0)
- DWORD - bits 31:7 source frame base address in 128 byte increments / bits 6:0 unused
- The color DWORD is formatted depending on the color mode as follows:
 - 24bpp: DWORD - 1 byte unused / 1 byte = Red / 1 byte = Green / 1 byte = Blue
 - 16bpp: DWORD - 2 bytes unused / 2 bytes = RGB in 565 format
 - 8bpp: DWORD - 3 bytes unused / 1 byte = color

ROP_RAW_RECTANGLE

| | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------|--------------------------------|----|---------------|--------------------------|----|----|----|----|----|----|----|----|----|----|---------------|-------------|-----------------------|--------------------------|----|----|----|----|---|---|--------|---|---|---|---|---|---|---|
| 0 | command | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | length | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 2 | unused | | | destination upper left Y | | | | | | | | | | | unused | | | destination upper left X | | | | | | | | | | | | | | |
| 3 | unused | | logic oper | height | | | | | | | | | | | logic oper | | | width | | | | | | | | | | | | | | |
| 4 | destination frame base address | | | | | | | | | | | | | | | | | | | | | | | | unused | | | | | | | |
| 5 | unused | | | vertical resolution | | | | | | | | | | | color mode | un- used | horizontal resolution | | | | | | | | | | | | | | | |
| 6 | unused | | | source upper left Y | | | | | | | | | | | unused | | | source upper left X | | | | | | | | | | | | | | |
| 7 | source frame base address | | | | | | | | | | | | | | | | | | | | | | | | unused | | | | | | | |
| 6 | packed pixel array (see below) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 7 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 8 etc | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

- Length = 24 plus length of pixel color data (including padding) (width x BPP + row padding) x height)
- WORD - bits 15:12 unused / bits 11:0 destination upper left X (top left of frame is 0)
- WORD - bits 15:12 unused / bits 11:0 destination upper left Y (top left of frame is 0)
- WORD - bits 15:12 logic operation[3:0] / bits 11:0 width (the maximum width supported is 2048 pixels)
- WORD - bits 15:13 unused / bit 12 logic operation[4] / bits 11:0 height
- DWORD - bits 31:7 destination frame base address in 128 byte increments / bits 6:0 unused
- WORD - bits 15:14 color mode (11 = 16bpp with up-converting, 10 = 24bpp, 01 = 16bpp, 00 = 8bpp) / bits 13:12 unused / bits 11:0 horizontal resolution

Note: The horizontal resolution must be a multiple of 8 while using 24bpp or 16bpp with up-converting modes.

- WORD - bits 15:12 unused / bits 11:0 vertical resolution
- WORD - bits 15:12 unused / bits 11:0 source upper left X (top left of frame is 0)
- WORD - bits 15:12 unused / bits 11:0 source upper left Y (top left of frame is 0)
- DWORD - bits 31:7 source frame base address in 128 byte increments / bits 6:0 unused
- The packed pixel array consists of one or more rows of pixels and is formatted depending on the color mode as follows:
 - 24bpp: 3 BYTES per pixel padded to DWORD at end of each row
 - 16bpp: 2 BYTES per pixel padded to DWORD at end of each row
 - 8bpp: 1 BYTE per pixel padded to DWORD at end of each row

11.5.1 Error Checking and Handling

Before any pixels are drawn, error checking is performed.

The bottom right corner of the destination rectangle must be within the bounds of the frame. (Upper left X plus width must be less than or equal to the horizontal resolution. Upper left Y plus height must be less than or equal to the vertical resolution.) Note that this also checks that the top left corner is within bounds.

For the ROP_RECTANGLE and ROP_RAW_RECTANGLE commands, the bottom right corner of the source rectangle must be within the bounds of the frame. (Source upper left X plus width must be less than or equal to the horizontal resolution. Source upper left Y plus height must be less than or equal to the vertical resolution.) Note that this also checks that the source top left corner is within bounds.

The height and width are checked for non-zero values. For the ROP_RECTANGLE and ROP_RAW_RECTANGLE commands, the width is also checked for a maximum size of 2048 pixels.

The color mode is checked for a non-supported value.

The length field is checked for a correct value. For the DRAW_FILLED_RECTANGLE and ROP_RECTANGLE commands, the length should be an exact match. For the DRAW_RAW_RECTANGLE and the ROP_RAW_RECTANGLE commands, the length should be at least the fixed portion (16 and 24 respectively) plus 4 (one pixel / pad).

Once the above checks have passed, length checking is done as the command is processed.

The co-processor will not read more data than indicated by the length field. An error is indicated if data runs out before the command is finished (the rectangle drawing reached the end X/Y coordinates).

Once the end X/Y coordinates are reached, there should not be any remaining data to be processed. The residual length should be zero. Note that in the event that the length field value is too large, it will be caught as an error. Also, the next command will most likely be caught as an error as well.

In the event of an command error or an error on the internal data path, the command is aborted and an error result is returned to the Command Parser. The BLT Co-processor then waits until it is deselected by the Command Parser.

11.5.1.1 Error Codes

The following command specific error codes are indicated in the [Graphics Engine Error Type \(GE_ERROR_TYPE\)](#) field in the [Graphics Engine Command State \(GE_CMD_STATE\)](#) register.

- 1 - Length field error
- 2 - Message length too short (ran out of data before command finished)
- 3 - Message length too long (residual data length not zero or pre-fetch FIFOs not empty)
- 4 - The color mode specified is not supported by the command
- 5 - The height or width was set to zero or (for ROP_RECTANGLE and ROP_RAW_RECTANGLE) the width was greater than 2048
- 6 - Some or all of the destination is outside the frame buffer
- 7 - Some or all of the source is outside the frame buffer

11.5.2 Aborting Command

In the event that the BLT Co-processor is deselected by the Command Parser before the command is completed, the Co-processor will abort the current command and return to idle.

11.5.3 Writing to Display Frame Buffer

As each pixel is created, it is written to an internal Pixel to QWORD Converter. The Pixel to QWORD Converter handles all issues associated with the display frame buffer access to permit this module to operate at the pixel level. The Pixel to QWORD Converter is given the color mode, horizontal resolution and frame base address, that was specified in the command. Based on the color mode, the appropriate number and combination of bits from the drawn pixel are selected. Based on the X and Y coordinates, horizontal resolution, color mode and frame base address, the correct memory address is calculated.

16bpp with up-converting mode is handle by the Pixel to QWORD Converter and the Pixel Retriever. The Pixel to QWORD Converter will convert the 16 bits of color (in 565 format) to 24 bits by right padding each color component as needed. The Pixel Retriever will convert the 24 bits of color from the display buffer to 16 bits by stripping off right bits as needed. Note that this will result in a loss of color resolution for operations such as a copy.

11.5.4 DRAW_FILLED_RECTANGLE

This command generates a solid filled rectangle. Pixels and their colors are generated from left to right and from top to bottom.

The color data within the command is a single DWORD which is interpreted based on the color mode, However this block needs just to pass 24 bits to the Pixel to QWORD Converter, which will select the appropriate bits.

This command operates at approximately one clock per pixel, assuming sufficient system memory bandwidth.

11.5.5 DRAW_RAW_RECTANGLE

This command generates a colored rectangle. Pixels and their colors are generated from left to right and either top to bottom or bottom to top. The drawing direction bit specifies if the rectangle is to be drawn top down or bottom up. In both cases, the rectangle is drawn left to right and the starting pixel parameter refers to the upper left corner of the rectangle.

The color data contained in the command consists of an array of pixels colors per row. The array is 1, 2 or 3 bytes (8, 16 or 24 bpp) and is padded at the end of each row to a DWORD boundary. Since the FCT interface is always read as DWORDs, each pixel must be extracted with any left over byte(s) used for the next pixel. In 8bpp mode, the lowest 8 bits represent the left most pixel. In 16bpp mode, the lowest 16 bits represent the left most pixel and the bit positions match the format in the display frame buffer. In 24bpp mode, three bytes are used per pixel, the lowest byte is the blue value, the middle byte is the green value and the left most byte is the red value, all matching the format in the display frame buffer. In 24bpp mode, a pixel might be split across two DWORDs.

The order of the row data matches the vertical direction specified in the command.

This command operates at approximately one clock per pixel, assuming sufficient USB and system memory bandwidth.

11.5.5.1 Logic Operations

The logic operations supported are as follows. C is the color from the command.

| op code | operation |
|---------|---|
| 0x0 | C (unconditional draw) |
| 0x1 | C if C is in-range (conditional draw) |
| 0x2 | C if C is out-of-range (conditional draw) |
| 0x3-0xF | reserved |

Included in the logic operations is a conditional copy. Each pixel from the command data is compared to the chroma key value ([Graphic Engine Chroma Key High Register \(GE_CHROMA_KEY_HIGH\)](#), [Graphic Engine Chroma Key Low Register \(GE_CHROMA_KEY_LOW\)](#)) and checked if it is in-range or out-of-range. Depending on the logic operation specified (in-range or out-of-range), the pixel source pixel is conditionally written to the destination memory. The range checking is done for each color component of the source pixel and the results ANDed. A component can be disabled by setting the low value to 0 and the high value to 255. 16 bpp mode uses 5 bits of red and blue and 6 bits of green. 8 bpp mode uses only the blue value.

11.5.6 ROP_RECTANGLE

This command reads a source rectangle, optionally performs a logic operation with the pixel data specified in the command and writes the rectangle into the display frame buffer. The source and destination can be the same, overlapping or independent areas. Different source and destination frame base addresses can be specified.

Maximum width is 2048 pixels. Top down or bottom up drawing is chosen based on the starting Y location of the source and destination.

11.5.6.1 Logic Operations

The logic operations supported are as follows. S is the source rectangle and C is the constant color from the command.

| op code | operation |
|---------|--|
| 0x0 | S (unconditional copy) - note that the command pixel data (C) is ignored but still present in the command |
| 0x1 | S if S is in-range (conditional copy) - note that the command pixel data (C) is ignored but still present in the command |
| 0x2 | S if S is out-of-range (conditional copy) - note that the command pixel data (C) is ignored but still present in the command |
| 0x3-0x5 | reserved |
| 0x6 | C XOR S |
| 0x7 | C XNOR S |
| 0x8 | C AND S |
| 0x9 | ~C AND S |

| | |
|-----|-----------|
| 0xA | C AND ~S |
| 0xB | ~C AND ~S |
| 0xC | C OR S |
| 0xD | ~C OR S |
| 0xE | C OR ~S |
| 0xF | ~C OR ~S |

The color data within the command is a single DWORD which is interpreted based on the color mode. Included in the logic operations is a conditional copy. Each pixel from the source rectangle is compared to the chroma key value ([Graphic Engine Chroma Key High Register \(GE_CHROMA_KEY_HIGH\)](#), [Graphic Engine Chroma Key Low Register \(GE_CHROMA_KEY_LOW\)](#)) and checked if it is in-range or out-of-range. Depending on the logic operation specified (in-range or out-of-range), the source pixel is conditionally written to the destination memory. The range checking is done for each color component of the source pixel and the results ANDed. A component can be disabled by setting the low value to 0 and the high value to 255. 16 bpp mode uses 5 bits of red and blue and 6 bits of green. 8 bpp mode uses only the blue value.

11.5.7 ROP_RAW_RECTANGLE

This command reads a source rectangle, optionally performs a logic operation with the pixel data specified in the command and writes the rectangle into the display frame buffer. The source and destination can be the same, overlapping, or independent areas. Different source and destination frame base addresses can be specified.

As with the ROP_RECTANGLE command, the 2048 width limit also applies.

The color data consists of an array of pixels colors per row in the same format as the DRAW_RAW_RECTANGLE command.

Note the host S/W must ensure that the order of the rows of data must match the vertical direction that the BLT engine determines.

This command operates at approximately two clocks per pixel, assuming sufficient USB and system memory bandwidth.

11.5.7.1 Logic Operations

The logic operations supported are as follows. S is the source rectangle and C is the color from the command.

| op code | operation |
|---------|---|
| 0x0 | S if C != 0 (masked copy) |
| 0x1 | S if S is in-range and C != 0 (masked conditional copy) |
| 0x2 | S if S is out-of-range and C != 0 (masked conditional copy) |
| 0x3 | S if C is in-range (conditional copy) |
| 0x4 | S if C is out-of-range (conditional copy) |
| 0x5 | reserved |
| 0x6 | C XOR S |
| 0x7 | C XNOR S |

| | |
|---------|---|
| 0x8 | C AND S |
| 0x9 | ~C AND S |
| 0xA | C AND ~S |
| 0xB | ~C AND ~S |
| 0xC | C OR S |
| 0xD | ~C OR S |
| 0xE | C OR ~S |
| 0xF | ~C OR ~S |
| 0x10 | C if S != 0 (masked draw) |
| 0x11 | C if C is in-range and S != 0 (masked conditional draw) |
| 0x12 | C if C is out-of-range and S != 0 (masked conditional draw) |
| 0x13 | C if S is in-range (conditional draw) |
| 0x14 | C if S is out-of-range (conditional draw) |
| 0x15-1F | reserved |

Included in the logic operations are conditional copies and conditional draws. Chroma key matching and pixel masking are supported using both the source rectangle and command data. Either the source rectangle (conditional copy) or the command data (conditional draw) is written to the destination.

For chroma key matching (conditional copy/draw), each pixel (from the source rectangle or the command data) is compared to the chroma key value ([Graphic Engine Chroma Key High Register \(GE_CHROMA_KEY_HIGH\)](#), [Graphic Engine Chroma Key Low Register \(GE_CHROMA_KEY_LOW\)](#)) and checked if it is in-range or out-of-range. Depending on the logic operation specified (in-range or out-of-range), the pixel is conditionally written to the destination memory. The range checking is done for each color component of the source pixel (or command data) and the results ANDed. A component can be disabled by setting the low value to 0 and the high value to 255. 16 bpp mode uses 5 bits of red and blue and 6 bits of green. 8 bpp mode uses only the blue value.

For pixel masking (masked copy/draw), either the source rectangle or command data can be used as a mask.

Masking and chroma key matching may be combined (masked conditional copy/draw).

11.6 JPEG Co-processor

The JPEG Co-processor operates on a JPEG compressed rectangle image and supports various sub-sampling formats. The JPEG Co-processor block takes source data, consisting of header information and the compressed image organized in Minimum Coded Unit (MCU) sized rectangles, and uses it to create pixel data which is written into the display frame buffer. The destination frame base addresses, and image location, as well as the horizontal and vertical resolutions are specified within the command as required.

The JPEG image is specified as 24 bits in the YCbCr color space. Re-sampling of the sub-sampled Cb and Cr values is supported by linear interpolation. Color space conversion is then performed resulting in 24 bit RGB data. 16bpp mode is supported by down-converting the 24bit color.

A bypass mode is supported where the JPEG decompression is skipped. An un-compressed image, still organized as MCUs, is taken and converted to a rectangle using the remaining processes (re-sampling, color space conversion, etc.).

This command operates at approximately 2 clocks per pixel at 4:4:4, 1.4 clocks per pixel at 4:2:2 and 1 clocks per pixel at 4:2:0 sub-sampling, assuming sufficient USB and system memory bandwidth, a system clock of 167 MHz and a jpeg clock of 240-250MHz (or frequencies with a similar clock ratio).

Bypass mode operates at approximately 1 clock per pixel assuming sufficient USB and system memory bandwidth, a system clock of 167 MHz and a jpeg clock of 240-250MHz (or frequencies with a similar clock ratio). For HS operation, the system clock is set at 166.667 MHz and the jpeg clock is set at 240 MHz. for SS operation, the system clock is set at 167 MHz and the jpeg clock is set to 250 MHz. The [Clock JPEG Source Select \(CLK_JPG_SRC_SEL\)](#) bit of the [CPM Control Register \(CPM_CTL\)](#) is used to set the jpeg clock speed.

DECOMP_JPEG

| | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------|----------------------------------|----------|---------------------|----|----|----|----|----|----|----|----|----|----|------------|---------|-----------------------|----|----|----|----|----|----|---|---|--------|---|---|---|---|---|---|---|
| 0 | command | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | length | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 2 | unused | | upper left Y | | | | | | | | | | | unused | | upper left X | | | | | | | | | | | | | | | | |
| 3 | unused | | height | | | | | | | | | | | unused | | width | | | | | | | | | | | | | | | | |
| 4 | frame base address | | | | | | | | | | | | | | | | | | | | | | | | unused | | | | | | | |
| 5 | un-used | sub-samp | vertical resolution | | | | | | | | | | | color mode | un-used | horizontal resolution | | | | | | | | | | | | | | | | |
| 6 | JPEG image (see below) / padding | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 7 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 8 etc | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

- Length = 16 plus length of JPEG image
- WORD - bits 15:12 unused / bits 11:0 upper left X (top left of frame is 0)
- WORD - bits 15:12 unused / bits 11:0 upper left Y (top left of frame is 0)
- WORD - bits 15:12 unused / bits 11:0 width
- WORD - bits 15:12 unused / bits 11:0 height
- DWORD - bits 31:7 frame base address in 128 byte increments / bits 6:0 unused
- WORD - bits 15:14 color mode (11 = reserved, 10 = 24bpp, 01 = 16bpp, 00 = reserved) / bits 13:12 unused / bits 11:0 horizontal resolution

Note: The horizontal resolution must be a multiple of 8 while using 24bpp mode.

- WORD - bits 15:14 unused / bits 13:12 sub-sampling mode (10 = 4:2:0, 01 = 4:2:2, 00 = 4:4:4) / bits 11:0 vertical resolution
- The JPEG image consisting of a sequence of segments each beginning with a marker. A segment may contain:
 - Start of Image indication (SOI)
 - Start of Frame indication (SOF₀ baseline DCT mode)
 - Huffman table(s) (DHT)
 - Quantization table(s) (DQT)
 - The image data
 - Restart interval (DRI) (not supported)
 - Restart indication (RST_m)
 - Number of Lines (DNL)

Comments (COM)
 JPEG Extensions (JPG_n)
 Application Specific (APP_n)
 End of Image indication (EOI)

- 0 to 3 BYTES - padding

BYPASS_JPEG

| | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|----------|--------------------------------------|----------|---------------------|----|----|----|----|----|----|----|----|----|----|----|------------|---------|-----------------------|----|----|----|----|----|---|---|--------|---|---|---|---|---|---|---|
| 0 | command | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | length | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 2 | unused | | upper left Y | | | | | | | | | | | | unused | | upper left X | | | | | | | | | | | | | | | |
| 3 | unused | | height | | | | | | | | | | | | unused | | width | | | | | | | | | | | | | | | |
| 4 | frame base address | | | | | | | | | | | | | | | | | | | | | | | | unused | | | | | | | |
| 5 | un-used | sub-samp | vertical resolution | | | | | | | | | | | | color mode | un-used | horizontal resolution | | | | | | | | | | | | | | | |
| 6 | un-compressed JPEG image (see below) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 7 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 8 etc | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

- Length = 16 plus length of un-compressed JPEG image
- WORD - bits 15:12 unused / bits 11:0 upper left X (top left of frame is 0)
- WORD - bits 15:12 unused / bits 11:0 upper left Y (top left of frame is 0)
- WORD - bits 15:12 unused / bits 11:0 width
- WORD - bits 15:12 unused / bits 11:0 height
- DWORD - bits 31:7 frame base address in 128 byte increments / bits 6:0 unused
- WORD - bits 15:14 color mode (11 = reserved, 10 = 24bpp, 01 = 16bpp, 00 = reserved) / bits 13:12 unused / bits 11:0 horizontal resolution

Note: The horizontal resolution must be a multiple of 8 while using 24bpp mode.

- WORD - bits 15:14 unused / bits 13:12 sub-sampling mode (10 = 4:2:0, 01 = 4:2:2, 00 = 4:4:4) / bits 11:0 vertical resolution
- The un-compressed JPEG image consists of repeated blocks formatted as follows (note that an integral number of blocks is included even if the bonding rectangle is smaller than the data - clipping is performed on the bottom and right side as needed):
 - 4:4:4: 8x8 array of Y values (64 bytes), 8x8 array of Cr values (64 bytes), 8x8 array of Cb values (64 bytes) - representing an 8x8 rectangle
 - 4:2:2: two 8x8 arrays of Y values (left and right) (128 bytes), 8x8 array of Cr values (64 bytes), 8x8 array of Cb values (64 bytes) - representing a 16x8 rectangle. Interpolation of the Cr and Cb values is performed in the X direction.
 - 4:2:0: four 8x8 arrays of Y values (top left, top right, bottom left and bottom right) (256 bytes), 8x8 array of Cr values (64 bytes), 8x8 array of Cb values (64 bytes) - representing a 16x16 rectangle. Interpolation of the Cr and Cb values is performed in the X and Y directions.

11.6.1 Error Checking and Handling

Before any pixels are drawn, error checking is performed

The bottom right corner of the drawn rectangle must be within the bounds of the frame. (Upper left X plus width must be less than or equal to the horizontal resolution. Upper left Y plus height must be less than or equal to the vertical resolution.) Note that this also checks that the top left corner is within bounds.

The height and width are checked for non-zero values.

The color mode is checked for non-supported value.

The length field is checked for a correct value. The length should be at least the fixed portion (16 bytes) plus 1 byte of image (even though 1 byte of image data is insufficient for any image, the length checking (described next) will catch those types of errors.

Once the above checks have passed, length checking is done as the command is processed.

The co-processor should not read more data than indicated by the length field. An error is indicated if data runs out before the command is finished (the rectangle drawing reached the end X/Y coordinates).

Once the end X/Y coordinates are reached, there should not be any remaining data to be processed. The residual length should be zero. Note that in the event that the length field value is too large, it will be caught as an error. Also, the next command will most likely be caught as an error as well.

Other error conditions that are handled are:

- Image too big
- Image too small

In the event of an command error or an error on the internal data path, the command is aborted and an error result is returned to the Command Parser. The JPEG Co-processor then waits until it is deselected by the Command Parser.

11.6.1.1 Error codes

- 1 - Length field error
- 2 - Message length too short (ran out of data before command finished)
- 3 - Message length too long (residual data length not zero or pre-fetch FIFOs not empty)
- 4 - The color mode specified is not supported by the command
- 5 - The height or width was set to zero
- 6 - Some or all of the destination is outside the frame buffer
- 7 - Image too big
- 8 - Image too small

11.6.2 Aborting Command

In the event that the JPEG Co-processor is deselected by the Command Parser before the command is completed, the Co-processor should abort the current command and return to idle.

11.6.3 Block Diagram

The basic flow of the JPEG Co-processor is shown below.

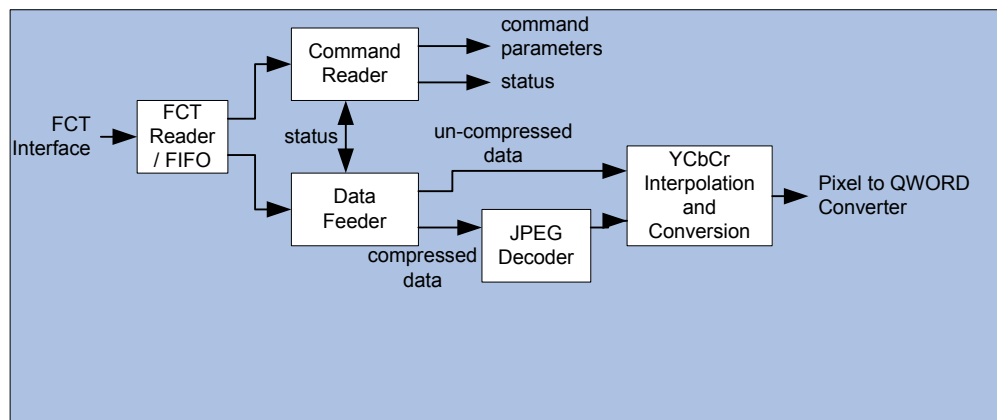


Figure 11.1 JPEG Co-processor Block Diagram

11.6.4 FCT Reader / FIFO

This block reads data from the FCT interface and sends it to the Command Reader and Data Feeder blocks.

The total length of the JPEG command is provided by the Command Parser module. Once this block reads the appropriate amount of data, it stops requesting from the FCT.

11.6.5 Command Reader

This block reads data from the FIFO in the FCT Reader and extracts the upper left x, upper left y, width, height, horizontal and vertical resolutions, MCU sub-sampling mode (0=444, 1=422, 2=420), frame base address and color depth.

Command parameter error checking is done in this block.

Once the command header is read, this block instructs the Data Feeder block to move the JPEG image data.

11.6.6 Data Feeder

This block reads data from the FIFO in the FCT Reader and sends it to the JPEG decoder. If processing the BYPASS_JPEG command, data is sent directly to the YCbCr Interpolation and Conversion block.

Message length and image size error checking are done in this block as the data is fed.

11.6.7 JPEG Decoder

The JPEG decoder offers the following features:

- Compatibility: 100 percent baseline ISO/IEC 10918-1 JPEG compliant
- 8-bit/channel pixel depths
- Support for JPEG header parsing
- Up to four programmable quantization tables
- Fully programmable Huffman tables (two AC and two DC)
- Fully programmable minimum coded unit (MCU)

- Single-cycle Huffman decoding

11.6.8 YCbCr Interpolation and Conversion

This block receives the Y, Cb and Cr values from the JPEG Decoder or, for JPEG Bypass mode, directly from the Data Feeder and converts them into RGB data. The following video formats are supported: 4:4:4, 4:2:2, and 4:2:0.

11.6.8.1 Writing to Display Frame Buffer

As each pixel is created, it is written to an internal Pixel to QWORD Converter. The Pixel to QWORD Converter handles all issues associated with the display frame buffer access to permit this module to operate at the pixel level. The Pixel to QWORD Converter is given the color mode, horizontal resolution and frame base address that was specified in the command. Based on the color mode, the appropriate number of bits from the drawn pixel are selected. Based on the X and Y coordinates, horizontal resolution, color mode and frame base address, the correct memory address is calculated.

Since JPEG images operate in the YCbCr color space, there is no direct support for 16bpp. In order to support 16bpp operation, the S/W driver will right pad the 5:6:5 RGB data into 8:8:8 format. Therefore when in 16bpp mode, this module strips the right most 3, 2 and 3 bits, respectively, from the 8:8:8 data RGB before sending it to the Pixel to QWORD Converter

11.7 Hextile Co-processor

The Hextile Co-processor takes source data from the FCT, organized in tiles and sub-tiles, and uses it to create pixel data which is written into the display frame buffer. The destination frame base addresses, and image location, as well as the horizontal and vertical resolutions, are specified within the command, as required.

24bpp, 16bpp, 16bpp with up-converting and 8bpp modes are supported.

DECOMP_HEXTILE

| | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-----|-------------------------------------|----|---------------------|----|----|----|----|----|----|----|----|----|------------|---------|-----------------------|----|----|----|----|----|----|----|---|---|--------|---|---|---|---|---|---|---|
| 0 | command | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | length | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 2 | unused | | upper left Y | | | | | | | | | | unused | | upper left X | | | | | | | | | | | | | | | | | |
| 3 | unused | | height | | | | | | | | | | unused | | width | | | | | | | | | | | | | | | | | |
| 4 | frame base address | | | | | | | | | | | | | | | | | | | | | | | | unused | | | | | | | |
| 5 | un-used | | vertical resolution | | | | | | | | | | color mode | un-used | horizontal resolution | | | | | | | | | | | | | | | | | |
| 6 | Hextile image (see below) / padding | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 7 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 8 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| etc | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

- Length = 16 plus length of JPEG image
- WORD - bits 15:12 unused / bits 11:0 upper left X (top left of frame is 0)
- WORD - bits 15:12 unused / bits 11:0 upper left Y (top left of frame is 0)
- WORD - bits 15:12 unused / bits 11:0 width
- WORD - bits 15:12 unused / bits 11:0 height

- DWORD - bits 31:7 frame base address in 128 byte increments / bits 6:0 unused
- WORD - bits 15:14 color mode (11 = 16bpp with up-converting, 10 = 24bpp, 01 = 16bpp, 00 = 8bpp) / bits 13:12 unused / bits 11:0 horizontal resolution

Note: The horizontal resolution must be a multiple of 8 while using 24bpp or 16bpp with up-converting modes.

- WORD - bits 15:12 unused / bits 11:0 vertical resolution
- The Hextile image, which consists of:
Subencoding type consisting of 1 byte with the following value definitions - 1 = Raw / 2 = BackgroundSpecified / 4 = ForegroundSpecified / 8 = AnySubrects / 16 = SubrectsColoured

Pixel array consisting of 256 pixels each with 1, 2 or 3 BYTES - only present if Raw bit set in subencoding type

Background pixel color consisting of 1, 2 or 3 bytes - only present if BackgroundSpecified bit set in subencoding type

Foreground pixel color consisting of 1, 2 or 3 bytes - only present if ForegroundSpecified bit set in subencoding type

Number of subrectangles consisting of 1 byte - only present if AnySubrects bit set in subencoding type

Subrectangle array consisting of "number of subrectangles" of the following array - only present if AnySubrects bit set in subencoding type:

Subrectangle pixel color consisting of 1, 2 or 3 bytes - only present if SubrectsColoured bit set in subencoding type

Subrectangle x and y offsets consisting of 1 byte. The 4 msbs are the x value and the 4 lsbs are the y value

Subrectangle width and height consisting of 1 byte. The 4 msbs specify the width minus one and the 4 lsbs specify the height minus one

The data stream starts at bits [7:0] of the DWORD from the FCT, followed by bits [15:8], then bits [23:16], then bits [31:24]. The fifth byte is at bits [7:0] in the next DWORD.

In 24bpp mode, when a pixel is encountered in the data stream, it is formatted with the blue byte first, followed by green, followed by red. For example, if the pixel started at bits [7:0], [7:0] would be blue, [15:8] would be green and [23:16] would be red. It is possible that the pixel is split across two DWORDs. For example, if the pixel started at bits [23:16], [23:16] would be blue, [31:24] would be green and [7:0] of the next DWORD would be red.

In 16bpp and 16bpp with up-converting modes, when a pixel is encountered in the data stream, it is formatted with the 5 blue bits first, followed by the 3 lsb green bits, followed by the 3 msb green bits in the next byte, followed by the 5 red bits. For example, if the pixel started at bits [15:8], [12:8] would be blue, [18:13] would be green and [23:19] would be red. It is possible that the pixel is split across two DWORDs. For example, if the pixel started at bits [31:24], [28:24] would be blue, [2:0] of the next DWORD along with [31:29] would be green and [7:3] of the next DWORD would be red. In 8bpp, each byte contains a full pixel.

- 0 to 3 BYTES - padding

11.7.1 Error Checking and Handling

Before any pixels are drawn, error checking is performed.

The bottom right corner of the drawn rectangle must be within the bounds of the frame. (Upper left X plus width must be less than or equal to the horizontal resolution. Upper left Y plus height must be

less than or equal to the vertical resolution.) Note that this also checks that the top left corner is within bounds.

The height and width are checked for non-zero values.

The color mode is checked for a non-supported value.

The length field is checked for a correct value. The length should be at least the fixed portion (16 bytes) plus 2 bytes of image.

As the command is processed, the image is checked for valid combinations of mask bits in the subencoding type byte, for valid foreground & background colors and for a valid number of subrectangles (if appropriate).

Length checking is also done as the command is processed.

The co-processor should not read more data than indicated by the length field. An error is indicated if data runs out before the command is finished (the rectangle drawing reached the end X/Y coordinates).

Once the end X/Y coordinates are reached, there should not be any remaining data to process.

The residual length should be zero. Note that in the event that the length field value is too large, this will be caught as an error. Also, the next command will most likely be caught as an error as well.

In the event of an command error or an error on the internal data path, the command is aborted and an error result is returned to the Command Parser. The Hextile Co-processor then waits until it is deselected by the Command Parser.

11.7.1.1 Error codes

- 1 - Length field error (length less than 18)
- 2 - Message length too short (ran out of data before command finished)
- 3 - Message length too long (residual data length not zero or pre-fetch FIFOs not empty)
- 4 - Reserved
- 5 - The height or width was set to zero
- 6 - Some or all of the destination is outside the frame buffer
- 7 - Tile with Raw mask set and with one or more of BackgroundSpecified, ForegroundSpecified, AnySubrects or SubrectsColoured masks also set
- 8 - Tile with BackgroundSpecified mask not set and background has not yet been set for this command
- 9 - Tile with ForegroundSpecified mask set but without AnySubrects mask set
- 0xA - Tile with ForegroundSpecified mask set but SubrectsColoured mask also set
- 0xB - Tile with AnySubrects mask set, both ForegroundSpecified and SubrectsColoured masks not set and foreground has not yet been set for this command
- 0xC - Tile with SubrectsColoured mask set but without AnySubrects mask set
- 0xD - Tile with AnySubrects but the number of subrectangles is 0

11.7.2 Aborting Command

In the event that the Hextile Co-processor is deselected by the Command Parser before the command is completed, the Hextile Co-processor will abort the current command and return to idle.

11.7.3 Block Diagram

The basic flow of the Hextile Co-processor is shown below.

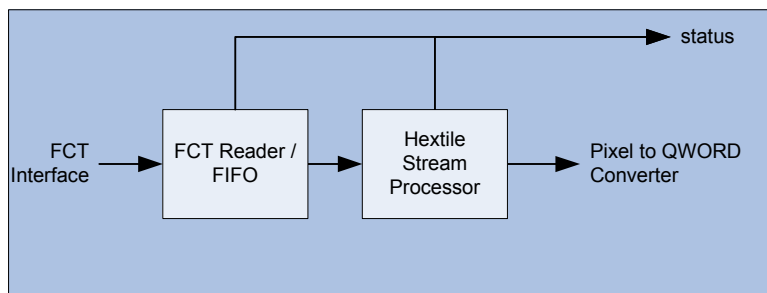


Figure 11.2 Hextile Co-processor Block Diagram

11.7.4 FCT Reader / FIFO

This block reads data from the FCT interface and sends it to the Hextile Stream Processor. The FCT Reader takes complete DWORDS from the FCT and presents 5 bytes to the Hextile Stream Processor. The Hextile Stream Processor can consume 1 to 5 bytes per cycle. Based on the number of bytes used, the remaining bytes are shifted and new bytes are added. The total length of the Hextile command is provided by the Command Parser module. Once this block reads the appropriate amount of data, it stops requesting from the FCT.

Command errors 1, 2 and 3 from [Section 11.7.1.1](#) are checked for in this block.

11.7.5 Hextile Stream Processor

The first function that this block performs is to extract the upper left x, upper left y, width, height, horizontal and vertical resolutions, frame base address and color depth from the command header. Command errors 5 and 6 from [Section 11.7.1.1](#) are checked for in this block.

Following the command header parsing, the Hextile Stream Processor parses 16 x16 tiles and determines if clipping is required. Normally, the width and height of a tile is 16 x 16. However, it is likely that the total image size is not a multiple of 16. Therefore at the right and bottom edges, clipping is performed with the width and height adjusted as necessary.

11.7.5.1 Tile Decoding

Tiles are decoded per the RFB (remote frame buffer) Protocol. The first byte is the subencoding type which contains Raw, BackgroundSpecified, ForegroundSpecified, AnySubrects and SubrectsColoured masks.

If the Raw bit is set:

256 pixels follow ordered from left to right, top to bottom. If this bit is set then the BackgroundSpecified, ForegroundSpecified, AnySubrects and SubrectsColoured bits must be zero.

Note: The RFB Protocol states that the BackgroundSpecified, ForegroundSpecified, AnySubrects and SubrectsColoured bits are irrelevant for when the Raw bit is set. These requirements are more restrictive.

If the BackgroundSpecified bit is set:

1, 2 or 3 bytes (depending on the color mode) follow that specify the background color. All pixels in the tile are set to this color unless changed by a subsequent subtitle.

If the BackgroundSpecified bit is not set:

The background is the same as the previous tile. However the background pixel value may not be carried over if the previous tile was Raw.

If the `ForegroundSpecified` bit is set:

1, 2 or 3 bytes (depending on the color mode) follow that specify the foreground color to be used in subsequent subtiles. If this bit is set then the `SubrectsColoured` bit must be zero and the `AnySubrects` bit must be one.

If the `ForegroundSpecified` bit is not set:

The foreground is the same as the previous tile. However the foreground pixel value may not be carried over if the previous tile had the `Raw` or `SubrectsColoured` bits set.

If the `AnySubrects` bit is set:

1 byte follows specifying the number of subrectangles. The number of subrectangles must not be zero.

Following the “number of subrectangles” is an array of x, y, width, height and, optionally, pixel colors.

If the `SubrectsColoured` bit is set:

The subrectangle also includes (preceding the x, y, width and height) 1, 2 or 3 bytes (depending on the color mode) specifying the pixel color. If this bit is set then the `ForegroundSpecified` bit must be zero.

If the `SubrectsColoured` bit is not set:

All subrectangles receive the foreground color (either set above or carried over).

Subrectangle x and y offsets consisting of 1 byte. The 4 msbs are the x value and the 4 lsbs are the y value.

Subrectangle width and height consisting of 1 byte. The 4 msbs specify the width minus one and the 4 lsbs specify the height minus one.

If the `AnySubrects` bit is not set:

There are no subrectangles (the entire tile is solid background color). If this bit is not set, then the `SubrectsColoured` must be zero.

Command errors 7 through 0XD from [Section 11.7.1.1](#) are checked for in this block.

11.7.5.2 Writing to Display Frame Buffer

As each pixel is created, it is written to an internal Pixel to QWORD Converter. The Pixel to QWORD Converter handles all issues associated with the display frame buffer access to permit this module to operate at the pixel level. The Pixel to QWORD Converter is given the color mode, horizontal resolution and frame base address, that was specified in the command. Based on the color mode, the appropriate number and combinations of bits from the drawn pixel are selected. Based on the X and Y coordinates, horizontal resolution, color mode and frame base address, the correct memory address is calculated.

16bpp with up-converting mode is handle by the Pixel to QWORD Converter. It will convert the 16 bits of color (in 565 format) to 24 bits by right padding each color component as needed.

11.8 JPEG-LS Co-processor

The JPEG-LS Co-processor operates on a JPEG-LS compressed rectangle image. The JPEG-LS Co-processor block takes source data, consisting of header information and the compressed image, and uses it to create pixel data which is written into the display frame buffer. The destination frame base addresses, and image location, as well as the horizontal and vertical resolutions, are specified within the command, as required.

The JPEG-LS image is specified in the RGB color space using three 8 bit components. 24bpp, 16bpp and 16bpp with up-converting modes are supported. 16bpp and 16bpp with up-converting modes use the upper 5 red, 6 green and 5 blue bits of the 24 bit RGB value.

The JPEG-LS decoder supports Regular and Run (constant color) modes.

Note that the maximum image width supported is 2048 pixels.

This command operates at approximately 3 clocks per pixel while decoding in regular mode and at approximately 1 clock per pixel while decoding in run mode, both assuming sufficient USB and system memory bandwidth.

DECOMP_JPEG_LS

| | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|-----|-------------------------------------|----|---------------------|----|----|----|----|----|----|----|----|----|------------|---------|-----------------------|----|----|----|----|----|----|----|---|--------|---|---|---|---|---|---|---|---|
| 0 | command | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | length | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 2 | unused | | upper left Y | | | | | | | | | | unused | | upper left X | | | | | | | | | | | | | | | | | |
| 3 | unused | | height | | | | | | | | | | unused | | width | | | | | | | | | | | | | | | | | |
| 4 | frame base address | | | | | | | | | | | | | | | | | | | | | | | unused | | | | | | | | |
| 5 | un-used | | vertical resolution | | | | | | | | | | color mode | un-used | horizontal resolution | | | | | | | | | | | | | | | | | |
| 6 | JPEG-LS image (see below) / padding | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 7 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 8 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| etc | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

- Length = 16 plus length of JPEG image
- WORD - bits 15:12 unused / bits 11:0 upper left X (top left of frame is 0)
- WORD - bits 15:12 unused / bits 11:0 upper left Y (top left of frame is 0)
- WORD - bits 15:12 unused / bits 11:0 width (the maximum width supported is 2048 pixels)
- WORD - bits 15:12 unused / bits 11:0 height
- DWORD - bits 31:7 frame base address in 128 byte increments / bits 6:0 unused
- WORD - bits 15:14 color mode (11 = 16bpp with up-converting, 10 = 24bpp, 01 = 16bpp, 00 = reserved) / bits 13:12 unused / bits 11:0 horizontal resolution

Note: The horizontal resolution must be a multiple of 8 while using 24bpp or 16bpp with up-converting modes.

- WORD - bits 15:12 unused / bits 11:0 vertical resolution
- The JPEG-LS image consisting of a sequence of markers and marker segments:
 - Start of Image (SOI) marker
 - Start of Frame (SOF₅₅) marker and marker segment
 - Start of Scan (SOS) marker and marker segment
 - Image data segment - The compressed image data is always formatted as 24 bits per pixel. In 16bpp and 16bpp up-convert modes, the upper bits of each component are not used in which case the format is 3 unused, 5 bits red, 2 unused, 6 bits green, 3 unused, 5 bits blue).
 - End of Image marker (EOI)
 - Note that the JPEG-LS specification also includes JPEG-LS preset parameters (LSE), Define Number of Lines (DNL), Define Restart Interval (DRI), Restart (RST_m), Application (APP_n) and Comment (COM) markers, however these are not supported.
- 0 to 3 BYTES - padding

11.8.1 Error Checking and Handling

Before any pixels are drawn, error checking is performed.

The bottom right corner of the drawn rectangle must be within the bounds of the frame. (Upper left X plus width must be less than or equal to the horizontal resolution. Upper left Y plus height must be less than or equal to the vertical resolution.) Note that this also checks that the top left corner is within bounds.

The height and width are checked for non-zero values. The width is also checked for a maximum size of 2048 pixels.

The color mode is checked for non-supported value.

The length field is checked for a correct value. The length should be at least the fixed portion (16 bytes) plus 1 byte of image (even though 1 byte of image data is insufficient for any image, the length checking (described next) will catch those types of errors).

As the command is processed, the image is checked for the correct sequence of markers and parameters. The order of markers and marker segments is: SOI, SOF, SOS, image data & EOF. The expected parameters are listed in [Section 11.8.6](#).

Length checking is also done as the command is processed.

The co-processor should not read more data than indicated by the length field. An error is indicated if data runs out before the command is finished (the rectangle drawing reached the end X/Y coordinates).

Once the end X/Y coordinates are reached, there should not be any remaining data to process. The residual length should be zero. Note that in the event that the length field value is too large, this will be caught as an error. Also, the next command will most likely be caught as an error as well.

Checking is provided to detect end of Image mismatch - either when the EOI marker is missing or in the wrong spot. In the event of a command error or an error on the internal data path, the command is aborted and an error result is returned to the Command Parser. The JPEG-LS Co-processor then waits until it is deselected by the Command Parser.

11.8.1.1 Error codes

- 1 - Length field error
- 2 - Message length too short (ran out of data before command finished)
- 3 - Message length too long (residual data length not zero or pre-fetch FIFOs not empty)
- 4 - The color mode specified is not supported by the command
- 5 - The height or width was set to zero or the width was greater than 2048
- 6 - Some or all of the destination is outside the frame buffer
- 7 - End of Image mismatch (EOI marker missing or in the wrong spot)
- 8 - Header mismatches (header fields mismatch expected values for LS sample interleaved mode or width/height does not match the drawing command header width/height)

11.8.2 Aborting Command

In the event that the JPEG-LS Co-processor is deselected by the Command Parser before the command is completed, the JPEG-LS Co-processor should abort the current command and return to idle.

11.8.3 Block Diagram

The basic flow of the JPEG-LS Co-processor is shown below. Decoding is done in accordance with ISO/IEC International Standards 14495-1 and 10918-1.

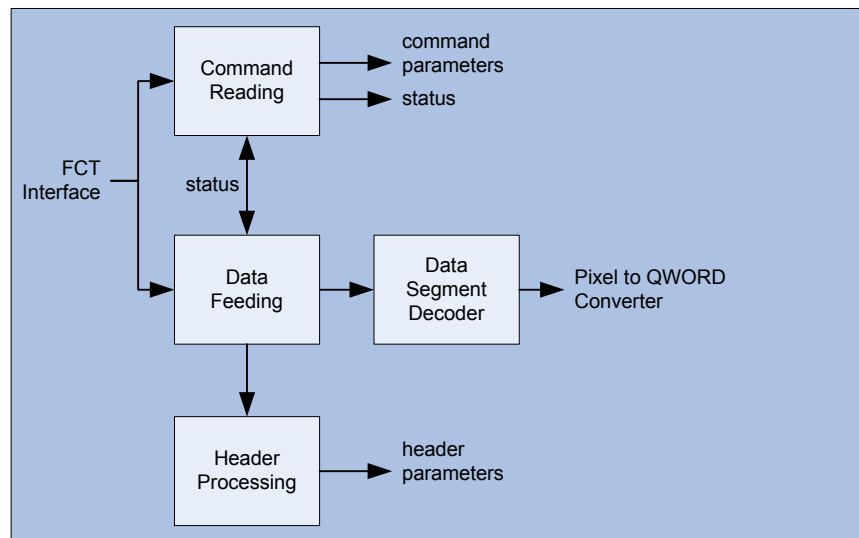


Figure 11.3 JPEG-LS Co-processor Block Diagram

11.8.4 Command Reading

This block reads data from the FCT and extracts the upper left x, upper left y, width, height, horizontal and vertical resolutions, frame base address, and color depth.

Command parameter error checking is done in this block.

11.8.5 Data Feeding

This block reads data from the FCT and sends it to either the Header Processor or the Data Segment Decoder. Bytes from the FCT are processed low byte (FCT bits 7-0) to high byte (FCT bits 24:31).

Message length and image size error checking are done in this block.

11.8.6 Header Processing

The Header Processor receives the JPEG-LS header from the Data Feeder and parses the various marker and marker segments. The markers and marker segments are checked for errors and the parameters are checked for their expected values.

It is expected that the following JPEG-LS marker and marker segments are used as specified and in the order specified.

- Start of Image (SOI - 0xFFD8)
- Start of Frame (SOF₅₅ - 0xFFF7)
 - Frame header length (Lf) must be 0x0011
 - Sample precision (P) must be 0x08
 - Number of lines (Y) must equal that specified in the DECOMP_JPEG_LS command and must not be zero (DNL and LSE are not supported)
 - Number of samples per line (X) must equal that specified in the DECOMP_JPEG_LS command and must not be zero (LSE is not supported)

Number of image components in frame (Nf) must equal 3
 Component identifier 1 (C_1) must equal 1
 Horizontal sampling factor for component 1 (H_1) must equal 1
 Vertical sampling factor for component 1 (V_1) must equal 1
 Quantization table destination selector for component 1 (Tq_1) must equal 0
 Component identifier 2 (C_2) must equal 2
 Horizontal sampling factor for component 2 (H_2) must equal 1
 Vertical sampling factor for component 2 (V_2) must equal 1
 Quantization table destination selector for component 2 (Tq_2) must equal 0
 Component identifier 3 (C_3) must equal 3
 Horizontal sampling factor for component 3 (H_3) must equal 1
 Vertical sampling factor for component 3 (V_3) must equal 1
 Quantization table destination selector for component 3 (Tq_3) must equal 0

- Start of Scan (SOS - 0xFFDA)
 - Scan header length (L_s) must be 0x000C
 - Number of image components in scan (N_s) must equal 3
 - Scan component selector 1 (Cs_1) must equal 1
 - Mapping table selector for component 1 (Tm_1) must equal 0
 - Scan component selector 2 (Cs_2) must equal 2
 - Mapping table selector for component 2 (Tm_2) must equal 0
 - Scan component selector 3 (Cs_3) must equal 3
 - Mapping table selector for component 3 (Tm_3) must equal 0
 - NEAR parameter (NEAR) must equal 0 (lossless)
 - ILV parameter (ILV) must equal 2 (sample interleaved scan)
 - Successive approximation bit position high (A_h) must equal 0 (no meaning in lossless)
 - Successive approximation bit position low (A_l) must equal 0 (no point transform)
- Image Data Segment
- End of Image (EOI - 0xFFD9)

JPEG-LS preset parameters (LSE), Define Number of Lines (DNL), Define Restart Interval (DRI), Restart (RST_m), Application (APP_n) and Comment (COM) markers are not supported.

Since the LSE marker is not supported, the quantization threshold values (T_1 , T_2 and T_3) are fixed at 3, 7 and 21 respectively and the value of RESET is fixed at 64.

11.8.7 Data Segment Decoder

The Data Segment Decoder involves the following steps:

- Reading of neighbor pixels
- Context modeling and determining decode mode
- Regular mode decoding
 - Pixel prediction
 - Pixel prediction correction and clamping
 - Golomb decoding
 - Pixel reconstruction
 - Context updating
- Run mode decoding
 - Run length decode
 - Run interrupt sample decode
 - Golomb decoding
 - Interrupt sample reconstruction
 - Context updating

11.8.8 Writing to Display Frame Buffer

As each pixel is created, it is written to an internal Pixel to QWORD Converter. The Pixel to QWORD Converter handles all issues associated with the display frame buffer access to permit this module to operate at the pixel level. The Pixel to QWORD Converter is given the color mode, horizontal resolution and frame base address that was specified in the command. Based on the color mode, the appropriate number of bits from the drawn pixel are selected. Based on the X and Y coordinates, horizontal resolution, color mode, and frame base address, the correct memory address is calculated.

16bpp with up-converting mode is handled by the Pixel to QWORD Converter. It will convert the 16 bits of color (in 565 format) to 24 bits by right padding each color component as needed.

11.9 General Error Checking and Handling

In the event of any internal data path errors, the command being processed is aborted.

All holding registers, counters, etc are cleared when [Graphics Engine Restart \(GE_RESTART\)](#) bit in [Graphic Engine Control Register \(GE_CTRL\)](#) is set.

11.10 Pixel Retriever

This block, when commanded, reads a block of pixels from the display frame buffer. When finished, the pixels can be read in sequential order. Up to 2K pixels can be stored.

11.10.1 Pixel Retrieving

The input to this block is by way of a command which specifies the frame base address, starting X/Y pixel (zero being the top left), number of pixels, horizontal resolution and color mode. The corresponding starting memory address is calculated and the display frame buffer memory is read.

As the display frame buffer is read, up to 2, 4, or 8 pixels (depending on the color mode) are extracted from each QWORD and are placed into shared scratch pad memory. Note that the first pixel may have any pixel alignment within the QWORD and its position depends upon the starting X coordinate.

The pixels are stored in the scratch pad memory unpacked, with each pixel consuming a DWORD regardless of the color mode.

The scratch pad memory is a shared resource and is connected to this block during BLT commands.

11.10.2 Pixel Reading

Once the command is finished and all of the pixels have been read from the display frame buffer, the shared scratch pad memory acts as a read only FIFO. The first data is placed on the output. When it has been read, a read signal increments a pointer and the next pixel is presented. When the last pixel is read, the output becomes undefined.

In the event that a new command is issued before the last pixel has been read, the remaining pixels are discarded (the read pointer is set back to zero).

11.10.3 Pixel Color Data

When 16bpp with up-converting mode is indicated, the Pixel Retriever will convert the 24 bits of color from the display buffer to 16 bits by stripping off right bits as needed. Note that this will result in a loss of color resolution for operations such as a copy.

11.10.4 Error Checking and Handling

In the event of an error on the internal memory bus, the command is aborted

All holding registers, counters, etc are cleared when [Graphics Engine Restart \(GE_RESTART\)](#) bit in [Graphic Engine Control Register \(GE_CTRL\)](#) is set.

11.11 Pixel to QWORD Converter

This block receives pixels (X, Y and color) and calculates the memory address. It assembles multiple pixels into a QWORD worth of data and then writes the data into the system memory.

When pixels are received, they can be flagged as masked pixels. These are pixels that are not to be written into system memory.

The Pixel to QWORD Converter can be accessed by the various Graphic Engine co-processors. The access to the Pixel to QWORD Converter is statically enabled based on the Graphic Engine co-processor currently invoked.

11.11.1 Memory Address

The memory address of a pixel is equal to $((Y \times \text{horizontal resolution} + X) \times \text{bytes per pixel}) + \text{frame base address}$. Bytes per pixel is either 1, 2 or 3 for 8bpp, 16bpp and 24bpp (and 16bpp with up-converting modes.) respectively.

11.11.2 Pixel Color Data

Based on the color mode set by the selected co-processor, the correct bits and byte lanes of the color are used as the data to be written into memory.

When 16bpp with up-converting mode is indicated, the Pixel to QWORD Converter will convert the 16 bits of color (in 565 format) to 24 bits by right padding each color component as needed.

11.12 Control and Status Registers

Table 11.1 Graphics Engine Control and Status Register Map

| ADDRESS OFFSET | REGISTER NAME (SYMBOL) |
|-------------------|--|
| 000h | Graphic Engine Control Register (GE_CTRL) |
| 004h | Graphic Engine Command Status Register (GE_CMD_STATUS) |
| 008h | Graphic Engine Command Info Register (GE_CMD_INFO) |
| 00Ch | Graphic Engine Command Length Register (GE_CMD_LEN) |
| 010h | Graphic Engine Command Offset Register (GE_CMD_OFFSET) |
| 014h | Graphic Engine Previous Command Info Register (GE_PREV_CMD_INFO) |
| 018h | Graphic Engine Previous Command Length Register (GE_PREV_CMD_LEN) |
| 01Ch | Graphic Engine Previous Command Offset Register (GE_PREV_CMD_OFFSET) |
| 020h | Graphic Engine Chroma Key High Register (GE_CHROMA_KEY_HIGH) |
| 024h | Graphic Engine Chroma Key Low Register (GE_CHROMA_KEY_LOW) |
| Reserved | |
| 028h – FFFh | Reserved for future expansion |

11.12.1 Graphic Engine Control Register (GE_CTRL)

Offset: 000h Size: 32 bits

This register is used to configure and control the Graphics Engine.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|--|------|---------|
| 31:16 | Graphics Engine Initial Command Sequence Number (GE_INIT_CMD_SEQ_NUM) This field is used to initialize the expect sequence number of the next command. It can be written at any point but only takes affect when the Graphics Engine Restart (GE_RESTART) bit is set or when the Graphics Engine Enable (GE_ENABLE) bit changes from low to high. This is used when the Graphics Engine Command Sequence Number Enable (GE_CMD_SEQ_EN) bit is set. | R/W | 0000h |
| 15:5 | RESERVED | RO | - |
| 4 | Graphics Engine Command Ones Complement Enable (GE_CMD_ONECOMP_EN) When set, ones complement checking is performed by the Command Parser. | R/W | 0b |
| 3 | Graphics Engine Command Sequence Number Enable (GE_CMD_SEQ_EN) When set, sequence number checking is performed by the Command Parser. | R/W | 0b |
| 2 | Graphics Engine Abort (GE_ABORT) Write a 1 to abort an active command. This will cause an endpoint stall and should be followed by setting the Graphics Engine Restart (GE_RESTART) bit. Note: This bit should not be set at the same time as the Graphics Engine Restart (GE_RESTART) bit. | SC | 0b |
| 1 | Graphics Engine Restart (GE_RESTART) Write a 1 to restart the Graphics Engine in the event of an error that causes an endpoint stall. This bit will also flush the Pixel Retriever and the Pixel to QWORD Converter. Note: This bit should not be set at the same time as the Graphics Engine Abort (GE_ABORT) bit. Note: This bit should not be set unless the Graphics Engine Command State (GE_CMD_STATE) field in Graphic Engine Command Status Register (GE_CMD_STATUS) indicates Stalled. | SC | 0b |
| 0 | Graphics Engine Enable (GE_ENABLE) When enabled, the Graphics Engine will execute commands. When disabled, the current command will finish and once reaching IDLE, the Graphics Engine will stop. If previously changed from enabled to disabled, this bit should not be re-enabled unless the Graphics Engine Read Master State (GE_RD_MASTER_STATE) , Graphics Engine Write Master State (GE_WR_MASTER_STATE) and Graphics Engine Command State (GE_CMD_STATE) fields in Graphic Engine Command Status Register (GE_CMD_STATUS) all indicate IDLE. | R/W | 0b |

11.12.2 Graphic Engine Command Status Register (GE_CMD_STATUS)

Offset: 004h Size: 32 bits

This register indicates the current state of the Command Parser.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|--|------|---------|
| 31:16 | Graphics Engine Next Command Sequence Number (GE_NEXT_CMD_SEQ_NUM) This field indicates the expect sequence number for the next command. It is initialized from the Graphics Engine Initial Command Sequence Number (GE_INIT_CMD_SEQ_NUM) field in Graphic Engine Control Register (GE_CTRL) and is incremented at the completion of each command. | RO | 0000h |
| 15:12 | RESERVED | RO | - |
| 11:8 | Graphics Engine Error Type (GE_ERROR_TYPE) When the Graphics Engine Error Status (GE_ERROR_STAT) field indicates "bad command specific check", this field indicates the command check that failed. The command specific errors are listed in the respective command descriptions. This field is cleared when the Graphics Engine Restart (GE_RESTART) bit in Graphic Engine Control Register (GE_CTRL) is set. | RO | 0000b |
| 7:4 | Graphics Engine Error Status (GE_ERROR_STAT) When the Graphics Engine Command State (GE_CMD_STATE) field indicates an error condition, this field indicates the reason for the error. This field is cleared when the Graphics Engine Restart (GE_RESTART) bit in Graphic Engine Control Register (GE_CTRL) is set. <ul style="list-style-type: none"> ■ 1 = Invalid command value ■ 2 = Invalid command value ones complement ■ 3 = Bad sequence number ■ 4 = Bad command specific check ■ 5 = Internal data path error 1 ■ 6 = Internal data path error 2 ■ 7 = Internal data path error 3 ■ 8 - 13 reserved ■ 14 = Graphics Engine Abort (GE_ABORT) bit was set while the Command Parser was waiting for a command (Graphics Engine Command State (GE_CMD_STATE) was equal to Idle) ■ 15 = Graphics Engine Abort (GE_ABORT) bit was set while a command was being executed (Graphics Engine Command State (GE_CMD_STATE) was equal to Executing) | RO | 0000b |
| 3 | Graphics Engine Read Master State (GE_RD_MASTER_STATE) Indicates the current state of the Read Command FIFO and internal read data path <ul style="list-style-type: none"> ■ 0 = Idle (no read commands pending and command FIFO empty) ■ 1 = Active | RO | 0b |

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|---|------|---------|
| 2 | <p>Graphics Engine Write Master State (GE_WR_MASTER_STATE) Indicates the current state of the Pixel to QWORD Converter, Data and Command FIFOs and internal write data path.</p> <ul style="list-style-type: none"> ■ 0 = Idle (no pixels being converted, no commands pending and command and data FIFOs empty) ■ 1 = Active <p>Note: When the Graphics Engine Command State (GE_CMD_STATE) field indicates Stalled, the Pixel to QWORD Converter and Data FIFO states are not taken into account.</p> | RO | 0b |
| 1:0 | <p>Graphics Engine Command State (GE_CMD_STATE) Indicates the current state of the Command Parser.</p> <ul style="list-style-type: none"> ■ 0 = Idle ■ 1 = Executing Command ■ 2 = Stalled (Errored) | RO | 00b |

11.12.3 Graphic Engine Command Info Register (GE_CMD_INFO)

Offset: 008h Size: 32 bits

This register reflects the sequence number and command values of the current drawing command.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|---|------|---------|
| 31:16 | <p>Graphics Engine Command Sequence Number (GE_CMD_SEQ_NUM) Indicates the sequence number of the current command. This is valid even if the Graphics Engine Command Sequence Number Enable (GE_CMD_SEQ_EN) bit is cleared.</p> <p>This field is cleared when the command completes.</p> <p>This field is cleared when the Graphics Engine Restart (GE_RESTART) bit in Graphic Engine Control Register (GE_CTRL) is set or when the Graphics Engine Enable (GE_ENABLE) bit in Graphic Engine Control Register (GE_CTRL) changes from low to high.</p> | RO | 0000h |
| 15:8 | <p>Graphics Engine Command Ones Complement (GE_CMD_ONES_COMP) The ones complement of the current command executed. See Graphics Engine Commands for the command values. This is valid even if the Graphics Engine Command Ones Complement Enable (GE_CMD_ONECOMP_EN) bit is cleared.</p> <p>This field is cleared when the command completes.</p> <p>This field is cleared when the Graphics Engine Restart (GE_RESTART) bit in Graphic Engine Control Register (GE_CTRL) is set or when the Graphics Engine Enable (GE_ENABLE) bit in Graphic Engine Control Register (GE_CTRL) changes from low to high.</p> | RO | 00h |
| 7:0 | <p>Graphics Engine Command (GE_CMD) Indicates the current command executed. See Graphics Engine Commands for the command values.</p> <p>This field is cleared when the command completes.</p> <p>This field is cleared when the Graphics Engine Restart (GE_RESTART) bit in Graphic Engine Control Register (GE_CTRL) is set or when the Graphics Engine Enable (GE_ENABLE) bit in Graphic Engine Control Register (GE_CTRL) changes from low to high.</p> | RO | 00h |

11.12.4 Graphic Engine Command Length Register (GE_CMD_LEN)

Address: 00Ch Size: 32 bits

This register reflects the length field of the current drawing command.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|--|------|-----------|
| 31:0 | Graphics Engine Command Length (GE_CMD_LEN) Indicates the length field of the current command. This field is cleared when the command completes. This field is cleared when the Graphics Engine Restart (GE_RESTART) bit in Graphic Engine Control Register (GE_CTRL) is set or when the Graphics Engine Enable (GE_ENABLE) bit in Graphic Engine Control Register (GE_CTRL) changes from low to high. | RO | 00000000h |

11.12.5 Graphic Engine Command Offset Register (GE_CMD_OFFSET)

Offset: 010h Size: 32 bits

This register reflects the number of bytes (including the Command and Length fields) that have been read from the FCT for the current drawing command. It is incremented by four for each FCT read.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|---|------|-----------|
| 31:0 | <p>Graphics Engine Command Offset (GE_CMD_OFFSET) Indicates the number of bytes read from the FCT for the current command. In the event of a command error, this field can be used to determine what part of the command was in error.</p> <p>This field is cleared when the command completes.</p> <p>This field is cleared when the Graphics Engine Restart (GE_RESTART) bit in Graphic Engine Control Register (GE_CTRL) is set or when the Graphics Engine Enable (GE_ENABLE) bit in Graphic Engine Control Register (GE_CTRL) changes from low to high.</p> | RO | 00000000h |

11.12.6 Graphic Engine Previous Command Info Register (GE_PREV_CMD_INFO)

Offset: 014h Size: 32 bits

This register reflects the sequence number and command values of the last completed drawing command.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|---|------|---------|
| 31:16 | Graphics Engine Previous Command Sequence Number (GE_PREV_CMD_SEQ_NUM) Indicates the sequence number of the last completed command. This is valid even if the Graphics Engine Command Sequence Number Enable (GE_CMD_SEQ_EN) bit is cleared. This field is cleared when the Graphics Engine Restart (GE_RESTART) bit in Graphic Engine Control Register (GE_CTRL) is set or when the Graphics Engine Enable (GE_ENABLE) bit in Graphic Engine Control Register (GE_CTRL) changes from low to high. | RO | 0000h |
| 15:8 | Graphics Engine Previous Command Ones Complement (GE_PREV_CMD_ONES_COMP) The ones complement of the last completed command executed. See Graphics Engine Commands for the command values. This is valid even if the Graphics Engine Command Ones Complement Enable (GE_CMD_ONECOMP_EN) bit is cleared. This field is cleared when the Graphics Engine Restart (GE_RESTART) bit in Graphic Engine Control Register (GE_CTRL) is set or when the Graphics Engine Enable (GE_ENABLE) bit in Graphic Engine Control Register (GE_CTRL) changes from low to high. | RO | 00h |
| 7:0 | Graphics Engine Previous Command (GE_PREV_CMD) Indicates the last completed command executed. See Graphics Engine Commands for the command values. This field is cleared when the Graphics Engine Restart (GE_RESTART) bit in Graphic Engine Control Register (GE_CTRL) is set or when the Graphics Engine Enable (GE_ENABLE) bit in Graphic Engine Control Register (GE_CTRL) changes from low to high. | RO | 00h |

11.12.7 Graphic Engine Previous Command Length Register (GE_PREV_CMD_LEN)

Offset: 018h Size: 32 bits

This register reflects the length field of the last completed drawing command.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|--|------|-----------|
| 31:0 | Graphics Engine Previous Command Length (GE_PREV_CMD_LEN) Indicates the length field of the last completed command. This field is cleared when the Graphics Engine Restart (GE_RESTART) bit in Graphic Engine Control Register (GE_CTRL) is set or when the Graphics Engine Enable (GE_ENABLE) bit in Graphic Engine Control Register (GE_CTRL) changes from low to high. | RO | 00000000h |

11.12.8 Graphic Engine Previous Command Offset Register (GE_PREV_CMD_OFFSET)

Offset: 01Ch Size: 32 bits

This register reflects the number of bytes (including the Command and Length fields) that have been read from the for the last completed command. It is incremented by four for each FCT read.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|--|------|-----------|
| 31:0 | <p>Graphics Engine Command Offset (GE_PREV_CMD_OFFSET) Indicates the number of bytes read from the FCT for the last completed command.</p> <p>This field is cleared when the Graphics Engine Restart (GE_RESTART) bit in Graphic Engine Control Register (GE_CTRL) is set or when the Graphics Engine Enable (GE_ENABLE) bit in Graphic Engine Control Register (GE_CTRL) changes from low to high.</p> | RO | 00000000h |

11.12.9 Graphic Engine Chroma Key High Register (GE_CHROMA_KEY_HIGH)

Offset: 020h Size: 32 bits

This register specifies the high compare values for chroma key operations.

A color match is true when it is less than or equal to the chroma high key and greater than or equal to the chroma low key. An single key value can be tested by setting the high and low values to the same value. Each color component is individually tested and the results ANDed, if enabled. In 24 bpp mode, all 8 bits of each component are used. In 16 bpp mode, the upper 5 bits of red and blue and the upper 6 bits of green are used. In 8 bpp mode, only the blue value is used.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|--|------|---------|
| 31:24 | RESERVED | RO | - |
| 23:16 | Graphics Engine Chroma Key Red High (GE_CHROMA_KEY_RED_HIGH) This is the red component of the chroma key high value. | RW | 00h |
| 15:8 | Graphics Engine Chroma Key Green High (GE_CHROMA_KEY_GREEN_HIGH) This is the green component of the chroma key high value. | RW | 00h |
| 7:0 | Graphics Engine Chroma Key Blue High (GE_CHROMA_KEY_BLUE_HIGH) This is the blue component of the chroma key high value. | RW | 00h |

11.12.10 Graphic Engine Chroma Key Low Register (GE_CHROMA_KEY_LOW)

Offset: 024h Size: 32 bits

This register specifies the low compare values for chroma key operations.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|---|------|---------|
| 31:24 | RESERVED | RO | - |
| 23:16 | Graphics Engine Chroma Key Red Low (GE_CHROMA_KEY_RED_LOW) This is the red component of the chroma key low value. | RW | 00h |
| 15:8 | Graphics Engine Chroma Key Green Low (GE_CHROMA_KEY_GREEN_LOW) This is the green component of the chroma key low value. | RW | 00h |
| 7:0 | Graphics Engine Chroma Key Blue Low (GE_CHROMA_KEY_BLUE_LOW) This is the blue component of the chroma key low value. | RW | 00h |

11.13 Error Recovery

11.13.1 USB 3-Strike Error Recovery

USB could have 3 errored transfers (3 strikes) which would cause the host to abort and stop further transfers. This is not the same as when the device initiates a STALL. From the device standpoint, it looks as if host just stopped sending data, there is no other error indication. It is the S/W driver responsibility to recover from this event

State of the H/W:

At the moment of the 3 strikes, the USB FIFOs (URXs and FCT - external to the Graphics Engine)

- 1 - could be empty
- 2 - could have an unfinished command (maybe header and a partial command body)
- 3 - could have one of more whole commands
- 4 - could have one of more whole commands and a partial command (header, partial body)
- 5 - etc.

In the absence of a command related error, the USB FIFOs would eventually drain to empty. The interruption in the data stream could occur at various points.

- 1 - cleanly between commands (the Command Parser is waiting for a new command). [Graphics Engine Command State \(GE_CMD_STATE\)](#) = Idle
- 2- in the command header of a new command (the Command Parser has read the command but is waiting for the length). [Graphics Engine Command State \(GE_CMD_STATE\)](#) = Idle
- 3 - within the body of a command (the Command Parser has read the length and is waiting for the active co-processor to finish) [Graphics Engine Command State \(GE_CMD_STATE\)](#) = Active

Step 1 - The S/W driver may optionally wait until the FCT is empty (via a register external to the Graphics Engine) or until [Graphics Engine Command State \(GE_CMD_STATE\)](#) indicates Stall (a command related error could cause a Graphics Engine Stall with data remaining in the FIFO).

Step 2 - The S/W driver should set [Graphics Engine Abort \(GE_ABORT\)](#) no matter what the above H/W condition. It is acceptable to abort an "in process" command.

- [Graphics Engine Command State \(GE_CMD_STATE\)](#) would then indicate Stall with [Graphics Engine Error Status \(GE_ERROR_STAT\)](#) would indicating either 14 or 15
- [Graphic Engine Command Info Register \(GE_CMD_INFO\)](#), [Graphic Engine Command Length Register \(GE_CMD_LEN\)](#) and [Graphic Engine Command Offset Register \(GE_CMD_OFFSET\)](#) would show the last command that was started or NULL (if [Graphics Engine Error Status \(GE_ERROR_STAT\)](#) = 14)
- [Graphic Engine Previous Command Info Register \(GE_PREV_CMD_INFO\)](#), [Graphic Engine Previous Command Length Register \(GE_PREV_CMD_LEN\)](#) and [Graphic Engine Previous Command Offset Register \(GE_PREV_CMD_OFFSET\)](#) would show the last completed command

Step 3 - The S/W driver should wait until [Graphics Engine Write Master State \(GE_WR_MASTER_STATE\)](#) and [Graphics Engine Read Master State \(GE_RD_MASTER_STATE\)](#) indicate idle.

Step 4 - The S/W driver may optionally clear [Graphics Engine Enable \(GE_ENABLE\)](#).

Step 5 - The S/W driver should flush the USB FIFO by setting the [FCT RX FIFO Reset \(FCT_RX_RESET\)](#) bit of the [FIFO Status Register \(FIFO_STATUS\)](#). This should always be done just in case the Graphics Engine stalled due to command related error in step 1.

Step 6 - The S/W driver should perform a ClearFeature STALL for BULK EP on the host. This will re-sync the USB seq# and Data Toggle between the host and device.



Step 7 - The S/W driver should re-initialize the command sequence number via [Graphics Engine Initial Command Sequence Number \(GE_INIT_CMD_SEQ_NUM\)](#).

Note that the S/W driver should restart commands following the last successful command in order to avoid repeating any commands that have already been executed. This is especially important for Wait For Event commands which specify an event that might have already happened.

Step 8 - The S/W driver should set [Graphics Engine Restart \(GE_RESTART\)](#).

Step 9 - The S/W driver should set [Graphics Engine Enable \(GE_ENABLE\)](#) if it was cleared in step 4.

Step 10 - The S/W driver should restart USB transfers.

Chapter 12 I²C Controller

12.1 I²C Controller Overview

The device includes a dedicated I²C controller with a simple two-wire master/slave serial interface. The I²C controller conforms to the Philips *I²C-Bus Specification*, consisting of a serial data line (I2CSDA), and a serial clock (I2CSCL). These wires carry information between the devices connected to the bus. The source/destination of the I2CSDA and I2CSCL signals are external pins, corresponding to two separate I²C channels. The setting of the [I2C Interface Selection Register \(I2C_SEL\)](#) determines which set of external data and clock pins (I2CSDA0, I2CSCL0 or I2CSDA1, I2CSCL1), corresponding to the I²C channels, will be connected to the I2CSDA and I2CSCL signals. Please refer to [Figure 12.1 I2C Block Diagram on page 268](#) for details.

Each device on the I²C bus, including the UFX7000, is recognized by a unique address and can operate as either a transmitter or receiver, depending on the function of the device. The UFX7000 can be configured as a master, a slave, or both. When in master mode, it is the responsibility of the I²C controller to generate the I2CSCL clock. Both slave and master devices are capable of being either a transmitter or receiver. Thus, four distinct modes are available: slave-receiver, slave-transmitter, master-receiver, and master-transmitter. These modes can be selected via the I²C control and status registers (CSRs).

The I²C controller communicates at the I²C standard mode speed of 100 Kb/s. Because fast mode (400Kb/s) and high-speed mode (3.4Mb/s) devices are downward compatible with the standard mode speed of 100 Kb/s, they are able to communicate with the device's I²C controller on 100 Kb/s I²C bus systems. However, as noted in the *I2C-Bus Specification*, standard mode devices are not upward compatible and should not be incorporated in fast or high-speed mode I²C bus systems.

The I²C controller supports multiple master arbitration. Any I²C device can be attached to an I²C bus and every device can talk with any master, passing information back and forth. There needs to be at least one master on the bus, but multiple masters are allowable by requiring each master to arbitrate for ownership. The I²C controller also supports the 7-bit and 10-bit addressing modes of the I²C specification. These modes can be configured independently for the slave and master via CSRs.

All functions of the I²C are configurable via their respective CSRs and data to be transmitted or received via the I²C interface is accessible here.

The I²C controller contains many interruptible conditions that can be individually masked and cleared via their respective CSRs. When one or more interrupt conditions are met, a single interrupt output I2C_INT is sent to the USB interrupt endpoint. The source of an interrupt can be determined via the interrupt status registers. The I²C interrupts are explained in detail in [Section 12.4.6, "Interrupt Operation"](#).

For more information about the I²C specification, including addressing protocols, transmitting and receiving data protocols, multiple master arbitration and clock synchronization, refer to the Philips *I2C-Bus Specification*.

12.2 I²C Sub-Modules

A top level block diagram of the I²C controller, including its various sub-modules, is provided in [Figure 12.1](#). The I²C controller is responsible for the control and flow of data to/from the I²C serial interface and the generation of interrupts based on configurable conditions. There are 5 main sub-modules of the I²C; the control block, I²C mode logic, I²C data control logic, clock generator, and interrupt controller. Detailed descriptions of each module's functions and various I/O signals are provided in the following sections.

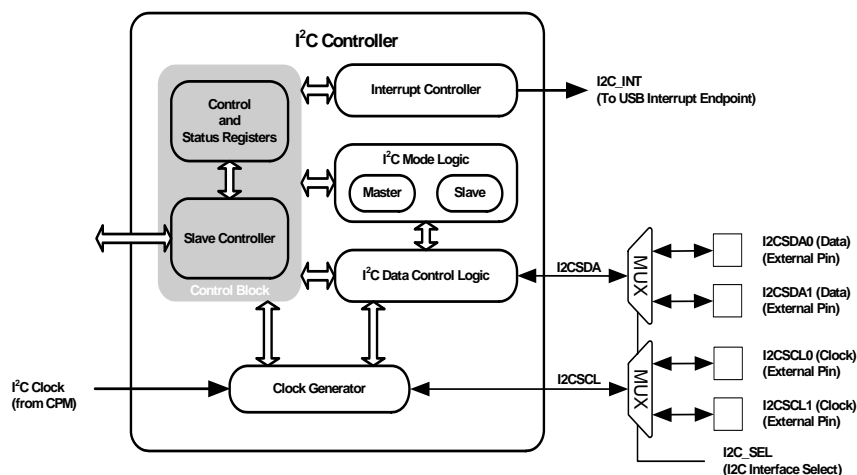


Figure 12.1 I²C Block Diagram

12.2.1 Control Block

The control block consists of a slave controller and a bank of control and status registers. These registers are responsible for all I²C related communication and configuration. Control and status signals from the control block are applied throughout the I²C controller sub-modules. For a detailed description of each I²C register, refer to [Section 12.5, "I2C Register Descriptions"](#).

12.2.2 Interrupt Controller

The interrupt controller block is used to generate an interrupt based upon various configurable conditions within the I²C controller. The I²C interrupt (I2C_INT) indicates to the USB interrupt control endpoint that an interrupt condition occurred. The specific cause of the I²C interrupt can be determined by reading the [Masked Interrupt Status Register \(I2C_MIS\)](#).

Each interrupt can be individually enabled by setting the appropriate mask bit to 1 in the [Interrupt Mask Register \(I2C_INTR_MASK\)](#). Conditions that can be configured to generate interrupts include:

- I²C line special condition detection (START/RESTART, STOP, General Call)
- Rx/Tx conditions (read request, Rx acknowledge error, Tx abort)
- Rx/Tx FIFO buffer conditions (empty, full, threshold level)

Interrupts are clearable via their respective interrupt clear registers. Some interrupts are clearable via hardware only. For more information on how to configure and operate the I²C interrupts, refer to [Section 12.4.6, "Interrupt Operation"](#).

12.2.3 I²C Mode Logic

The I²C mode logic block is responsible for following the I²C protocol of both a slave and master device. This block contains two controllers, a master device controller and a slave device controller. The master controller is responsible for generating the I²C protocol for master transfers, while the slave controller follows the protocol for a slave and monitors the bus for an address match. The master and slave controllers are configured and enabled/disabled via the I²C CSRs. The I²C mode logic sends and receives the I²C data to/from the I²C data control logic block, where it is transported across the I2CSDA line.

12.2.4 I²C Data Control Logic

The I²C data control logic block is responsible for transporting the I²C data between the I2CSDA data line and the I2C mode logic slave and master controllers. The I²C data logic block contains two FIFO buffers, one for storing received data (RX FIFO) and one for storing data to be transmitted (TX FIFO). The RX FIFO is 8-bits wide and has a buffer depth of 16. The TX FIFO is 9-bits wide (1 extra bit for command type) and has a buffer depth of 16.

Shifting logic is also contained within this block. The Rx shift logic receives incoming serialized data from the I2CSDA line and converts it to byte format for loading into the RX FIFO. The Tx shift logic converts the TX FIFO byte formatted data into a serialized format for transmission over the I2CSDA line.

12.2.5 Clock Generator

The clock generator logic block controls and monitors the I2CSCL line and calculates the required timing to do the following:

- Generate the I2CSCL clock when configured as a master
- Check for bus idle
- Generate a START and STOP
- Setup the data and hold the data

The clock generator is configured via the I²C CSR's. For more information on configuring the I2CSCL clock, refer to [Section 12.4.5, "I2CSCL Frequency Configuration"](#).

12.3 I²C Terminology

The I²C specification contains its own unique set of specific terminology. The following terms are used throughout this chapter and are defined as follows:

12.3.1 I²C Bus Terms

The following terms relate to the role of the I²C device and how it interacts with other I²C devices on the bus.

Transmitter - A device that sends data to the bus. A transmitter can either be a device that initiates the data transmission to the bus (a master-transmitter) or responds to a request from the master to send data to the bus (a slave-transmitter).

Receiver - A device that receives data from the bus. A receiver can either be a device that receives data on its own request (a master-receiver) or in response to a request from the master (a slave receiver).

Master - The master initializes a transfer (START command), generates the clock signal (I2CSCL) and terminates the transfer (STOP command). A master can be either a transmitter or a receiver.

Slave - A device addressed by the master. A slave can be either a receiver or transmitter.

12.3.2 I2C Bus Transfer Terms

The following terms are specific to data transfers that occur to/from the I²C bus.

START (RESTART) - Data transfer begins with a START or RESTART condition. The level of the SDA data line changes from high to low, while the SCL clock line remains high. When this occurs, the bus becomes busy.

Note: The RESTART condition of the I²C controller is the same behavior as the REPEAT START condition described in the Philips *I2C-Bus Specification*. START and RESTART conditions are functionally identical.

STOP - Data transfer is terminated by a STOP condition. This occurs when the level on the SDA data line passes from the low state to the high state, while the I2CSCL clock line remains high. When the data transfer has been terminated, the bus is free or idle once again. The bus stays busy if a RESTART is generated instead of a STOP condition.

12.4 I²C Operation

The following sections describe the operation of the I²C controller:

- [Slave Mode Operation](#)
- [Master Mode Operation](#)
- [Prioritizing Slave-Transmit Operation Over Master-Receiver Operation](#)
- [Disabling the I2C Controller](#)
- [I2CSCL Frequency Configuration](#)
- [Interrupt Operation](#)

12.4.1 Slave Mode Operation

The following sections describe the initial configuration, slave-transmitter and slave-receiver operations.

12.4.1.1 Initial Configuration

To use the I²C controller as a slave, perform the following steps:

1. Disable the I²C controller by writing a 0 to bit 0 of the I2C_ENABLE register.
2. Write to the I2C_SAR register (bits 9:0) to set the slave address. This is the address to which the I²C controller responds.
3. Write to the I2C_CON register to specify which type of addressing is supported (7 or 10-bit by setting bit 3) and whether the I²C controller is in slave only (by writing a 0 to bit 6) or master slave mode (by writing a 0 to bit 6 and a 1 to bit 0).

Note: Slaves and masters do not have to be programmed with the same type of addressing 7 or 10-bit address. For instance, a slave can be programmed with 7-bit addressing and a master with 10-bit addressing, and vice versa.

4. Enable the I²C controller by writing a 1 in bit 0 of the I2C_ENABLE register.

12.4.1.2 Slave-Transmitter Operation for a Single Byte

When another I²C master device on the bus addresses the I²C and requests data, the I²C controller acts as a slave transmitter and the following steps occur:

1. The other I²C master device initiates an I²C transfer with an address that matches the slave address in the I2C_SAR register of the I²C controller.

2. The I²C controller acknowledges the sent address and recognizes the direction of the transfer to indicate that it is acting as a slave transmitter.
3. The I²C controller asserts the RD_REQ interrupt (bit 5 of the I2C_RIS register) and holds the I2CSCL line low. It is in a wait state until software responds.

If the RD_REQ interrupt has been masked, due to the I2C_INTR_MASK[5] register (M_RD_REQ bit field) being cleared to 0, then it is recommended that a hardware and/or software timing routine be used to instruct the CPU to perform periodic reads of the I2C_RIS register.

- a. Reads that indicate I2C_RIS[5] (R_RD_REQ bit field) being set to 1 must be treated as the equivalent of the RD_REQ interrupt being asserted.
- b. Software must then act to satisfy the I²C transfer.
- c. The timing interval used should be in the order of 10 times the I2CSCL clock period. For 100 kb/s, the clock period is 10µs. Therefore, a timing interval on the order of 100µs should be used.
4. If there is any data remaining in the TX FIFO before receiving the read request, then the I²C controller asserts a TX_ABRT interrupt (bit 6 of the I2C_RIS register) to flush the old data from the TX FIFO.

If the TX_ABRT interrupt has been masked, due to the I2C_INTR_MASK[6] register (M_TX_ABRT bit field) being set to 0, it is recommended to re-use the timing routine (described in the previous step), or a similar one, to read the I2C_RIS register.

- a. Reads that indicate bit 6 (R_TX_ABRT) being set to 1 must be treated as the equivalent of the TX_ABRT interrupt being asserted.
- b. There is no further action required from software.
- c. The timing interval used should be similar to that described in the previous step for the I2C_RIS[5] register.
5. Software writes the I2C_DATA_CMD register with the data to be written (by writing a 0 in bit 8).
6. Software must clear the RD_REQ and TX_ABRT interrupts (bits 5 and 6, respectively) of the I2C_RIS register before proceeding.

If the RD_REQ and/or TX_ABRT interrupts have been masked, then clearing of the I2C_RIS register will have already been performed when either the R_RD_REQ or R_TX_ABRT bit has been read as 1.

7. The I²C controller releases the I2CSCL and transmits the byte.
8. The master may hold the I²C bus by issuing a RESTART condition or release the bus by issuing a STOP condition.

12.4.1.3 Slave-Receiver Operation for a Single Byte

When another I²C master device on the bus addresses the I²C controller and is sending data, the I²C controller acts as a slave receiver and the following steps occur:

1. The other I²C master device initiates an I²C transfer with an address that matches the I²C controller's slave address in the I2C_SAR register.
2. The I²C controller acknowledges the sent address and recognizes the direction of the transfer to indicate that the I²C controller is acting as a slave receiver.
3. The I²C controller receives the transmitted byte and places it in the receive buffer, assuming there is enough space.
4. The I²C controller asserts the RX_FULL interrupt (I2C_RIS[2] register).

Note: For the RX_FULL interrupt to operate correctly, when 1 byte is received, the I2C_RX_TL register must be written with and/or reset to a value of 0 before the Slave-Receiver operation begins.

If the RX_FULL interrupt has been masked, due to setting I2C_INTR_MASK[2] register to 0 or setting I2C_RX_TL to a value larger than 0, then it is recommended that a timing routine (described in [Section 12.4.1.2, "Slave-Transmitter Operation for a Single Byte"](#)) be implemented for periodic reads of the

I2C_STATUS register. Reads of the I2C_STATUS register, with bit 3 (RFNE) set at 1, must then be treated by software as the equivalent of the RX_FULL interrupt being asserted.

5. Software may read the byte from the I2C_DATA_CMD register (bits 7:0).
6. The other master device may hold the I²C bus by issuing a RESTART condition or release the bus by issuing a STOP condition.

12.4.1.4 Slave Bulk Transmit Mode

In the standard I²C protocol, all transactions are single byte transactions and the programmer responds to a remote master read request by writing one byte into the slave's TX FIFO. When a slave (slave-transmitter) is issued a read request from the remote master (master-receiver), at a minimum there should be one entry placed into the slave-transmitter's TX FIFO. The I²C controller is designed to handle more data in the TX FIFO so that subsequent read requests can take that data without raising an interrupt to get more data. Ultimately, this eliminates the possibility of significant latencies being incurred between raising the interrupt for data each time, had there been a restriction of having only one entry placed in the TX FIFO.

This mode only occurs when the I²C controller is acting as a slave-transmitter. If the slave-transmitter acknowledges the data sent by the remote master and there is no data in the slave's TX FIFO, the slave must hold the I2CSCL line low while it raises the read request interrupt (RD_REQ) and waits for data to be populated in the TX FIFO before it can be sent to the master.

If the RD_REQ interrupt is masked, due to bit 5 (M_RD_REQ) of the I2C_INTR_MASK register being set to 0, then it is recommended that a timing routine be used to activate periodic reads of the I2C_RIS register. Reads of I2C_RIS that return bit 5 (R_RD_REQ) set to 1, must be treated as the equivalent of the RD_REQ interrupt referred to in this section. This timing routine is similar to that described in [Section 12.4.1.2, "Slave-Transmitter Operation for a Single Byte"](#).

The RD_REQ interrupt is raised upon a read request, and like interrupts, must be cleared when exiting the interrupt service handling routine (ISR). The ISR should allow for the writing of 1 byte or more into the TX FIFO. During the transmission of these bytes to the master, if the master acknowledges the last byte, then the slave must re-raise the RD_REQ because the master is looking for more data.

If the programmer knows in advance that the remote master is requesting a packet of n bytes, then when another master addresses the I²C controller and requests data, the TX FIFO can be written with n number bytes and the remote master receives it as a continuous stream of data. For example, the I²C slave controller continues to send data to the remote master as long as the remote master is acknowledging the data sent, and there is data available in the TX FIFO. There is no need to hold the I2CSCL line low or to issue RD_REQ again.

If the remote master is to receive n bytes from the I²C controller but the programmer wrote a number of bytes larger than n to the TX FIFO, when the slave finishes sending the requested n bytes, it clears the TX FIFO and ignores any excess bytes.

There is no interrupt to signify the clearing of the TX FIFO in this example. At the time an ACK/NACK is expected, if a NACK is received, then the remote master has all the data it wants. At this time, a flag is raised within the slave controller to clear the leftover data. The contents of the TX FIFO are cleared at that time. This prevents the master controller within the I²C controller from thinking it has data to send when it does not.

12.4.2 Master Mode Operation

The following sections describe the initial configuration and master transmit and receive operations.

12.4.2.1 Initial Configuration

To use the I²C controller as a master, perform the following steps:

1. Disable the I²C controller by writing 0 to the I2C_ENABLE register.

2. Write to the I2C_SAR register to set the slave address, which is the address to which the I²C controller responds.

Note: Slaves and masters do not have to be programmed with the same type of addressing (7 or 10-bit). For instance, a slave can be programmed with 7-bit addressing and a master with 10-bit addressing, and vice versa.

3. Write to the I2C_TAR register the address of the I²C device to be addressed (bits 9:0). This register also indicates whether a General Call or a START BYTE command is going to be performed by the I²C controller.
4. Enable the I²C controller by writing a 1 in bit 0 of the I2C_ENABLE register.
5. Now write the transfer direction command bit (CMD) and the data to be sent into the I2C_DATA_CMD register. If the I2C_DATA_CMD register is written before the I²C controller is enabled, the data and commands are lost, as the buffers are kept cleared when the I²C controller is disabled. This step generates the START condition and the address byte. Once the I²C controller is enabled and there is data in the TX FIFO, the I²C controller starts reading the data.

12.4.2.2 Master-Transmitter and Master-Receiver Operation

The I²C controller supports switching back and forth between reading and writing dynamically. To transmit data, write the data to be written to the lower byte of the [Rx/Tx Data Buffer and Command Register \(I2C_DATA_CMD\)](#). The CMD bit [8] should be written to 0 for write operations. Subsequently, a read command may be issued by writing "don't cares" to the lower byte of the I2C_DATA_CMD register, and a 1 should be written to the CMD bit. As data is transmitted and received, the transmit and receive buffer status bits and interrupts change. When the I²C controller is enabled by writing 1 in bit 0 of the I2C_ENABLE register and there is data in the TX FIFO, a START is generated. When the TX FIFO becomes empty, a STOP is generated.

12.4.3 Prioritizing Slave-Transmit Operation Over Master-Receiver Operation

In the event that the I2C_DATA_CMD register is written to in order to perform a master-receive operation (I2C_DATA_CMD[8] is set to 1) while a coincident I²C slave-transmit operation occurs in the I²C controller, the component's hardware treats the slave-transmit operation to be of a higher priority. In this type of scenario, the following occurs:

1. The I2C_DATA_CMD write (bit 8 set to 1) is ignored.
2. A TX_ABRT interrupt (if not masked) is generated.
3. I2C_TX_ABRT_SRC[15] (ABRT_SLVRD_INTX) is set to 1.
4. I2C_RIS[6] (TX_ABRT) is set to 1.
5. A RD_REQ interrupt (if not masked) is generated.

To summarize, the write to the I2C_DATA_CMD is ignored, and I²C controller requires servicing as per any normal I²C Slave-Transmit operation. Note that due to the latter, the I²C controller forces I2CSCL low.

12.4.4 Disabling the I²C Controller

To properly disable the I²C controller, follow these steps:

1. Define a timer interval equal to 10 times the I2CSCL clock period. For 100 kb/s, the clock period is 10μs. Therefore, a timing interval on the order of 100μs should be used.
2. Define a maximum time-out parameter, MAX_T_POLL_COUNT, such that if any repeated polling operation exceeds this maximum value, an error is reported.
3. Execute a blocking thread/process/function that prevents any further I²C master transactions from being started by software, but allows any pending transfers to be completed.
4. Create a variable POLL_COUNT and initialize it to zero.

5. Read the I2C_ENABLE_STATUS register and test the I2C_EN bit (bit 0). Increment POLL_COUNT by one. If POLL_COUNT >= MAX_T_POLL_COUNT, exit with the relevant error code. If I2C_EN is 0, then the I²C controller is disabled and the routine may be exited with a relevant success code.
6. If I2C_ENABLE_STATUS[0] is 1, then sleep for the specified timer interval and proceed to the previous step.

12.4.5 I2CSCL Frequency Configuration

When the I²C controller is configured as a master, the *CNT registers must be set before any I²C bus transaction can take place to ensure proper I2CSCL clock timing. The *CNT registers are:

- I2C_SS_SCL_HCNT
- I2C_SS_SCL_LCNT

Setting the I2C_SS_SCL_LCNT register configures the number of I²C Clock cycles that are required for setting the low time of the I2CSCL clock. Setting the I2C_SS_SCL_HCNT register configures the number of I²C Clock cycles that are required for setting the high time of the I2CSCL clock. The recommended values of I2C_SS_SCL_HCNT and I2C_SS_SCL_LCNT should be set for 100Kb/s standard speed communication at the frequency of the input I²C Clock, which is fixed at 2.08 MHz. The recommended values are stated in the text accompanying the definition of these registers.

Note: The count registers can be written only when the I²C interface has been disabled. This is accomplished by clearing bit 0 of the I2C_ENABLE register. Writes at other times have no effect.

12.4.6 Interrupt Operation

The I²C controller contains the ability to generate interrupts based on a variety of conditions within the I²C controller. The I²C interrupt (I2C_INT) indicates to the interrupt control endpoint that an interrupt condition occurred. The specific cause of the I²C interrupt can be determined by reading the [Masked Interrupt Status Register \(I2C_MIS\)](#).

Each interrupt can be individually enabled by setting the appropriate mask bit to 1 in the [Interrupt Mask Register \(I2C_INTR_MASK\)](#). At any time, the status of the interrupt mask can be read from I2C_INTR_MASK. Both the raw interrupt status (prior to masking) and the final interrupt status (after masking) can be read from the [Raw Interrupt Status Register \(I2C_RIS\)](#) and the [Masked Interrupt Status Register \(I2C_MIS\)](#) respectively. Conditions that can be configured to generate interrupts include:

- I²C line special condition detection (START/RESTART, STOP, General Call)
- Rx/Tx conditions (read request, Rx acknowledge error, Tx abort)
- Rx/Tx FIFO buffer conditions (empty, full, threshold level)

For a detailed explanation of each configurable interrupt source, refer to the [Raw Interrupt Status Register \(I2C_RIS\)](#) in [Section 12.5, "I2C Register Descriptions"](#).

Upon detection of an interrupt event, as configured by in I2C_INTR_MASK, the I2C_INT interrupt is generated. The I2C_INT signal is sent to the USB interrupt endpoint for further processing. If the interrupt is software clearable, it can be cleared by reading its corresponding interrupt clear register, or the [Clear Interrupt Register \(I2C_CLR_INTR\)](#) which clears all software interrupts. If the interrupt is clearable by hardware only, the interrupt will be cleared automatically when the interrupt conditions have changed. All I²C interrupts are generated by hardware. The majority of interrupt bits are cleared by software, however some interrupt bits are cleared by hardware only. [Table 12.1](#) shows how each I²C interrupt is cleared. For interrupts that are cleared by software, the corresponding clear interrupt registers that will clear the interrupt are listed. Interrupts that are cleared by hardware will be specified as such.

Table 12.1 Clearing Method of I2C Interrupts

| INTERRUPT BIT FIELD | CLEARING METHOD |
|---------------------|-----------------------------------|
| GEN_CALL | I2C_CLR_GEN_CALL or I2C_CLR_INTR |
| START_DET | I2C_CLR_START_DET or I2C_CLR_INTR |
| STOP_DET | I2C_CLR_STOP_DET or I2C_CLR_INTR |
| ACTIVITY | I2C_CLR_ACTIVITY or Hardware |
| RX_DONE | I2C_CLR_RX_DONE or I2C_CLR_INTR |
| TX_ABRT | I2C_CLR_TX_ABRT or I2C_CLR_INTR |
| RD_REQ | I2C_CLR_RD_REQ or I2C_CLR_INTR |
| TX_EMPTY | Hardware |
| TX_OVER | I2C_CLR_TX_OVER or I2C_CLR_INTR |
| RX_FULL | Hardware |
| RX_OVER | I2C_CLR_RX_OVER or I2C_CLR_INTR |
| RX_UNDER | I2C_CLR_RX_UNDER or I2C_CLR_INTR |

Note: Refer to bit 9 of the [Transmit Abort Source Register \(I2C_TX_ABRT_SRC\)](#) for an exception to clearing the TX_ABRT interrupt.

Note: The ACTIVITY interrupt can be cleared via the [Clear ACTIVITY Interrupt Register \(I2C_CLR_ACTIVITY\)](#) but will also be automatically cleared by hardware when the I²C controller is disabled and there is no further activity on the bus.

12.5 I²C Register Descriptions

This section describes the various I²C control and status registers. Some registers may be written only when the I²C is disabled, as programmed by the I2C_ENABLE register. Software should not disable the I²C while it is active. If the I²C is in the process of transmitting when it is disabled, it stops and deletes the contents of the transmit buffer after the current transfer is complete. In slave mode, the I²C continues receiving until the remote master aborts the transfer, in which case the I²C will then be disabled. Registers that cannot be written to when the I²C is enabled are indicated in their descriptions.

[Table 12.2](#) lists the control and status registers of the I²C controller and their corresponding addresses. All I²C control and status registers are reset to their default values on assertion of a chip-level reset. I²C registers must be read and written using naturally aligned 32-bit transfers.

Table 12.2 I²C Register Map

| ADDRESS | SYMBOL | REGISTER NAME |
|---------|-------------|---|
| 1000h | I2C_CONTROL | Control Register, Section 12.5.1 |
| 1004h | I2C_TAR | Target Address Register, Section 12.5.2 |
| 1008h | I2C_SAR | Slave Address Register, Section 12.5.3 |
| 100Ch | RESERVED | Reserved for future expansion |

Table 12.2 I²C Register Map (continued)

| ADDRESS | SYMBOL | REGISTER NAME |
|---------------|-------------------|--|
| 1010h | I2C_DATA_CMD | Rx/Tx Data Buffer and Command Register, Section 12.5.4 |
| 1014h | I2C_SCL_HCNT | SCL Clock High Count Register, Section 12.5.5 |
| 1018h | I2C_SCL_LCNT | SCL Clock Low Count Register, Section 12.5.6 |
| 101Ch - 1028h | RESERVED | Reserved for future expansion |
| 102Ch | I2C_MIS | Masked Interrupt Status Register, Section 12.5.7 |
| 1030h | I2C_INTR_MASK | Interrupt Mask Register, Section 12.5.8 |
| 1034h | I2C_RIS | Raw Interrupt Status Register, Section 12.5.9 |
| 1038h | I2C_RX_TL | Receive FIFO Threshold Register, Section 12.5.10 |
| 103Ch | I2C_TX_TL | Transmit FIFO Threshold Register, Section 12.5.11 |
| 1040h | I2C_CLR_INTR | Clear Interrupt Register, Section 12.5.12 |
| 1044h | I2C_CLR_RX_UNDER | Clear RX_UNDER Interrupt Register, Section 12.5.13 |
| 1048h | I2C_CLR_RX_OVER | Clear RX_OVER Interrupt Register, Section 12.5.14 |
| 104Ch | I2C_CLR_TX_OVER | Clear TX_OVER Interrupt Register, Section 12.5.15 |
| 1050h | I2C_CLR_RD_REQ | Clear RD_REQ Interrupt Register, Section 12.5.16 |
| 1054h | I2C_CLR_TX_ABRT | Clear TX_ABRT Interrupt Register, Section 12.5.17 |
| 1058h | I2C_CLR_RX_DONE | Clear RX_DONE Interrupt Register, Section 12.5.18 |
| 105Ch | I2C_CLR_ACTIVITY | Clear ACTIVITY Interrupt Register, Section 12.5.19 |
| 1060h | I2C_CLR_STOP_DET | Clear STOP_DET Interrupt Register, Section 12.5.20 |
| 1064h | I2C_CLR_START_DET | Clear START_DET Interrupt Register, Section 12.5.21 |
| 1068h | I2C_CLR_GEN_CALL | Clear GEN_CALL Interrupt Register, Section 12.5.22 |
| 106Ch | I2C_ENABLE | Enable Register, Section 12.5.23 |
| 1070h | I2C_STATUS | Status Register, Section 12.5.24 |
| 1074h | I2C_TX_FLR | Transmit FIFO Level Register, Section 12.5.25 |
| 1078h | I2C_RX_FLR | Receive FIFO Level Register, Section 12.5.26 |
| 107Ch | RESERVED | Reserved for future expansion |
| 1080h | I2C_TX_ABRT_SRC | Transmit Abort Source Register, Section 12.5.27 |
| 1084h - 1090h | RESERVED | Reserved for future expansion |
| 1094h | I2C_SDA_SETUP | SDA Setup Register, Section 12.5.28 |
| 1098h | I2C_ACK_GEN_CALL | ACK General Call Register, Section 12.5.29 |
| 109Ch | I2C_ENABLE_STATUS | Enable Status Register, Section 12.5.30 |
| 10A0h - 10FCh | RESERVED | Reserved for future expansion |
| 1100h | I2C_ACCESS | Access Module Register, Section 12.5.31 |

Table 12.2 I²C Register Map (continued)

| ADDRESS | SYMBOL | REGISTER NAME |
|---------------|-----------------|--|
| 1104h | I2C_PIN | Pin Control Register, Section 12.5.32 |
| 1108h - 110Ch | RESERVED | Reserved for future expansion |
| 1110h | I2C_SEQ_DATA_0 | Sequential Data 0 Register, Section 12.5.33 |
| 1114h | I2C_SEQ_DATA_1 | Sequential Data 1 Register, Section 12.5.34 |
| 1118h | I2C_SEQ_DATA_2 | Sequential Data 2 Register, Section 12.5.35 |
| 111Ch | I2C_SEQ_DATA_3 | Sequential Data 3 Register, Section 12.5.36 |
| 1120h | I2C_SEQ_DATA_4 | Sequential Data 4 Register, Section 12.5.37 |
| 1124h | I2C_SEQ_DATA_5 | Sequential Data 5 Register, Section 12.5.38 |
| 1128h | I2C_SEQ_DATA_6 | Sequential Data 6 Register, Section 12.5.39 |
| 112Ch | I2C_SEQ_DATA_7 | Sequential Data 7 Register, Section 12.5.40 |
| 1130h | I2C_SEQ_DATA_8 | Sequential Data 8 Register, Section 12.5.41 |
| 1134h | I2C_SEQ_DATA_9 | Sequential Data 9 Register, Section 12.5.42 |
| 1138h | I2C_SEQ_DATA_10 | Sequential Data 10 Register, Section 12.5.43 |
| 113Ch | I2C_SEQ_DATA_11 | Sequential Data 11 Register, Section 12.5.44 |
| 1140h | I2C_SEQ_DATA_12 | Sequential Data 12 Register, Section 12.5.45 |
| 1144h | I2C_SEQ_DATA_13 | Sequential Data 13 Register, Section 12.5.46 |
| 1148h | I2C_SEQ_DATA_14 | Sequential Data 14 Register, Section 12.5.47 |
| 114Ch | I2C_SEQ_DATA_15 | Sequential Data 15 Register, Section 12.5.48 |

12.5.1 Control Register (I2C_CONTROL)

Offset: 0000h Size: 32 bits

This read/write register is responsible for enabling or disabling the master and slave modes as well as the selection of 7 or 10-bit addressing for each mode. The ability to send RESTART conditions when acting as a master is also selectable here. This register can be written only when the I²C interface has been disabled. This is accomplished by clearing bit 0 of the I2C_ENABLE register.

See the bit descriptions below for additional information on these modes.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|--|------|---------|
| 31:7 | RESERVED | RO | - |
| 6 | Slave Mode (I2C_SLAVE_DISABLE) This bit controls the enabling and disabling of the I ² C slave mode. If this bit is set, the I ² C functions only as a master and does not perform any action that requires a slave. The slave mode is enabled (set to 0) on reset by default. This bit needs only to be written after reset if slave mode is to be disabled. 0: Slave enabled 1: Slave disabled | R/W | 0b |
| 5 | RESTART Enable (I2C_RESTART_EN) This bit determines whether RESTART conditions may be sent when acting as a master. 0: RESTART disabled 1: RESTART enabled When RESTART is disabled, the master is prohibited from performing the following functions: <ul style="list-style-type: none"> ■ Change direction within a transfer (split) ■ Send a START byte ■ Combined format transfers in 7-bit addressing modes ■ Read operation with a 10-bit address ■ Send multiple bytes per transfer By replacing RESTART conditions with a STOP and a subsequent START condition, split operations are broken down into multiple I ² C transfers. If the above operations are performed, it will result in setting bit 6 (TX_ABRT) of the I2C_RIS register. Note: Some older slaves do not support handling RESTART conditions. However, RESTART conditions are used in several I ² C operations. | R/W | 1b |
| 4 | Master Address Mode (I2C_10BITADDR_MASTER) This bit controls the addressing mode of the I ² C when acting as a master. 0: 7-bit addressing 1: 10-bit addressing | R/W | 1b |
| 3 | Slave Address Mode (I2C_10BITADDR_SLAVE) This bit controls the addressing mode of the I ² C when acting as a slave. In 7-bit addressing mode, the I ² C responds only to 7-bit addressing transfers that match the lower 7 bits of the I2C_SAR register. In 10-bit addressing mode, the I ² C responds only to 10-bit addressing transfers that match the full 10 bits of the I2C_SAR register. 0: 7-bit addressing 1: 10-bit addressing | R/W | 1b |

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|--|------|---------|
| 2:1 | Internal configuration only (INT_CFG_ONLY) These bits must always be written as 01b. | R/W | 01b |
| 0 | Master Mode (MASTER_MODE) This bit controls the enabling and disabling of the I ² C master mode. 0: Master disabled 1: Master enabled | R/W | 1b |

12.5.2 Target Address Register (I2C_TAR)

Offset: 0004h Size: 32 bits

This read/write register contains the target address for a master I²C transmission and controls whether a START byte or a General Call will be transmitted.

This register can be written only when the I²C interface has been disabled. This is accomplished by clearing bit 0 of the I2C_ENABLE register. Writes at other times have no effect.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|--|------|---------|
| 31:12 | RESERVED | RO | - |
| 11 | Special Command Selection (SPECIAL) This bit indicates whether a General Call or START byte command will be performed. 0: Ignore bit 10 (GC_OR_START) and use I2C_TAR normally 1: Perform special I ² C command as specified in GC_OR_START bit | R/W | 0b |
| 10 | General Call or START BYTE Selection (GC_OR_START) If bit 11 (SPECIAL) is set to 1, this bit indicates whether a General Call or START byte command is to be performed by the I ² C. 0: General Call Address After issuing a General Call, only writes may be performed. Attempting to issue a read command results in setting bit 6 (TX_ABRT) of the I2C_RIS register. The I ² C remains in General Call mode until the SPECIAL bit is cleared. 1: START byte | R/W | 0b |
| 9:0 | Target Address (I2C_TAR) This field represents the target address for any master transaction. When transmitting a General Call, these bits are ignored. To generate a START byte, the CPU needs to write only once into this field. If the I2C_TAR and I2C_SAR are the same value, loopback exists. However, the FIFOs are shared between the master and slave, so full loopback is not feasible. Only one direction loopback mode is supported (simplex), not duplex. A master cannot transmit to itself, it can transmit only to a slave. | R/W | 055h |

12.5.3 Slave Address Register (I2C_SAR)

Offset: 0008h Size: 32 bits

This read/write register contains the slave address used for when the I²C is operating as a slave.

This register can be written only when the I²C interface has been disabled. This is accomplished by clearing bit 0 of the I2C_ENABLE register. Writes at other times have no effect.

Note: The correct operation of the device is not guaranteed if I2C_SAR or I2C_TAR are programmed to the reserved address values of 00h, 07h, or 78h to 7Fh. Refer to the Philips *I2C_Specifications* for a complete list of these reserved values.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|--|------|---------|
| 31:10 | RESERVED | RO | - |
| 9:0 | Slave Address (I2C_SAR) This field represents the slave address of the I ² C slave controller. For 7-bit addressing, only I2C_SAR[6:0] is used. | R/W | 055h |

12.5.4 Rx/Tx Data Buffer and Command Register (I2C_DATA_CMD)

Offset: 0010h Size: 32 bits

This read/write register is used to read or write the RX FIFO and TX FIFO respectively. The CPU should write to this register when filling the TX FIFO and read from this register when retrieving bytes from the RX FIFO. When writing, this register has 9 valid bits comprised of the command bit and 8 data bits. When reading, the register only contains the 8 data bits.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|--|------|---------|
| 31:9 | RESERVED | RO | - |
| 8 | <p>Command Bit (CMD) This bit controls whether a read or a write is performed. This bit controls the direction only when the I²C is acting as a master, not a slave.</p> <p>0: Write 1: Read</p> <p>When a command is entered in the TX FIFO, this bit distinguishes the read and write commands. In slave-receiver mode, this bit is a “don’t care” because writes to this register are not required. In slave-transmitter mode, a 0 indicates that the CPU data written to the DAT field is to be transmitted.</p> <p>Attempting to perform a read operation after a General Call command has been sent will result in a TX_ABRT interrupt (bit 6 of the I2C_RIS register), unless the SPECIAL bit (bit 11 of the I2C_TAR register) has been cleared.</p> <p>If a 1 is written to this bit after receiving a RD_REQ interrupt, then a TX_ABRT interrupt will occur.</p> <p>Note: It is possible that while attempting a master I²C read transfer, a RD_REQ interrupt may have occurred simultaneously due to a remote I²C master addressing the I²C controller. In this type of scenario, the I²C controller ignores the I2C_DATA_CMD write, generates a TX_ABRT interrupt, and waits to service the RD_REQ interrupt. For more details, see Section 12.4, "I2C Operation".</p> | R/W | 0b |
| 7:0 | <p>Tx/Rx Data (DAT) This field contains the data to be transmitted or received on the I²C bus. If writing to this register to perform a read, bits 7:0 (DAT) are ignored by the I²C controller. However, when this register is read, this field will return the value of the data received on the I²C interface.</p> | R/W | 00h |

12.5.5 SCL Clock High Count Register (I2C_SCL_HCNT)

Offset: 0014h Size: 32 bits

This read/write register sets the I2CSCL clock high-period count. The I2CSCL clock is based off the I²C Clock (2.08 MHz) from the clock factory. Based on this frequency, it is recommended that the high-period count value be set to 0009h. For more information on configuring the I2CSCL clock, refer to [Section 12.4.5, "I2CSCL Frequency Configuration"](#).

This register can be written only when the I²C interface has been disabled. This is accomplished by clearing bit 0 of the I2C_ENABLE register. Writes at other times have no effect.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|--|------|---------------------------|
| 31:16 | RESERVED | RO | - |
| 15:0 | I²C SCL High Count Value (I2C_SCL_HCNT) This field sets the high-period count value for the I2CSCL clock. | R/W | Note 12.1 |

Note 12.1 While the default value of this field is 0008h, it is recommended that this field be set to 0009h for proper operation.

12.5.6 SCL Clock Low Count Register (I2C_SCL_LCNT)

Offset: 0018h Size: 32 bits

This read/write register sets the I2CSCL clock low-period count. The I2CSCL clock is based off the I²C Clock (2.08 MHz) from the clock factory. Based on this frequency, it is recommended that the low-period count value be set to 000Ah. For more information on configuring the I2CSCL clock, refer to [Section 12.4.5, "I2CSCL Frequency Configuration"](#).

This register can be written only when the I²C interface has been disabled. This is accomplished by clearing bit 0 of the I2C_ENABLE register. Writes at other times have no effect.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|--|------|---------------------------|
| 31:16 | RESERVED | RO | - |
| 15:0 | I²C SCL Low Count Value (I2C_SCL_LCNT) This field sets the low-period count value for the I2CSCL clock. | R/W | Note 12.2 |

Note 12.2 The default value and the recommended value of this field is 000Ah.

12.5.7 Masked Interrupt Status Register (I2C_MIS)

Offset: 002Ch Size: 32 bits

This read-only register indicates the masked interrupt status of the I²C interrupts. The masked interrupt status is the status of each corresponding interrupt as masked by the [Interrupt Mask Register \(I2C_INTR_MASK\)](#). A bit value of 1 reflects that an interrupt trigger condition has been detected. A bit value of 0 indicates that the corresponding condition has not been met, or is masked. The unmasked raw versions of these bits are available in the I2C_RIS register. An interrupt is cleared by reading the corresponding interrupt clear register, or the [Clear Interrupt Register \(I2C_CLR_INTR\)](#). For more information on configuring the I²C interrupts, refer to [Section 12.4.6, "Interrupt Operation"](#). At reset, all bits are cleared.

Note: See [Raw Interrupt Status Register \(I2C_RIS\)](#) for a detailed description of each of these bits.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|---|------|---------|
| 31:12 | RESERVED | RO | - |
| 11 | General Call Masked Interrupt Status (R_GEN_CALL) | RO | 0b |
| 10 | START Detection Masked Interrupt Status (R_START_DET) | RO | 0b |
| 9 | STOP Detection Masked Interrupt Status (R_STOP_DET) | RO | 0b |
| 8 | Activity Masked Interrupt Status (R_ACTIVITY) | RO | 0b |
| 7 | Receive Complete Masked Interrupt Status (R_RX_DONE) | RO | 0b |
| 6 | Transmit Abort Masked Interrupt Status (R_TX_ABRT) | RO | 0b |
| 5 | Read Request Masked Interrupt Status (R_RD_REQ) | RO | 0b |
| 4 | TX FIFO Empty Masked Interrupt Status (R_TX_EMPTY) | RO | 0b |
| 3 | TX FIFO Over Masked Interrupt Status (R_TX_OVER) | RO | 0b |
| 2 | RX FIFO Full Masked Interrupt Status (R_RX_FULL) | RO | 0b |
| 1 | RX FIFO Over Masked Interrupt Status (R_RX_OVER) | RO | 0b |
| 0 | RX FIFO Under Masked Interrupt Status (R_RX_UNDER) | RO | 0b |

12.5.8 Interrupt Mask Register (I2C_INTR_MASK)

Offset: 0030h Size: 32 bits

This read/write register is responsible for masking the various I²C interrupts. A value of 1 enables the corresponding interrupt for output to the I2C_INT interrupt, while a value of 0 disables the corresponding interrupt. The bits of this register mask their corresponding interrupt status bits in the [Masked Interrupt Status Register \(I2C_MIS\)](#). An interrupt is cleared by reading the corresponding interrupt clear register, or the [Clear Interrupt Register \(I2C_CLR_INTR\)](#). For more information on configuring the I²C interrupts, refer to [Section 12.4.6, "Interrupt Operation"](#).

Note: See [Raw Interrupt Status Register \(I2C_RIS\)](#) for a detailed description of each of these bits.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|--|------|---------|
| 31:12 | RESERVED | RO | - |
| 11 | General Call Interrupt Mask (M_GEN_CALL) | R/W | 1b |
| 10 | START Detection Interrupt Mask (M_START_DET) | R/W | 0b |
| 9 | STOP Detection Interrupt Mask (M_STOP_DET) | R/W | 0b |
| 8 | Activity Interrupt Mask (M_ACTIVITY) | R/W | 0b |
| 7 | Receive Complete Interrupt Mask (M_RX_DONE) | R/W | 1b |
| 6 | Transmit Abort Interrupt Mask (M_TX_ABRT) | R/W | 1b |
| 5 | Read Request Interrupt Mask (M_RD_REQ) | R/W | 1b |
| 4 | TX FIFO Empty Interrupt Mask (M_TX_EMPTY) | R/W | 1b |
| 3 | TX FIFO Over Interrupt Mask (M_TX_OVER) | R/W | 1b |
| 2 | RX FIFO Full Interrupt Mask (M_RX_FULL) | R/W | 1b |
| 1 | RX FIFO Over Interrupt Mask (M_RX_OVER) | R/W | 1b |
| 0 | RX FIFO Under Interrupt Mask (M_RX_UNDER) | R/W | 1b |

12.5.9 Raw Interrupt Status Register (I2C_RIS)

Offset: 0034h Size: 32 bits

This read-only register indicates the raw interrupt status of the I²C interrupts. The raw interrupt status is the status of each corresponding interrupt regardless if it has been masked by the [Interrupt Mask Register \(I2C_INTR_MASK\)](#). A bit value of 1 reflects that an interrupt trigger condition has been detected. A bit value of 0 indicates that the corresponding condition has not been met. The masked versions of these bits are available in the [Masked Interrupt Status Register \(I2C_MIS\)](#). An interrupt is cleared by reading the corresponding interrupt clear register, or the [Clear Interrupt Register \(I2C_CLR_INTR\)](#). For more information on configuring the I²C interrupts, refer to [Section 12.4.6, "Interrupt Operation"](#). At reset, all bits are cleared.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|---|------|---------|
| 31:12 | RESERVED | RO | - |
| 11 | General Call Raw Interrupt Status (GEN_CALL) This interrupt bit is set only when a General Call address is received and is acknowledged. It stays set until it is cleared either by disabling the I ² C controller (set bit 0 of I2C_ENABLE = 0) or when the CPU reads bit 0 of the I2C_CLR_GEN_CALL register. The I ² C controller stores the received data in the Rx buffer. | RO | 0b |
| 10 | START Detection Raw Interrupt Status (START_DET) This interrupt bit indicates whether a START or RESTART condition has occurred on the I ² C interface regardless of whether the I ² C controller is operating in slave or master mode. Note: There is no status bit for a RESTART condition because it is detected as a normal start condition. The I ² C protocol does not care whether it is a START or RESTART because both conditions start from the IDLE state and send the message to all the slaves on the bus. | RO | 0b |
| 9 | STOP Detection Raw Interrupt Status (STOP_DET) This interrupt bit indicates whether a STOP condition has occurred on the I ² C interface regardless of whether the I ² C controller is operating in slave or master mode. | RO | 0b |
| 8 | Activity Raw Interrupt Status (ACTIVITY) This bit captures I ² C activity and stays set until it is cleared. There are four ways to clear it: <ul style="list-style-type: none"> ■ Disabling the I²C controller (set bit 0 of I2C_ENABLE = 0) ■ Reading the I2C_CLR_ACTIVITY register ■ Reading the I2C_CLR_INTR register ■ Chip-level reset Once this bit is set, it stays set unless one of the four methods is used to clear it. Even if the I ² C controller is idle, it remains set until cleared, indicating that there was activity on the bus. | RO | 0b |
| 7 | Receive Complete Raw Interrupt Status (RX_DONE) When the I ² C controller is acting as a slave transmitter, this bit is set to 1 if the master does not acknowledge a transmitted byte. Acknowledgement occurs on the last byte of the transmission, indicating that the transmission is done. | RO | 0b |

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|--|------|---------|
| 6 | <p>Transmit Abort Raw Interrupt Status (TX_ABRT)</p> <p>This bit indicates if the I²C controller, acting as an I²C transmitter, is unable to complete the intended actions on the contents of the transmit FIFO. This situation can occur both as an I²C master or an I²C slave, and is referred to as a “transmit abort”.</p> <p>When this bit is set to 1, the I2C_TX_ABRT_SRC register indicates the reason why the transmit abort has taken place.</p> <p>Note: The I²C controller flushes/empties all of its FIFOs (transmit and receive) whenever this bit is set to indicate the occurrence of a transmit abort.</p> | RO | 0b |
| 5 | <p>Read Request Raw Interrupt Status (RD_REQ)</p> <p>This bit is set to 1 when the I²C controller is acting as a slave and another I²C master is attempting to read data from it. The I²C controller holds the I²C bus in a wait state (I2CSCL=0) until this interrupt is serviced, which means that the slave has been addressed by a remote master that is asking for data to be transferred. The processor must respond to this interrupt and then write the requested data to the I2C_DATA_CMD register. This interrupt bit is cleared upon a read of bit 0 in the I2C_CLR_RD_REQ register.</p> | RO | 0b |
| 4 | <p>TX FIFO Empty Raw Interrupt Status (TX_EMPTY)</p> <p>This bit is set to 1 when the transmit buffer is at or below the threshold value set in the I2C_TX_TL register. It is automatically cleared by hardware when the buffer level goes above the threshold. When the I2C_ENABLE bit 0 is cleared, disabling the I²C controller, the TX FIFO is flushed and held in reset. In this case the TX FIFO appears as if it has no data within it, so this bit is set to 1, provided there is still activity in the master or slave controllers. When there is no longer activity, this interrupt bit is cleared.</p> | RO | 0b |
| 3 | <p>TX FIFO Over Raw Interrupt Status (TX_OVER)</p> <p>This bit is set during a transmit if the transmit buffer is filled to the maximum TX FIFO buffer depth of 16 and the processor attempts to issue another I²C command by writing to the I2C_DATA_CMD register. When the I²C controller is disabled (I2C_ENABLE[0]=0), this bit keeps its level until the master or slave controllers activity is completed. When there is no longer activity, this interrupt bit is cleared.</p> | RO | 0b |
| 2 | <p>RX FIFO Full Raw Interrupt Status (RX_FULL)</p> <p>This bit is set when the receive buffer reaches or goes above the RX_TL threshold in the I2C_RX_TL register. It is automatically cleared by hardware when the buffer level returns below the threshold. If the I²C controller is disabled (I2C_ENABLE[0]=0), the RX FIFO is flushed and held in reset. therefore the RX FIFO is not full. Therefore, this bit is cleared once the I2C_ENABLE bit 0 is programmed with a 0, regardless of any I²C activity that continues.</p> | RO | 0b |
| 1 | <p>RX FIFO Over Raw Interrupt Status (RX_OVER)</p> <p>This bit is set if the receive buffer is completely filled to the maximum RX FIFO buffer depth of 16 and an additional byte is received from an external I²C device. The I²C controller acknowledges this additional received byte, but any data bytes received after the RX FIFO is full are lost. If the I²C controller is disabled (I2C_ENABLE[0]=0), this bit keeps its level until the master or slave controller activity is completed. When there is no longer activity, this interrupt bit is cleared.</p> | RO | 0b |
| 0 | <p>RX FIFO Under Raw Interrupt Status (RX_UNDER)</p> <p>This bit is set if the processor attempts to read the RX FIFO when it is empty by reading from the I2C_DATA_CMD register. If the I²C controller is disabled (I2C_ENABLE[0]=0), this bit keeps its level until the master or slave controller activity is completed. When there is no longer activity, this interrupt is cleared.</p> | RO | 0b |

12.5.10 Receive FIFO Threshold Register (I2C_RX_TL)

Offset: 0038h Size: 32 bits

This read/write register controls the level of entries (or above) into the RX FIFO that triggers the RX_FULL interrupt (bit 2 of the I2C_RIS register). The valid range is 0 to 15, since the total buffer depth of the RX FIFO is 16. A value of 0 will set the threshold for 1 entry, while a value of 15 will set the threshold for 16 entries. If an attempt is made to write a value larger than the depth of the buffer, the actual value set will be 15, the maximum depth of the buffer.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|--|------|---------|
| 31:4 | RESERVED | RO | - |
| 3:0 | RX FIFO Threshold Level (RX_TL) This field is used to set the level of entries into the RX FIFO that triggers the RX_FULL interrupt. | R/W | 7h |

12.5.11 Transmit FIFO Threshold Register (I2C_TX_TL)

Offset: 003Ch Size: 32 bits

This read/write register controls the level of entries (or below) into the TX FIFO that triggers the TX_EMPTY interrupt (bit 4 of the I2C_RIS register). The valid range is 0 to 15, since the total buffer depth of the TX FIFO is 16. A value of 0 will set the threshold for 1 entry, while a value of 15 will set the threshold for 16 entries. If an attempt is made to write a value larger than the depth of the buffer, the actual value set will be 15, the maximum depth of the buffer.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|---|------|---------|
| 31:4 | RESERVED | RO | - |
| 3:0 | TX FIFO Threshold Level (TX_TL) This field is used to set the level of entries into the TX FIFO that triggers the TX_EMPTY interrupt. | R/W | 7h |

12.5.12 Clear Interrupt Register (I2C_CLR_INTR)

Offset: 0040h Size: 32 bits

When read, this read-only register clears the combined interrupt I2C_INT, all individual interrupts, and the I2C_ABRT_SRC register.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|--|------|---------|
| 31:1 | RESERVED | RO | - |
| 0 | Clear Interrupts (CLR_INTR) When read, clears the combined interrupt I2C_INT, all individual interrupts, and the I2C_TX_ABRT_SRC register. Note: This bit does not clear hardware clearable interrupts. Refer to bit 9 of the I2C_ABRT_SRC register for an exception to clearing I2C_TX_ABRT_SRC. Refer to Table 12.1 for a full list of hardware and software clearable interrupts. | RO | 0b |

12.5.13 Clear RX_UNDER Interrupt Register (I2C_CLR_RX_UNDER)

Offset: 0044h Size: 32 bits

When read, this read-only register clears the RX_UNDER interrupt.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|--|------|---------|
| 31:1 | RESERVED | RO | - |
| 0 | Clear RX_UNDER Interrupt (CLR_RX_UNDER) Read this bit to clear the RX_UNDER interrupt (bit 0) of the I2C_RIS register. | RO | 0b |

12.5.14 Clear RX_OVER Interrupt Register (I2C_CLR_RX_OVER)

Offset: 0048h Size: 32 bits

When read, this read-only register clears the RX_OVER interrupt.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|---|------|---------|
| 31:1 | RESERVED | RO | - |
| 0 | Clear RX_OVER Interrupt (CLR_RX_OVER) Read this bit to clear the RX_OVER interrupt (bit 1) of the I2C_RIS register. | RO | 0b |

12.5.15 Clear TX_OVER Interrupt Register (I2C_CLR_TX_OVER)

Offset: 004Ch Size: 32 bits

When read, this read-only register clears the TX_OVER interrupt.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|---|------|---------|
| 31:1 | RESERVED | RO | - |
| 0 | Clear TX_OVER Interrupt (CLR_TX_OVER) Read this bit to clear the TX_OVER interrupt (bit 3) of the I2C_RIS register. | RO | 0b |

12.5.16 Clear RD_REQ Interrupt Register (I2C_CLR_RD_REQ)

Offset: 0050h Size: 32 bits

When read, this read-only register clears the RD_REQ interrupt.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|--|------|---------|
| 31:1 | RESERVED | RO | - |
| 0 | Clear RD_REQ Interrupt (CLR_RD_REQ) Read this bit to clear the RD_REQ interrupt (bit 5) of the I2C_RIS register. | RO | 0b |

12.5.17 Clear TX_ABRT Interrupt Register (I2C_CLR_TX_ABRT)

Offset: 0054h Size: 32 bits

When read, this read-only register clears the TX_ABRT interrupt.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|---|------|---------|
| 31:1 | RESERVED | RO | - |
| 0 | Clear TX_ABRT Interrupt (CLR_TX_ABRT) Read this bit to clear the TX_ABRT interrupt (bit 6) of the I2C_RIS register, and the I2C_TX_ABRT_SRC register. Refer to bit 9 of the I2C_ABRT_SRC register for an exception to clearing I2C_TX_ABRT_SRC. | RO | 0b |

12.5.18 Clear RX_DONE Interrupt Register (I2C_CLR_RX_DONE)

Offset: 0058h Size: 32 bits

When read, this read-only register clears the RX_DONE interrupt.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|---|------|---------|
| 31:1 | RESERVED | RO | - |
| 0 | Clear RX_DONE Interrupt (CLR_RX_DONE) Read this bit to clear the RX_DONE interrupt (bit 7) of the I2C_RIS register. | RO | 0b |

12.5.19 Clear ACTIVITY Interrupt Register (I2C_CLR_ACTIVITY)

Offset: 005Ch Size: 32 bits

When read, this read-only register clears the ACTIVITY interrupt.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|--|------|---------|
| 31:1 | RESERVED | RO | - |
| 0 | Clear ACTIVITY Interrupt (CLR_ACTIVITY) Reading this bit clears the ACTIVITY interrupt (bit 8) of the I2C_RIS register if the I ² C is no longer active. If the I ² C controller is still active on the bus, the ACTIVITY interrupt bit continues to be set. This bit is automatically cleared by hardware if the I ² C controller is disabled and there is no further activity on the bus. | RO | 0b |

12.5.20 Clear STOP_DET Interrupt Register (I2C_CLR_STOP_DET)

Offset: 0060h Size: 32 bits

When read, this read-only register clears the STOP_DET interrupt.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|--|------|---------|
| 31:1 | RESERVED | RO | - |
| 0 | Clear STOP_DET Interrupt (CLR_STOP_DET) Read this bit to clear the STOP_DET interrupt (bit 9) of the I2C_RIS register. | RO | 0b |

12.5.21 Clear START_DET Interrupt Register (I2C_CLR_START_DET)

Offset: 0064h Size: 32 bits

When read, this read-only register clears the START_DET interrupt.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|--|------|---------|
| 31:1 | RESERVED | RO | - |
| 0 | Clear START_DET Interrupt (CLR_START_DET) Read this bit to clear the START_DET interrupt (bit 10) of the I2C_RIS register. | RO | 0b |

12.5.22 Clear GEN_CALL Interrupt Register (I2C_CLR_GEN_CALL)

Offset: 0068h Size: 32 bits

When read, this read-only register clears the GEN_CALL interrupt.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|---|------|---------|
| 31:1 | RESERVED | RO | - |
| 0 | Clear GEN_CALL Interrupt (CLR_GEN_CALL) Read this bit to clear the GEN_CALL interrupt (bit 11) of the I2C_RIS register. | RO | 0b |

12.5.23 I²C Enable Register (I2C_ENABLE)

Offset: 006Ch Size: 32 bits

This read/write register controls the enabling and disabling of the I²C controller.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|--|------|---------|
| 31:1 | RESERVED | RO | - |
| 0 | <p>I²C Enable (ENABLE) This bit controls whether the I²C controller is enabled.</p> <p>0: Disables I²C controller (TX/RX FIFOs are held in erased state) 1: Enables I²C controller</p> <p>Software can disable the I²C controller while it is active. However, it is important that care be taken to ensure that it is disabled properly. A recommended procedure is described in Section 12.4.4, "Disabling the I2C Controller".</p> <p>When the I²C controller is disabled, the following occurs:</p> <ul style="list-style-type: none"> ■ The TX FIFO and RX FIFO get flushed. ■ Status bits in the I2C_INTR_STAT register are still active until the I²C controller goes into IDLE state. <p>If the I²C controller is transmitting, it stops as well as deletes the contents of the transmit buffer after the current transfer is complete. If the I²C controller is receiving, the current transfer is stopped at the end of the current byte and does not acknowledge the transfer.</p> <p>Note: There is a two clock cycle delay when enabling or disabling the I²C controller.</p> | R/W | 0b |

12.5.24 Status Register (I2C_STATUS)

Offset: 0070h Size: 32 bits

This read-only register is used to indicate the current transfer and FIFO status. This register may be read at any time.

When the I²C controller is disabled (I2C_ENABLE[0]=0):

- Bits 1 and 2 are set to 1
- Bits 3 and 4 are set to 0

When the I²C controller is disabled (I2C_ENABLE[0]=0) and the master and slave controllers are idle:

- Bits 5 and 6 are set to 0

Note: None of the bits in this register request an interrupt.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|---|------|---------|
| 31:7 | RESERVED | RO | - |
| 6 | Slave Activity Status (SLV_ACTIVITY) When the slave controller is not in the IDLE state, this bit is set. 0: Slave controller is in IDLE state. The slave portion of the I ² C controller is not active. 1: Slave controller is not in IDLE state. The slave portion of the I ² C controller is active. | RO | 0b |
| 5 | Master Activity Status (MST_ACTIVITY) When the master controller is not in the IDLE state, this bit is set. 0: Master controller is in IDLE state. The master portion of the I ² C controller is not active. 1: Master controller is not in IDLE state. The master portion of the I ² C controller is active. | | 0b |
| 4 | RX FIFO Full Status (RFF) When the RX FIFO is completely full, this bit is set. When the RX FIFO contains one or more empty locations, this bit is cleared. 0: RX FIFO is not full 1: RX FIFO is full | | 0b |
| 3 | RX FIFO Not Empty Status (RFNE) This bit is set when the RX FIFO contains one or more entries and is cleared when the RX FIFO is empty. This bit can be polled by software to completely empty the RX FIFO. 0: RX FIFO is empty 1: RX FIFO is not empty | | 0b |
| 2 | TX FIFO Empty Status (TFE) When the TX FIFO is completely empty, this bit is set. When it contains one or more valid entries, this bit is cleared. This bit field does not request an interrupt. 0: TX FIFO is not empty 1: TX FIFO is empty | | 1b |

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|--|------|---------|
| 1 | TX FIFO Not Full Status (TFNF) This bit is set when the TX FIFO contains one or more empty locations, and is cleared when the TX FIFO is full. 0: TX FIFO is full 1: TX FIFO is not full | | 1b |
| 0 | I²C Activity Status (ACTIVITY) 0: No I ² C activity detected 1: I ² C activity detected | | 0b |

12.5.25 Transmit FIFO Level Register (I2C_TXFLR)

Offset: 0074h Size: 32 bits

This read-only register contains the number of valid data entries in the TX FIFO buffer. This register increments whenever data is placed into the TX FIFO and decrements when data is taken from the TX FIFO. It is cleared by the following conditions:

- The I²C controller is disabled (I2C_ENABLE[0]=0)
- There is a transmit abort (TX_ABRT bit is set in the I2C_RIS register)
- The slave bulk transmit mode is aborted

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|--|------|---------|
| 31:5 | RESERVED | RO | - |
| 4:0 | TX FIFO Level (TXFLR) This field contains the number of valid data entries in the TX FIFO. | RO | 0h |

12.5.26 Receive FIFO Level Register (I2C_RXFLR)

Offset: 0078h Size: 32 bits

This read-only register contains the number of valid data entries in the RX FIFO buffer. This register increments whenever data is placed into the RX FIFO and decrements when data is taken from the RX FIFO. It is cleared by the following conditions:

- The I²C controller is disabled (I2C_ENABLE[0]=0)
- There is a transmit abort caused by any of the events tracked in the I2C_TX_ABRT_SRC register

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|--|------|---------|
| 31:5 | RESERVED | RO | - |
| 4:0 | RX FIFO Level (RXFLR) This field contains the number of valid data entries in the RX FIFO. | RO | 0h |

12.5.27 Transmit Abort Source Register (I2C_TX_ABRT_SRC)

Offset: 0080h Size: 32 bits

This read/write register contains 14 bits that indicate the source of the transmit abort interrupt (TX_ABRT). Except for bit 9, this register is cleared whenever the I2C_CLR_TX_ABRT register or the I2C_CLR_INTR register is read. To clear bit 9, the source of the ABRT_SBYTE_NORSTRT must be attended to; RESTART must be enabled (I2C_CON [5]=1), the SPECIAL bit must be cleared (I2C_TAR [11]), or the GC_OR_START bit must be cleared (I2C_TAR[10]). Once the source of the ABRT_SBYTE_NORSTRT is attended to, this bit can be cleared in the same manner as other bits of this register. If the source of the ABRT_SBYTE_NORSTRT is not attended to before attempting to clear this bit, bit 9 clears for one cycle and is then re-asserted.

Note: As can be seen below, each bit description contains the additional field “I²C MODE” which contains the applicable I²C modes for each bit (master-transmitter, master-receiver, slave-transmitter, slave-receiver).

| BITS | DESCRIPTION | TYPE | DEFAULT | I ² C MODE |
|-------|---|------|---------|---|
| 31:16 | RESERVED | RO | - | - |
| 15 | (ABRT_SLVRD_INTX) This bit is set when the I ² C controller responds to a slave mode request for data to be transmitted to a remote master and a 0 is written in CMD (bit 8) of the I2C_DATA_CMD register. | RO | 0b | Slave-Transmitter |
| 14 | (ABRT_SLV_ARBLOST) This bit is set when the slave controller loses the bus while transmitting data to a remote master. I2C_TX_ABRT_SRC[12] is set at the same time. Note: Even though the slave never “owns” the bus, something could go wrong on the bus. This is a fail safe check. For instance, during a data transmission at the low-to-high transition of I2CSCL, if what is on the data bus is not what is supposed to be transmitted, then the I ² C controller no longer own the bus. | RO | 0b | Slave-Transmitter |
| 13 | (ABRT_SLVFLUSH_TXFIFO) This bit is set when the slave has received a read command and data exists in the TX FIFO. The slave issues a TX_ABRT interrupt to flush the old data in the TX FIFO. | RO | 0b | Slave-Transmitter |
| 12 | (ARB_LOST) This bit is set when the master has lost arbitration, or if I2C_TX_ABRT_SRC[14] is also set, then the slave transmitter has lost arbitration. Note: I ² C can be both a master and a slave at the same time. | RO | 0b | Master-Transmitter or Slave-Transmitter |
| 11 | (ABRT_MASTER_DIS) This bit is set when a master operation is attempted with the master mode disabled. | RO | 0b | Master-Transmitter or Master-Receiver |
| 10 | (ABRT_10B_RD_NORSTRT) This bit is set when the restart is disabled (I2C_RESTART_EN bit (I2C_CON[5]) = 0) and the master sends a read command in 10-bit addressing mode. | RO | 0b | Master-Receiver |

| BITS | DESCRIPTION | TYPE | DEFAULT | I ² C MODE |
|------|--|------|---------|---------------------------------------|
| 9 | (ABRT_SBYTE_NORSTRT) This bit is set when the restart is disabled (I2C_RESTART_EN bit (I2C_CON[5]) = 0) and a START byte send is attempted. To clear Bit 9, the source of the ABRT_SBYTE_NORSTRT must be attended to; restart must be enabled (I2C_CON[5]=1), the SPECIAL bit must be cleared (I2C_TAR[11]), or the GC_OR_START bit must be cleared (I2C_TAR[10]). Once the source of the ABRT_SBYTE_NORSTRT is attended to, this bit can be cleared in the same manner as other bits in this register. If the source of the ABRT_SBYTE_NORSTRT is not attended to before attempting to clear this bit, bit 9 clears for one cycle and then gets re-asserted. | RO | 0b | Master-Transmitter or Master-Receiver |
| 8 | RESERVED | RO | - | - |
| 7 | (ABRT_SBYTE_ACKDET) This bit is set when the master has sent a START byte and the START byte was acknowledged (wrong behavior). | RO | 0b | Master-Transmitter or Master-Receiver |
| 6 | RESERVED | RO | - | - |
| 5 | (ABRT_GCALL_READ) This bit is set when in master mode a General Call is sent, but the user programmed the byte following the General Call to be a read from the bus (I2C_DATA_CMD[8] is set to 1). | RO | 0b | Master-Transmitter |
| 4 | (ABRT_GCALL_NOACK) This bit is set when in master mode a General Call is sent, and no slave on the bus acknowledged the General Call. | RO | 0b | Master-Transmitter |
| 3 | (ABRT_TXDATA_NOACK) This master-mode only bit is set when the master has received an acknowledgement for the address, but the sent data byte(s) following the address did not receive an acknowledgement from the remote slave(s) | RO | 0b | Master-Transmitter |
| 2 | (ABRT_10ADDR2_NOACK) This bit is set when the master is in 10-bit address mode and the second address byte of the 10-bit address was not acknowledged by any slave. | RO | 0b | Master-Transmitter or Master-Receiver |
| 1 | (ABRT_10ADDR1_NOACK) This bit is set when the master is in 10-bit address mode and the first 10-bit address byte was not acknowledged by any slave. | RO | 0b | Master-Transmitter or Master-Receiver |
| 0 | (ABRT_7B_ADDR_NOACK) This bit is set when the master is in 7-bit addressing mode and the address sent was not acknowledged by any slave. | RO | 0b | Master-Transmitter or Master-Receiver |

12.5.28 SDA Setup Register (I2C_SDA_SETUP)

Offset: 0094h Size: 32 bits

This read/write register controls the amount of time delay (in terms of number of I²C clock periods) introduced in the rising edge of I2CSCL, relative to I2CSDA changing, when the I²C controller services a read request in a slave-receiver operation. The relevant I²C requirement is tSU;DAT (Note 4) as detailed in the Philips *I2C-Bus Specification*.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|--|------|---------|
| 31:8 | RESERVED | RO | - |
| 7:0 | SDA Setup Value (SDA_SETUP) Controls the amount of delay (in clock periods) introduced in the rising edge of I2CSCL. | R/W | 64h |

12.5.29 ACK General Call Register (I2C_ACK_GEN_CALL)

Offset: 0098h Size: 32 bits

This read/write register controls whether the I²C controller responds with an ACK or NACK when it receives an I²C General Call address.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|---|------|---------|
| 31:1 | RESERVED | RO | - |
| 0 | ACK General Call Enable (ACK_GEN_CALL) When set to 1, the I ² C controller responds with an ACK when it receives a General Call. Otherwise, the I ² C controller responds with a NACK. 0: General Call ACK response 1: General Call NACK response | R/W | 1b |

12.5.30 Enable Status Register (I2C_ENABLE_STATUS)

Offset: 009Ch Size: 32 bits

This read-only register is used to report the I²C controller hardware status when the ENABLE bit of the I2C_ENABLE register changes from 1 to 0 (I²C controller disabled). Once ENABLE has been set to 1, bits [2:1] will be forced to 0.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|--|------|---------|
| 31:3 | RESERVED | RO | - |
| 2 | <p>Slave FIFO Filled and Flushed (SLV_FIFO_FF) This bit indicates if a slave-receiver operation has been aborted with at least 1 data byte received from an I²C transfer due to the setting of I2C_ENABLE from 1 to 0.</p> <p>When read as 1, the I²C controller is deemed to have been actively engaged in an aborted I²C transfer (with matching address) and the data phase of the I²C transfer has been entered, even though the first data byte has been responded with a NACK.</p> <p>When read as 0, the I²C controller is deemed to have been disabled without being actively involved in any slave-receiver transfer.</p> <p>Note: The CPU can safely read this bit when I2C_EN (bit 0) is read as 0.</p> | RO | 0b |
| 1 | <p>Slave-Receiver Operation Aborted (SLV_RX_ABORTED) This bit indicates if a potential or active slave-receiver operation has been aborted due to the setting of the I2C_ENABLE register from 1 to 0. This bit is set when the CPU writes a 0 to the I2C_ENABLE register while the I²C controller is receiving the address byte of the slave-transmitter operation from a remote master. This bit is tied to a slave-receiver operation only.</p> <p>When read as 1, the I²C controller is deemed to have forced a NACK during any part of an I²C transfer, irrespective of whether the I²C address matches the slave address set in the I2C_SAR register OR if the transfer is completed before I2C_ENABLE is set to 0 but has not taken effect.</p> <p>When read as 0, the I²C controller is deemed to have been disabled without actively involved in any slave-receiver transfer.</p> <p>Note: This bit does not reflect any master or slave transmit operation. The CPU can safely read this bit when I2C_EN (bit 0) is read as 0.</p> | RO | 0b |
| 0 | <p>I²C Controller Enabled Status (I2C_EN) This bit reflects the I²C controller status. When read as 1, the I²C controller is deemed to be actively involved in an I²C transfer, irrespective of whether being in an address or data phase for all master or slave modes. When read as 0, the I²C controller is deemed completely inactive.</p> <p>Note: The CPU can safely read this bit anytime. When this bit is read as 0, the CPU can safely read SLV_FIFO_FILLED_AND_FLUSHED (bit 2) and SLV_RX_ABORTED (bit 1).</p> | RO | 0b |

12.5.31 Access Module Register (I2C_ACCESS)

Offset: 0100h Size: 32 bits

This register may be used in conjunction with the I2C_SEQ_DATA_x registers for sequential reads/writes. The access module allows a user to perform complex I²C accesses via a single register. It can perform non-sequential accesses, where it writes an address, followed by either a read or write of data (via its I2C_DATA field), or it can perform sequential accesses where it writes an address followed by 1 to 64-byte writes or reads of data (using the I2C_SEQ_DATA_x registers). Once an access is complete, its status should be read and checked by the user. Note that when the I2C_SEQ_DATA_x registers are employed in a sequential read/write, the LSB of the data being accessed is in the LSB of I2C_SEQ_DATA_0, while the location of the MSB depends on the value of I2C_ACCESS_SIZE. When I2C_ACCESS_SIZE is 3Fh, the MSB of the data is in the MSB of I2C_SEQ_DATA_15.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|---|--------|---------|
| 31 | Busy (I2C_ACCESS_BUSY) This bit must be set by software when initiating an I ² C transfer. It is cleared by the hardware on completion of the transfer. A new transfer must not be attempted while this bit or the I2C_ACCESS_DONE bit is set. Once a transfer is initiated via a write to this register, the State (I2C_ACCESS_DONE) bit may be checked to determine that the transfer has completed. Refer to Note 12.3 for further information. | R/W/SC | 0b |
| 30 | Reset (I2C_ACCESS_RST) Writing 1 to this bit causes a soft reset of the I2C Access Module. | WO | 0b |
| 29 | Error (I2C_ACCESS_ERROR) When set, this bit indicates that an error occurred during the previous I ² C access attempted using this register. Refer to Note 12.3 for further information. | R/WC | 0b |
| 28 | RESERVED | RO | - |
| 27 | State (I2C_ACCESS_DONE) Access is complete when set. | R/WC | 0b |
| 26 | RESERVED | RO | - |
| 25:20 | Size (I2C_ACCESS_SIZE) The contents of this field specifies the number of bytes in a sequential access. | R/W | 00h |
| 19 | RESERVED | RO | - |
| 18 | Skip Status (I2C_SKIP_STATUS) This bit informs the hardware to skip the status check after address in read operations. | R/W | 0b |
| 17 | Sequential Access Enable (I2C_SEQ_ACCESS) When set, this bit enables sequential access via the I2C_SEQ_DATA_x registers. When clear, a non-sequential access will be performed using the I2C_DATA field of this register. | R/W | 0b |
| 16 | Read/Write Select (I2C_READ_nWRITE) When set, a read access will be performed. When cleared, a write will occur. | R/W | 0b |
| 15:8 | Address (I2C_ADDRESS) This field specifies the address of the I ² C access. | R/W | 00h |
| 7:0 | Data (I2C_DATA) This field is used to access read/write data for a non-sequential access. | R/W | 00h |



Note 12.3 The [Error \(I2C_ACCESS_ERROR\)](#) bit will not indicate an error when the video cable (VGA/HDMI/DVI) is unplugged and software attempts to read the EDID of the monitor. In this case, the [State \(I2C_ACCESS_DONE\)](#) bit is not set nor is the [Busy \(I2C_ACCESS_BUSY\)](#) bit cleared by hardware. Software should utilize a timer with a suitable timeout period to force reset of the I2C Access Module when a condition of this sort occurs. This issue only occurs when the [Sequential Access Enable \(I2C_SEQ_ACCESS\)](#) bit is set.

12.5.32 I2C Pin Control Register (I2C_PIN)

Offset: 0104h Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|---|------|---------|
| 31:5 | RESERVED | RO | - |
| 4 | I2C Clock pin output enable (I2C_CLK_OEN) | R/W | 0b |
| 3 | I2C Clock pin input value (I2C_CLK_IN) | RO | X |
| 2 | I2C Data pin output enable (I2C_DATA_OEN) | R/W | 0b |
| 1 | I2C Data pin input value (I2C_DATA_IN) | RO | X |
| 0 | I2C Raw pin control enable (I2C_RAW_EN) | R/W | 0b |

12.5.33 Sequential Data 0 Register (I2C_SEQ_DATA_0)

Offset: 0110h Size: 32 bits

This register is used in conjunction with the I2C_ACCESS register for sequential reads/writes. It should be written with data before a write, and after a read will contain the read data.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|------------------------------------|------|------------|
| 31:0 | SEQUENTIAL DATA 0 (I2C_SEQ_DATA_0) | R/W | 0000_0000h |

12.5.34 Sequential Data 1 Register (I2C_SEQ_DATA_1)

Offset: 0114h Size: 32 bits

This register is used in conjunction with the I2C_ACCESS register for sequential reads/writes. It should be written with data before a write, and after a read will contain the read data.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|------------------------------------|------|------------|
| 31:0 | SEQUENTIAL DATA 1 (I2C_SEQ_DATA_1) | R/W | 0000_0000h |

12.5.35 Sequential Data 2 Register (I2C_SEQ_DATA_2)

Offset: 0118h Size: 32 bits

This register is used in conjunction with the I2C_ACCESS register for sequential reads/writes. It should be written with data before a write, and after a read will contain the read data.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|------------------------------------|------|------------|
| 31:0 | SEQUENTIAL DATA 2 (I2C_SEQ_DATA_2) | R/W | 0000_0000h |

12.5.36 Sequential Data 3 Register (I2C_SEQ_DATA_3)

Offset: 011Ch Size: 32 bits

This register is used in conjunction with the I2C_ACCESS register for sequential reads/writes. It should be written with data before a write, and after a read will contain the read data.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|------------------------------------|------|------------|
| 31:0 | SEQUENTIAL DATA 3 (I2C_SEQ_DATA_3) | R/W | 0000_0000h |

12.5.37 Sequential Data 4 Register (I2C_SEQ_DATA_4)

Offset: 0120h Size: 32 bits

This register is used in conjunction with the I2C_ACCESS register for sequential reads/writes. It should be written with data before a write, and after a read will contain the read data.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|------------------------------------|------|------------|
| 31:0 | SEQUENTIAL DATA 4 (I2C_SEQ_DATA_4) | R/W | 0000_0000h |

12.5.38 Sequential Data 5 Register (I2C_SEQ_DATA_5)

Offset: 0124h Size: 32 bits

This register is used in conjunction with the I2C_ACCESS register for sequential reads/writes. It should be written with data before a write, and after a read will contain the read data.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|------------------------------------|------|------------|
| 31:0 | SEQUENTIAL DATA 5 (I2C_SEQ_DATA_5) | R/W | 0000_0000h |

12.5.39 Sequential Data 6 Register (I2C_SEQ_DATA_6)

Offset: 0128h Size: 32 bits

This register is used in conjunction with the I2C_ACCESS register for sequential reads/writes. It should be written with data before a write, and after a read will contain the read data.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|------------------------------------|------|------------|
| 31:0 | SEQUENTIAL DATA 6 (I2C_SEQ_DATA_6) | R/W | 0000_0000h |



12.5.40 Sequential Data 7 Register (I2C_SEQ_DATA_7)

Offset: 012Ch Size: 32 bits

This register is used in conjunction with the I2C_ACCESS register for sequential reads/writes. It should be written with data before a write, and after a read will contain the read data.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|------------------------------------|------|------------|
| 31:0 | SEQUENTIAL DATA 7 (I2C_SEQ_DATA_7) | R/W | 0000_0000h |

12.5.41 Sequential Data 8 Register (I2C_SEQ_DATA_8)

Offset: 0130h Size: 32 bits

This register is used in conjunction with the I2C_ACCESS register for sequential reads/writes. It should be written with data before a write, and after a read will contain the read data.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|------------------------------------|------|------------|
| 31:0 | SEQUENTIAL DATA 8 (I2C_SEQ_DATA_8) | R/W | 0000_0000h |



12.5.42 Sequential Data 9 Register (I2C_SEQ_DATA_9)

Offset: 0134h Size: 32 bits

This register is used in conjunction with the I2C_ACCESS register for sequential reads/writes. It should be written with data before a write, and after a read will contain the read data.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|------------------------------------|------|------------|
| 31:0 | SEQUENTIAL DATA 9 (I2C_SEQ_DATA_9) | R/W | 0000_0000h |

12.5.43 Sequential Data 10 Register (I2C_SEQ_DATA_10)

Offset: 0138h Size: 32 bits

This register is used in conjunction with the I2C_ACCESS register for sequential reads/writes. It should be written with data before a write, and after a read will contain the read data.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|--------------------------------------|------|------------|
| 31:0 | SEQUENTIAL DATA 10 (I2C_SEQ_DATA_10) | R/W | 0000_0000h |

12.5.44 Sequential Data 11 Register (I2C_SEQ_DATA_11)

Offset: 013Ch Size: 32 bits

This register is used in conjunction with the I2C_ACCESS register for sequential reads/writes. It should be written with data before a write, and after a read will contain the read data.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|--------------------------------------|------|------------|
| 31:0 | SEQUENTIAL DATA 11 (I2C_SEQ_DATA_11) | R/W | 0000_0000h |

12.5.45 Sequential Data 12 Register (I2C_SEQ_DATA_12)

Offset: 0140h Size: 32 bits

This register is used in conjunction with the I2C_ACCESS register for sequential reads/writes. It should be written with data before a write, and after a read will contain the read data.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|--------------------------------------|------|------------|
| 31:0 | SEQUENTIAL DATA 12 (I2C_SEQ_DATA_12) | R/W | 0000_0000h |



12.5.46 Sequential Data 13 Register (I2C_SEQ_DATA_13)

Offset: 0144h Size: 32 bits

This register is used in conjunction with the I2C_ACCESS register for sequential reads/writes. It should be written with data before a write, and after a read will contain the read data.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|--------------------------------------|------|------------|
| 31:0 | SEQUENTIAL DATA 13 (I2C_SEQ_DATA_13) | R/W | 0000_0000h |

12.5.47 Sequential Data 14 Register (I2C_SEQ_DATA_14)

Offset: 0148h Size: 32 bits

This register is used in conjunction with the I2C_ACCESS register for sequential reads/writes. It should be written with data before a write, and after a read will contain the read data.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|--------------------------------------|------|------------|
| 31:0 | SEQUENTIAL DATA 14 (I2C_SEQ_DATA_14) | R/W | 0000_0000h |

12.5.48 Sequential Data 15 Register (I2C_SEQ_DATA_15)

Offset: 014Ch Size: 32 bits

This register is used in conjunction with the I2C_ACCESS register for sequential reads/writes. It should be written with data before a write, and after a read will contain the read data.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|--------------------------------------|------|------------|
| 31:0 | SEQUENTIAL DATA 15 (I2C_SEQ_DATA_15) | R/W | 0000_0000h |

Chapter 13 System Control

13.1 Overview

This section contains entities that control system hardware operation and whose Control and Status registers operate solely on `clk_xtal`. As a result, these entities may be configured before the System PLL is operable.

13.2 I2C Selection

I2C port selection is accomplished via the [I2C Interface Select \(I2C_SEL\)](#) bit of the [I2C Interface Selection Register \(I2C_SEL\)](#). When this bit is clear, the HDMI transmitter is connected to I2C Port 0 and the I2C Controller is connected to I2C Port 1. When the bit is set, the HDMI transmitter is connected to I2C Port 1, while the I2C Controller is connected to I2C Port 0.

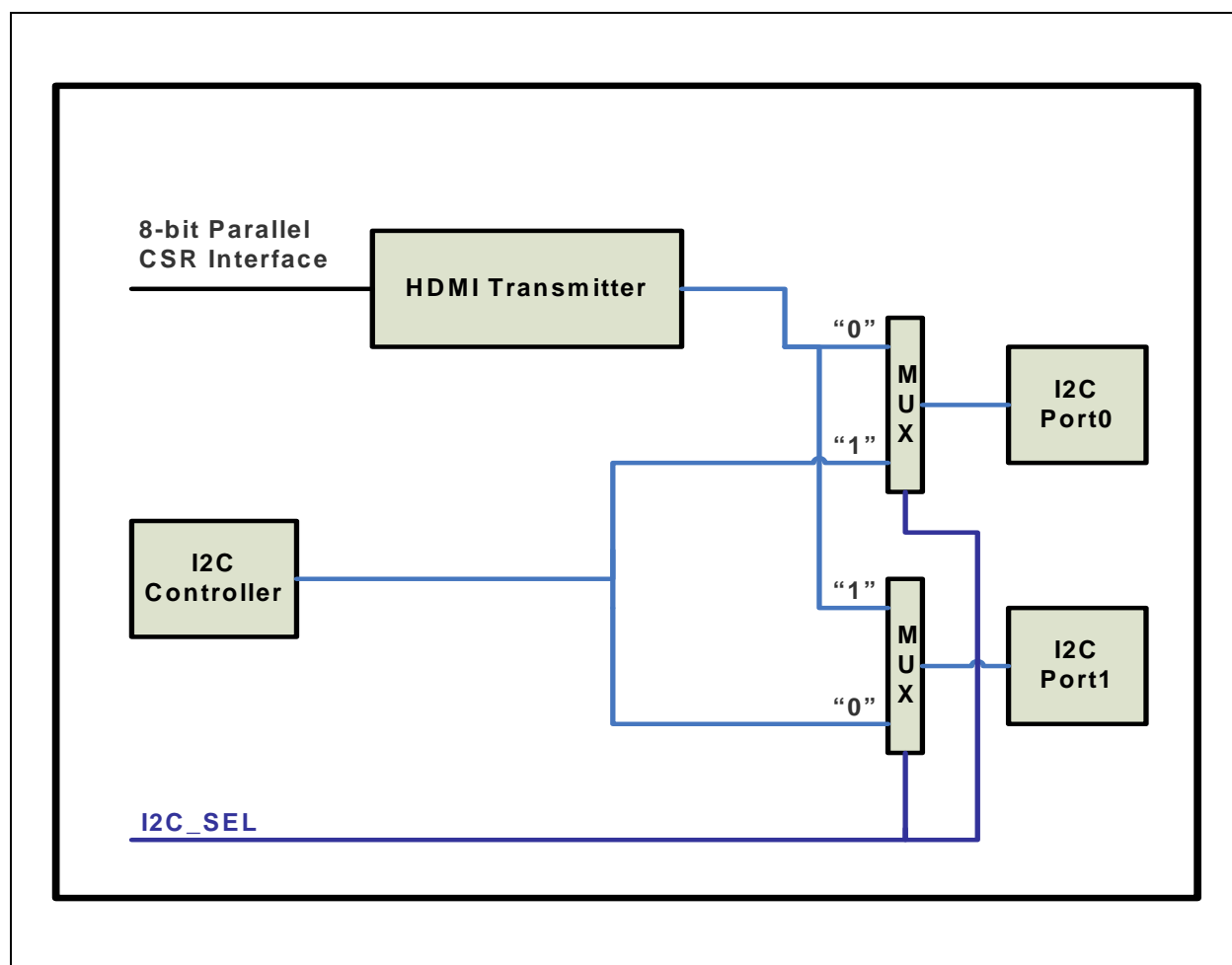


Figure 13.1 I2C Interface Selection

13.3 Control and Status Registers

The System Control CSRs are capable of configuring various system settings. This block operates on clk_xtal, therefore, its CSRs are available before the System PLL is enabled.

Table 13.1 System Control Register Map

| ADDRESS | SYMBOL | REGISTER NAME |
|---------------|----------|---------------------------------|
| 3000h | ID_REV | Device ID Register |
| 3004h | FPGA_REV | FPGA Revision Register |
| 3008h | HW_CFG | Hardware Configuration Register |
| 300Ch | RESERVED | Reserved for future expansion |
| 3010h | I2C_SEL | I2C Selection Register |
| 3010h – 3FFFh | RESERVED | Reserved for future expansion |

13.3.1 Device ID (ID_REV)

Offset: 0000h Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|-----------------------|------|---------------------------|
| 31:16 | Device ID (ID) | RO | 9D00h |
| 15:0 | Device Revision (REV) | RO | Note 13.1 |

Note 13.1 Default value is dependent on device revision.



13.3.2 FPGA (FPGA_REV)

Offset: 0004h Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|--------------------------|------|---------|
| 31:16 | RESERVED | RO | - |
| 15:0 | FPGA Revision (FPGA_REV) | RO | 0000h |

13.3.3 Hardware Configuration Register (HW_CFG)

Offset: 0008h Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|--|------|---------------------------|
| 31:7 | RESERVED | RO | - |
| 6 | SW_MODE Polarity (SW_MODE_POL) This bit selects the polarity of the nSW_MODE pin. 0 = Active low. 1 = Active high. Note: This field is protected by Reset Protection (RST_PROTECT) | R/W | Note 13.2 |
| 5 | Interrupt Pin Polarity (INT_POL) This bit indicates the polarity of the INT pin. When set, the INT pin is active-high. Otherwise it is active-low. Note: This field is protected by Reset Protection (RST_PROTECT) . | R/W | Note 13.3 |
| 4 | EEPROM Emulation Enable (EEM) This bit is used to select the source of descriptor information and configuration flags when no EEPROM is present. 0 = Use defaults as specified in Section 8.3, "EEPROM Defaults," on page 163 . 1 = Use Descriptor RAM and Attributes Registers Note: This bit affects operation only when a EEPROM is not present. This bit has no effect when a EEPROM is present. Note: This field is protected by Reset Protection (RST_PROTECT) . | R/W | 0b |
| 3 | Reset Protection (RST_PROTECT) Setting this bit protects select fields of certain registers from being affected by resets other than POR. Note: This field is protected by Reset Protection (RST_PROTECT) . | R/W | 0b |
| 2 | EEPROM Time-out Control (ETC) This bit controls the length of time used by the EEPROM controller to detect a time-out. 0 = Time-out occurs if no response received from EEPROM after 30 ms. 1 = Time-out occurs if no response received from EEPROM after 1.28 us. | R/W | 0b |
| 1 | Soft Reset (SRST) Writing 1 generates a software initiated reset of the device. If an external Ethernet PHY is used, it will be reset as well. A software reset will result in the contents of the EEPROM being reloaded. While the reset sequence is in progress, the USB PHY will be disconnected. After the device has been reinitialized, it will take the PHY out of the disconnect state and be visible to the Host. | SC | 0b |
| 0 | Soft Lite Reset (LRST) Writing 1 generates the lite software reset of the device. A lite reset will not affect the UDC. Additionally, the contents of the EEPROM will not be reloaded. This reset will not cause the USB PHY to be disconnected. This bit clears after the reset sequence has completed. | SC | 0b |

- Note 13.2** The default value of this field is determined by the value of the [SW_MODE Polarity \(CFG0_SW_MODE_POL\)](#) field of the [Configuration Flags 0](#) contained within the EEPROM, if present. If no EEPROM is present, 0b is the default. A USB Reset or Lite Reset (LRST) will cause this field to be restored to the image value last loaded from EEPROM, or to be set to 0b if no EEPROM is present.
- Note 13.3** The default value of this bit is determined by the value of the [Interrupt Pin Polarity \(CFG0_INT_PIN_POL\)](#) bit of [Configuration Flags 0](#) contained within the EEPROM, if present. If no EEPROM is present, 0b is the default. A USB Reset or Lite Reset (LRST) will cause this field to be restored to the image value last loaded from EEPROM, or to be set to 0b if no EEPROM is present.

13.3.4 I2C Interface Selection Register (I2C_SEL)

Offset: 0010h Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|---|------|---------|
| 31:1 | RESERVED | RO | - |
| 0 | I2C Interface Select (I2C_SEL) Controls I2C selection mux | R/W | 0 |

See [Section 13.2, "I2C Selection"](#) for a description of the operation of this register.

Chapter 14 Miscellaneous

14.1 Overview

This section describes miscellaneous registers that are present in the device.

14.2 LED Configuration

The device supports an LED which can be configured to visually indicate the amount of USB Bulk Out data packet traffic it is handling. When the [LED Enable \(LED_EN\)](#) bit of [LED Configuration Register 0 \(LED_CFG0\)](#) is set, and the [SW LED Control \(SW_LED_CTRL\)](#) bit is clear, the LED is enabled for traffic indication. When [LED Enable \(LED_EN\)](#) is cleared, the LED is disabled. When [LED Enable \(LED_EN\)](#) and [SW LED Control \(SW_LED_CTRL\)](#) are both set, software is in control of the LED blinking and the amount of USB Bulk Out packet traffic has no effect on LED operation.

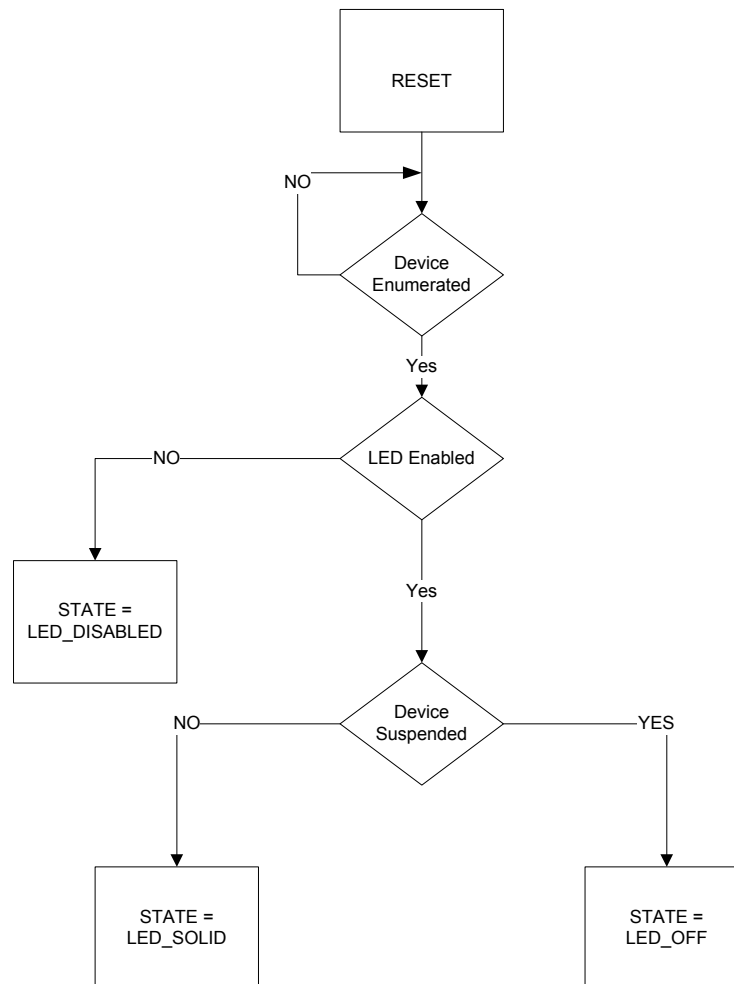
The following assumptions hold when determining the amount of USB Bulk Out data packet traffic:

- The total number of received bulk packets are counted.
- Only the size of the data stage is counted.

[Figure 14.1](#) illustrates how the initial state of the LED is determined. From the initial state, the transitions indicated in [Figure 14.2](#) occur, depending on the events encountered during operation.

Note: The operations illustrated in [Figure 14.1](#) and [Figure 14.2](#) are VALID ONLY WHEN [SW LED Control \(SW_LED_CTRL\)](#) IS CLEAR.

Note: If [I2S Enable \(I2S_EN\)](#) is set and [I2SCLKALT1](#) is selected, then [LED Enable \(LED_EN\)](#) must not be set, otherwise unpredictable results and untoward operation will result.

**Figure 14.1 LED Initial State Determination**

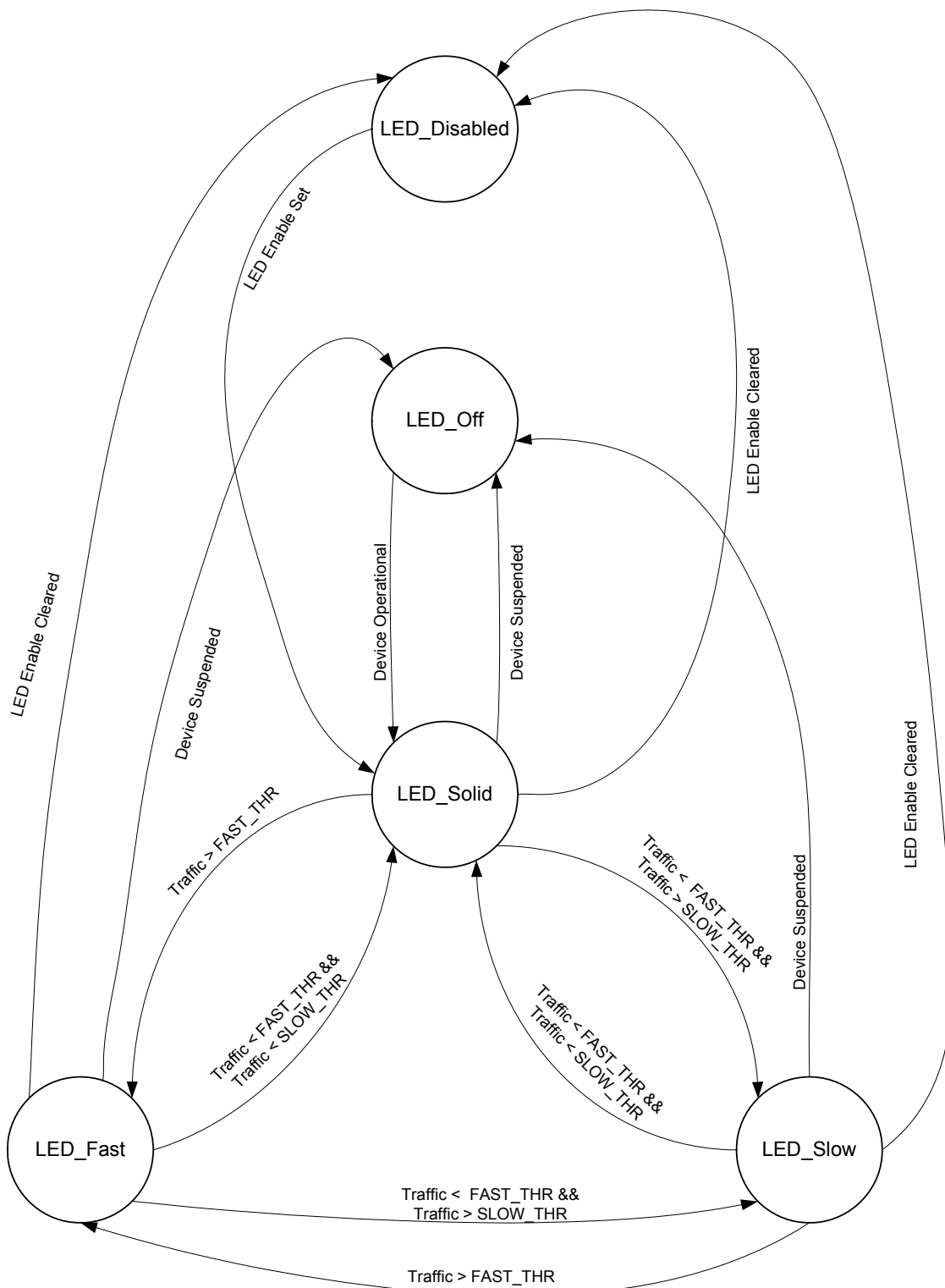


Figure 14.2 LED State Transitions

The [LED Configuration Register 1 \(LED_CFG1\)](#) specifies the traffic rates for the LED fast and slow blink thresholds. The hardware will periodically check the traffic rate against these thresholds whenever the [LED Enable \(LED_EN\)](#) bit of the [LED Configuration Register 0 \(LED_CFG0\)](#) is set, and the state is LED_Fast, LED_Slow, or LED_Solid. The [Fast Blink Threshold \(FAST_THR\)](#) has precedence and is always checked first by the hardware. If the traffic rate is greater than the [Fast Blink Threshold \(FAST_THR\)](#), then the LED is flashed at the rate dictated by the [Fast Threshold Blink Time \(FAST_THR_BLINK\)](#) field in [LED Configuration Register 2 \(LED_CFG2\)](#). If the traffic rate does not exceed the [Fast Blink Threshold \(FAST_THR\)](#), then the traffic rate is compared to the [Slow Blink Threshold \(SLOW_THR\)](#). If the traffic rate exceeds the [Slow Blink Threshold \(SLOW_THR\)](#), then the LED is flashed at the rate dictated by the [Slow Threshold Blink Time \(SLOW_THR_BLINK\)](#) field in [LED Configuration Register 2 \(LED_CFG2\)](#). LED_Fast and LED_Slow are the LED states entered when the corresponding blink threshold is exceeded. LED_Solid is entered on any one of the following conditions:

1. When the LED is initially enabled.
2. On transitions from the SUSPEND state.
3. On a transition from LED_Fast or LED_Slow when the traffic fails to exceed the [Fast Blink Threshold \(FAST_THR\)](#) and the [Slow Blink Threshold \(SLOW_THR\)](#), respectively.
4. Whenever either threshold value contained in the [LED Configuration Register 1 \(LED_CFG1\)](#) is altered by SW.
5. Whenever the contents of [LED Configuration Register 2 \(LED_CFG2\)](#) is updated.

Upon entering the LED_Solid state, traffic rate measurement and comparison with the thresholds commences.

One or both threshold checks may be disabled by writing the value FFFFh into threshold fields of [LED Configuration Register 1 \(LED_CFG1\)](#). Whenever the LED is enabled and both threshold checks are disabled, the LED will remain in the LED_SOLID state, with the LED being continuously illuminated.

Note: The default values for FAST_THR and SLOW_THR of FFFFh disable both threshold checks. In order to perform one or both of the threshold checks, and enable blinking according to the threshold met, set the threshold field corresponding to the desired check(s) to a value other than FFFFh.

14.3 Audio Configuration

14.3.1 I2S Audio

Three registers are provided in this section for audio configuration when the I2S audio pathway is to be utilized. Note that only single channel I2S is supported. The [Audio Configuration Register \(AUDIO_CFG\)](#) is used to enable the I2S audio pathway, select the pin on which the I2S clock input is present, and to enable I2S frequency measurement. The [I2S Enable \(I2S_EN\)](#) bit of this register is set to enable usage of the I2S audio path. When [I2S Enable \(I2S_EN\)](#) is set, the setting of the [I2SCLK Select \(I2SCLK_SEL\)](#) bit determines the pin ([I2SCLKALT0](#) or [I2SCLKALT1](#)) on which the I2S clock signal is connected. Since [I2SCLKALT1](#) shares pin functionality with the [LED](#) signal, both [I2S Enable \(I2S_EN\)](#) and [LED Enable \(LED_EN\)](#), or enable GPIO24, may not be set at the same time, otherwise unpredictable results and untoward operation will result. Likewise it is not valid to enable single ended RGB mode, or GPIO0 enable, when using [I2SCLKALT1](#).

Note: [I2SCLKALT0](#) is used if single data rate RGB is required. Otherwise [I2SCLKALT1](#) is used in the typical case.

Hardware dedicated to measuring the sampling rate of the CODEC and to determine when audio data starts/stops is enabled via the [I2S Frequency Measurement Enable \(I2S_FREQ_EN\)](#) bit of the [Audio Configuration Register \(AUDIO_CFG\)](#). When this is set, the results of the measurements are available in the [Audio Frequency 0 Register \(AUDIO_FREQ0\)](#) and [Audio Frequency 1 Register \(AUDIO_FREQ1\)](#) 1 ms later and are subsequently updated every 1ms. No interrupt is available to signal software that a result is available. Software must poll the registers to obtain updated results. The [Audio Frequency 0 Register \(AUDIO_FREQ0\)](#) provides measurements of the frequency at which the WS signal and I2S Clock toggle at. The units are transitions/ms. The [I2S WS Frequency \(WS_FREQ\)](#) field toggles one clock cycle before the MSB of a new audio word, thus it provides an accurate metric for sampling audio frequency. [I2S Clock Frequency \(I2SCLK_FREQ\)](#) contains the value of the frequency the I2S Clock toggles at. This field may be checked to determine that the clock signal is present.

The [Audio Frequency 1 Register \(AUDIO_FREQ1\)](#) provides measurements of the frequency at which the [MCLK](#) pin and the [I2SDATA](#) pin toggle. They are useful in determining the presence of the MCLK signal and for determining when audio data starts/stops.

Note: An existing GPIO pin may be used to force the disconnect of the partner USB audio CODEC. It is suggested that [GPIO30](#) be used, as it is available in all configurations of the part.

The results of the frequency measurements are used by the software to program the HDMI channel status CSRs appropriately.

14.3.2 SDPIF Audio

When SPDIF audio is in use, the [I2S Enable \(I2S_EN\)](#) bit of the [Audio Configuration Register \(AUDIO_CFG\)](#) must be cleared. Minimum programming of the HDMI CSRs is required to facilitate operation in this mode.

14.4 Control and Status Registers

These CSRs are only accessible after the System PLL is programmed and sys_clk is available.

Table 14.1 Miscellaneous Control and Status Register Map

| ADDRESS | SYMBOL | REGISTER NAME |
|---------------|-------------|---------------------------------------|
| 8000h | DP_SEL | Data Port Select Register |
| 8004h | DP_CMD | Data Port Command Register |
| 8008h | DP_ADDR | Data Port Address Register |
| 800Ch | DP_DATA0 | Data Port Data Register 0 |
| 8010h | DP_DATA1 | Data Port Data Register 1 |
| 8014h – 801Fh | RESERVED | Reserved for future expansion |
| 8020h | FIFO_STATUS | FIFO Status Register |
| 8024h | VDAC_CFG | Video DAC Configuration Register |
| 8028h | RGB_CFG | RGB Configuration Register |
| 802Ch | EXT_RST_CFG | External Reset Configuration Register |
| 8030h | AUDIO_CFG | Audio Configuration Register |
| 8034h | AUDIO_FREQ0 | Audio Frequency 0 Register |
| 8038h | AUDIO_FREQ1 | Audio Frequency 1 Register |
| 803Ch – 803Fh | RESERVED | Reserved for future expansion |
| 8040h | HDMI_CFG | HDMI Configuration Register |
| 8044h – 807Fh | RESERVED | Reserved for future expansion |
| 8080h | LED_CFG0 | LED Configuration Register 0 |
| 8084h | LED_CFG1 | LED Configuration Register 1 |
| 8088h | LED_CFG2 | LED Configuration Register 2 |
| 808Ch – 809Fh | RESERVED | Reserved for future expansion |
| 80A0h | GPIO_ENABLE | GPIO Enable Register |
| 80A4h | GPIO_BUF | GPIO Buffer Type Register |
| 80A8h | GPIO_DIR | GPIO Direction Register |
| 80ACh | GPIO_DATA | GPIO Data Register |
| 80B0h – 80FFh | RESERVED | Reserved for future expansion |
| 8100h | BOS_ATTR | BOS Attributes Register |
| 8104h | SS_ATTR | Super Speed Attributes Register |
| 8108h | HS_ATTR | High Speed Attributes Register |
| 810Ch | FS_ATTR | Full Speed Attributes Register |
| 8110h | STRNG_ATTR0 | String Attributes Register 0 |
| 8114h | STRNG_ATTR1 | String Attributes Register 0 |
| 8118h – 8FFFh | RESERVED | Reserved for future expansion |

14.4.1 Data Port Select Register (DP_SEL)

Offset: 0000h Size: 32 bits

Before accessing the internal RAMs, the [RAM Test Mode Enable \(DP_SEL_TESTEN\)](#) bit must be set. It is not valid to use the RAM data port during run time.

The [Data Port Select \(DP_SEL_RSEL\)](#) field chooses which internal RAM to access.

The [Data Port Ready \(DP_SEL_DPRDY\)](#) bit indicates when the data port RAM access has completed. In the case of a read operation, this indicates when the read data has been stored in the DP_DATA register.

Note: All writes to the Data Port Registers are ignored when the data port is busy.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|---|------|---------|
| 31 | Data Port Ready (DP_SEL_DPRDY) 0 = Data port is busy processing a transaction 1 = Data port is ready | RO | 1b |
| 30 | Data Port Done (DP_DONE) This bit asserts after the Data Port transaction successfully completes. | R/WC | 0b |
| 29:24 | RESERVED | RO | - |
| 23:20 | Transaction Response ID (DP_SEL_ID) This field corresponds to the Data Port Command ID (DP_CMD_ID) written when the transaction was issued. | RO | 0 |
| 19:18 | RESERVED | RO | - |
| 17:16 | Transaction Response Status (DP_SEL_RESP) AXI protocol response. 00 = OKAY - Indicates if a normal access has been successful. Can also indicate an exclusive access failure. 01 = EXOKAY - indicates that either the read or write portion of an exclusive access has been successful. 10 = SLVERR - The access has reached the slave successfully, but the slave wishes to return an error condition to the originating master. 11 = DECERR - Decode error - indicates that there is no slave at the transaction address. | RO | 0 |
| 15:10 | RESERVED | RO | - |
| 9:4 | Data Port Select (DP_SEL_RSEL) Selects target of the dataport access. See Table 14.2 for mapping. | R/W | 0 |
| 3:2 | RESERVED | RO | - |
| 1 | Auto Address Increment Enable (DP_SEL_AINC) Determines whether or not the Data Port Address Register (DP_ADDR) is incremented after a transaction completes. 0 = Address incrementing disabled 1 = After each transaction completes, the Data Port Address Register (DP_ADDR) is automatically incremented depending on the transaction size (incremented by 8 for DDR2 accesses). | R/W | 0 |

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|---|------|---------|
| 0 | RAM Test Mode Enable (DP_SEL_TESTEN) When set, put all test accessible RAMs in test mode. | R/W | 0 |

Table 14.2 Data Port Select Mapping Table

| DATA PORT SELECT | DATA PORT TARGET |
|------------------|--------------------|
| 00h | DDR2 RAM |
| 01h - 11h | Reserved |
| 12h | USB Descriptor RAM |
| 13h - 28h | Reserved |
| 29h | SSP PHY CSRs |
| 2Ah | HDMI CSRs |
| 2Bh - 3Fh | Reserved |

14.4.2 Data Port Command Register (DP_CMD)

Offset: 0004h Size: 32 bits

This register commences the data port access. Writing a one to this register will enable a write access, while writing a zero will do a read access.

The address and data registers need to be configured appropriately for the desired read or write operation before accessing the register.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|--|------|---------|
| 31:17 | RESERVED | RO | - |
| 16:8 | Byte Enables (DP_CMD_BE) Byte Enables for a write command. | R/W | 0 |
| 7:4 | Data Port Command ID (DP_CMD_ID) This field may be used to write an identifier for the command. When the transaction is complete, the value of the Transaction Response ID (DP_SEL_ID) field in Data Port Select Register (DP_SEL) should agree with this field. | R/W | 0 |
| 3:1 | RESERVED | RO | - |
| 0 | Data Port Write Or Read (DP_CMD_WOR) Selects operation. Writing to this bit initiates the data port access. 0 = Read operation 1 = Write operation | R/W | 0 |

14.4.3 Data Port Address Register (DP_ADDR)

Offset: 0008h Size: 32 bits

Indicates the address to be used for the data port access.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|-----------------------------|------|-----------|
| 31:26 | RESERVED | RO | - |
| 25:0 | Data Port Address (DP_ADDR) | R/W | 000_0000h |

14.4.4 Data Port Data 0 Register (DP_DATA0)

Offset: 000Ch Size: 32 bits

The Data Port Data register holds the write data for a write access and the resultant read data for a read access.

Before reading this register for the result of a read operation, the [Data Port Ready \(DP_SEL_DPRDY\)](#) bit should be checked. The [Data Port Ready \(DP_SEL_DPRDY\)](#) bit must indicate the data port is ready. Otherwise, the read operation is still in progress.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|---|------|------------|
| 31:0 | Data Port Data [31:0] (DP_DATA0) | R/W | 0000_0000h |

14.4.5 Data Port Data 1 Register (DP_DATA1)

Offset: 0010h Size: 32 bits

The Data Port Data register holds the write data for a write access and the resultant read data for a read access.

Before reading this register for the result of a read operation, the [Data Port Ready \(DP_SEL_DPRDY\)](#) bit should be checked. The [Data Port Ready \(DP_SEL_DPRDY\)](#) bit must indicate the data port is ready. Otherwise, the read operation is still in progress.

This register is required when accessing the DDR2 DRAM.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|--|------|------------|
| 31:0 | Data Port Data [63:32] (DP_DATA1) | R/W | 0000_0000h |

14.4.6 FIFO Status Register (FIFO_STATUS)

Offset: 0020h Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|--|------|---------|
| 31 | FCT RX FIFO Reset (FCT_RX_RESET) When set, the FCT RX FIFO pointers are reset. Any remnant data from the FIFO stored in the datapath is also cleared. | SC | 0b |
| 30:16 | RESERVED | RO | - |
| 15:0 | RX Data FIFO Free Space (RXDFREE) The amount of free space, in bytes, in the RX Data FIFO. The value returned is rounded down to the nearest QWORD to take into account any payload bytes that do not end on a QWORD boundary. | RO | 4800h |

14.4.7 Video DAC Configuration Register (VDAC_CFG)

Offset: 0024h Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|---|------|---------|
| 31:8 | RESERVED | RO | - |
| 7 | Enable External Reference Current (BYP_IDAC) | R/W | 0b |
| 6 | Enable 34 mA Output Full-Scale (EN_34) | R/W | 0b |
| 5 | Enable Analog Biasing Test Control 1 (EN_CTR1) | R/W | 0b |
| 4 | Enable Analog Biasing Test Control 0 (EN_CTR0) | R/W | 0b |
| 3 | RESERVED | RO | - |
| 2 | Enable DAC Channel 2 (EN_DAC2) Powers up DAC Channel 2. | R/W | 0b |
| 1 | Enable DAC Channel 1 (EN_DAC1) Powers up DAC Channel 1. | R/W | 0b |
| 0 | Enable DAC Channel 0 (EN_DAC0) Powers up DAC Channel 0. | R/W | 0b |

14.4.8 RGB Configuration Register (RGB_CFG)

Offset: 0028h Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|--|------|---------|
| 31:2 | RESERVED | RO | - |
| 1 | VDAC Pad Enable (VDACPAD_EN) This bit disables the VDAC_HSYNC and VDAC_VSYNC pads when cleared. | R/W | 0b |
| 0 | RGB Pad Enable (RGBPAD_EN) This bit disables the RGB pads (VDATA0 - VDATA23, RGB_VSYNC, RGB_HSYNC, nBLANK, and VCLK) when cleared. | R/W | 0b |

14.4.9 External Reset Configuration Register (EXT_RST_CFG)

Offset: 002Ch Size: 32 bits

See [Table 6.1, "Chip Level Resets"](#) for additional description about this reset.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|---|------|---------------------------|
| 31:5 | RESERVED | RO | - |
| 4 | External Reset Force (EXT_RST_FORCE) When set, the external reset is asserted. When clear, the external reset is released. Note: This function is a logical or with the Assert External Reset (EXT_RST_AST) . | R/W | 0b |
| 3:2 | External Reset Period (EXT_RST_PERIOD) Defines the amount of time the nEXTRST pin is asserted. 00 = Assert reset for 50 us 01 = Assert reset for 1 ms 10 = Assert reset for 32 ms 11 = Assert reset for 128 ms | R/W | 00b |
| 1 | Assert External Reset (EXT_RST_AST) Asserts the external reset pin for the amount of time specified by EXT_RST_PERIOD and clears after reset has completed. | R/SC | 0b |
| 0 | External Reset Polarity (EXT_RST_POL) Determines the polarity of the external reset pin (nEXTRST). 0 = active-low 1 = active-high Note: This field is protected by Reset Protection (RST_PROTECT) . | R/W | Note 14.1 |

Note 14.1 The default value of this bit is determined by the value of the [External Reset Polarity \(CFG0_EXT_RST_POL\)](#) bit of [Configuration Flags 0](#) contained within the EEPROM, if present. If no EEPROM is present, 0b is the default. A USB Reset or Lite Reset (LRST) will cause this field to be restored to the image value last loaded from EEPROM, or to be set to 0b if no EEPROM is present.

14.4.10 Audio Configuration Register (AUDIO_CFG)

Offset: 0030h Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|--|------|---------|
| 31 | LED 1 μS TESTMODE (LED_TEST_1US) Setting this test bit causes the audio frequency counters to operate at 1 μ S intervals rather than 1 mS for testing purposes. | R/W | 0b |
| 30:5 | RESERVED | RO | - |
| 4 | I2SCLK Select (I2SCLK_SEL) When I2S Enable (I2S_EN) is set, this bit specifies the pin functioning as the I2SCLK input. 0 = I2SCLKALT0 1 = I2SCLKALT1 | R/W | 0b |
| 3:2 | RESERVED | RO | - |
| 1 | I2S Frequency Measurement Enable (I2S_FREQ_EN) When set, the frequency that each of the I2S pins (WS , I2SCLKALT0/I2SCLKALT1 (depending on setting of I2SCLK Select (I2SCLK_SEL)), MCLK , and I2SDATA) toggles is measured. Once this bit is set, it takes 1ms before a measurement is available. All subsequent measurements are updated at a 1 ms rate. Note: I2S Enable (I2S_EN) must be set in order for the measurement to be occur. | R/W | 0b |
| 0 | I2S Enable (I2S_EN) When this bit is set, I2S is enabled and the I2SCLK Select (I2SCLK_SEL) bit determines the I2SCLK input. | R/W | 0b |

Note: If **I2S Enable (I2S_EN)** is set and **I2SCLKALT1** is selected, then **LED Enable (LED_EN)** must not be set, otherwise unpredictable results and untoward operation will result.

14.4.11 Audio Frequency 0 Register (AUDIO_FREQ0)

Offset: 0034h Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|---|------|---------|
| 31:16 | I2S WS Frequency (WS_FREQ) Measures the frequency that the I2S WS pin toggles at. The units are transitions/ms. Note: The typical audio frequency supported will be 44.1 or 48 KHz. The WS_FREQ is the most appropriate option to determine the sample frequency, since it toggles one clock cycle before the MSB of a new audio word. The hardware only counts positive edges, therefore, the expected values for this field will be 43, 44, or 45 for the 44.1 KHz case and 47, 48, or 49 for the 48 KHz case. Note: I2S Enable (I2S_EN) and I2S Frequency Measurement Enable (I2S_FREQ_EN) bits must be set in order for the measurement to be taken. | RO | X |
| 15:0 | I2S Clock Frequency (I2SCLK_FREQ) Measures the frequency that the I2SCLKALT0/I2SCLKALT1 (depending on setting of I2SCLK Select (I2SCLK_SEL)) pin toggles at. The units are transitions/ms. Note: Only positive clock edges are counted. Note: I2S Enable (I2S_EN) and I2S Frequency Measurement Enable (I2S_FREQ_EN) bits must be set in order for the measurement to be taken. | RO | X |

14.4.12 Audio Frequency 1 Register (AUDIO_FREQ1)

Offset: 0038h Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|--|------|---------|
| 31:16 | I2S MCLK Frequency (MCLK_FREQ) Measures the frequency that the I2S MCLK pin toggles. The units are transitions/ms. Note: Only positive clock edges are counted. Note: I2S Enable (I2S_EN) and I2S Frequency Measurement Enable (I2S_FREQ_EN) bits must be set in order for the measurement to be taken. | RO | X |
| 15:0 | I2S Data Frequency (SD_FREQ) Measures the frequency that the I2SDATA pin toggles. The units are transitions/ms. Note: Only positive clock edges are counted. Note: I2S Enable (I2S_EN) and I2S Frequency Measurement Enable (I2S_FREQ_EN) bits must be set in order for the measurement to be taken. | RO | X |

14.4.13 HDMI Configuration Register (HDMI_CFG)

Offset: 0040h Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|--|------|---------|
| 31:3 | RESERVED | RO | - |
| 2 | Reset HDMI Access Module (RST_HDMI_ACCESS) Resets the HDMI Access Module. | R/SC | 0b |
| 1 | Reset HDMI (RST_HDMI) Asserts hardware resets to HDMI controller for 1 us. | R/SC | 0b |
| 0 | Enable HDMI PHY (EN_HDMI) This register controls the power down pin of the HDMI PHY. | R/W | 0b |

14.4.14 LED Configuration Register 0 (LED_CFG0)

Offset: 0080h Size: 32 bits

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|---|------|---------------------------|
| 31 | LED 1 μS TESTMODE (LED_TEST_1US) Setting this test bit causes the LED to operate at 1 μ S intervals rather than 1 mS for testing purposes. | R/W | 0b |
| 30:4 | RESERVED | RO | - |
| 3 | LED Buffer Type (LED_TYPE) When set, the output buffer for the LED pin is configured as a push/pull driver. When cleared, the LED signal is configured as an open-drain driver. Note: This field is protected by Reset Protection (RST_PROTECT) . | R/W | Note 14.2 |
| 2 | LED Polarity (LED_POL) When set, the LED pin is active high. When cleared, the pin is active low. Note: This field is protected by Reset Protection (RST_PROTECT) . | R/W | Note 14.3 |
| 1 | SW LED Control (SW_LED_CTRL) When asserted while LED Enable (LED_EN) is set, software controls LED blinking. Hardware control of the blinking rate, as a function of traffic rate, is disabled. The value of the Slow Threshold Blink Time (SLOW_THR_BLINK) field of the LED Configuration Register 2 (LED_CFG2) controls the blink rate. Note: If it is desired to utilize the LED as a non-blinking entity under software control, then LED Enable (LED_EN) must not be enabled and the LED pin should be programmed as a GPIO. | R/W | 0b |
| 0 | LED Enable (LED_EN) When asserted, the LED is enabled. Note: This field is protected by Reset Protection (RST_PROTECT) . Note: If I2S Enable (I2S_EN) is set and I2SCLKALT1 is selected, then LED Enable (LED_EN) must not be set, otherwise unpredictable results and untoward operation will result. | R/W | Note 14.4 |

Note 14.2 The default value of this bit is determined by the value of the [LED Buffer Type \(CFG0_LED_TYPE\)](#) bit of [Configuration Flags 0](#) contained within the EEPROM, if present. If no EEPROM is present, 0b is the default. A USB Reset or Lite Reset (LRST) will cause this field to be restored to the image value last loaded from EEPROM, or to be set to 0b if no EEPROM is present.

Note 14.3 The default value of this bit is determined by the value of the [LED Polarity \(CFG0_LED_POL\)](#) bit of [Configuration Flags 0](#) contained within the EEPROM, if present. If no EEPROM is present, 0b is the default. A USB Reset or Lite Reset (LRST) will cause this field to be restored to the image value last loaded from EEPROM, or to be set to 0b if no EEPROM is present.

Note 14.4 The default value of this bit is determined by the value of the [LED Enable \(CFG0_LED_ENABLE\)](#) bit of [Configuration Flags 0](#) contained within the EEPROM, if present. If no EEPROM is present, 1b is the default. A USB Reset or Lite Reset (LRST) will cause this field to be restored to the image value last loaded from EEPROM, or to be set to 1b if no EEPROM is present.

14.4.15 LED Configuration Register 1 (LED_CFG1)

Offset: 0084h Size: 32 bits

This register specifies the traffic rates in units of 64 Bytes/mSec for the LED fast and slow blink thresholds. The hardware will periodically check the traffic rate against these thresholds whenever the **LED Enable (LED_EN)** bit of the **LED Configuration Register 0 (LED_CFG0)** is set and the **SW LED Control (SW_LED_CTRL)** bit is clear. The **Fast Blink Threshold (FAST_THR)** has precedence and is always checked first by the hardware. If the traffic rate is greater than the **Fast Blink Threshold (FAST_THR)**, then the LED is flashed at the rate dictated by the **Fast Threshold Blink Time (FAST_THR_BLINK)** field in **LED Configuration Register 2 (LED_CFG2)**. If the traffic rate does not exceed the **Fast Blink Threshold (FAST_THR)**, then the traffic rate is compared to the **Slow Blink Threshold (SLOW_THR)**. If the traffic rate exceeds the **Slow Blink Threshold (SLOW_THR)**, then the LED is flashed at the rate dictated by the **Slow Threshold Blink Time (SLOW_THR_BLINK)** field in **LED Configuration Register 2 (LED_CFG2)**.

Note: The default values for FAST_THR and SLOW_THR of FFFFh disable both threshold checks. In this case, the LED is steady on. In order to perform one or both of the threshold checks, set these fields to a value other than FFFFh.

Note: The value of this register is not loaded from EEPROM or preserved across resets. Upon reset, in EEPROM emulation mode, the SW must check the **LED Enable (LED_EN)** bit of the **LED Configuration Register 0 (LED_CFG0)**. If it is set, the SW must re-initialize the value of this register.

Note: The 16-bit threshold field size and units of 64 Bytes/mSec allows for a traffic rate range of 512 Kbits/sec to 33 Gigabits/sec. The maximum value allowed here is approximately 10 times the expected maximum rate.

Note: This register provides no functionality when LED blinking rate is under software control (**LED Enable (LED_EN)** and **SW LED Control (SW_LED_CTRL)** both set).

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|---|------|---------|
| 31:16 | Fast Blink Threshold (FAST_THR) When the LED is enabled for hardware traffic monitoring (LED Enable (LED_EN) set, SW LED Control (SW_LED_CTRL) clear), and the traffic rate exceeds this value, the hardware will blink the LED according to the Fast Threshold Blink Time (FAST_THR_BLINK) specified in LED Configuration Register 2 (LED_CFG2) . Note: A value of FFFFh disables the threshold check. If both threshold checks are disabled, the LED will be steady on. Note: This threshold has precedence. Note: Units = 64-bytes/mSec | R/W | FFFFh |
| 15:0 | Slow Blink Threshold (SLOW_THR) When the LED is enabled for hardware traffic monitoring (LED Enable (LED_EN) set, SW LED Control (SW_LED_CTRL) clear), and the traffic rate exceeds this value, the hardware will blink the LED according to the Slow Threshold Blink Time (SLOW_THR_BLINK) specified in LED Configuration Register 2 (LED_CFG2) . Note: A value of FFFFh disables the threshold check. If both threshold checks are disabled, the LED will be steady on. Note: Units = 64-bytes/mSec | R/W | FFFFh |

14.4.16 LED Configuration Register 2 (LED_CFG2)

Offset: 0088h Size: 32 bits

This register specifies the parameters that control the blinking rate of the LED.

When hardware controls the blinking rate as a function of traffic ([LED Enable \(LED_EN\)](#) set, [SW LED Control \(SW_LED_CTRL\)](#) clear) and the traffic rate exceeds the [Slow Blink Threshold \(SLOW_THR\)](#) or the [Fast Blink Threshold \(FAST_THR\)](#), the LED is cycled on and off repeatedly while either condition is in effect.

When software is in control of the blinking rate ([LED Enable \(LED_EN\)](#) and [SW LED Control \(SW_LED_CTRL\)](#) both set), the LED is blinked at the rate specified by [Slow Threshold Blink Time \(SLOW_THR_BLINK\)](#).

Note: The value of this register is not loaded from EEPROM or preserved across resets. Upon reset, in EEPROM emulation mode, the SW must check the [LED Enable \(LED_EN\)](#) bit of the [LED Configuration Register 0 \(LED_CFG0\)](#). If it is set, the SW must re-initialize the value of this register.

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|---|------|---------|
| 31:16 | Fast Threshold Blink Time (FAST_THR_BLINK) Controls the blinking rate when the traffic rate exceeds the Fast Blink Threshold (FAST_THR) and hardware control of LED blinking is in effect. Specifies the time, in mSec, to alternately turn the LED on, then off. | R/W | 00h |
| 15:0 | Slow Threshold Blink Time (SLOW_THR_BLINK) Controls the blinking rate when the traffic rate exceeds the Slow Blink Threshold (SLOW_THR) and hardware control of LED blinking is in effect. Also controls the blinking rate when software control of LED blinking is in effect. Specifies the time, in mSec, to alternately turn the LED on, then off. | R/W | 00h |

14.4.17 GPIO Enable Register (GPIO_ENABLE)

Offset: 00A0h Size: 32 bits

Note: This register is protected by [Reset Protection \(RST_PROTECT\)](#).

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|---|------|---------------------------|
| 31 | RESERVED | RO | - |
| 30:0 | <p>GPIO Enable 0-30 (nGPIOEN) A '1' sets the associated pin to use the default function. When cleared low, the pin functions as a GPIO signal.</p> <p>nGPIOEN0 - bit 0 nGPIOEN1 - bit 1 nGPIOEN2 - bit 2 nGPIOEN3 - bit 3 nGPIOEN4 - bit 4 nGPIOEN5 - bit 5 nGPIOEN6 - bit 6 nGPIOEN7 - bit 7 nGPIOEN8 - bit 8 nGPIOEN9 - bit 9 nGPIOEN10 - bit 10 nGPIOEN11 - bit 11 nGPIOEN12 - bit 12 nGPIOEN13 - bit 13 nGPIOEN14 - bit 14 nGPIOEN15 - bit 15 nGPIOEN16 - bit 16 nGPIOEN17 - bit 17 nGPIOEN18 - bit 18 nGPIOEN19 - bit 19 nGPIOEN20 - bit 20 nGPIOEN21 - bit 21 nGPIOEN22 - bit 22 nGPIOEN23 - bit 23 nGPIOEN24 - bit 24 nGPIOEN25 - bit 25 nGPIOEN26 - bit 26 nGPIOEN27 - bit 27 nGPIOEN28 - bit 28 nGPIOEN29 - bit 29 nGPIOEN30 - bit 30</p> <p>Note: These GPIOs are disabled after a reset.</p> | R/W | Note 14.5 |

Note 14.5 The default value of this register is determined by the value of the following fields within the EEPROM, if present:

[GPIO Enable Config Byte 0 \(GPIOEN_CFG0\)](#)
[GPIO Enable Config Byte 1 \(GPIOEN_CFG1\)](#)
[GPIO Enable Config Byte 2 \(GPIOEN_CFG2\)](#)
[GPIO Enable Config Byte 3 \(GPIOEN_CFG3\)](#)

If no EEPROM is present, 7FFF_FFFFh is the default. A USB Reset or Lite Reset (LRST) will cause this field to be restored to the image value last loaded from EEPROM, or to be set to 7FFF_FFFFh if no EEPROM is present.

14.4.18 GPIO Buffer Type Register (GPIO_BUF)

Offset: 00A4h Size: 32 bits

Note: This register is protected by [Reset Protection \(RST_PROTECT\)](#).

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|---|------|---------------------------|
| 31 | RESERVED | RO | - |
| 30:0 | GPIO Buffer Type 0-30 (GPIOBUF) When set, the output buffer for the corresponding GPIO signal is configured as a push/pull driver. When cleared, the corresponding GPIO set configured as an open-drain driver. GPIOBUF0 - bit 0 GPIOBUF1 - bit 1 GPIOBUF2 - bit 2 GPIOBUF3 - bit 3 GPIOBUF4 - bit 4 GPIOBUF5 - bit 5 GPIOBUF6 - bit 6 GPIOBUF7 - bit 7 GPIOBUF8 - bit 8 GPIOBUF9 - bit 9 GPIOBUF10 - bit 10 GPIOBUF11 - bit 11 GPIOBUF12 - bit 12 GPIOBUF13 - bit 13 GPIOBUF14 - bit 14 GPIOBUF15 - bit 15 GPIOBUF16 - bit 16 GPIOBUF17 - bit 17 GPIOBUF18 - bit 18 GPIOBUF19 - bit 19 GPIOBUF20 - bit 20 GPIOBUF21 - bit 21 GPIOBUF22 - bit 22 GPIOBUF23 - bit 23 GPIOBUF24 - bit 24 GPIOBUF25 - bit 25 GPIOBUF26 - bit 26 GPIOBUF27 - bit 27 GPIOBUF28 - bit 28 GPIOBUF29 - bit 29 GPIOBUF30 - bit 30 | R/W | Note 14.6 |

Note 14.6 The default value of this register is determined by the value of the following fields within the EEPROM, if present:

[GPIO Buffer Type Config Byte 0 \(GPIOBUF_CFG0\)](#)
[GPIO Buffer Type Config Byte 1 \(GPIOBUF_CFG1\)](#)
[GPIO Buffer Type Config Byte 2 \(GPIOBUF_CFG2\)](#)
[GPIO Buffer Type Config Byte 3 \(GPIOBUF_CFG3\)](#)

If no EEPROM is present, 0000_0000h is the default. A USB Reset or Lite Reset (LRST) will cause this field to be restored to the image value last loaded from EEPROM, or to be set to 0000_0000h if no EEPROM is present.

14.4.19 GPIO Direction Register (GPIO_DIR)

Offset: 00A8h Size: 32 bits

Note: This register is protected by [Reset Protection \(RST_PROTECT\)](#).

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|---|------|---------------------------|
| 31 | RESERVED | RO | - |
| 30:0 | GPIO Direction 0-30 (GPDIR) When set, enables the corresponding GPIO as output. When cleared the GPIO is enabled as an input. GPIODIR0 - bit 0 GPIODIR1 - bit 1 GPIODIR2 - bit 2 GPIODIR3 - bit 3 GPIODIR4 - bit 4 GPIODIR5 - bit 5 GPIODIR6 - bit 6 GPIODIR7 - bit 7 GPIODIR8 - bit 8 GPIODIR9 - bit 9 GPIODIR10 - bit 10 GPIODIR11 - bit 11 GPIODIR12 - bit 12 GPIODIR13 - bit 13 GPIODIR14 - bit 14 GPIODIR15 - bit 15 GPIODIR16 - bit 16 GPIODIR17 - bit 17 GPIODIR18 - bit 18 GPIODIR19 - bit 19 GPIODIR20 - bit 20 GPIODIR21 - bit 21 GPIODIR22 - bit 22 GPIODIR23 - bit 23 GPIODIR24 - bit 24 GPIODIR25 - bit 25 GPIODIR26 - bit 26 GPIODIR27 - bit 27 GPIODIR28 - bit 28 GPIODIR29 - bit 29 GPIODIR30 - bit 30 | R/W | Note 14.7 |

Note 14.7 The default value of this register is determined by the value of the following fields within the EEPROM, if present:

[GPIO Direction Config Byte 0 \(GPIODIR_CFG0\)](#)
[GPIO Direction Config Byte 1 \(GPIODIR_CFG1\)](#)
[GPIO Direction Config Byte 2 \(GPIODIR_CFG2\)](#)
[GPIO Direction Config Byte 3 \(GPIODIR_CFG3\)](#)

If no EEPROM is present, 0000_0000h is the default. A USB Reset or Lite Reset (LRST) will cause this field to be restored to the image value last loaded from EEPROM, or to be set to 0000_0000h if no EEPROM is present.

14.4.20 GPIO Data Register (GPIO_DATA)

Offset: 00ACh Size: 32 bits

Note: This register is protected by [Reset Protection \(RST_PROTECT\)](#).

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|---|------|---------------------------|
| 31 | RESERVED | RO | - |
| 30:0 | GPIO Data 0-30 (GPIOD) When enabled as an output, the value written is reflected on GPIODx. When read, GPIODx reflects the current state of the corresponding GPIO pin. GPIOD0 - bit 0 GPIOD1 - bit 1 GPIOD2 - bit 2 GPIOD3 - bit 3 GPIOD4 - bit 4 GPIOD5 - bit 5 GPIOD6 - bit 6 GPIOD7 - bit 7 GPIOD8 - bit 8 GPIOD9 - bit 9 GPIOD10 - bit 10 GPIOD11 - bit 11 GPIOD12 - bit 12 GPIOD13 - bit 13 GPIOD14 - bit 14 GPIOD15 - bit 15 GPIOD16 - bit 16 GPIOD17 - bit 17 GPIOD18 - bit 18 GPIOD19 - bit 19 GPIOD20 - bit 20 GPIOD21 - bit 21 GPIOD22 - bit 22 GPIOD23 - bit 23 GPIOD24 - bit 24 GPIOD25 - bit 25 GPIOD26 - bit 26 GPIOD27 - bit 27 GPIOD28 - bit 28 GPIOD29 - bit 29 GPIOD30 - bit 30 | R/W | Note 14.8 |

Note 14.8 When programmed as an outputs, the default value of enabled output bits of this register is determined by the value of the corresponding bits in the following fields within the EEPROM, if present:

[GPIO Data Config Byte 0 \(GPIOD_CFG0\)](#)
[GPIO Data Config Byte 1 \(GPIOD_CFG1\)](#)
[GPIO Data Config Byte 2 \(GPIOD_CFG2\)](#)
[GPIO Data Config Byte 3 \(GPIOD_CFG3\)](#)

If no EEPROM is present, the pins are not enabled. A USB Reset or Lite Reset (LRST) will cause pins of this register to be disabled if no EEPROM is present, or, if programmed as outputs, to be restored to the image value last loaded from EEPROM.

14.4.21 BOS Attributes Register (BOS_ATTR)

Offset: 0100h Size: 32 bits

This register sets the length values for BOS Block contents that have been loaded into Descriptor RAM via the Data Port registers. The Descriptor RAM images may be used, in conjunction with this register, to facilitate customized operation when no EEPROM is present.

Note: If the block does not exist in Descriptor RAM, its size value must be written as 00h.

Note: This register only affects system operation when an EEPROM is not present and the [EEPROM Emulation Enable \(EEM\)](#) bit indicates Descriptor RAM and the Attributes Registers are to be used for descriptor processing.

Note: Writing to this register when an EEPROM is present is prohibited and will result in untoward operation and unexpected results.

Note: Writing to this register in UFX6000 when [LPM Enable \(LPM_ENABLE\)](#) is clear in the [USB Configuration Register \(USB_CFG\)](#) is prohibited and will result in untoward operation and unexpected results.

Note: This register is protected by [Reset Protection \(RST_PROTECT\)](#).

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|---|------|---------|
| 31:8 | RESERVED | RO | - |
| 7:0 | BOS Block Size (BOS_BLOCK_SIZE) Note 14.9 | R/W | 00h |

Note 14.9 In UFX6000 ([LPM Enable \(LPM_ENABLE\)](#) set) this block may include Binary Object Store (BOS) Descriptor, USB 2.0 Extension Descriptor, and Container ID Descriptor.

In UFX7000, this block may include Binary Object Store (BOS) Descriptor, USB 2.0 Extension Descriptor, Super-Speed Device Capabilities Descriptor, and Container ID Descriptor.

14.4.22 SS Attributes Register (SS_ATTR)

Offset: 0104h Size: 32 bits

This register sets the length values for an SS Device Descriptor and/ or SS Configuration Block elements that have been loaded into Descriptor RAM via the Data Port registers. The SS Polling interval is also defined by a field within this register. The Descriptor RAM images may be used, in conjunction with this register, to facilitate customized operation when no EEPROM is present.

Note: If the Device Descriptor descriptor or the Configuration Block does not exist in Descriptor RAM, its size value must be written as 00h.

Note: This register only affects system operation when an EEPROM is not present and the [EEPROM Emulation Enable \(EEM\)](#) bit indicates Descriptor RAM and the Attributes Registers are to be used for descriptor processing.

Note: Writing to this register when an EEPROM is present is prohibited and will result in untoward operation and unexpected results.

Note: This register is protected by [Reset Protection \(RST_PROTECT\)](#).

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|--|------|---------|
| 31:24 | RESERVED | RO | - |
| 23:16 | SS Polling Interval (SS_POLL_INT) | R/W | 06h |
| 15:8 | SS Device Descriptor Size (SS_DEV_DESC_SIZE) Note 14.10 | R/W | 00h |
| 7:0 | SS Configuration Block Size (SS_CFG_BLOCK_SIZE) Note 14.11 | R/W | 00h |

Note 14.10 The only legal values are 0 and 0x12h. Writing any other values will result in untoward behavior and unexpected results.

Note 14.11 The only legal values are 0 and 0x1Eh. If present in Descriptor RAM, this block **must** include the following descriptors in the following order:
SS Configuration descriptor
SS Interface descriptor
Bulk-Out Endpoint Companion descriptor
Interrupt Endpoint Companion descriptor.

14.4.23 HS Attributes Register (HS_ATTR)

Offset: 0108h Size: 32 bits

This register sets the length values for HS descriptors that have been loaded into Descriptor RAM via the Data Port registers. The HS Polling interval is also defined by a field within this register. The Descriptor RAM images may be used, in conjunction with this register, to facilitate customized operation when no EEPROM is present.

Note: If a descriptor does not exist in Descriptor RAM, its size value must be written as 00h.

Note: This register only affects system operation when an EEPROM is not present and the [EEPROM Emulation Enable \(EEM\)](#) bit indicates Descriptor RAM and the Attributes Registers are to be used for descriptor processing.

Note: Writing to this register when an EEPROM is present is prohibited and will result in untoward operation and unexpected results.

Note: This register is protected by [Reset Protection \(RST_PROTECT\)](#).

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|--|------|---------|
| 31:24 | RESERVED | RO | - |
| 23:16 | HS Polling Interval (HS_POLL_INT) | R/W | 04h |
| 15:8 | HS Device Descriptor Size (HS_DEV_DESC_SIZE) Note 14.12 | R/W | 00h |
| 7:0 | HS Configuration Descriptor Size (HS_CFG_DESC_SIZE) Note 14.13 | R/W | 00h |

Note 14.12 The only legal values are 0 and 0x12h. Writing any other values will result in untoward behavior and unexpected results.

Note 14.13 The only legal values are 0 and 0x12h. Writing any other values will result in untoward behavior and unexpected results.

14.4.24 FS Attributes Register (FS_ATTR)

Offset: 010Ch Size: 32 bits

This register sets the length values for FS descriptors that have been loaded into Descriptor RAM via the Data Port registers. The FS Polling interval is also defined by a field within this register. The Descriptor RAM images may be used, in conjunction with this register, to facilitate customized operation when no EEPROM is present.

Note: If a descriptor does not exist in Descriptor RAM, its size value must be written as 00h.

Note: This register only affects system operation when an EEPROM is not present and the [EEPROM Emulation Enable \(EEM\)](#) bit indicates Descriptor RAM and the Attributes Registers are to be used for descriptor processing.

Note: Writing to this register when an EEPROM is present is prohibited and will result in untoward operation and unexpected results.

Note: This register is protected by [Reset Protection \(RST_PROTECT\)](#).

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|--|------|---------|
| 31:24 | RESERVED | RO | - |
| 23:16 | FS Polling Interval (FS_POLL_INT) | R/W | 01h |
| 15:8 | FS Device Descriptor Size (FS_DEV_DESC_SIZE) Note 14.14 | R/W | 00h |
| 7:0 | FS Configuration Descriptor Size (FS_CFG_DESC_SIZE) Note 14.15 | R/W | 00h |

Note 14.14 The only legal values are 0 and 0x12h. Writing any other values will result in untoward behavior and unexpected results.

Note 14.15 The only legal values are 0 and 0x12h. Writing any other values will result in untoward behavior and unexpected results.

14.4.25 String Attributes Register 0 (STRNG_ATTR0)

Offset: 0110h Size: 32 bits

This register sets the length values for the named string descriptors that have been loaded into Descriptor RAM via the Data Port registers. The Descriptor RAM images may be used, in conjunction with this register, to facilitate customized operation when no EEPROM is present.

Note: If a descriptor does not exist in Descriptor RAM, its size value must be written as 00h.

Note: This register only affects system operation when an EEPROM is not present and the [EEPROM Emulation Enable \(EEM\)](#) bit indicates Descriptor RAM and the Attributes Registers are to be used for descriptor processing.

Note: Writing to this register when an EEPROM is present is prohibited and will result in untoward operation and unexpected results.

Note: This register is protected by [Reset Protection \(RST_PROTECT\)](#).

| BITS | DESCRIPTION | TYPE | DEFAULT |
|-------|---|------|---------|
| 31:24 | Configuration String Descriptor Size (CFGSTR_DESC_SIZE) | R/W | 00h |
| 23:16 | Serial Number String Descriptor Size (SERSTR_DESC_SIZE) | R/W | 00h |
| 15:8 | Product Name String Descriptor Size (PRODSTR_DESC_SIZE) | R/W | 00h |
| 7:0 | Manufacturing String Descriptor Size (MANUF_DESC_SIZE) | R/W | 00h |

14.4.26 String Attributes Register 1 (STRNG_ATTR1)

Offset: 0114h Size: 32 bits

This register sets the length values for the named string descriptors that have been loaded into Descriptor RAM via the Data Port registers. The Descriptor RAM images may be used, in conjunction with this register, to facilitate customized operation when no EEPROM is present.

Note: If a descriptor does not exist in Descriptor RAM, its size value must be written as 00h.

Note: This register only affects system operation when an EEPROM is not present and the [EEPROM Emulation Enable \(EEM\)](#) bit indicates Descriptor RAM and the Attributes Registers are to be used for descriptor processing.

Note: Writing to this register when an EEPROM is present is prohibited and will result in untoward operation and unexpected results.

Note: This register is protected by [Reset Protection \(RST_PROTECT\)](#).

| BITS | DESCRIPTION | TYPE | DEFAULT |
|------|---|------|---------|
| 31:8 | RESERVED | RO | - |
| 7:0 | Interface String Descriptor Size (INTSTR_DESC_SIZE) | R/W | 00h |

Chapter 15 Operational Characteristics

15.1 Absolute Maximum Ratings*

| | |
|---|--------------------------------|
| +3.3V Supply Voltage (VDD33IO, VDD33USB, VDD33USB3, VDD33VDAC, SYSPLLP) (Note 15.1) | 0V to +3.6V |
| +1.8V Supply Voltage (VDD18DDR) (Note 15.1) | 0V to +1.9V |
| +1.2V Supply Voltage (VDD12CORE, VDD12USB3, VDD12USBPLL, VDD12HDMI) (Note 15.1) | 0V to +1.32V |
| Positive voltage on XI, with respect to ground. | +4.6V |
| Positive voltage on XO, with respect to ground. | +2.5V |
| Storage Temperature | -55°C to +150°C |
| Lead Temperature Range | Refer to JEDEC Spec. J-STD-020 |
| HBM ESD Performance | JEDEC Class 2 |

Note 15.1 When powering this device from laboratory or system power supplies, it is important that the absolute maximum ratings not be exceeded or device failure can result. Some power supplies exhibit voltage spikes on their outputs when AC power is switched on or off. In addition, voltage transients on the AC power line may appear on the DC output. If this possibility exists, it is suggested that a clamp circuit be used.

*Stresses exceeding those listed in this section could cause permanent damage to the device. This is a stress rating only. Exposure to absolute maximum rating conditions for extended periods may affect device reliability. Functional operation of the device at any condition exceeding those indicated in [Section 15.2, "Operating Conditions**"](#), [Section 15.5, "DC Specifications"](#), or any other applicable section of this specification is not implied. Note, device signals are *NOT* 5 volt tolerant unless specified otherwise.

15.2 Operating Conditions**

| | |
|---|--------------|
| +3.3V Supply Voltage (VDD33IO, VDD33USB, VDD33USB3, VDD33VDAC, SYSPLLP) | +3.3V +/- 5% |
| +1.8V Supply Voltage (VDD18DDR) | +1.8V +/- 5% |
| +1.2V Supply Voltage (VDD12CORE, VDD12USB3, VDD12USBPLL, VDD12HDMI) | +1.2V +/- 5% |
| Ambient Operating Temperature in Still Air (T _A) | 0°C to +70°C |
| Maximum Junction Temperature | +110°C |

**Proper operation of the device is guaranteed only within the ranges specified in this section.

15.3 Package Thermal Specifications

Table 15.1 Package Thermal Parameters

| PARAMETER | SYMBOL | VALUE | UNITS | COMMENTS |
|----------------------------|---------------|-------|-------|---|
| Thermal Resistance | Θ_{JA} | 28.7 | °C/W | Measured in still air from the die to ambient air |
| Thermal Resistance | Θ_{JC} | 10.4 | °C/W | Measured from the die to the case |
| Junction-to-Top-of-Package | Ψ_{JT} | 0.38 | °C/W | Measured in still air |

Note: Thermal parameters are measured or estimated for devices in a multi-layer 2S2P PCB per JESD51.

15.4 Current Consumption

This section details the current consumption of the device as measured during various modes of operation and power states. Current consumption values are provided for each power rail (+3.3V, +1.8V, +1.2V). Power dissipation is determined by temperature, supply voltage, and external source/sink requirements.

Note: All current consumption values were measured with power supplies at nominal voltages unless otherwise noted.

15.4.1 SUSPEND Power State

Table 15.2 SUSPEND Supply Current

| PARAMETER | TYPICAL | UNIT |
|---|---------|------|
| +3.3V Supply Current (Device Only) (VDD33IO, VDD33USB, VDD33USB3, VDD33VDAC, SYSPLL) | 0.7 | mA |
| +1.8V Supply Current (Device Only) (VDD18DDR) | 0.0 | mA |
| +1.2V Supply Current (Device Only) (VDD12CORE, VDD12USB3, VDD12USBPLL, VDD12HDMI) | 1.5 | mA |

15.4.2 Operational

15.4.2.1 Super-Speed

Table 15.3 Typical Super-Speed Operational Supply Current (mA)

| PARAMETER | STATIC IMAGE | | | FULL SCREEN VIDEO | | |
|--|-------------------------|-------------------------|-------------------------|-------------------------|-------------------------|-------------------------|
| | 1280X1024 (DDR2-400) | 1600X1200 (DDR2-533) | 1920X1200 (DDR2-667) | 1280X1024 (DDR2-400) | 1600X1200 (DDR2-533) | 1920X1200 (DDR2-667) |
| Video DAC Interface Enabled | | | | | | |
| +3.3V Supply Current (Dev. Only) (VDD33IO, VDD33USB, VDD33USB3, VDD33VDAC, SYSPLL) | 102 | 102 | 104 | 102 | 102 | 104 |
| +1.8V Supply Current (Dev. Only) (VDD18DDR) | 39 | 45 | 48 | 59 | 65 | 74 |
| +1.2V Supply Current (Dev. Only) (VDD12CORE, VDD12USB3, VDD12USBPLL, VDD12HDMI) | 287 | 295 | 298 | 310 | 319 | 325 |
| HDMI Interface Enabled (with Audio) | | | | | | |
| +3.3V Supply Current (Dev. Only) (VDD33IO, VDD33USB, VDD33USB3, VDD33VDAC, SYSPLL) | 29 | 29 | 29 | 29 | 29 | 29 |
| +1.8V Supply Current (Dev. Only) (VDD18DDR) | 39 | 45 | 49 | 58 | 67 | 74 |
| +1.2V Supply Current (Dev. Only) (VDD12CORE, VDD12USB3, VDD12USBPLL, VDD12HDMI) | 307 | 323 | 323 | 327 | 346 | 350 |

15.4.2.2 High-Speed

Table 15.4 Typical High-Speed Operational Supply Current (mA)

| PARAMETER | STATIC IMAGE | | | FULL SCREEN VIDEO | | |
|--|-------------------------|-------------------------|-------------------------|-------------------------|-------------------------|-------------------------|
| | 1280X1024 (DDR2-400) | 1600X1200 (DDR2-533) | 1920X1200 (DDR2-667) | 1280X1024 (DDR2-400) | 1600X1200 (DDR2-533) | 1920X1200 (DDR2-667) |
| Video DAC Interface Enabled | | | | | | |
| +3.3V Supply Current (Dev. Only) (VDD33IO, VDD33USB, VDD33USB3, VDD33VDAC, SYSPLL) | 77 | 78 | 79 | 78 | 78 | 79 |
| +1.8V Supply Current (Dev. Only) (VDD18DDR) | 39 | 45 | 48 | 58 | 67 | 75 |
| +1.2V Supply Current (Dev. Only) (VDD12CORE, VDD12USB3, VDD12USBPLL, VDD12HDMI) | 181 | 189 | 192 | 203 | 215 | 222 |
| HDMI Interface Enabled (with audio) | | | | | | |
| +3.3V Supply Current (Dev. Only) (VDD33IO, VDD33USB, VDD33USB3, VDD33VDAC, SYSPLL) | 5.0 | 5.0 | 5.0 | 5.2 | 5.2 | 5.2 |
| +1.8V Supply Current (Dev. Only) (VDD18DDR) | 39 | 45 | 49 | 60 | 67 | 76 |
| +1.2V Supply Current (Dev. Only) (VDD12CORE, VDD12USB3, VDD12USBPLL, VDD12HDMI) | 201 | 218 | 218 | 223 | 242 | 255 |

15.5 DC Specifications

Table 15.5 I/O Buffer Characteristics

| PARAMETER | SYMBOL | MIN | TYP | MAX | UNITS | NOTES |
|---|--------------|--------------------------|-----------|---------------------|-------|---------------------------|
| IS Type Input Buffer | | | | | | |
| Low Input Level | V_{ILI} | -0.3 | | | V | |
| High Input Level | V_{IHI} | | | 3.6 | V | |
| Negative-Going Threshold | V_{ILT} | 1.01 | 1.19 | 1.39 | V | Schmitt trigger |
| Positive-Going Threshold | V_{IHT} | 1.39 | 1.59 | 1.8 | V | Schmitt trigger |
| SchmittTrigger Hysteresis ($V_{IHT} - V_{ILT}$) | V_{HYS} | 336 | 399 | 485 | mV | |
| O8 Type Output Buffer | | | | | | |
| Low Output Level | V_{OL} | | | 0.4 | V | $I_{OL} = 8\text{mA}$ |
| High Output Level | V_{OH} | $V_{DD33IO} - 0.4$ | | | V | $I_{OH} = -8\text{mA}$ |
| OD8 Type Output Buffer | | | | | | |
| Low Output Level | V_{OL} | | | 0.4 | V | $I_{OL} = 8\text{mA}$ |
| RGB Type Output Buffer | | | | | | |
| Low Output Level | V_{OL} | | | 0.4 | V | $I_{OL} = 8\text{mA}$ |
| High Output Level | V_{OH} | $V_{DD33IO} - 0.4$ | | | V | $I_{OH} = -8\text{mA}$ |
| DDR2I Type Input Buffer | | | | | | Note 15.2 |
| Termination Voltage | V_{TT} | $V_{REF} - 0.04$ | V_{REF} | $V_{REF} + 0.04$ | V | Note 15.3 |
| Low Input Level (DC) | $V_{IL(dc)}$ | -0.3 | | $V_{REF} - 0.125$ | V | Note 15.3 |
| High Input Level (DC) | $V_{IH(dc)}$ | $V_{REF} + 0.125$ | | $V_{DD18DDR} + 0.3$ | V | Note 15.3 |
| Low Input Level (AC) | $V_{IL(ac)}$ | | | $V_{REF} - 0.25$ | V | Note 15.3 |
| High Input Level (AC) | $V_{IH(ac)}$ | $V_{REF} + 0.25$ | | | V | Note 15.3 |
| DDR2O Type Output Buffer | | | | | | Note 15.2 |
| Termination Voltage | V_{TT} | $V_{REF} - 0.04$ | V_{REF} | $V_{REF} + 0.04$ | V | Note 15.3 |
| Low Output Level (DC) | $V_{OL(dc)}$ | | | 0.28 | V | Note 15.4 |
| High Output Level (DC) | $V_{OH(dc)}$ | $V_{DD18DDR(min)} - 0.2$ | | | V | Note 15.5 |
| Low Output Level (AC) | $V_{OL(ac)}$ | | | $V_{TTmin} - 0.603$ | V | Note 15.4 |
| High Output Level (AC) | $V_{OH(ac)}$ | $V_{TTmax} + 0.603$ | | | V | Note 15.5 |
| ICLK Type Buffer (XI Input) | | | | | | Note 15.6 |
| Low Input Level | V_{ILI} | -0.3 | | 0.5 | V | |
| High Input Level | V_{IHI} | 1.08 | | 1.32 | V | |

Note 15.2 All values apply to both full-strength and half-strength operation unless otherwise stated.

Note 15.3 V_{REF} equals $DDR_{VREF}[0:2]$.

Note 15.4 I_{OL} equals 13.4mA for full-strength operation and 6.7mA for half-strength operation.

Note 15.5 I_{OH} equals -13.4mA for full-strength operation and -6.7mA for half-strength operation.

Note 15.6 XI can optionally be driven from a 25MHz single-ended clock oscillator. A 25MHz oscillator, or other single-ended clock source that meets the ICLK DC buffer characteristics and the specifications in [Section 15.7, "Clock Circuit," on page 383](#), is required when utilizing the Digital RGB interface. Do not use a crystal when operating in Digital RGB mode.

Table 15.6 Video DAC - DC Characteristics

| PARAMETER | SYMBOL | MIN | TYP | MAX | UNITS |
|----------------------------------|--------|-----|------|--------|-------|
| Output Voltage | - | | 1.28 | | V |
| Output Current per Channel | - | | 17 | | mA |
| Video DAC Resolution | - | | 10 | | bits |
| Integral Non-linearity Error | INL | | | +/-2 | LSB |
| Differential Non-linearity Error | DNL | | | +/-0.5 | LSB |

15.6 AC Specifications

This section details the various AC timing specifications of the device.

Note: The USB interface timing adheres to the USB 3.0 Specification. Refer to the Universal Serial Bus Revision 3.0 Specification for detailed USB timing information.

Note: The DDR2 interface timing adheres to the JESD79-2E Specification. Refer to the JESD79-2E Specification for detailed DDR2 timing information.

Note: The HDMI interface timing adheres to the HDMI 1.3 Specification. Refer to the HDMI 1.3 Specification for detailed HDMI timing information.

Note: The S/PDIF interface timing adheres to the IEC 60958 2-channel PCM Specification. Refer to the IEC 60958 2-channel PCM Specification for detailed S/PDIF timing information.

Note: The I²S interface timing adheres to the NXP I²S Bus Specification. Refer to the NXP I²S Bus Specification for detailed I²S timing information.

Note: The I²C interface timing adheres to the NXP I²C-Bus Specification. Refer to the I²C-Bus Specification for detailed I²C timing information.

15.6.1 Power Sequence Timing

Power supplies must adhere to the following rules:

- All power supplies of the same voltage must be powered up/down together.
- There is no power-up sequencing requirement, however all power supplies must reach operational levels within the time periods specified in [Table 15.7](#).
- There is no power-down sequencing or timing requirement, however the device must not be powered for an extended period of time without all supplies at operational levels.
- Following power-on, or if a power supply brownout occurs (i.e., one or more supplies drops below operational limits), a power-on reset must be executed once all power supplies reach operational levels. Refer to section [Section 15.6.2, "Power-On Reset Timing," on page 377](#) for power-on reset requirements.
- With the exception of HPD, VBUS_DET, I2CSDA[0:1], and I2CSCL[0:1], do not drive input signals without power supplied to the device.

Note: Violation of these specifications may damage the device.

Note: Power sequencing requirements are preliminary and subject to change.

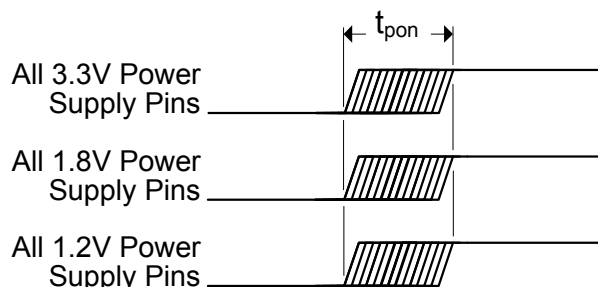


Figure 15.1 Power-On Timing

Table 15.7 Power-On Timing Values

| SYMBOL | DESCRIPTION | MIN | TYP | MAX | UNITS |
|-----------|---------------------------|-----|-----|-----|-------|
| t_{pon} | Power supply turn on time | 0 | | 25 | mS |

15.6.2 Power-On Reset Timing

This diagram illustrates the nRESET timing requirements in relation to power-on. A hardware reset (nRESET assertion) is required following power-up. For proper operation, nRESET must be asserted for no less than t_{rstia} . The nRESET pin can be asserted at any time, but must not be deasserted before t_{purstd} after all external power supplies have reached operational levels.

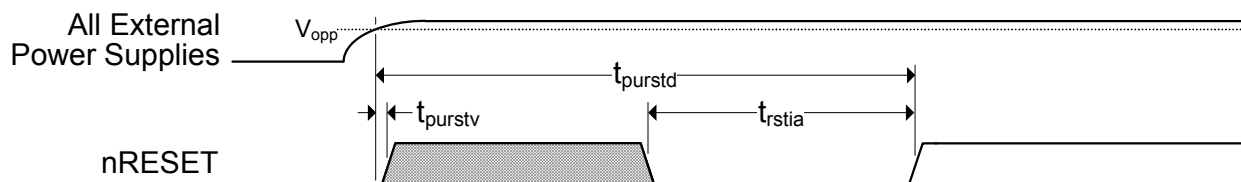


Figure 15.1 nRESET Power-On Timing

Table 15.8 nRESET Power-On Timing Values

| SYMBOL | DESCRIPTION | MIN | TYP | MAX | UNITS |
|--------------|---|-----|-----------|-----|---------|
| t_{purstd} | External power supplies at operational levels to nRESET deassertion | 25 | Note 15.7 | | mS |
| t_{purstv} | External power supplies at operational levels to nRESET valid | 0 | | | nS |
| t_{rstia} | nRESET input assertion time | 100 | | | μ S |

Note: nRESET deassertion must be monotonic.

Note 15.7 For bus-powered applications, a typical value of 200 mS is recommended to allow time for connector mating. Permanently attached and/or self-powered applications do not require this longer reset time.

15.6.3 Reset Timing

Figure 15.1 illustrates the nRESET pin timing requirements. When used, nRESET must be asserted for no less than t_{rstia} .

Note: A hardware reset (nRESET assertion) is required following power-on. Refer to [Section 15.6.2, "Power-On Reset Timing,"](#) on page 377 for additional information.

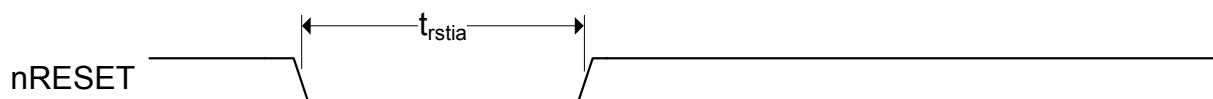


Figure 15.1 nRESET Timing

Table 15.9 nRESET Timing Values

| SYMBOL | DESCRIPTION | MIN | TYP | MAX | UNITS |
|-------------|-----------------------------|-----|-----|-----|-------|
| t_{rstia} | nRESET input assertion time | 1 | | | uS |

15.6.4 Video DAC Timing

The following table specifies the Video DAC timing characteristics for the device. All values are measured with the Video DAC in 17mA full scale mode.

Table 15.10 Video DAC - AC Characteristics

| PARAMETER | MIN | TYP | MAX | UNITS |
|-----------------------------|-----|------|-----|-------|
| Frequency | 25 | | 200 | MHz |
| Analog Output Delay | 0.4 | 0.5 | 0.8 | nS |
| Analog Output Rise Time | | 0.31 | | nS |
| Analog Output Fall Time | | 0.5 | | nS |
| Analog Output Settling Time | | 0.7 | | nS |

15.6.5 Digital RGB Timing

The following sub-sections specify the Digital RGB timing requirements for the device in DDR and SDR modes of operation.

15.6.5.1 DDR Mode

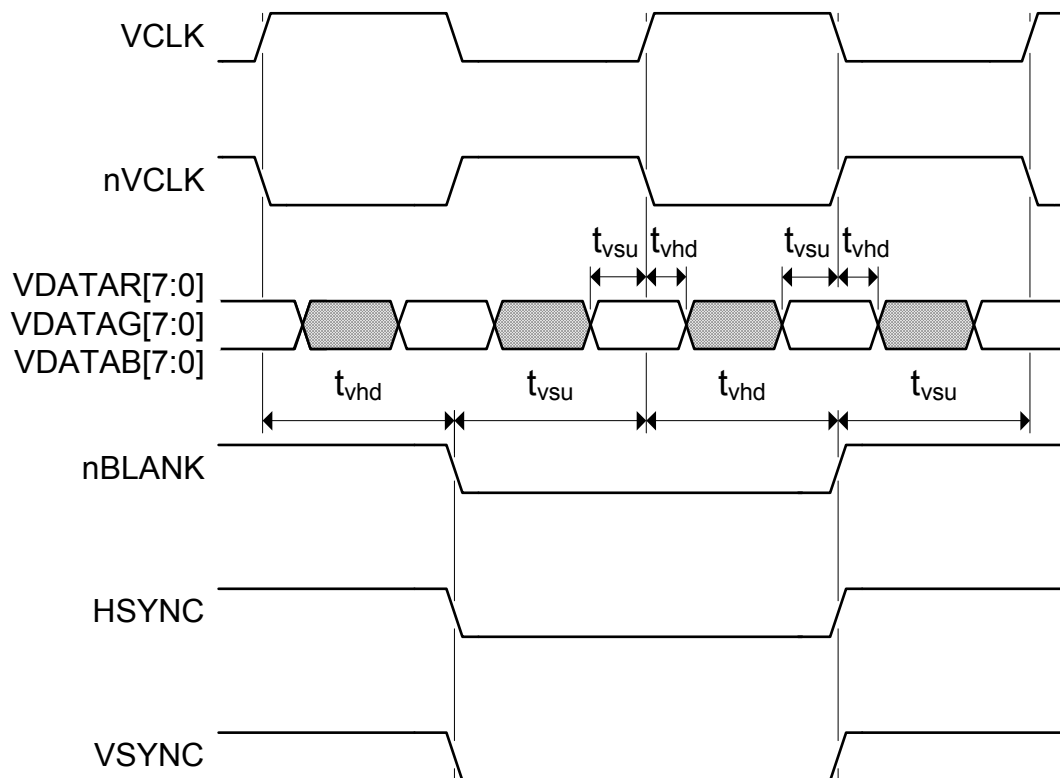


Figure 15.2 Digital RGB Timing - DDR Mode

Table 15.11 Digital RGB Timing Values - DDR Mode

| SYMBOL | DESCRIPTION | MIN | TYP | MAX | UNITS |
|------------|--------------------------|-----|-----|-----|-------|
| f_{vclk} | VCLK Frequency | | | 165 | MHz |
| t_{vsu} | Video Setup Output Delay | 0.8 | | | nS |
| t_{vhd} | Video Hold Output Delay | 0.5 | | | nS |

Note: RGB timing values are with respect to an equivalent test load of 5 pF.

15.6.5.2 SDR Mode

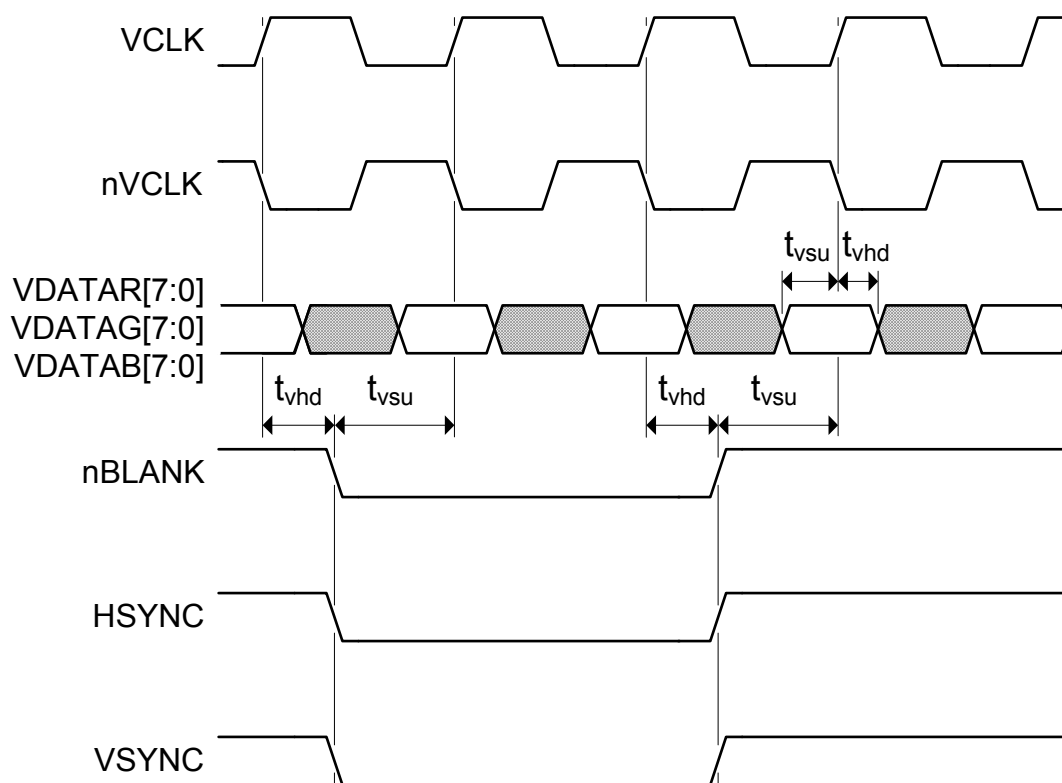


Figure 15.3 Digital RGB Timing - SDR Mode

Table 15.12 Digital RGB Timing Values - SDR Mode

| SYMBOL | DESCRIPTION | MIN | TYP | MAX | UNITS |
|------------|--------------------------|-----|-----|-----|-------|
| f_{vclk} | VCLK Frequency | | | 165 | MHz |
| t_{vsu} | Video Setup Output Delay | 2.5 | | | nS |
| t_{vhd} | Video Hold Output Delay | 1.5 | | | nS |

Note: RGB timing values are with respect to an equivalent test load of 5 pF.

15.6.6 EEPROM Timing

The following specifies the EEPROM timing requirements for the device:

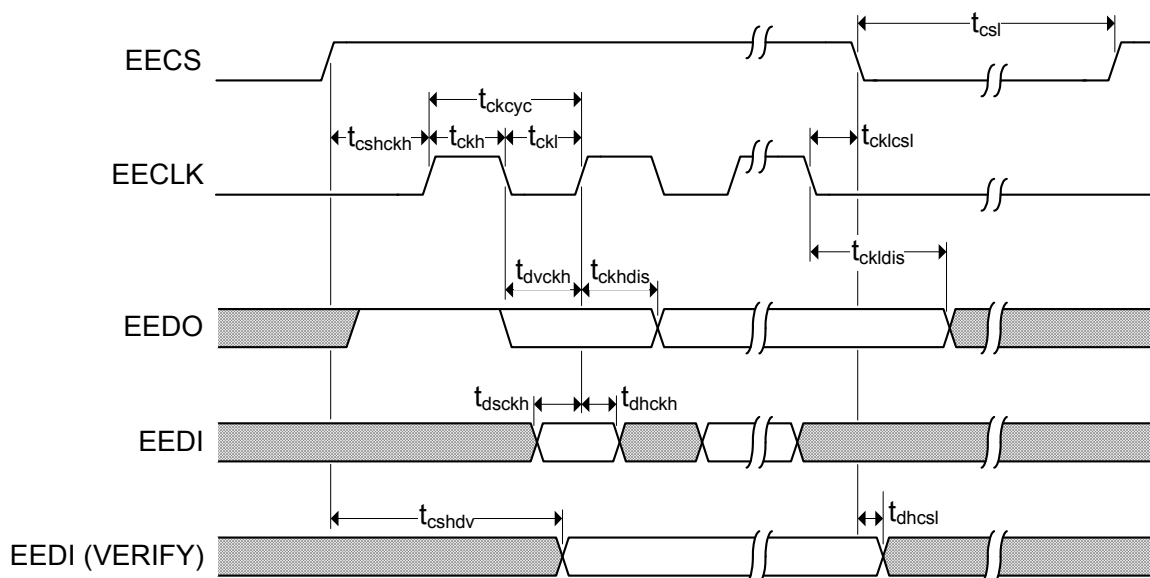


Figure 15.4 EEPROM Timing

Table 15.13 EEPROM Timing Values

| SYMBOL | DESCRIPTION | MIN | TYP | MAX | UNITS |
|-----------------|--|------|-----|------|-------|
| t_{ckcyc} | EECLK Cycle time | 1110 | | 1130 | ns |
| t_{ckh} | EECLK High time | 550 | | 570 | ns |
| t_{ckl} | EECLK Low time | 550 | | 570 | ns |
| t_{cshckh} | EECS high before rising edge of EECLK | 1070 | | | ns |
| t_{cklcs1} | EECLK falling edge to EECS low | 30 | | | ns |
| t_{dvckh} | EEDO valid before rising edge of EECLK | 550 | | | ns |
| $t_{ckhinvald}$ | EEDO invalid after rising edge EECLK | 550 | | | ns |
| t_{dsckh} | EEDI setup to rising edge of EECLK | 90 | | | ns |
| t_{dhckh} | EEDI hold after rising edge of EECLK | 0 | | | ns |
| t_{ckldis} | EECLK low to data disable (OUTPUT) | 580 | | | ns |
| t_{cshdv} | EEDIO valid after EECS high (VERIFY) | | | 600 | ns |
| t_{dhcsl} | EEDIO hold after EECS low (VERIFY) | 0 | | | ns |
| t_{cs1} | EECS low | 1070 | | | ns |

Note: EEPROM timing values are with respect to an equivalent test load of 25 pF.

15.6.7 JTAG Timing

This section specifies the JTAG timing of the device.

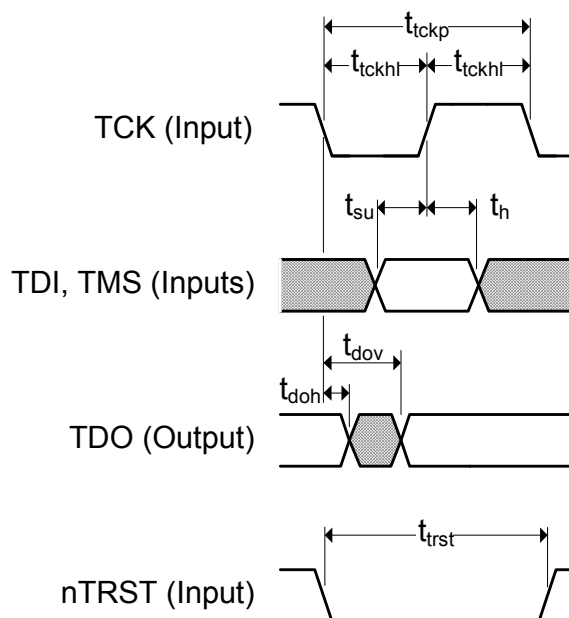


Figure 15.5 JTAG Timing

Table 15.14 JTAG Timing Values

| SYMBOL | DESCRIPTION | MIN | MAX | UNITS | NOTES |
|---------------|--|------------------|------------------|-------|-------|
| t_{tckp} | TCK clock period | 66.67 | 80 | nS | |
| t_{tckhl} | TCK clock high/low time | $t_{tckp} * 0.4$ | $t_{tckp} * 0.6$ | nS | |
| t_{su} | TDI, TMS setup to TCK rising edge | 10 | | nS | |
| t_h | TDI, TMS hold from TCK rising edge | 10 | | nS | |
| t_{dov} | TDO output valid from TCK falling edge | | 16 | nS | |
| $t_{doinvld}$ | TDO output invalid from TCK falling edge | 0 | | nS | |
| t_{trst} | nTRST assertion time | 10 | | mS | |

Note: JTAG timing values are with respect to an equivalent test load of 25 pF.

15.7 Clock Circuit

The device can accept either a 25MHz crystal (preferred) or a 25MHz single-ended clock oscillator (+/- 50ppm) input. If the single-ended clock oscillator method is implemented, XO should be left unconnected and XI should be driven with a nominal 0-1.2V clock signal. The input clock duty cycle is 40% minimum, 50% typical and 60% maximum.

It is recommended that a crystal utilizing matching parallel load capacitors be used for the crystal input/output signals (XI/XO). See [Table 15.15](#) for the recommended crystal specifications.

Note: A 25MHz oscillator, or other single-ended clock source that meets the specifications of this section and [Section 15.5, "DC Specifications," on page 374](#), is required when utilizing the Digital RGB interface. Do not use a crystal when operating in Digital RGB mode.

Table 15.15 Crystal Specifications

| PARAMETER | SYMBOL | MIN | NOM | MAX | UNITS | NOTES |
|-------------------------------|------------------------|-----|-----------|--------|-------|----------------------------|
| Crystal Cut | AT, typ | | | | | |
| Crystal Oscillation Mode | Fundamental Mode | | | | | |
| Crystal Calibration Mode | Parallel Resonant Mode | | | | | |
| Frequency | F_{fund} | - | 25.000 | - | MHz | |
| Frequency Tolerance @ 25°C | F_{tol} | - | - | +/-50 | PPM | Note 15.8 |
| Frequency Stability Over Temp | F_{temp} | - | - | +/-100 | PPM | Note 15.8 |
| Frequency Deviation Over Time | F_{age} | - | +/-3 to 5 | - | PPM | Note 15.9 |
| Total Allowable PPM Budget | | - | - | +/-150 | PPM | |
| Shunt Capacitance | C_O | - | 7 typ | - | pF | |
| Load Capacitance | C_L | - | 20 typ | - | pF | |
| Drive Level | P_W | 300 | - | - | uW | |
| Equivalent Series Resistance | R_1 | - | - | 50 | Ohm | |
| Operating Temperature Range | | 0 | - | 70 | °C | |
| XI Pin Capacitance | | - | 3 typ | - | pF | Note 15.10 |
| XO Pin Capacitance | | - | 3 typ | - | pF | Note 15.10 |

Note 15.8 The maximum allowable values for Frequency Tolerance and Frequency Stability are application dependant.

Note 15.9 Frequency Deviation Over Time is also referred to as Aging.

Note 15.10 This number includes the pad, the bond wire and the lead frame. PCB capacitance is not included in this value. The XO/XI pin and PCB capacitance values are required to accurately calculate the value of the two external load capacitors. These two external load capacitors determine the accuracy of the 25.000 MHz frequency.

Chapter 16 Package Outline

16.1 225-LFBGA Package

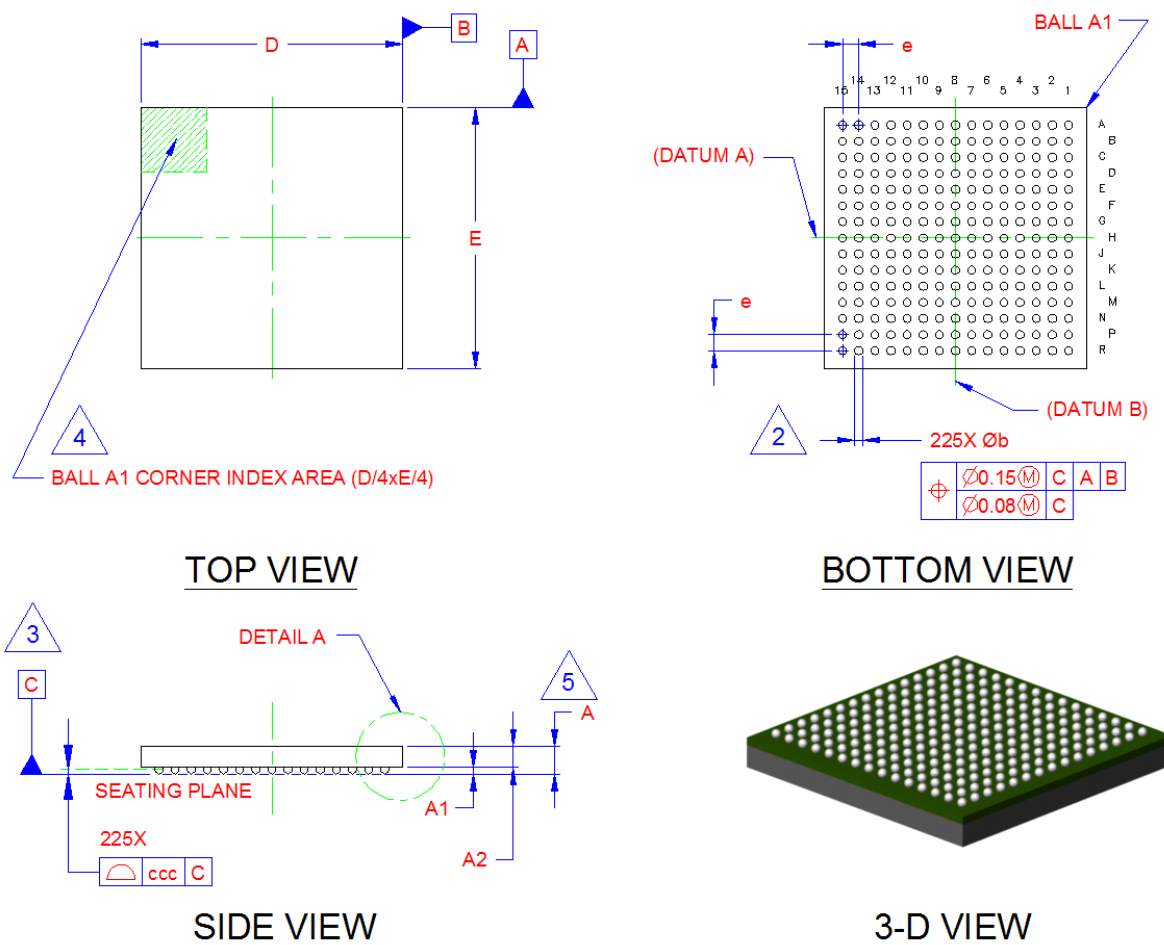


Figure 16.1 UFX7000 225-LFBGA Package Definition

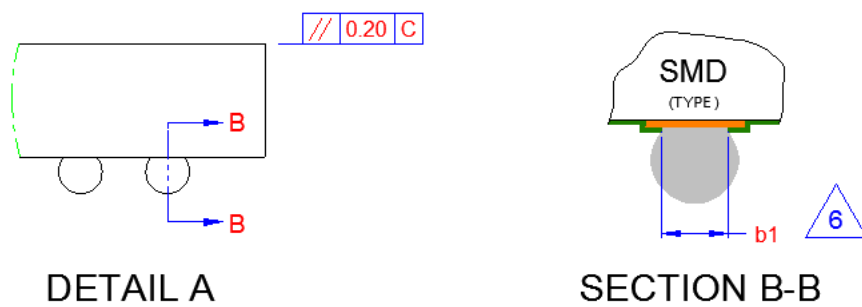


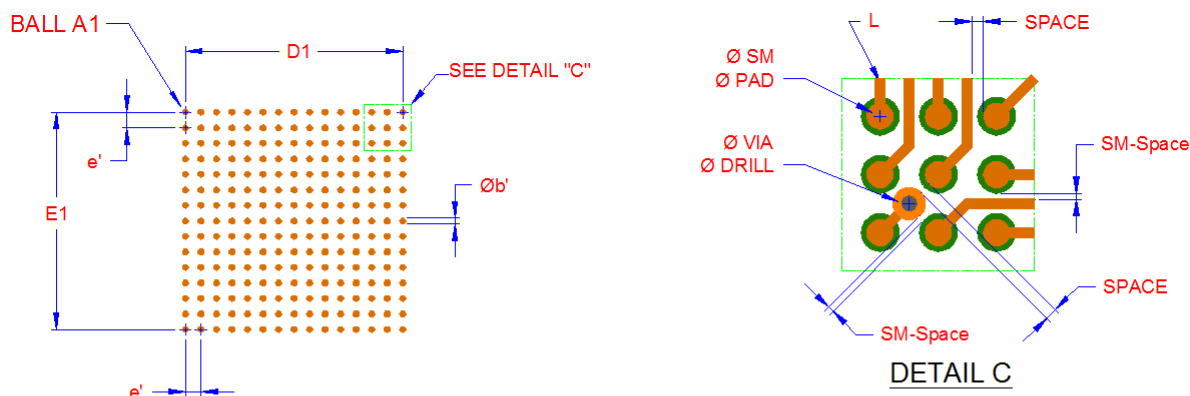
Figure 16.1 UFX7000 225-LFBGA Package Ball Detail

Table 16.1 UFX7000 225-LFBGA Package Parameters

| | MIN | NOMINAL | MAX | REMARKS |
|-----|----------|---------|-------|------------------------------|
| A | - | 1.30 | 1.40 | Overall Package Height |
| A1 | 0.25 | - | 0.40 | Standoff |
| A2 | 0.65 | 0.96 | - | Package Body Thickness |
| D/E | 12.90 | 13.00 | 13.10 | Overall Package Size |
| b | 0.40 | 0.45 | 0.50 | Ball Diameter |
| b1 | 0.35 | 0.40 | 0.45 | Finished Solder Mask Opening |
| b2 | 0.45 | 0.50 | 0.55 | Finished Ball Pad Diameter |
| e | 0.80 BSC | | | Ball Pitch |
| ccc | - | - | 0.20 | Coplanarity |

Notes:

1. All dimensions are in millimeters.
2. Dimension "b" is measured at the maximum ball diameter, parallel to primary datum "C".
3. Primary datum "C" (seating plane) is defined by the spherical crowns of the contact balls.
4. The ball A1 identifier may vary, but is always located within the zone indicated.
5. Dimension "A" does not include attached external features, such as heat sink or chip capacitors.
6. The package ball solderable surface is Solder-Mask-Defined (SMD) type.



| PCB LAND PATTERN DIMENSIONS | | | |
|-----------------------------|----------|-------|-----|
| SYMBOL | MIN | NOM | MAX |
| D1/E1 | - | 11.20 | - |
| e' | 0.80 BSC | | |

THE USER MAY MODIFY THE PCB LAND PATTERN & ROUTING DIMENSIONS, BASED ON THEIR EXPERIENCE AND/OR PROCESS CAPABILITY

| ROUTING DIMENSIONS | |
|--------------------|-------|
| SYMBOL | NOM |
| Ø PAD | 0.40 |
| Ø SM | 0.50 |
| L (Trace) | 0.125 |
| SPACE | 0.135 |
| SM-Space | 0.09 |
| Ø VIA PAD | 0.45 |
| Ø DRILL | 0.25 |

Figure 16.2 UFX7000 225-LFBGA Recommended PCB Land Pattern

Chapter 17 Power Connections

Figure 17.1 illustrates the power connections for UFX7000.

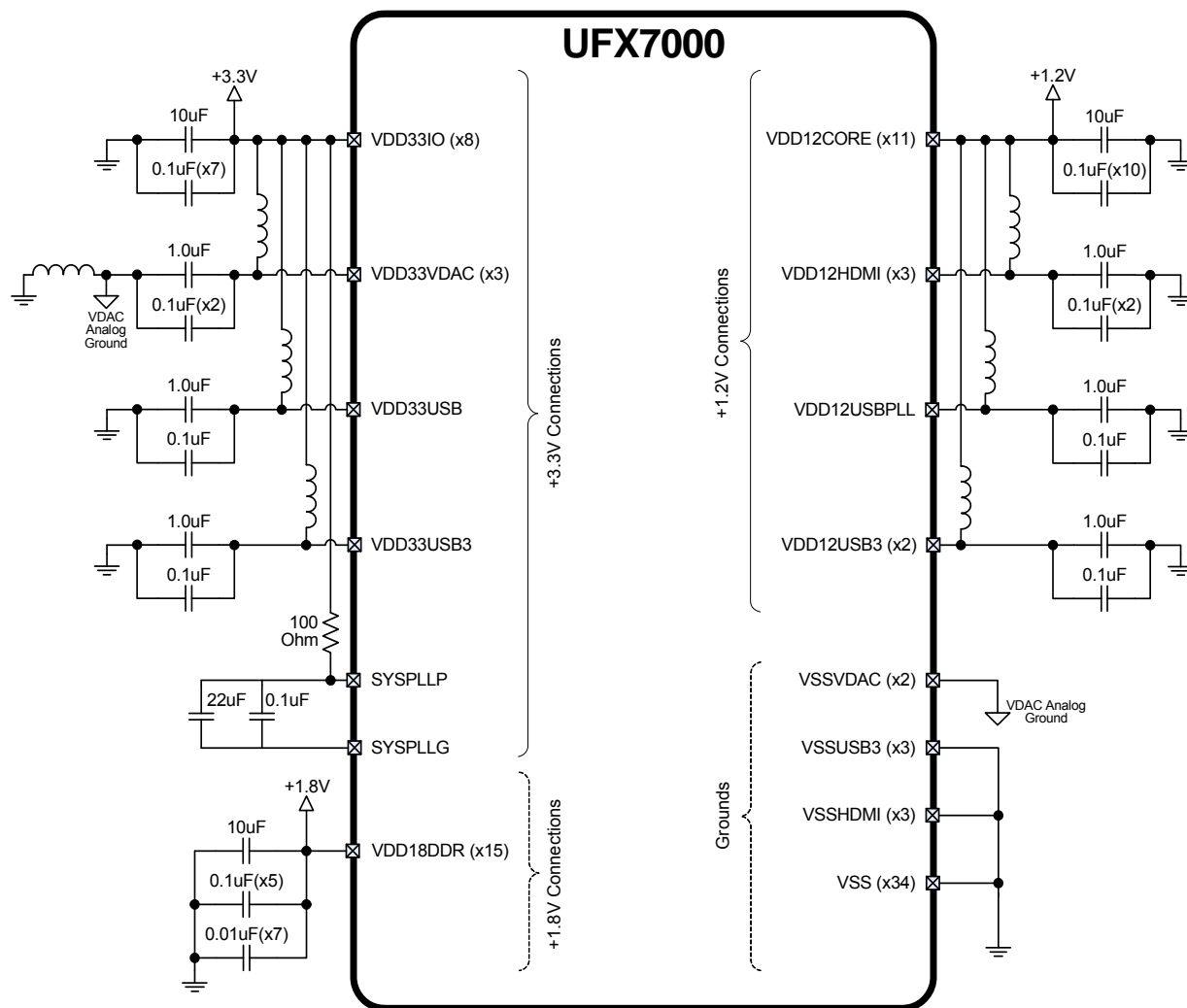


Figure 17.1 Power Connections

Note: For additional power connection information, refer to the UFX7000 reference schematic.

Chapter 18 Databook Revision History

Table 18.1 Customer Revision History

| REVISION LEVEL & DATE | SECTION/FIGURE/ENTRY | CORRECTION |
|-----------------------|----------------------|-----------------|
| Rev. 1.0 (04-22-13) | All | Initial Release |