# MG3500/MG2580 HD H.264 CODEC
# DATA SHEET
# Advance Information

July 23, 2010
Document Release 1.4
Document Number: PN1100

**For pricing, delivery, and ordering information, please contact Maxim Direct at 1-888-629-4642, or visit Maxim's website at www.maximintegrated.com.**

**1.0: Description** **13**
1.1: Hardware Overview 14
1.2: Support Tools 18
**2.0: Device Overview, Pin Assignments** **21**
2.1: Naming Conventions 21
2.2: Pinout Diagrams 22
2.3: Pin Descriptions (by Interface) 27
2.4: Power and Ground Pins 52
2.5: Pin List by Power Group 55
2.6: Hookup Recommendations when Interfaces Are Unused 57
**3.0: Device Configuration** **67**
3.1: Reset 67
3.2: Boot modes for the MMEs and the ARM 67
3.3: Firmware Loader 68
3.4: API Configuration 68
3.5: Pin Multiplexing, GPIOs, etc. 68
3.6: Debug Mode 68
3.7: JTAG ID Register 68
**4.0: Device Operating Conditions** **71**
4.1: Absolute Maximum Ratings 71
4.2: Recommended Operation Conditions 71
4.3: Essential Characteristics 72
4.4: Power Supply Currents for the Different Power Domains 73
4.5: AC Timing 73
**5.0: Block Level Operation** **97**
5.1: Detailed Block Diagram 97
5.2: Reset Logic 98
5.3: Clocks and PLLs 99
5.4: Video Interfaces 103
5.5: Video Scaling 111
5.6: Audio Interfaces 118
5.7: Host Interfaces 120
5.8: Configuration and Status Register (CSR) Definition 132
5.9: DMA Engine Register Definition 136
5.10: Bitstream Write Register Definition 144
5.11: Special Registers 145
5.12: Memory Interfaces 152
5.13: Serial Interfaces 167
5.14: USB 2.0 On-the-Go Interface 178
5.15: Ethernet Media Access Controller 184
5.16: High-Speed Bitstreams 188
**6.0: System Design and Applications** **193**
6.1: Power Supply Design and Recommendations 193
6.2: Power Supply Sequencing 193
6.3: Reset timing Diagrams 193
6.4: Oscillator Connections, Values and Formulas 194
**7.0: Ordering Information** **197**
7.1: Product Information 197
7.2: MG3500 Family Reflow Profile 198
**8.0: Packaging Information** **201**
8.1: Package Diagram 201
8.2: Thermal Data 202
**9.0: Marking** **203**

## Change Log

This section of the data sheet lists the changes that have occurred since the last release. Customers should be aware that not all releases are public, and therefore they might see gaps in the release numbering system.

### Change Log

| Revision | Page | Section | Change |
|---|---|---|---|
| 0.18 | 71 | 4.1 | The Core power supply current was changed from 1000 mA. typical to 1000 mA. maximum. |
| | 195 | 6.4.3 | A new section was added describing use when only an external clock is used. |
| 0.19 | 43 | 2.3.8 | Specified the resistance of the external USB Bias Current resistor. |
| | 43 | 2.3.9 | Modified the timing specification for the SD and MMC Interface. |
| | 46 | 2.3.14 | Specified the resistance of the internal pull-up and pull-down resistors |
| | 89 | 4.5.6 | Added a new section showing the bitstream timing. |
| | 99 | 5.3 | This entire section was re-written to clarify the clock structure. |
| | 111 | 5.5 | The definition of the VOUT register was altered. |
| | 145 | 5.11 | The specification for the ChipID register was added. |
| | 152 | 5.12 | The SDRAM Requirements for Various Profiles table was updated. |
| | 184 | 5.15 | Added note regarding the use of an external switch. |
| | 190 | 5.16.5 | Updated the Bitstream Register section. |
| 0.20 | Throughout | | Made minor changes throughout the book. |
| | 120 | 5.7.1 | Added timing diagrams showing Master Host Interface (MHIF) access timing. |
| 0.21 | Throughout | | Valid value for EWait of EM1Config is 1; EM1Cmd register set to 0x00; removal of several registers, Slave Host Interface; Valid value for EM1 is 0; Corrections to S/PDIF and I2S I/Os; BFifostatus changed to EM1fifostatus. |
| 1.0 | 14 | 1.1.2 | The heading title and description has been changed. |
| | 15 | 1.1.2 | Figure 1-2, only one independent video output is supported. |
| | 16 | 1.1.3 | The maximum pixel rate that VIP can process corresponds to the video input of resolution 1920x1080i at 30 frames per seconds. |
| | Throughout | | Made editorial changes, added definitions, made updates and corrections to several diagrams and tables throughout the book. |

## Change Log

| Revision | Page | Section | Change |
|---|---|---|---|
| 1.1 | 71 | 4.1 | The core supply voltage has been changed to 1.05v ± 5%. |
| | 71 | 4.2 | The minimum, typical, and maximum ranges of the core supply voltage have been re-adjusted. |
| | 78 | 4.5.2 | Modified the Video Interface Timing Diagram to incorporate the new $t_{VCQ}$ parameter. |
| | 78 | 4.5.2 | Modified Table 4-7 to indicate that VIDOUT_DATA is an output delay from VID_CLK for the standard definition video interface AC timing values. It also specifies the minimum and maximum timing value in ns. |
| | 78 | 4.5.2 | Modified Table 4-8 to indicate that VIDOUT_DATA is an output delay from VID_CLK for the high definition video interface AC timing values. It also specifies the minimum and maximum timing value in ns. |
| | 78 | 4.5.2 | Modified Table 4-9 to indicate that VIDOUT_DATA is an output delay from VID_CLK for the high-speed video interface AC timing values. It also specifies the minimum and maximum timing value in ns. |
| | 144 | 5.10 | Made corrections to the bit settings of the BiFiStatus and BiFiConfig registers. |
| | 188 | 5.16.1 | Changed the description for the bitstream interface. |
| | 190 | 5.16.5 | Removed the old section “High-Speed Bitstream” since it is not supported. |
| | 189 | 5.16.4 | Modified Figure 5-39 to show signal BS-DATA is 8 bits long.<br>Changed the definition for Cycle 7 of the waveform diagram. |
| | 189 | 5.16.4 | Changed the definition for Cycle 7 of the waveform diagram. |
| | 190 | 5.16.5 | In Bitstream Control 2, removed the unsupported value 1 for BSClkEnMode, BSStopCond, and BSStrobeModeEn signals. |
| | 190 | 5.16.5 | Bitstream Interface Control registers 8, A, and C have been removed. |
| | 193 | 6.2 | The second half of the RESETn signal in Figure 6-1has been removed. |
| | 201 | 8.0 | Added a new Ordering Information section. |
| 1.2 | 14 | 1.1 | Added Hardware Description section back. |
| | 78 | 4.5.2 | Corrected the note below Table 4-7 to say the clock should be supplied by MG3500. |
| | 78 | 4.5.2 | Re-adjusted the $t_{VCQ}$ parameter in Figure 4-5. |
| | 78 | 4.5.2 | Re-adjusted the $t_{VIH}$ parameter in Figure 4-5. |
| | 81 | 4.5.3 | Inverted the AUD_LRCK signal in Figure 4-6. |
| | 81 | 4.5.3 | Re-adjusted the shaded area for ETH_RXDV and ETH_RXER signals in Figure 4-10. |
| | 81 | 4.5.3 | Modified the description for Figure 4-7. |
| | 81 | 4.5.2 | Completely re-drew Figure 4-7. |
| | 84 | 4.5.4 | The ETH_TXD signal was corrected to ETH_RXD in Table 4-12. |
| | 84 | 4.5.4 | Changed the description for $t_{ETH}$ to indicate a clock High time in Table 4-12. |
| | 86 | 4.5.4 | Changed signals ETH_RXDV and ETH_RXER in Table 4-14. |
| | 86 | 4.5.4 | Changed the description for $t_{ETH}$ to indicate a clock High time in Table 4-14. |
| | 88 | 4.5.4 | Changed signals ETH_RXDV and ETH_RXER in Table 4-16. |
| | 88 | 4.5.4 | Changed the ETH_RXDV signal in Table 4-16. |
| | 81 | 4.5.3 | Re-adjusted the shaded area for ETH_RXDV and ETH_RXER signals in Figure 4-14. |
| | 203 | 9.0 | Removed the approval table from the Marking section. |
| | 197 | 7.1.3 | Changed the maximum height. |

## Change Log

| Revision | Page | Section | Change |
|---|---|---|---|
| 1.3 | 11 | “SoC Features” | Separated Definitions of output ports for Audio Codecs and Decoders. |
| | 11 | “SoC Features” | Changed SDRAM voltage for DDR_VDD. |
| | 13 | 1.0 | Changed the description to provide a summary of both MG3500 and MG2580. |
| | 13 | 1.0 | Changed Table 1-1 to specify the features of both MG3500 and MG2580. |
| | 13 | 1.0 | Added the slave mode support for MG2580 in Table 1-1. |
| | 14 | 1.1.1 | Added the total number of macroblocks required for the H.264 Codec. |
| | 14 | 1.1.2 | Changed the description for video processors and interfaces for clarity. |
| | 15 | 1.1.2 | Added a note following the text describing Figure 1-2 to indicate that video composition features are not available when two VIPs are used as inputs. |
| | 15 | 1.1.2 | Added a column to Table 1-2 that shows video modes for the video composition. |
| | 16 | 1.1.4 | Added a note to indicate the supported formats for video output processor. |
| | 16 | 1.1.7 | Modified the description for valid input and outputs for audio interfaces. |
| | 17 | 1.1.12 | Removed IDE from the list of external devices that can communicate with the host interface. |
| | 18 | 1.2 | Removed the evaluation applications and demonstration product applications supplied by Mobilygen. |
| | 19 | 1.2 | Changed the middle box at the bottom of Figure 1-4 to say Mobilygen Software. |
| | 31 | 2.3.3 | Added pin descriptions for MG2580 to Table 2-4. |
| | 46 | 2.3.14 | Indicated in Table 2-17 that GPIO_2-12, -13, and -15 are not applicable for MG2580. |
| | 57 | 2.6 | Indicated in Table 2-24 that VID23_MISO, _MOSI, and _MSS are not available for MG2580 for VIDEO_PORT 2/3. |
| | 57 | 2.6 | Removed the descriptions for the USB and Ethernet pin names in Table 2-24. |
| | 57 | 2.6 | Specified that USB_ADD is recommended for USB_REXT in Table 2-24. |
| | 57 | 2.6 | Added a footnote to Table 2-24 that recommends a two-step procedure on how to connect the USB pins when the USB block is not used on MG3500. |
| | 71 | 4.0 | Changed the range for the DDR_VDD IO voltage in Table 4-2. |
| | 71 | 4.0 | Changed the operating conditions range for DDR_VREF in Table 4-2. |
| | 86 | 4.5.4 | Changed ETH_RXER[3:0] to ETH_RXD[3:0] in Figure 4-12. |
| | 88 | 4.5.4 | Changed ETH_RXER[1:0] to ETH_RXD[1:0] in Figure 4-14. |
| | 125 | 5.7.1 | Specified the bit range for AddrInc and WEn in the “DevConfigAn Register” table. |
| | 126 | 5.7.1 | Specified the bit range for RHold in the “DevConfigBn Register” table. |
| | 168 | 5.13.3 | Added a new section to describe TWI on MG2580. This includes description as well as a new diagram, Figure 5-33. |
| | 169 | 5.13.2 | Changed Figure 5-33 to show that VID23_SDA is not applicable to MG2580. |
| | 170 | 5.13.4 | Indicated that this section is about SPI on MG3500. Added a note to imply that V23 SPI port is not available on MG2580. |
| | 171 | 5.13.5 | Changed Figure 5-35 to show that VID23_MSS, VID23_MCLK, VID23_MOSI, and VID23_MISO are not applicable to MG2580. |
| | 171 | 5.13.5 | Added a new section for SPI on MG2580. This includes descriptions as well as a modified diagram, Figure 5-35. |
| | 173 | 5.13.8 | Indicated in the Serial I/O Control table that V23_MCLK_AltSeL and V23_MOSI_AltSeL are not available on MG2580. |
| | 173 | 5.13.8 | Indicated in the GPIO 2 Sel that table GPIO_2_12, GPIO_2_13, and GPIO_2_15 fields will have no effect on MG2580 since GPIO pins are not connected. |
| | 173 | 5.13.8 | Indicated in the GPIO 2 Pull-up Enable table GPIO_2_12, GPIO_2_13, and GPIO_2_15 fields will have no effect on MG2580 since GPIO pins are not connected. |
| | 173 | 5.13.8 | Indicated in the GPIO 2 Pull-up Enable table GPIO_2_12, GPIO_2_13, and GPIO_2_15 fields will have no effect on MG2580 since GPIO pins are not connected. |
| | 197 | 7.1.3 | Removed the part number to order MG2580A2 since this part will no longer be built. |

## Change Log

| Revision | Page | Section | Change |
|---|---|---|---|
| 1.4 | 29/30 | 2.3.2 | Changed VID0_PIXCLK and VID1_PIXCLK to IO |
| | 40 | 2.3.6 | Added a note for DDR_DQ[31:16] and DDR_DQM[3:] pins are not connected in 16-bit mode |
| | 46 | 2.3.14 | Removed alternate functionality for GPIO_6 and GPIO_7 |
| | 56 | 2.5 | Removed signal USB_VBUS from 3.3V Power Group |
| | 71 | 4.2 | Updated ETH_VDD 3.3V +/- 5% |
| | 78 | 4.5.2 | Included tVH(min) and tVH(max) in Table 4-7, Table 4-8 and Table 4-9<br>Added HSYNC, VSYNC, and FRAME signals to the timing diagram. Clarified Setup and Hold time description with reference to VID_CLK for VID_DATA and VIDOUT_DATA. |
| | 81 | 4.5.3 | Updated figure 4-6 as it was not readable in v1.3 |
| | 87/88 | 4.5.4 | Updated RMII Transmit/Receive Timing Diagram in Figure 4-13 by replacing TXCLK with RX-CLK. Updated Tables 4-15 accordingly by removing transmit and replacing it with receive clock.. Changed Min. and Max. values in Table 4-16 for ECYC-ETH_CLK, ETL- Low and ETH-High Time signal. |
| | 103/ 104 | 5.4 | Included separate block diagrams for MG3500 and MG2580 video paths |
| | 165 | 5.12.5 | Removed 512 byte page size under NAND flash bulleted item |
| | 166 | 5.12.5 | Updated NAND/NOR Flash Interface connected to NOR Flash Memory Figure 5-30 |

**Table 1 Pin Name Changes**

| Old Name | New Name |
|---|---|
| ENET_COL | ETH_COL |
| ENET_RXD1 | ETH_RXD1 |
| ENET_RXERR | ETH_RXERR |
| ENET_RXDV | ETH_RXDV |
| ENET_CRS | ETH_CRS |
| ENET_RXCLK | ETH_RXCLK |
| ENET_RXD0 | ETH_RXD0 |
| ENET_RXD2 | ETH_RXD2 |
| ENET_RXD7 | ETH_RXD7 |
| ENET_MDCLK | ETH_MDCLK |
| ENET_RXD3 | ETH_RXD3 |
| ENET_MDIO | ETH_MDIO |
| ENET_RXD6 | ETH_RXD6 |
| ENET_RXD4 | ETH_RXD4 |
| ENET_TXD0 | ETH_TXD0 |
| ENET_RXD5 | ETH_RXD5 |
| ENET_TXD2 | ETH_TXD2 |
| ENET_TXD4 | ETH_TXD4 |
| ENET_TXEN | ETH_TXEN |
| ENET_TXD3 | ETH_TXD3 |
| ENET_TXD5 | ETH_TXD5 |
| ENET_TXERR | ETH_TXER |
| ENET_TXCLK | ETH_TXCLK |
| ENET_TXD1 | ETH_TXD1 |
| ENET_TXD6 | ETH_TXD6 |
| ENET_TXD7 | ETH_TXD7 |
| HOST_D08 | HOST_D8 |
| HOST_D09 | HOST_D9 |
| HOST_D04 | HOST_D4 |
| HOST_D05 | HOST_D5 |
| HOST_D06 | HOST_D6 |
| HOST_D02 | HOST_D2 |
| HOST_D03 | HOST_D3 |
| HOST_D07 | HOST_D7 |
| HOST_D01 | HOST_D1 |
| HOST_D00 | HOST_D0 |
| CF_IACKn | CF_INPACKn |

| Old Name | New Name |
|---|---|
| HOST_A08 | HOST_A8 |
| HOST_A05 | HOST_A5 |
| HOST_A06 | HOST_A6 |
| CF_WAITn | CF_WAITn |
| HOST_A02 | HOST_A2 |
| HOST_A09 | HOST_A9 |
| HOST_A07 | HOST_A7 |
| HOST_A03 | HOST_A3 |
| HOST_A04 | HOST_A4 |
| HOST_A01 | HOST_A1 |
| DDR_A03 | DDR_A3 |
| DDR_A02 | DDR_A2 |
| DDR_A01 | DDR_A1 |
| DDR_A00 | DDR_A0 |
| DDR_A08 | DDR_A8 |
| DDR_A04 | DDR_A4 |
| DDR_A07 | DDR_A7 |
| DDR_A06 | DDR_A6 |
| DDR_A09 | DDR_A9 |
| DDR_D05 | DDR_DQ5 |
| DDR_D00 | DDR_DQ0 |
| DDR_D07 | DDR_DQ7 |
| DDR_D04 | DDR_DQ4 |
| DDR_D01 | DDR_DQ1 |
| DDR_D06 | DDR_DQ6 |
| DDR_D02 | DDR_DQ2 |
| DDR_D03 | DDR_DQ3 |
| DDR_D14 | DDR_DQ14 |
| DDR_D08 | DDR_DQ8 |
| DDR_D15 | DDR_DQ15 |
| DDR_DQ9 | DDR_DQ9 |
| DDR_D10 | DDR_DQ10 |
| DDR_VREF1 | DDR_VREF |
| DDR_D13 | DDR_DQ13 |
| DDR_D12 | DDR_DQ12 |
| DDR_D11 | DDR_DQ11 |
| DDR_D25 | DDR_DQ25 |

| Old Name | New Name |
|---|---|
| DDR_D27 | DDR_DQ27 |
| DDR_D31 | DDR_DQ31 |
| DDR_D29 | DDR_DQ29 |
| DDR_D30 | DDR_DQ30 |
| DDR_D28 | DDR_DQ28 |
| DDR_D24 | DDR_DQ24 |
| DDR_D26 | DDR_DQ26 |
| DDR_D23 | DDR_DQ23 |
| DDR_D21 | DDR_DQ21 |
| DDR_D22 | DDR_DQ22 |
| DDR_D19 | DDR_DQ19 |
| DDR_D20 | DDR_DQ20 |
| DDR_D16 | DDR_DQ16 |
| DDR_D17 | DDR_DQ17 |
| DDR_D18 | DDR_DQ18 |
| JTAG_TEST | TEST |
| RESETn | RESETn |
| JTAG_T_SELL | JTAG_TAP_SEL |
| VID1_OCLK | VID1_OUTCLK |
| VID1_PXCLK | VID1_PIXCLK |
| VID0_OCLK | VID0_OUTCLK |
| VID0_PXCLK | VID0_PIXCLK |
| USB_VDD_D | USB_DVDD |
| USB_VDDA | USB_AVDD |
| USB_GNDA | USB_AGND |
| USB_GNDA | USB_AGND |
| USB_GNDA | USB_AGND |
| USB_VDDA | USB_AVDD |
| USB_VDDAC | USB_ACVDD |
| USB_GNDAC | USB_ACGND |
| USB_A_TST | USB_ANA_TEST |
| VID3_FIELD | VID23_GPIO |
| VID3_VSYNC | VID_DATA_16 |
| USB_VBUSD | USB_D_VBUS |
| VID3_HSYNC | VID_DATA_17 |
| VID2_PXCIK | VID2_PIXCLK |
| | |

## SoC Features

### *HD H.264 Codec*

- Dual-Stream High-Definition (HD) or Standard-Definition (SD) H.264 Codecs
  - Full-duplex HD or SD operation
  - Dual Encode HD or SD or
  - Dual Decode HD or SD
- H.264 Codec supports High, Main, and Baseline profiles
- H.264 Codec up to level 4.1
- H.264 Encoding or decoding up to 1920x1080i
- Programmable resolutions and frame rates
- Multi-stream SD encode or decode
- Video bit rates: 64 kbps – 62.5 Mbps
- Macro-Block Level Adaptive Frame/Field (MBAFF) support

### *MPEG-2 Decoder*

- HD and SD decoder
- Enables real-time HD MPEG-2 to HD H.264 transcoding
- Multi-stream SD MPEG-2 decoding

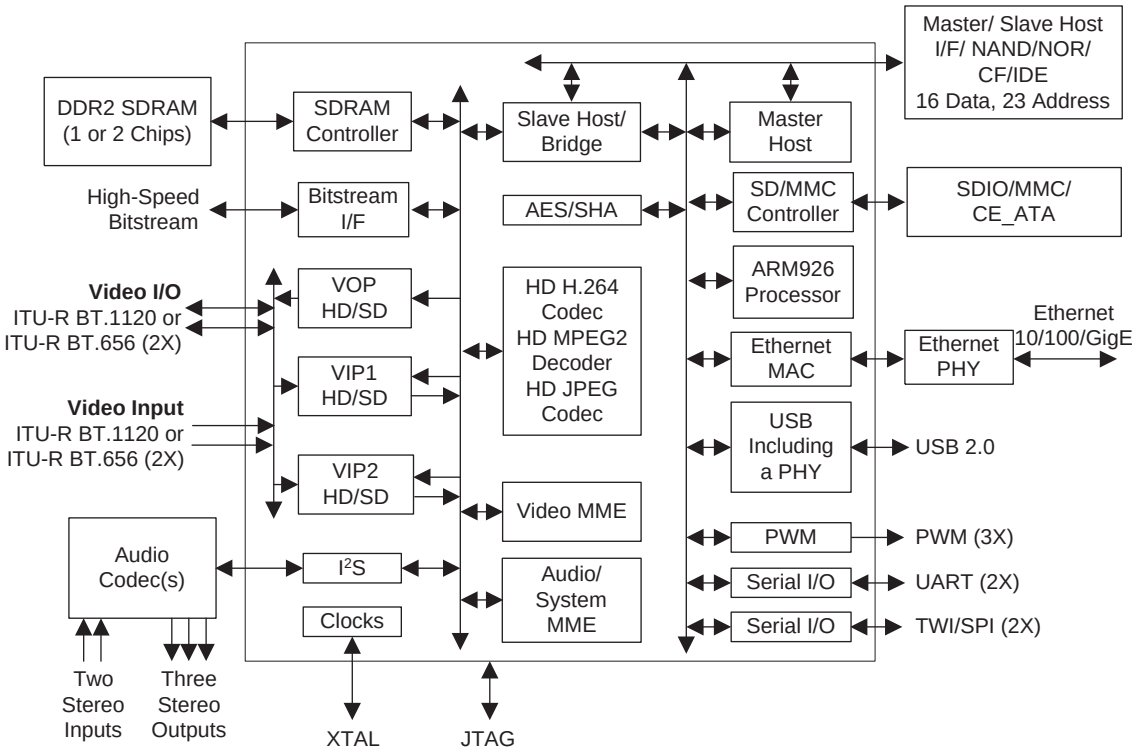
### *JPEG Codec*

- JPEG Encoder and Decoder
- HD or SD MJPEG Support
- Exchangeable Image File Format (EXIF) Support

### *Audio Codecs & Decoders*

- High-fidelity, 2-channel AAC-LC codec
- MPEG-1/2 Audio Layer II codec (MP2)
- MPEG-1/2 Audio Layer I and III decoder (MP1 and MP3)
- Dolby Digital 5.1 decode and down mix
- G.711 Codec
- Flexible bit rates and sample rates
- Additional codecs planned
- One S/PDIF output port
- Two $I^2S$ Audio input ports and three $I^2S$ Audio output ports

### *Block Diagram*



### *Video Input Processors (VIPs)*

- Flexible direct video inputs
  - Two ITU-R BT.1120 parallel interfaces
  - Four ITU-R BT.656 parallel interfaces
- Two advanced Video Input Processors (VIPs)
- Digital Image Stabilization
- Smooth Digital Zoom

### *Video Output Processor (VOP)*

- HD or SD output support via ITU-R BT.1120 or ITU-R BT.656
- Multi-stream decode supports scaled PIP and multi-channel compositing on video output
- LCD Interface, 16-Bit, 18-bit or 8-bit RGB
- High-quality Video output video scaling
- Two overlay planes with alpha blending and cursor
- Generates optional external sync signals

### *Integrated ARM926-EJ Processor*

- 240 MHz general purpose processor
- 16 kByte Data Cache
- 16 kByte Instruction Cache
- 16 kByte Scratch Pad Memory

### *System Connectivity*

- 10/100/GigE Ethernet MAC [1]
- USB 2.0 On-The-Go (OTG) ports including the physical layer
- High Speed Bit-stream I/O
- AES and SHA hardware acceleration

### *Peripheral Interfaces*

- Secure Digital (SD), Secure Digital Input/ Output (SDIO), Multi-Media Card (MMC), and Consumer Electronics AT Attachment (CE-ATA)
- Compact FLASH, IDE

### *General Purpose Interfaces*

- Two SPI or Two Wire Interface ports
- Three UARTs
- Three Pulse Width Modulators
- Up to 72 GPIO, 8 dedicated

### *System*

- Core Voltage: 1.05V ± 5%
- SDRAM Voltage: 1.8V ±0.1V
- I/O Voltages: 1.8V, 2.5V, 3.3V ±10%
- On-Chip A/V PLLs driven from single crystal

### *Power Consumption (MG3500+SDRAM)*

- H.264 HD 30fps + AAC Encode 750 mW

### *Packaging*

- 376-ball FPBGA, 18x18mm, 0.8mm pitch, RoHS compliant

1. When both 10/100 and GigE need to be enabled, an external switch must be installed to select the clock.

## 1.0 Description

The Maxim High-Profile H.264 Codecs currently comprises two devices: MG3500 HD H.264 Codec SoC and MG2580 720p30 H.264 Codec SoC. The MG3500 HD H.264 Codec SoC is a full HD 1080p30 H.264 Codec. It is the ideal choice for any 1080p30 H.264 application as well as multi-channel D1 applications as found in the security surveillance space. Similarly, the MG2580 SoC is the cost-reduced version of the High-Profile H.264 Codecs that performs 720p30 H.264 and MJPEG encoding operations. The MG2580 is particularly adapted for both IP camera and H.264 webcam designs. Both chips encompass an ARM926-EJ processor as well as a complete set of System-On-a-Chip (SoC) features.

Table 1-1 shows the features for each of the devices. Specific information for both of these devices are covered in this datasheet.

All references to MG3500 throughout this manual also apply to the MG2580 as well unless stated otherwise.

**Table 1-1 MG3000 Family of High-Definition H.264 Codecs**

| Feature | MG3500 | MG2580 |
|---|---|---|
| Standard Definition Codec | ✓ | ✓ |
| High Definition H.264 Codec | 1080p30 | 720p30 |
| MPEG-2 Decoder | ✓ | ✓ |
| JPEG Codec | ✓ | ✓ |
| Video Input Ports Supported (8-bit or 16-bit) | 2 | 1 |
| Frame Multiplexed Video Inputs | 4 | – |
| Video Input Processors | 2 | 2 |
| Video Output Ports Supported (8-bit or 16-bit)[1] | 1 | 1 |
| Video Output Prcessors | 1 | 1 |
| Audio Input Ports | 2 | 1 |
| Audio Codecs and Decoders | ✓ | ✓ |
| High-Speed Bitstream I/O | ✓ | – |
| Embedded ARM926-EJ Processor | ✓ | ✓ |
| Master Mode Operation | ✓ | ✓ |
| Slave Mode Operation | ✓ | ✓ |
| Embedded 10/100/GigE Ethernet MAC | ✓ | ✓ |
| USB On-The-Go including Physical Layer | ✓ | ✓ |
| SD, SDIO, MMC, CE-ATA Peripheral Interface | ✓ | ✓ |
| Compact Flash | ✓ | ✓ |
| 32-Bit SDRAM Interface | ✓ | ✓ |
| SPI or Two-Wire Interface | 3 | 2 |
| UARTs | 3 | 3 |
| Pulse Width Modulators | 3 | 3 |
| GPIO, Shared | 64 | 61 |
| GPIO, Dedicated | 8 | 8 |

1.The MG2580 supports 8-bit output only. MG3500 can support an 8-bit or 16-bit output.

## 1.1 Hardware Overview

This section provides an overview of each of the blocks in the MG3500 SoC. See “Block Diagram” on page 11.

### *1.1.1 Video Codecs*

The MG3500 SoC includes efficient hardware implementations of two HD encoders and three HD decoders:

- H.264 Encoder/Decoder
- MPEG2 Decoder
- JPEG/MJPEG Encoder/Decoder

As shown in Figure 1-2, the H.264 Codec, MPEG2 Decoder and JPEG/MJPEG Codec are implemented as separate elements in order to support real time trans-coding from one format to another.

The H.264 Codec hardware pipeline allows the highest processing power at the lowest power consumption to support all of the H.264 tools for the High, Main, and Baseline profiles. The processing power that enables HD Encoding or Decoding can also be applied to Encoding or Decoding multiple reduced resolution or SD streams.

The H.264 Codec is capable of encoding or decoding up to 1920 pixels per line (horizontal) and up to 2000 lines (vertical) as long as the total number of 16x16 macroblocks does not exceed 8192 and the macroblocks per second does not exceed 244800.

The HD MPEG2 Decoder is also capable of decoding up to a maximum of 1920 pixels per line (horizontal) and 2000 lines (vertical). It does not have encoding capabilities.

The JPEG/MJPEG Codec is also capable of decoding up to a maximum of 1920 pixels per line (horizontal) and 2000 lines (vertical) for real time video, but in addition, it can encode or decode up to 8k by 8k still images that reside in the memory.

### *1.1.2 Video Processors and Interfaces*

As shown in Figure 1-2, the MG3500 SoC video path has two Video Input Processors (VIP: VIP1 and VIP2) and one Video Output Processor (VOP).

The Codec has two 8-bit video inputs (VID0 and VID1) that can be used either as two individual 8-bit ITU-R BT 656 video inputs or a single 16-bit ITU-R BT 1120 video input for HD inputs from an HDMI receiver or other HD input.

Additionally, the Codec provides two bi-directional ports (VID2 and VID3) that can be used either as an HD input or as an output (one 8-bit or one 16-bit output).

These bi-directional ports can be clocked at higher frequency to support non-standard video interfaces. These two 8-bit interfaces can be combined to create a single 16-bit HD ITU-R BT 1120 interface.

The bi-directional video ports can also be used to drive an LCD display in one of two modes. As an standard output, it can drive an 8-bit RGB LCD interface or it can be used as a 16-bit HD output. Two additional bits are available to drive an 18-bit RGB LCD interface.

Each video input supports independent clocks and synchronization signals. The clock frequency can be driven over 100 MHz in order to support non-standard video inputs including HD sensors with 8-bit interfaces.

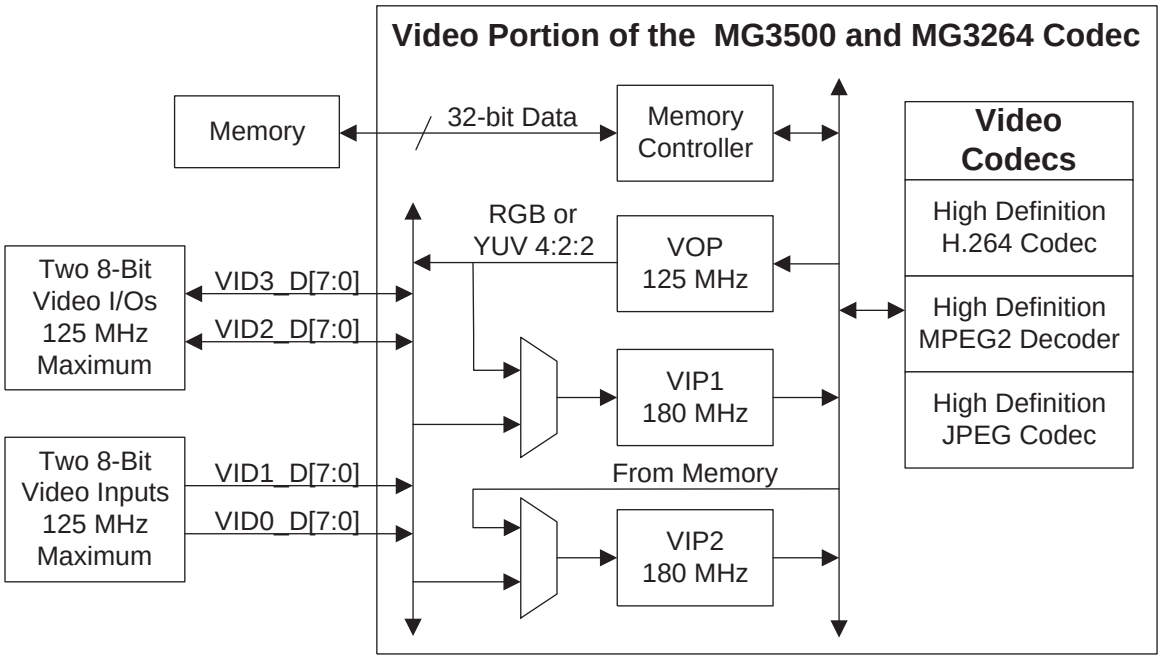



**Figure 1-2 Block Diagram of the Video Input Section**

The two VIPs and one VOP provide the capability of processing two independent video inputs and one independent video output. Together with the flexible Video Interfaces described above, the modes shown in Table 1-2 are supported.

**Note:** Video composition features, such as memory-based scaling or merging multiple videos into one screen are not available when two VIPs are both used as inputs.

**Table 1-2 Video Modes**

| Video Mode | Video Inputs | Video Outputs | Video Composition |
|---|---|---|---|
| 1 | 1 HD | 1 HD | Yes |
| 2 | 1 HD | 1 SD | Yes |
| 3 | 2 HD | None | Not Available |
| 4 | 2 SD | 1 HD | Not Available |
| 5 | 1 SD + 1 HD | 1 SD | Not Available |
| 6 | 2 SD | 1 SD | Not Available |

**Note:** The HD output can be used as an 18-bit LCD interface and an SD output can be used as an 8-bit LCD interface.

### 1.1.3 Video Input Processor

There are two identical Video Input Processors (VIPs) that perform high quality scaling, chroma and gamma adjustment, filtering, and the extraction of video analytics. The maximum pixel rate that the VIP can process corresponds to video input of resolution 1920x1080i at 30 frames per seconds.

### 1.1.4 Video Output Processor

The Video Output Processor (VOP) performs high quality scaling of un-compressed video, overlays it with two graphic planes, performs gamma and chroma adjustment, overlays a hardware cursor, and outputs the combined video to a video port. Each graphic plane can be from 1 to 32 bits. Graphic planes using less than eight bits use a Look-Up Table (LUT).

**Note:** Some formats may not be possible depending on the output resolution and available system memory bandwidth. For example 32 bits/pixel modes are not possible for 1080i60 resolution output.

**Table 1-3 Video Output Modes**

| Mode | Bits/Pixel | Format | Mode | Bits/Pixel | Format |
|---|---|---|---|---|---|
| 0 | 1 | Indexed | 16 | 16 | RGB 4:4:4 |
| 1 | 1 | Grayscale | 17 | 16 | RGBα 4:4:4:4 |
| 2 | 2 | Indexed | 18 | 16 | RGB 5:5:5 |
| 3 | 2 | Grayscale | 19 | 16 | RGBα 5:5:5:1 |
| 4 | 4 | Indexed | 20 | 16 | RGB 5:6:5 |
| 5 | 4 | Grayscale | 21 | 16 | RGBα 5:6:4:1 |
| 6 | 8 | Indexed | 24 | 32 | RGB 8:8:8 |
| 7 | 8 | Grayscale | 25 | 32 | RGBα 8:8:8:8 |

The video output can be either be YCbCr via an 8-bit ITU-R BT 656 interface, YCbCr via a 16-bit ITU-R BT 1120 interface, RGB via an 8-bit interface, or RGB via an 18-bit interface. In some of the output modes, the MG3500/MG2580 HD H.264 Codec SoC is also capable of generating optional external sync signals.

### 1.1.5 Video Multi-Media Engine

The Video Multi-Media Engine (MME) is a proprietary Reduced Instruction Set Computer (RISC) that has been optimized for single cycle context switching and low power. The Video MME controls all aspects of the VIPs, Video Cores, and the VOP (see the MG3500/MG2580 HD H.264 Codec SoC Block Diagram on page 3 for more information).

### 1.1.6 Audio Multi-Media Engine

The Audio MME implements all audio Codecs in firmware.

### 1.1.7 Audio Interfaces

There are two $I^2S$ inputs, three $I^2S$ outputs, and one S/PDIF output. One of the two $I^2S$ inputs is associated with one of the audio clocks. The other audio input, the three audio outputs, and the S/PDIF output must share a common clock and sample rate. The three $I^2S$ outputs and the S/PDIF output must also share a common format.

### 1.1.8 SDRAM

The MG3500/MG2580 HD H.264 Codec SoC has a high performance memory subsystem that uses either a 16- or 32-bit wide external SDRAM. The SDRAM is DDR2, and runs up to 264 MHz.

### 1.1.9 ARM926-EJ

The MG3500/MG2580 HD H.264 Codec SoC has an embedded ARM926-EJ processor that runs at speeds up to 240 MHz. This processor is not used for Audio or Video Codec functions, so it is completely available to implement any required system level functions. Mobilygen provides Codec and Data Streaming APIs under Linux 2.6.20.

### 1.1.10 Ethernet Media Access Controller

The Ethernet MAC supports 10/100/1000 Mbps Ethernet interfaces.[1] This is typically connected to an external Physical Layer (Phy) device but can also be connected directly to Ethernet switches that support Reverse MII interfaces.

### 1.1.11 USB 2.0

The USB interface is USB 2.0, High-Speed with the ability to operate as Device, Host, or On-The-Go (OTG) at speeds of up to 480 MHz. The USB interface includes the Physical Layer.

### 1.1.12 FLASH, IDE and Host Interface

The host interface can be used to communicate to external devices including NOR FLASH, NAND FLASH, and COMPACT FLASH, as well as other devices.

### 1.1.13 SD/MMC Interface

The SD/MMC interface is designed to support Secure Digital (SD), Secure Digital Input/Output (SDIO), Multi-Media Card (MMC), and Consumer Electronics AT Attachment (CE-ATA) devices. This four-bit wide interface supports up to a 25 MHz clock rate (100 Mbits/sec. transfer rate).

### 1.1.14 AES and SHA Hardware Acceleration

The MG3500 SoC design includes hardware acceleration for the Advanced Encryption Standard (AES) and Secure Hashing Algorithm (SHA). The AES accelerator supports CBC, CTR, ECB, and CCM modes with 128, 192, and 256 bit keys for secure data storage and transmission. The SHA accelerator supports the creation of 128, 224, and 256 bit digests for Digital Signatures and Digital Time Stamps.

### 1.1.15 Serial and Misc. IO

The MG3500 SoC has several UARTs for communication, Pulse Width Modulators (PWMs) for control, $I^2C$-compatible Two Wire Interfaces (TWIs) for device control, and Serial Peripheral Interfaces (SPIs) for device control.

The MG3500 SoC also has eight dedicated General Purpose Input/Output (GPIO) pins and up to 64 shared GPIO pins that can be used for system control. The shared GPIO pins are multiplexed with other functions and are only available when the primary function for the pin is not being used. For example, if your design does not require a SPI interface (see “SPI/Bitstream Interface Timing” on page 89), the four pins dedicated to that interface can be used as GPIO pins.

1. When both 10/100 and GigE need to be enabled, an external switch must be installed to select the clock.

## 1.2 Support Tools

This section provides an overview of the software and hardware development tools that are available to support the part.

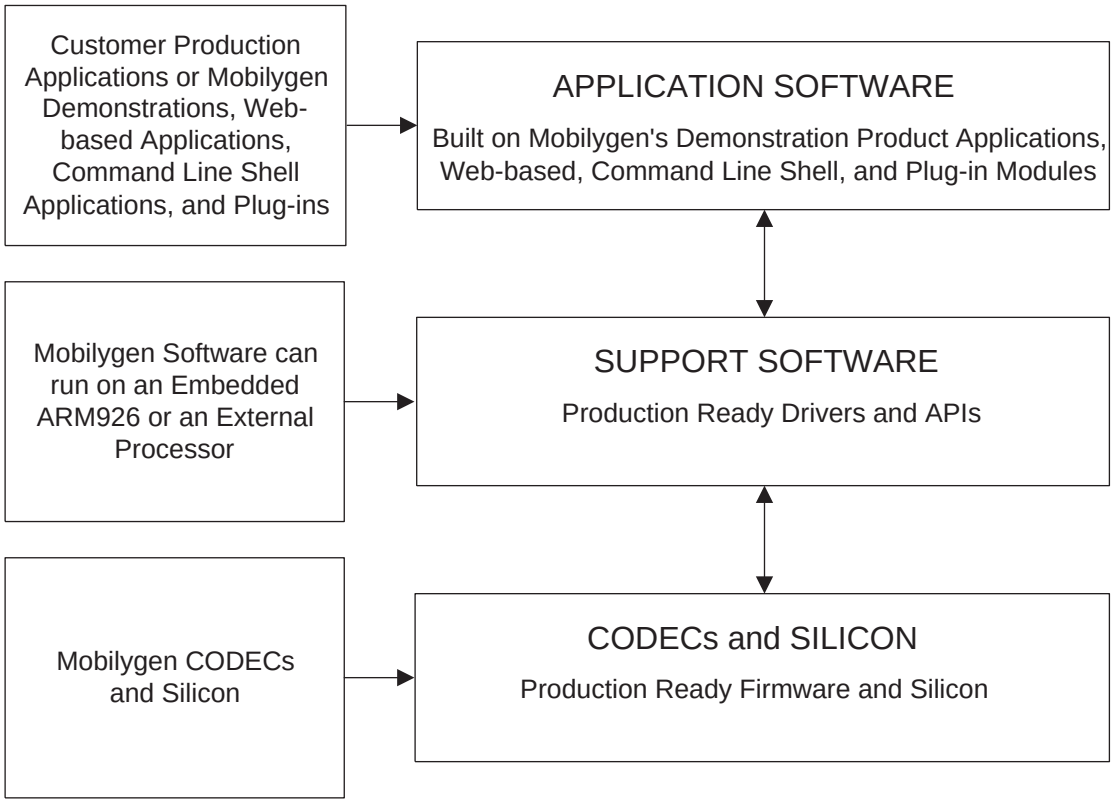



**Figure 1-3 Software Architecture**

The Mobilygen-developed MG3500 SoC software is developed for Linux 2.6.20. Mobilygen supplies these APIs and Drivers:

- Codec API
- qHAL Hardware Abstraction Layer
- Data Streaming API
- On-Chip Device Drivers

Figure 1-4 shows an expanded version of Figure 1-3 that has all of the elements of the system software included. In this figure:

- Blue boxes are applications, firmware, drivers, and silicon supplied by Mobilygen.
- Green boxes are applications that are available from third-party vendors (public domain or Linux vendors)
- White boxes are customer-written applications

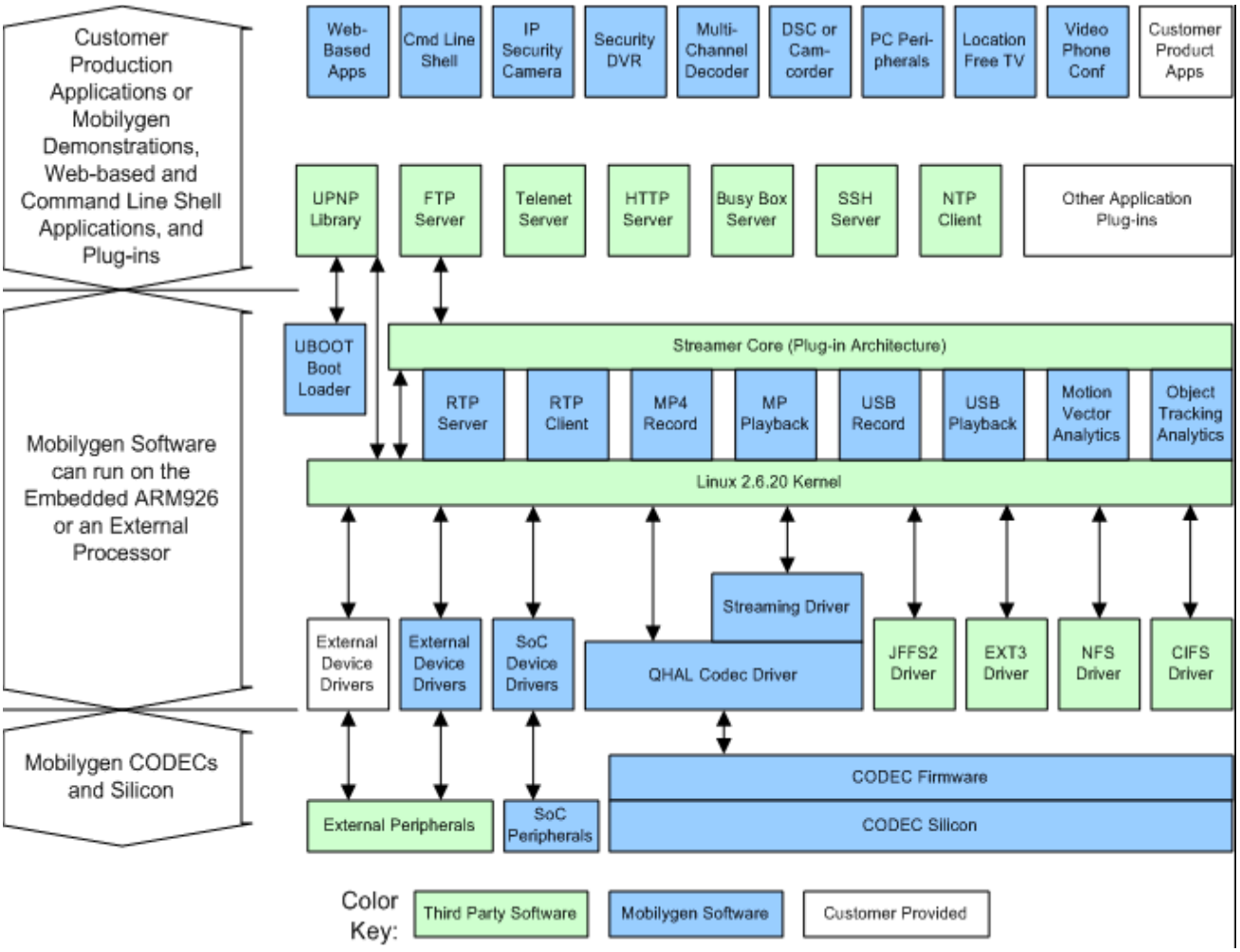



**Figure 1-4 Software Elements**

**Note:** As shown in Figure 1-4, the Mobilygen supplied drivers and higher-level functions (the lower two-thirds of Figure 1-4) are production ready and fully supported by Mobilygen. The Customer Production Applications and Mobilygen Demonstration programs (the upper third of Figure 1-4) are available for customers to use as an advanced starting point, but are not production ready.

# 2.0 Device Overview, Pin Assignments

## 2.1 Naming Conventions

The MG3500 SoC has both signal and power connections. Each signal has a unique name. Power connections do not necessarily have unique names.

The signals are organized by signal groups. The signal names typically have two parts separated by an underscore. When that is the case the first part represents the name of the signal group and the second part defines the function within that group. The signal group names do not have an underscore in them, so the first underscore separates the signal group name from the function name. The function name may have an underscore in it. Signals that are active low end with a lower case ‘n’.

Power connection names also have two parts separated by an underscore. The first part represents the power domain, and the second part represents the power type.

All pins have a Primary function, and the name that is assigned to the pin reflects that primary function. Many of the pins have an Alternate (ALT) function that can be used if the primary function is not used. Some pins are capable of being used as General Purpose I/O pins (GPIO) is neither their primary or their secondary functions are being used. These pins are available for customer-assigned uses.

The pinout diagrams and tables in this section list the pins by their Primary name. The pinout tables also show the Alternate and GPIO capabilities of the pins if any are assigned to them.

## 2.2 Pinout Diagrams

Figure 2-1 shows a map of all the signal positions.

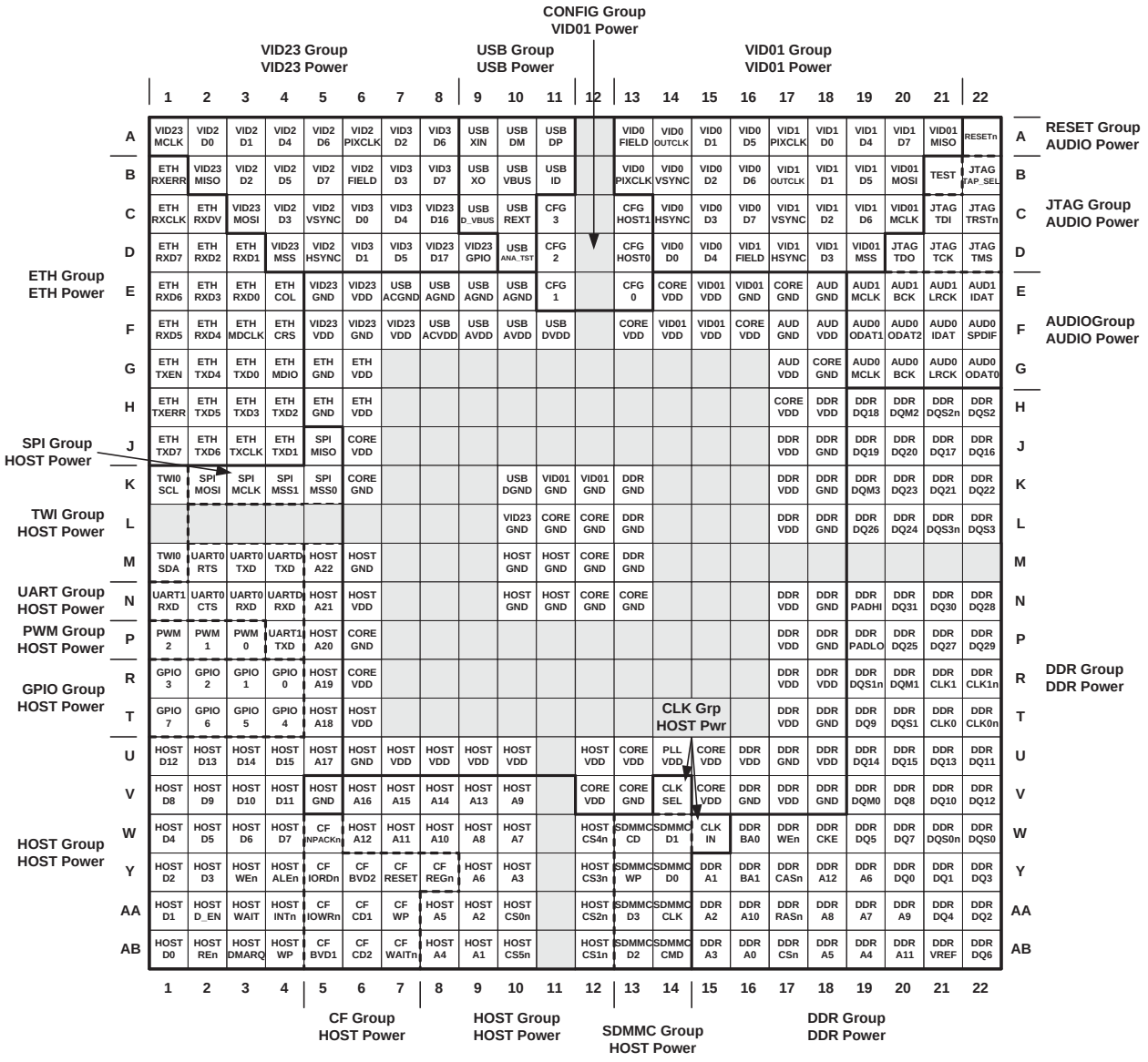
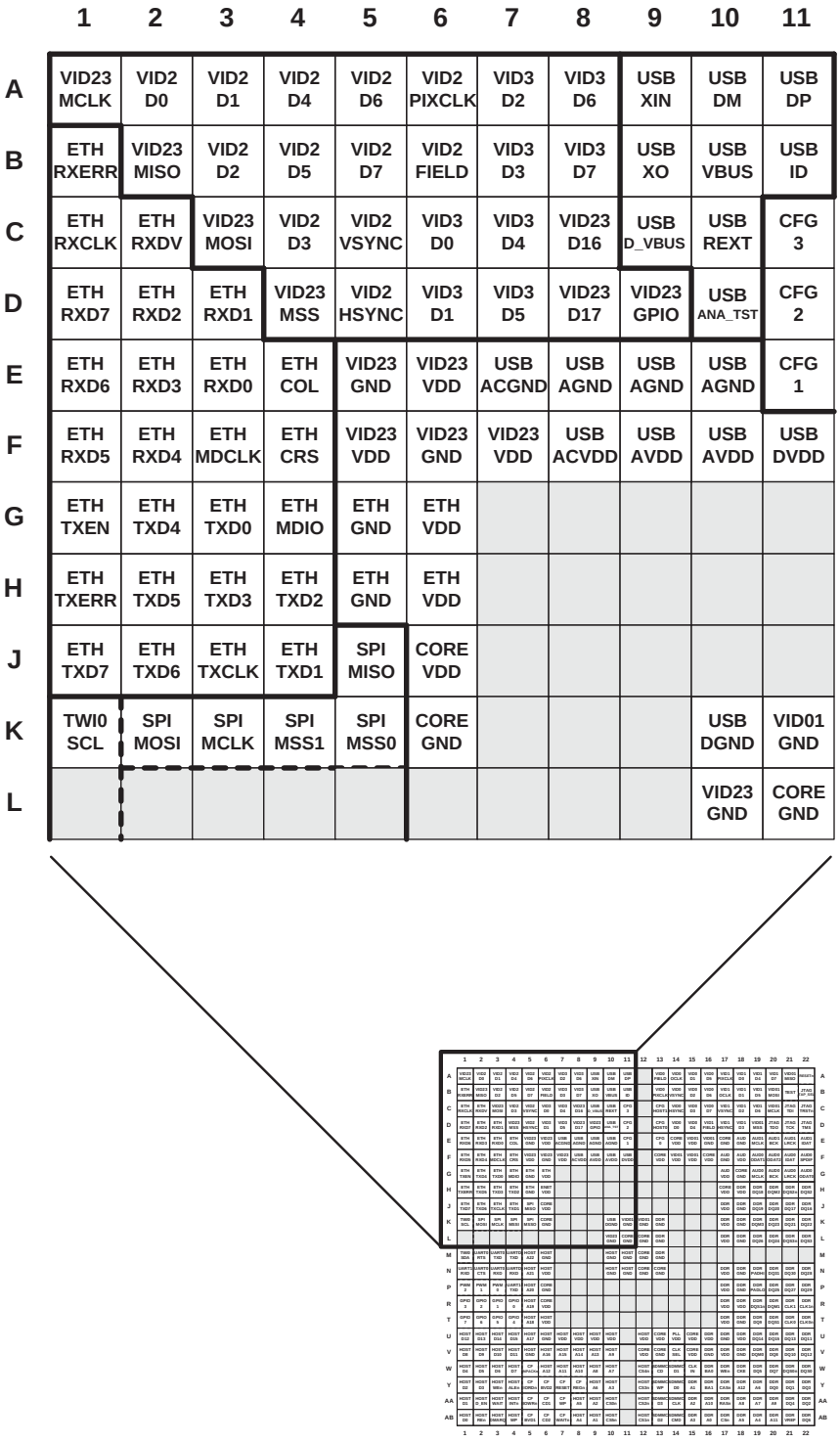



**Figure 2-1 Map of all the MG3500 SoC Signal Positions (Top View)**

Figure 2-2 is a map of the upper-left quadrant.



| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| A | VID23 MCLK | VID2 D0 | VID2 D1 | VID2 D4 | VID2 D6 | VID2 PIXCLK | VID3 D2 | VID3 D6 | USB XIN | USB DM | USB DP |
| B | ETH RXERR | VID23 MISO | VID2 D2 | VID2 D5 | VID2 D7 | VID2 FIELD | VID3 D3 | VID3 D7 | USB XO | USB VBUS | USB ID |
| C | ETH RXCLK | ETH RXDV | VID23 MOSI | VID2 D3 | VID2 VSYNC | VID3 D0 | VID3 D4 | VID23 D16 | USB D_VBUS | USB REXT | CFG 3 |
| D | ETH RXD7 | ETH RXD2 | ETH RXD1 | VID23 MSS | VID2 HSYNC | VID3 D1 | VID3 D5 | VID23 D17 | VID23 GPIO | USB ANA_TST | CFG 2 |
| E | ETH RXD6 | ETH RXD3 | ETH RXD0 | ETH COL | VID23 GND | VID23 VDD | USB ACGND | USB AGND | USB AGND | USB AGND | CFG 1 |
| F | ETH RXD5 | ETH RXD4 | ETH MDCLK | ETH CRS | VID23 VDD | VID23 GND | VID23 VDD | USB ACVDD | USB AVDD | USB AVDD | USB DVDD |
| G | ETH TXEN | ETH TXD4 | ETH TXD0 | ETH MDIO | ETH GND | ETH VDD | | | | | |
| H | ETH TXERR | ETH TXD5 | ETH TXD3 | ETH TXD2 | ETH GND | ETH VDD | | | | | |
| J | ETH TXD7 | ETH TXD6 | ETH TXCLK | ETH TXD1 | SPI MISO | CORE VDD | | | | | |
| K | TWI0 SCL | SPI MOSI | SPI MCLK | SPI MSS1 | SPI MSS0 | CORE GND | | | | USB DGND | VID01 GND |
| L | | | | | | | | | | VID23 GND | CORE GND |

**Figure 2-2 Map of the Upper-left Quadrant (Top View)**

Figure 2-3 is a map of the upper-right quadrant.

| 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | VID0 FIELD | VID0 OUTCLK | VID0 D1 | VID0 D5 | VID1 PIXCLK | VID1 D0 | VID1 D4 | VID1 D7 | VID01 MISO | RESETn | A |
| | VID0 PIXCLK | VID0 VSYNC | VID0 D2 | VID0 D6 | VID1 OUTCLK | VID1 D1 | VID1 D5 | VID01 MOSI | TEST | JTAG TAP_SEL | B |
| | CFG HOST1 | VID0 HSYNC | VID0 D3 | VID0 D7 | VID1 VSYNC | VID1 D2 | VID1 D6 | VID01 MCLK | JTAG TDI | JTAG TRSTn | C |
| | CFG HOST0 | VID0 D0 | VID0 D4 | VID1 FIELD | VID1 HSYNC | VID1 D3 | VID01 MSS | JTAG TDO | JTAG TCK | JTAG TMS | D |
| | CFG 0 | CORE VDD | VID01 VDD | VID01 GND | CORE GND | AUD GND | AUD1 MCLK | AUD1 BCK | AUD1 LRCK | AUD1 IDAT | E |
| | CORE VDD | VID01 VDD | VID01 VDD | CORE VDD | AUD GND | AUD VDD | AUD0 ODAT1 | AUD0 ODAT2 | AUD0 IDAT | AUD0 SPDIF | F |
| | | | | | AUD VDD | CORE GND | AUD0 MCLK | AUD0 BCK | AUD0 LRCK | AUD0 ODAT0 | G |
| | | | | | CORE VDD | DDR VDD | DDR DQ18 | DDR DQM2 | DDR DQS2n | DDR DQS2 | H |
| | | | | | DDR VDD | DDR GND | DDR DQ19 | DDR DQ20 | DDR DQ17 | DDR DQ16 | J |
| VID01 GND | DDR GND | | | | DDR VDD | DDR GND | DDR DQM3 | DDR DQ23 | DDR DQ21 | DDR DQ22 | K |
| CORE GND | DDR GND | | | | DDR VDD | DDR GND | DDR DQ26 | DDR DQ24 | DDR DQS3n | DDR DQS3 | L |

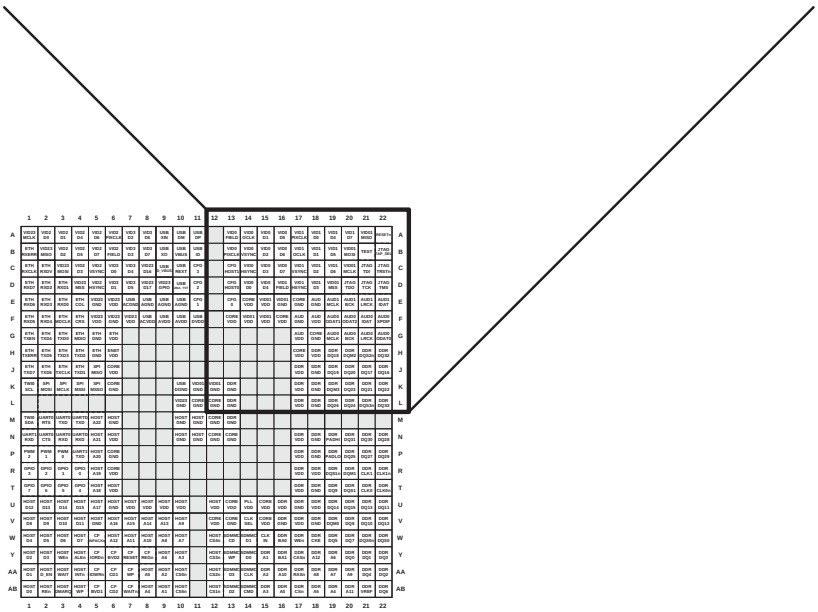


**Figure 2-3 Map of the Upper-Right Quadrant (Top View)**

Figure 2-4 is a map of the lower-right quadrant.

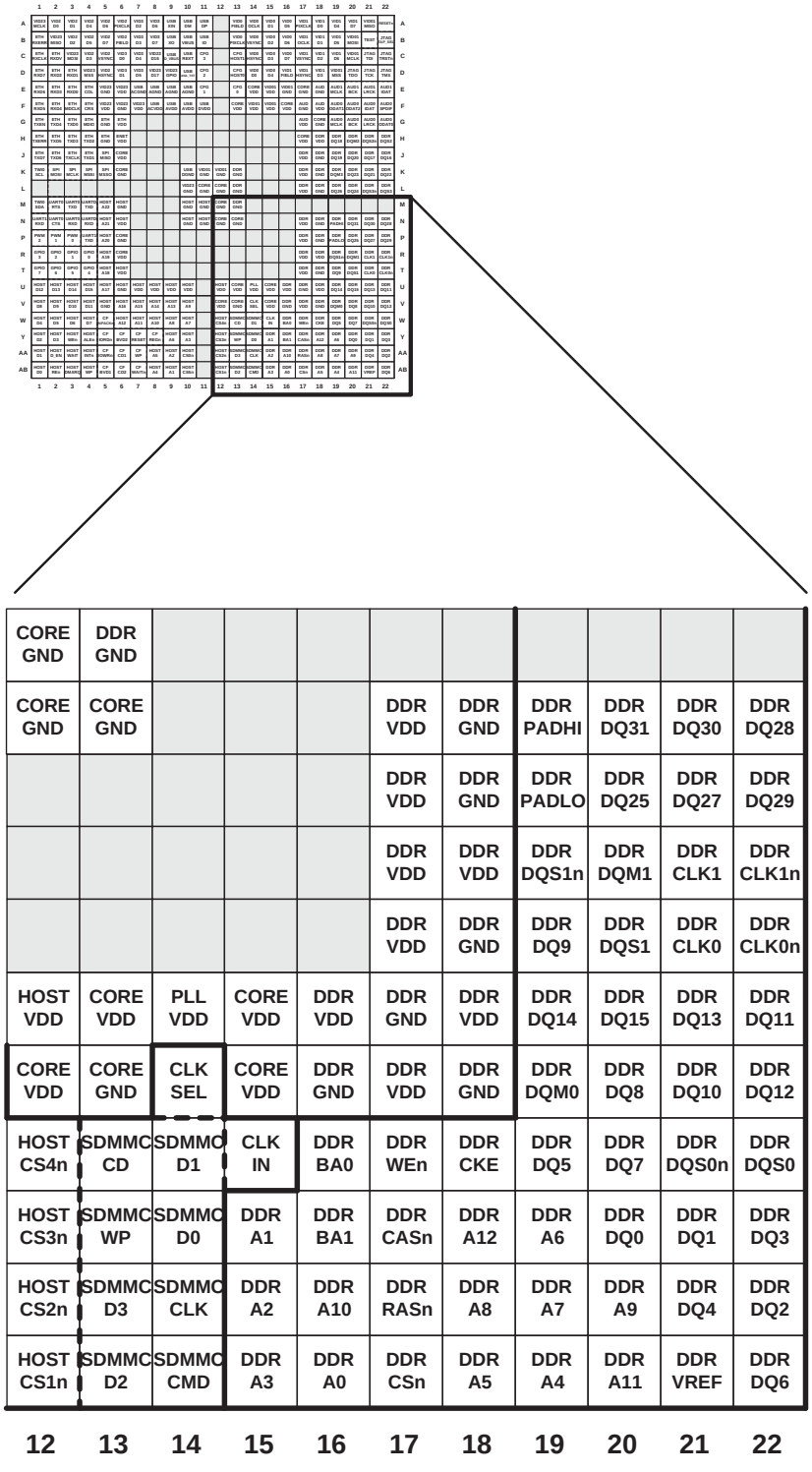



**Figure 2-4 Map of the Lower-Right Quadrant (Top View)**

Figure 2-5 is a map of the lower-left quadrant.

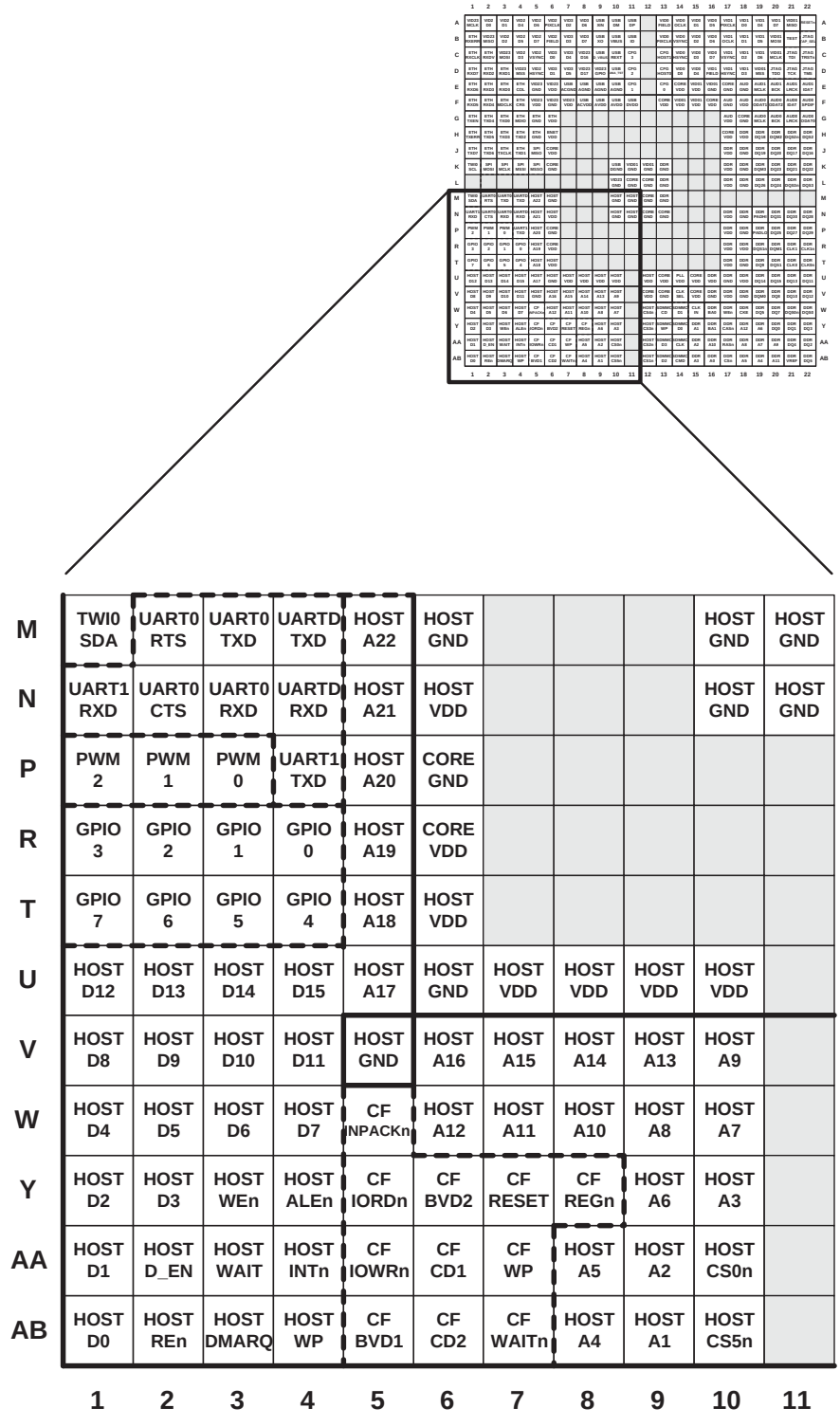



**Figure 2-5 Map of the Lower-left Quadrant (Top View)**

## 2.3 Pin Descriptions (by Interface)

This section provides a summary of the interfaces and corresponding signals of the MG3500 SoC. The 376 signals of the MG3500 SoC are divided into signal groups as shown in Table 2-1.

**Table 2-1 Signal Group Names**

| Signal Group | Group Name | Power Domain | Voltage Requirement | Signals |
|---|---|---|---|---|
| Core | CORE | CORE | 1.0 | 0 |
| Audio | AUDx | AUD | 1.8, 2.5, 3.3 | 12 |
| Video Ports 0 and 1 | VID01, VIDx | VID01 | 1.8, 2.5, 3.3 | 30 |
| Video Ports 2 and 3 | VID23, VIDx | VID23 | 1.8, 2.5, 3.3 | 27 |
| Host | HOST | HOST | 1.8, 2.5, 3.3 | 52 |
| Compact Flash | CF | HOST | 1.8, 2.5, 3.3 | 11 |
| DDR SDRAM | DDR | DDR | 1.8 | 71 |
| Ethernet | ETH | ETH | 3.3 | 26 |
| USB | USB | USB | 3.3 | 9 |
| SD/MMC | SD | HOST | 1.8, 2.5, 3.3 | 8 |
| UART | UART | HOST | 1.8, 2.5, 3.3 | 8 |
| SPI | SPI | HOST | 1.8, 2.5, 3.3 | 5 |
| TWI | TWI | HOST | 1.8, 2.5, 3.3 | 2 |
| PWM | PWM | HOST | 1.8, 2.5, 3.3 | 3 |
| GPIO | GPIO | HOST | 1.8, 2.5, 3.3 | 8 |
| Configuration | CFG | VID01 | 1.8, 2.5, 3.3 | 6 |
| Clock | CLK | HOST | 1.8, 2.5, 3.3 | 2 |
| Reset | RESET | AUD | 1.8, 2.5, 3.3 | 1 |
| JTAG | JTAG | AUD | 1.8, 2.5, 3.3 | 7 |
| Total Signals | | | | 288 |
| Power Connections | | | | 88 |
| Total Balls | | | | 376 |

In each group the signals are listed alphabetically by Primary signal name. The tables also include alternate functions for each signal that has either a secondary function (ALT column) or can be used as a GPIO (GPIO column). There is a column indicating the signal type: Input (I), Input/Output (I/O), Input/Open Drain output (IOD), or Analog (A).

The MG3500 SoC has independent power domains for various functions and the power domain for each Signal Group is also listed. The possible power domains are CORE, AUD, ETH, HOST, DDR, USB, VID01, and VID23.

### *2.3.1 Audio Signal Group*

The Audio Signal Group has 12 signals as shown in Table 2-2. It consists of two independent audio interfaces. Audio Group 0 contains one $I^2S$ input and three $I^2S$ outputs that share common clocking. Audio Group 1 contains one $I^2S$ input with independent clocking. These signals are all in the AUD power domain.

**Table 2-2 Audio Signals**

| Primary Signal | | ALT | GPIO | Ball | Description |
|---|---|---|---|---|---|
| Name | Type | | | | |
| AUD0_BCK | IO | – | – | G20 | Audio Port 0 $I^2S$ bit clock clocks input or output data |
| AUD0_IDAT | I | – | – | F21 | Audio Port 0 $I^2S$ input data |
| AUD0_LRCK | I/O | – | – | G21 | Audio Port 0 $I^2S$ left right clock indicates whether data is for the left or right channel |
| AUD0_MCLK | I/O | – | – | G19 | Audio Port 0 $I^2S$ Master clock (256 times the sampling clock) |
| AUD0_ODAT0 | O | – | – | G22 | Audio Port 0 $I^2S$ output data |
| AUD0_ODAT1 | O | – | – | F19 | Audio Port 0 $I^2S$ output data |
| AUD0_ODAT2 | O | – | GPIO_1_20 | F20 | Audio Port 0 $I^2S$ output data |
| AUD0_SPDIF | O | – | GPIO_1_21 | F22 | Audio Port 0 Sony/Philips digital interface |
| AUD1_BCK | I/O | – | GPIO_1_22 | E20 | Audio Port 1 $I^2S$ bit clock clocks input or output data |
| AUD1_IDAT | I | – | GPIO_1_24 | E22 | Audio Port 1 $I^2S$ input data |
| AUD1_LRCK | I/O | – | GPIO_1_23 | E21 | Audio Port 1 $I^2S$ left right clock indicates whether the data is for the left or right channel |
| AUD1_MCLK | I/O | – | – | E19 | Audio Port 1 $I^2S$ Master clock (256 times the sampling clock) |

### *2.3.2 Video Ports 0 and 1 Signal Group*

Video Ports 0 and 1 Signal Group include 30 signals to support two 8-bit video input ports or a single 16-bit video input port (see Table 2-3). The Signal Group also includes a serial control interface that can be used for configuring external video decoders, sensors, or other video interface devices. These signals are all in the VID01 power domain.

**Table 2-3 Video Ports 0 and 1 Signals**

| Primary Signal | | ALT | GPIO | Ball | Description |
|---|---|---|---|---|---|
| Name | Type | | | | |
| VID01_MCLK | O | VID01_SCL | GPIO_1_26 | C20 | Video Ports 0 and 1 Master Clock Video; Ports 0 and 1 serial clock |
| VID01_MISO | I | – | GPIO_1_28 | A21 | Video Ports 0 and 1 Master Input / Slave Output |
| VID01_MOSI | O | VID01_SDA | GPIO_1_27 | B20 | Video Ports 0 and 1 Master Output / Slave Input;<br>Video Ports 0 and 1 Serial Data |
| VID01_MSS | O | – | GPIO_1_25 | D19 | Video Ports 0 and 1 Slave Select |
| VID0_D7 | I | – | – | C16 | Video Port 0 Data [7:0] |
| VID0_D6 | I | – | – | B16 | |
| VID0_D5 | I | – | – | A16 | |
| VID0_D4 | I | – | – | D15 | |
| VID0_D3 | I | – | – | C15 | |
| VID0_D2 | I | – | – | B15 | |
| VID0_D1 | I | – | – | A15 | |
| VID0_D0 | I | – | – | D14 | |
| VID0_FIELD | I | – | GPIO_1_19 | A13 | Video Port 0 Field |
| VID0_HSYNC | I | – | GPIO_2_00 | C14 | Video Port 0 Hsync |
| VID0_OUTCLK | O | – | – | A14 | Video Port 0 Output Clock |
| VID0_PIXCLK | IO | – | – | B13 | Video Port 0 Pixel Clock |
| VID0_VSYNC | I | – | GPIO_2_01 | B14 | Video Port 0 Vsync |
| VID1_D7 | I | – | – | A20 | Video Port 1 Data [7:0] |
| VID1_D6 | I | – | – | C19 | |
| VID1_D5 | I | – | – | B19 | |
| VID1_D4 | I | – | – | A19 | |
| VID1_D3 | I | – | – | D18 | |
| VID1_D2 | I | – | – | C18 | |
| VID1_D1 | I | – | – | B18 | |
| VID1_D0 | I | – | – | A18 | |
| VID1_FIELD | I | – | GPIO_1_31 | D16 | Video Port 1 Field |
| VID1_HSYNC | I | – | GPIO_1_29 | D17 | Video Port 1 Hsync |
| VID1_OUTCLK | O | – | - | B17 | Video Port 1 Output Clock |

**Table 2-3 Video Ports 0 and 1 Signals**

| Primary Signal | | ALT | GPIO | Ball | Description |
|---|---|---|---|---|---|
| Name | Type | | | | |
| VID1_PIXCLK | IO | – | - | A17 | Video Port 1 Pixel Clock |
| VID1_VSYNC | I | – | GPIO_1_30 | C17 | Video Port 1 Vsync |

### *2.3.3 Video Ports 2 and 3 Signal Group*

The Video Ports 2 and 3 Signal Group includes 27 signals to support two 8-bit video input/output ports, or a single 16-bit video input/output port (see Table 2-4). They can also be combined to create an 18-bit wide RGB port to drive an LCD.

The signal group also includes a serial control interface that can be used for configuring external video decoders, sensors or other video interface devices. These signals are all in the VID23 power domain.

**Table 2-4 Video Ports 2 and 3 Signals**

| Primary Signal | | ALT | GPIO | Ball | Description |
|---|---|---|---|---|---|
| Name | Type | | | | |
| VID23_MCLK | O | VID23_SCL | GPIO_2_14 | A1 | Video Ports 2 and 3 Master Clock;<br>Video Ports 2 and 3 Serial Clock |
| VID23_MISO (for MG3500) | I | – | GPIO_2_12 | B2 | Video Ports 2 and 3 Master Input / Slave Output |
| VID23_MOSI (for MG3500) | O | VID23_SDA | GPIO_2_15 | C3 | Video Ports 2 and 3 Master Output / Slave Input;<br>Video Ports 2 and 3 Serial Data |
| VID23_MSS (for MG3500) | O | – | GPIO_2_13 | D4 | Video Ports 2 and 3 Slave Sync |
| VID23_MISO (for MG2580) | – | – | – | B2 | No connection |
| VID23_MOSI (for MG2580) | – | – | – | C3 | No connection |
| VID23_MSS (for MG2580) | – | – | – | D4 | No connection |
| VID2_D7 | I/O | – | – | B5 | Video Port 2 Data [7:0] |
| VID2_D6 | I/O | – | – | A5 | |
| VID2_D5 | I/O | – | – | B4 | |
| VID2_D4 | I/O | – | – | A4 | |
| VID2_D3 | I/O | – | – | C4 | |
| VID2_D2 | I/O | – | – | B3 | |
| VID2_D1 | I/O | – | – | A3 | |
| VID2_D0 | I/O | – | – | A2 | |
| VID2_FIELD | I/O | – | GPIO_2_09 | B6 | Video Port 2 Field |
| VID2_HSYNC | I/O | – | GPIO_2_10 | D5 | Video Port 2 Hsync |
| VID2_PIXCLK | I/O | – | – | A6 | Video Port 2 Pixel Clock |
| VID2_VSYNC | I/O | – | GPIO_2_11 | C5 | Video Port 2 Vsync |

**Table 2-4 Video Ports 2 and 3 Signals**

| Primary Signal | | ALT | GPIO | Ball | Description |
|---|---|---|---|---|---|
| Name | Type | | | | |
| VID3_D7 | I/O | – | – | B8 | Video Port 3 Data [7:0] |
| VID3_D6 | I/O | – | – | A8 | |
| VID3_D5 | I/O | – | – | D7 | |
| VID3_D4 | I/O | – | – | C7 | |
| VID3_D3 | I/O | – | – | B7 | |
| VID3_D2 | I/O | – | – | A7 | |
| VID3_D1 | I/O | – | – | D6 | |
| VID3_D0 | I/O | – | – | C6 | |
| VID23_D17 | I/O | – | GPIO_2_07 | D8 | Video Data [17] (for LCD with video Ports 2 and 3 data) |
| VID23_D16 | I/O | – | GPIO_2_08 | C8 | Video data [16] (for LCD with video Ports 2 and 3 data) |
| VID23_GPIO | I/O | – | GPIO_2_06 | D9 | Video Port 2/3 GPIO |

### *2.3.4 Host Signal Group*

The MG3500 HD H.264 Codec SoC Host Signal Group has 58 signals as shown in Table 2-5. When the MG3500 HD H.264 Codec SoC is in Master mode, the host bus is used to access external devices or memory including NAND Flash, NOR Flash, Compact Flash, or IDE drives. The use of Compact Flash or IDE also requires the use of the signals in the Compact Flash Signal Group (see page 38). When the MG3500 HD H.264 Codec SoC is in Slave mode, these signals are used to allow external processors to access resources inside the MG3500 HD H.264 Codec SoC. These signals are all in the HOST power domain.

In addition to the parallel host interface, the MG3500 HD H.264 Codec SoC can be accessed using a serial host interface that uses an interface similar to the Serial Peripheral Interface (SPI) with CPHA=1 and CPOL=1. The Host interface is described in detail in “Host Interfaces” .

The MG3500 SoC Host Interface connections in Master mode are shown in Figure 2-6.

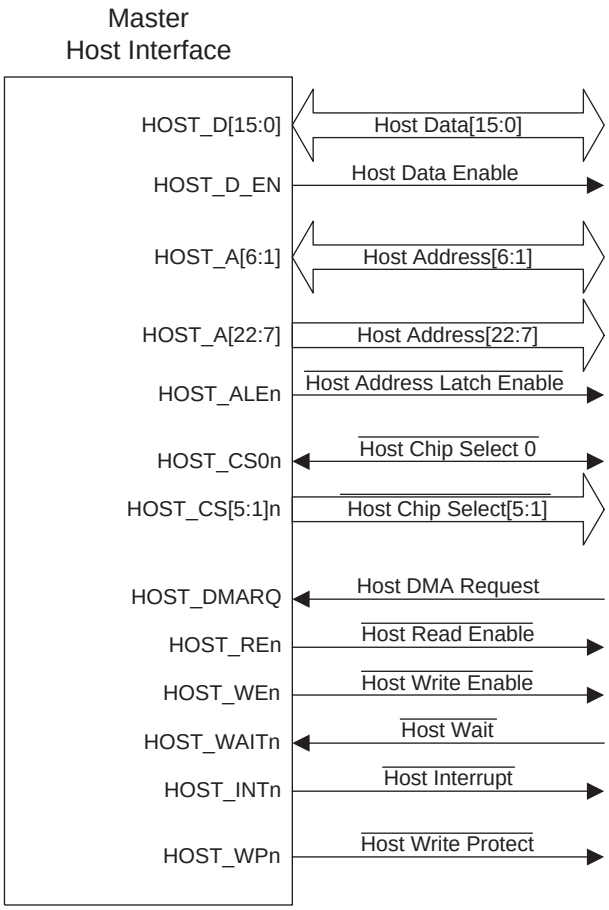



**Figure 2-6 Host Interface Master Mode Connections Diagram**

Figure 2-7 shows the connections when using the MG3500 SoC in Slave Host Interface mode.

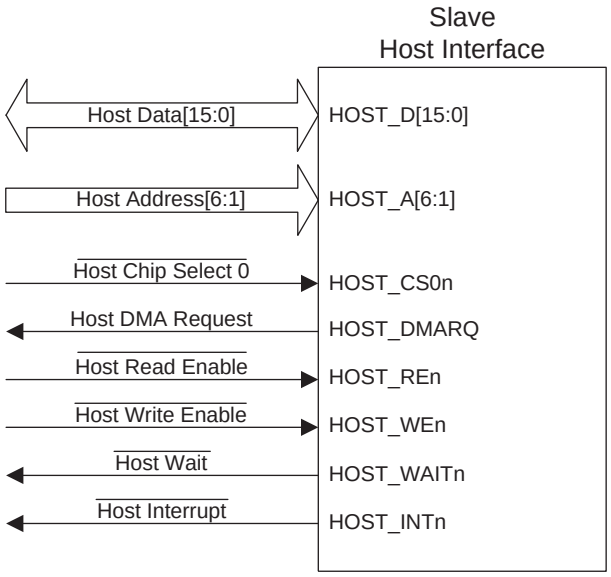



**Figure 2-7 Slave Host Connections**

Figure 2-8 shows the MG3500 SoC in Serial Host Interface mode.

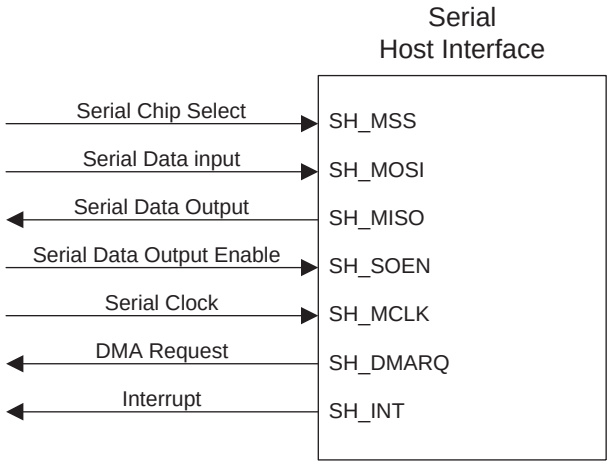



**Figure 2-8 Serial Host Connections**

As shown in Table 2-5, the signal type “IOD” refers to Input/Output open drain ports.

**Table 2-5 Host Signals**

| Primary Signal | | ALT | GPIO | Serial Host | Ball | Description |
|---|---|---|---|---|---|---|
| Name | Type | | | | | |
| HOST_A6 | I/O | SH_A6 | – | | Y9 | Host Address Bits [6:1] / Slave Host Address Bits [6:1] |
| HOST_A5 | I/O | SH_A5 | – | | AA8 | |
| HOST_A4 | I/O | SH_A4 | – | | AB8 | |
| HOST_A3 | I/O | SH_A3 | – | | Y10 | |
| HOST_A2 | I/O | SH_A2 | – | | AA9 | |
| HOST_A1 | I/O | SH_A1 | – | | AB9 | |
| HOST_A22 | O | – | – | | M5 | Host address bits [22:7] / Slave host address bits [22:7] |
| HOST_A21 | O | – | – | | N5 | |
| HOST_A20 | O | – | – | | P5 | |
| HOST_A19 | O | – | – | | R5 | |
| HOST_A18 | O | – | – | | T5 | |
| HOST_A17 | O | – | – | | U5 | |
| HOST_A16 | O | – | – | | V6 | |
| HOST_A15 | O | – | – | | V7 | |
| HOST_A14 | O | – | – | | V8 | |
| HOST_A13 | O | – | – | | V9 | |
| HOST_A12 | O | – | – | | W6 | |
| HOST_A11 | O | – | – | | W7 | |
| HOST_A10 | O | – | – | | W8 | |
| HOST_A9 | O | – | – | | V10 | |
| HOST_A8 | O | – | – | | W9 | |
| HOST_A7 | O | – | – | | W10 | |
| HOST_ALEn | O | – | – | | Y4 | Host Address Latch Enable |
| HOST_CS5n | O | | – | | AB10 | Host Chip Select/ Slave Host Chip Select [5:0] In Serial Host Mode, the HOST_CS0 pin acts as SH_MSS (Serial Host Chip Select). |
| HOST_CS4n | O | | – | | W12 | |
| HOST_CS3n | O | | – | | Y12 | |
| HOST_CS2n | O | | – | | AA12 | |
| HOST_CS1n | O | – | – | | AB12 | |
| HOST_CS0n | I/O | SH_CS0n | – | SH_ MSS | AA10 | |

**Table 2-5 Host Signals**

| Primary Signal | | ALT | GPIO | Serial Host | Ball | Description |
|---|---|---|---|---|---|---|
| Name | Type | | | | | |
| HOST_D15 | I/O | SH_D15 | – | | U4 | Host data bits [15:0] / Slave host data bits [15:0] In Serial Host mode, HOST_D1 acts as the serial data input, and HOST_D0 acts as the serial data output. |
| HOST_D14 | I/O | SH_D14 | – | | U3 | |
| HOST_D13 | I/O | SH_D13 | – | | U2 | |
| HOST_D12 | I/O | SH_D12 | – | | U1 | |
| HOST_D11 | I/O | SH_D11 | – | | V4 | |
| HOST_D10 | I/O | SH_D10 | – | | V3 | |
| HOST_D9 | I/O | SH_D9 | – | | V2 | |
| HOST_D8 | I/O | SH_D8 | – | | V1 | |
| HOST_D7 | I/O | SH_D7 | – | | W4 | |
| HOST_D6 | I/O | SH_D6 | – | | W3 | |
| HOST_D5 | I/O | SH_D5 | – | | W2 | |
| HOST_D4 | I/O | SH_D4 | – | | W1 | |
| HOST_D3 | I/O | SH_D3 | – | | Y2 | |
| HOST_D2 | I/O | SH_D2 | – | | Y1 | |
| HOST_D1 | I/O | SH_D1 | – | SH_ MOSI | AA1 | |
| HOST_D0 | I/O | SH_D0 | – | SH_ MISO | AB1 | |
| HOST_D_EN | O | – | – | SH_ SOEN | AA2 | Host Data Enable Serial Host Output Data Enable |
| HOST_DMARQ | I/O | SH_DMARQ | – | SH_ DMARQ | AB3 | Host DMA Request / Slave Host DMA Request Serial Host DMA Request |
| HOST_INTn | IOD | SH_INTn | GPIO_1_00 | SH_INT | AA4 | Host Interrupt / Slave Host Interrupt Serial Host Interrupt GPIO_1_0 In Host Slave mode, this signal is an open-collector output and requires a 1 kOhm pull-up resistor. |
| HOST_REn | I/O | SH_REn | – | | AB2 | Host Read Enable / Slave Host Read Enable |
| HOST_WAITn | IOD | SH_WAITn | – | | AA3 | Host Wait / Slave Host Wait: This signal is always active low in Slave mode, but the polarity is programmable in Host mode. In Host Slave mode, this signal is an open-collector output and requires a 1 KOhm pull-up resistor. |
| HOST_WEn | I/O | SH_WEn | – | SH_ MCLK | Y3 | Host Write Enable / Slave Host Write Enable Serial Host MCLK |

**Table 2-5 Host Signals**

| Primary Signal | | ALT | GPIO | Serial Host | Ball | Description |
|---|---|---|---|---|---|---|
| Name | Type | | | | | |
| HOST_WPn | O | – | – | | AB4 | Host Write Protect. Used with Flash memory. |

### *2.3.5 Compact Flash Signal Group*

The MG3500 SoC Compact Flash (CF) Signal Group has 11 signals as shown in Table 2-6. These signals are used in conjunction with the signals of the Host Signal Group to interface to Compact Flash or IDE devices. These signals are all in the HOST power domain.

**Table 2-6 CF Signals**

| Primary Signal | | ALT | GPIO | Ball | Description |
|---|---|---|---|---|---|
| Name | Type | | | | |
| CF_BVD1 | I/O | – | GPIO_1_04 | AB5 | Battery Voltage Detect 1 |
| CF_BVD2 | I/O | – | GPIO_1_05 | Y6 | Battery Voltage Detect 2 |
| CF_CD1 | I | – | GPIO_1_06 | AA6 | Card Detect 1 |
| CF_CD2 | I | – | GPIO_1_07 | AB6 | Card Detect 2 |
| CF_INPACKn | I | – | GPIO_1_01 | W5 | Input acknowledge |
| CF_IORDn | O | – | GPIO_1_02 | Y5 | I/O read strobe |
| CF_IOWRn | O | – | GPIO_1_03 | AA5 | I/O write strobe |
| CF_REGn | O | – | GPIO_1_11 | Y8 | Register select |
| CF_RESET | O | – | GPIO_1_08 | Y7 | Reset |
| CF_WAITn | I | – | GPIO_1_10 | AB7 | Wait |
| CF_WP | I | – | GPIO_1_09 | AA7 | Write Protect |

### *2.3.6 SDRAM Signal Group*

The MG3500 HD H.264 Codec SoC SDRAM Signal Group has 71 signals as shown in Table 2-7. The MG3500 HD H.264 Codec SoC supports both 1 x16 DDR2 SDRAM and 2 x16 DDR2 SDRAM configurations. These signals are all in the SDRAM power domain.

**Table 2-7 SDRAM Signals**

| Primary Signal | | ALT | GPIO | Ball | Description |
|---|---|---|---|---|---|
| Name | Type | | | | |
| DDR_PADHI | A | – | – | N19 | Driver compensation for DDR2 |
| DDR_PADLO | A | – | – | P19 | Driver compensation for DDR2 |
| DDR_A12 | O | – | – | Y18 | SDRAM Address Bits [12:0] |
| DDR_A11 | O | – | – | AB20 | |
| DDR_A10 | O | – | – | AA16 | |
| DDR_A9 | O | – | – | AA20 | |
| DDR_A8 | O | – | – | AA18 | |
| DDR_A7 | O | – | – | AA19 | |
| DDR_A6 | O | – | – | Y19 | |
| DDR_A5 | O | – | – | AB18 | |
| DDR_A4 | O | – | – | AB19 | |
| DDR_A3 | O | – | – | AB15 | |
| DDR_A2 | O | – | – | AA15 | |
| DDR_A1 | O | – | – | Y15 | |
| DDR_A0 | O | – | – | AB16 | |
| DDR_BA0 | O | – | – | W16 | Bank address bit [0] |
| DDR_BA1 | O | – | – | Y16 | Bank address bit [1] |
| DDR_CASn | O | – | – | Y17 | Column access strobe |
| DDR_CKE | O | – | – | W18 | Clock enable |
| DDR_CLK0 | I/O | – | – | T21 | Primary clock |
| DDR_CLK0n | O | – | – | T22 | Primary clock complement |
| DDR_CLK1 | I/O | – | – | R21 | Secondary clock |
| DDR_CLK1n | O | – | – | R22 | Secondary clock complement |
| DDR_CSn | O | – | – | AB17 | Chip select |

**Table 2-7 SDRAM Signals**

| Primary Signal | | ALT | GPIO | Ball | Description |
|---|---|---|---|---|---|
| Name | Type | | | | |
| DDR_DQ31 | I/O | – | – | N20 | SDRAM Data bits [31:0]<br><br>In 16-bit mode, DDR_DQ[31:16] and DDR_DQM[3:2] are not connected. |
| DDR_DQ30 | I/O | – | – | N21 | |
| DDR_DQ29 | I/O | – | – | P22 | |
| DDR_DQ28 | I/O | – | – | N22 | |
| DDR_DQ27 | I/O | – | – | P21 | |
| DDR_DQ26 | I/O | – | – | L19 | |
| DDR_DQ25 | I/O | – | – | P20 | |
| DDR_DQ24 | I/O | – | – | L20 | |
| DDR_DQ23 | I/O | – | – | K20 | |
| DDR_DQ22 | I/O | – | – | K22 | |
| DDR_DQ21 | I/O | – | – | K21 | |
| DDR_DQ20 | I/O | – | – | J20 | |
| DDR_DQ19 | I/O | – | – | J19 | |
| DDR_DQ18 | I/O | – | – | H19 | |
| DDR_DQ17 | I/O | – | – | J21 | |
| DDR_DQ16 | I/O | – | – | J22 | |
| DDR_DQ15 | I/O | – | – | U20 | |
| DDR_DQ14 | I/O | – | – | U19 | |
| DDR_DQ13 | I/O | – | – | U21 | |
| DDR_DQ12 | I/O | – | – | V22 | |
| DDR_DQ11 | I/O | – | – | U22 | |
| DDR_DQ10 | I/O | – | – | V21 | |
| DDR_DQ9 | I/O | – | – | T19 | |
| DDR_DQ8 | I/O | – | – | V20 | |
| DDR_DQ7 | I/O | – | – | W20 | |
| DDR_DQ6 | I/O | – | – | AB22 | |
| DDR_DQ5 | I/O | – | – | W19 | |
| DDR_DQ4 | I/O | – | – | AA21 | |
| DDR_DQ3 | I/O | – | – | Y22 | |
| DDR_DQ2 | I/O | – | – | AA22 | |
| DDR_DQ1 | I/O | – | – | Y21 | |
| DDR_DQ0 | I/O | – | – | Y20 | |
| DDR_DQM3 | O | – | – | K19 | Data masks for byte lanes 3:0<br><br>In 16-bit mode, DDR_DQ[31:16] and DDR_DQM[3:2] are not connected. |
| DDR_DQM2 | O | – | – | H20 | |
| DDR_DQM1 | O | – | – | R20 | |
| DDR_DQM0 | O | – | – | V19 | |

**Table 2-7 SDRAM Signals**

| Primary Signal | | ALT | GPIO | Ball | Description |
|---|---|---|---|---|---|
| Name | Type | | | | |
| DDR_DQS3 | I/O | – | – | L22 | Data strobes for byte lanes 3:0 |
| DDR_DQS2 | I/O | – | – | H22 | |
| DDR_DQS1 | I/O | – | – | T20 | |
| DDR_DQS0 | I/O | – | – | W22 | |
| DDR_DQS3n | I/O | – | – | L21 | Data strobe complements for byte lanes 3:0 |
| DDR_DQS2n | I/O | – | – | H21 | |
| DDR_DQS1n | I/O | – | – | R19 | |
| DDR_DQS0n | I/O | – | – | W21 | |
| DDR_RASn | O | – | – | AA17 | Row access strobe |
| DDR_VREF | A | – | – | AB21 | This pin should be set to ½ of VDD (0.9v) for DDR2 |
| DDR_WEn | O | – | – | W17 | Write enable |

### *2.3.7 Ethernet Signal Group*

The MG3500 HD H.264 Codec SoC Ethernet Signal Group has 26 signals as shown in Table 2-8. They support 10, 100, and GigaBit Ethernet connections via a Media Independent Interface (MII), Reduced Media Independent Interface (RMII), or a GigaBit Media Independent Interface (GMII) to an external Ethernet physical layer chip. The MG3500 HD H.264 Codec SoC may also be connected to an Ethernet switch chip if the switch supports the Reverse Media Independent Interface (RevMII). These signals are all in the ETH power domain.

**Table 2-8 Ethernet Signals**

| Primary Signal | | ALT | GPIO | Ball | Description |
|---|---|---|---|---|---|
| Name | Type | | | | |
| ETH_COL | I | – | – | E4 | Collision detect input |
| ETH_CRS | I | – | – | F4 | Carrier sense input |
| ETH_MDCLK | O | – | – | F3 | Management data clock |
| ETH_MDIO | I/O | – | – | G4 | Management data I/O |
| ETH_RXCLK | I | – | – | C1 | Receive clock input |
| ETH_RXD7 | I | – | – | D1 | Receive data input bits [7:0] |
| ETH_RXD6 | I | – | – | E1 | |
| ETH_RXD5 | I | – | – | F1 | |
| ETH_RXD4 | I | – | – | F2 | |
| ETH_RXD3 | I | – | – | E2 | |
| ETH_RXD2 | I | – | – | D2 | |
| ETH_RXD1 | I | – | – | D3 | |
| ETH_RXD0 | I | – | – | E3 | |
| ETH_RXDV | I | – | – | C2 | Receive data valid input |
| ETH_RXER | I | – | – | B1 | Receive error input |
| ETH_TXCLK | I | – | – | J3 | Transmit clock input |
| ETH_TXD7 | O | – | – | J1 | Transmit data output bits [7:0] |
| ETH_TXD6 | O | – | – | J2 | |
| ETH_TXD5 | O | – | – | H2 | |
| ETH_TXD4 | O | – | – | G2 | |
| ETH_TXD3 | O | – | – | H3 | |
| ETH_TXD2 | O | – | – | H4 | |
| ETH_TXD1 | O | – | – | J4 | |
| ETH_TXD0 | O | – | – | G3 | |
| ETH_TXEN | O | – | – | G1 | Transmit enable output |
| ETH_TXER | O | – | – | H1 | Transmit error output |

### *2.3.8 USB Signal Group*

The USB Signal group consists of 17 signals to support a USB 2.0 High-Speed On-The-Go (OTG), and a Host or Device interface (Table 2-9). These signals are all in the USB power domain.

**Table 2-9 USB Signals**

| Primary Signal | | ALT | GPIO | Ball | Description |
|---|---|---|---|---|---|
| Name | Type | | | | |
| USB_ANA_TST | A | – | – | D10 | Connect this signal to GND. Test mode signal for the USB analog sections. |
| USB_DM | A | – | – | A10 | USB D- signal |
| USB_DP | A | – | – | A11 | USB D+ signal |
| USB_D_VBUS | A | – | – | C9 | USB VBUS Drive signal. This active high signal is used to enable an external charge pump for USB_VBUS. |
| USB_ID | A | – | – | B11 | This signal differentiates a Mini-A from a Mini-B plug. The ID Detector senses the ID line's state to indicate which type of plug is connected. The ID Detector can differentiate the following conditions:<br>• ID pin floating (> 100 kilohms) = The connected plug is a mini-B plug.<br>• ID pin shorted to ground (< 10 ohms) = The connected plug is a mini-A plug. |
| USB_REXT | A | – | – | C10 | External 3.4 KOhm ±1% resistor connection that sets the bias current for the USB PHY. |
| USB_VBUS | A | – | – | B10 | Separate 5.0V supply for USB |
| USB_XIN | A | – | – | A9 | Crystal Oscillator XI pin. Connects a 12 MHz oscillator |
| USB_XO | A | – | – | B9 | Crystal Oscillator XO pin. Connects a 12 MHz oscillator |

### 2.3.9 SD and MMC Signal Group

The SD/MMC interface is designed to support Secure Digital (SD), Secure Digital Input/Output (SDIO), Multi-Media Card (MMC), and Consumer Electronics AT Attachment (CE-ATA) devices. This four-bit wide interface supports up to a 25 MHz clock rate (100 Mbits/sec. transfer rate). The SD/MMC Signal Group consists of eight signals as shown in Table 2-10. These signals are all in the HOST power domain.

**Table 2-10 SD and MMC Signals**

| Primary Signal | | ALT | GPIO | Ball | Description |
|---|---|---|---|---|---|
| Name | Type | | | | |
| SDMMC_CDn | I | – | GPIO_1_12 | W13 | Card detect |
| SDMMC_CLK | O | – | – | AA14 | Clock |
| SDMMC_CMD | I/O | – | GPIO_1_18 | AB14 | Command or response |
| SDMMC_D3 | I/O | – | GPIO_1_14 | AA13 | Data bit [3] |
| SDMMC_D2 | I/O | – | GPIO_1_15 | AB13 | Data bit [2] |
| SDMMC_D1 | I/O | – | GPIO_1_16 | W14 | Data bit [1] |
| SDMMC_D0 | I/O | – | GPIO_1_17 | Y14 | Data bit [0] |
| SDMMC_WP | I | – | GPIO_1_13 | Y13 | Write Protect |

**Note:** Use an SD card connector that includes the SD_WP and SD_CD signals or you will be limited to 1-bit mode.

### 2.3.10 UART Signal Group

Table 2-11 shows the Universal Asynchronous Receiver Transmitter (UART) Signal Group. These signals are all in the HOST power domain.

**Table 2-11 UART Signals**

| Primary Signal | | ALT | GPIO | Ball | Description |
|---|---|---|---|---|---|
| Name | Type | | | | |
| UARTD_RXD | I | MME_RXD[1] | – | N4 | Debug UART received data [2] |
| UARTD_TXD | O | MME_TXD[1] | – | M4 | Debug UART transmitted data |
| UART0_CTS | I | – | GPIO_2_26 | N2 | UART0 clear to send |
| UART0_RTS | O | – | GPIO_2_24 | M2 | UART0 request to send |
| UART0_RXD | I | – | GPIO_2_25 | N3 | UART0 received data |
| UART0_TXD | O | – | GPIO_2_23 | M3 | UART0 transmitted data |
| UART1_RXD | I | – | GPIO_2_27 | N1 | UART1 received data |
| UART1_TXD | O | – | GPIO_2_28 | P4 | UART1 transmitted data |

1. The alternate functions MME_RXD and MME_TXD are selected using the DBGUARTSel bit in the Serial I/O Control register. See “Serial Registers” for more information.
2. The Debug UART port is very useful in debugging the system and should always be connected.

### 2.3.11 SPI/Bitstream Signal Group

Table 2-12 shows the Serial Peripheral Interface/Bitstream (BS) Signal Group. These signals are all in the HOST power domain.

**Table 2-12 Serial Peripheral Interface/Bitstream Interface Signals**

| Primary Signal | | | | | |
|---|---|---|---|---|---|
| Name | Type | ALT | GPIO | Ball | Description |
| SPI_MCLK | I/O | BS_CLK[1] | GPIO_2_19 | K3 | SPI Master Clock<br>Bitstream Clock |
| SPI_MISO | I/O | | GPIO_2_16 | J5 | SPI Master In/ Slave Out |
| SPI_MOSI | I/O | BS_DATA | GPIO_2_20 | K2 | SPI Master Out / Slave In<br>Bitstream Data |
| SPI_MSS0 | I/O | BS_EN | GPIO_2_17 | K5 | SPI Master / Slave Select 0<br>Bitstream Data Enable |
| SPI_MSS1 | IO | BS_REQ | GPIO_2_18 | K4 | SPI Master / Slave Select 1<br>Bitstream Data Request |

1. The alternate function BS_CLK, BS_DATA, BS_EN, and BSREQ are selected using bits in the Serial I/O Control register. See “Serial Registers” for more information.

### 2.3.12 TWI Signal Group

Table 2-13 shows the $I^2C$-Compatible Two-Wire Interface (TWI) Signal Group. These signals are all in the HOST power domain.

**Table 2-13 Two-Wire Interface Signals**

| Primary Signal | | | | | |
|---|---|---|---|---|---|
| Name | Type | ALT | GPIO | Ball | Description |
| TWI0_SCL | IOD | TWI1_SCL[1] | GPIO_2_21 | K1 | TWI serial clock |
| TWI0_SDA | IOD | TWI1_SDA | GPIO_2_22 | M1 | TWI serial data |

1. The alternate functions TWI1_SCL and TWI1_SDA are selected using the TWI1Cfg bit in the Serial I/O Control register. See “Serial Registers” for more information.

### 2.3.13 PWM Signal Group

Table 2-14 shows the Pulse-Width Modulator (PWM) Signal Group. These signals are all in the HOST power domain.

**Table 2-14 Pulse Width Modulator Signals**

| Primary Signal | | ALT | GPIO | Ball | Description |
|---|---|---|---|---|---|
| Name | Type | | | | |
| PWM_0 | O | – | GPIO_2_29 | P3 | PMI Output 0 |
| PWM_1 | O | – | GPIO_2_30 | P2 | PMI Output 1 |
| PWM_2 | O | – | GPIO_2_31 | P1 | PMI Output 2 |

### 2.3.14 GPIO Signal Group

The GPIO Signal Group has eight signals as shown in Table 2-15. They are dedicated General Purpose Input/Output (GPIO) signals. These dedicated GPIO signals are all in the HOST power domain.

The I/O pins in the GPIO Signal Group have programmable 15 KOhm ±20% pull-up and pull-down resistors. The pull-up resistors are enabled by default, and can be disabled using the associated bit in the GPIO 0 Pull-up Enable register. The pull-down resistors are disabled by default, and can be enabled using the GPIO 0 Pull-down Enable register. See “Serial Registers” for more information.

**Table 2-15 GPIO Signals**

| Primary Signal | | ALT | GPIO | Ball | Description |
|---|---|---|---|---|---|
| Name | Type | | | | |
| GPIO_0 | I/O | – | GPIO_0_00 | R4 | GPIO bit [0] |
| GPIO_1 | I/O | – | GPIO_0_01 | R3 | GPIO bit [1] |
| GPIO_2 | I/O | – | GPIO_0_02 | R2 | GPIO bit [2] |
| GPIO_3 | I/O | – | GPIO_0_03 | R1 | GPIO bit [3] |
| GPIO_4 | I/O | – | GPIO_0_04 | T4 | GPIO bit [4] |
| GPIO_5 | I/O | – | GPIO_0_05 | T3 | GPIO bit [5] |
| GPIO_6 | I/O | – | GPIO_0_06 | T2 | GPIO bit [6] |
| GPIO_7 | I/O | – | GPIO_0_07 | T1 | GPIO bit [7] |

There are 64 other GPIO signals that are multiplexed with other signals. These pins can be used as GPIOs if neither their Primary function or their Alternate function (ALT) are not being used. These additional GPIO signals are broken into the two groups as shown in Table 2-16 and Table 2-17, and are not necessarily in the HOST power domain. Refer to the primary signal (listed under SIGNAL NAME) to check the power domain.

The multiplexed signals associated with GPIO_1 are disabled by default, and enabled using the associated bits in the GPIO 1 Sel register (see “Serial Registers” ). When enabled, the I/O function has priority over both the Primary and the Alternate function (ALT). The I/O pins in the GPIO1 Signal Group have programmable 15 KOhm ±20% pull-up and pull-down resistors. The pull-up resistors are enabled by default, and can be disabled using the associated bit in the GPIO 1 Pull-up Enable register.

The pull-down resistors are disabled by default, and can be enabled using the GPIO 1 Pull-down Enable register.

**Table 2-16 GPIO Signals**

| GPIO | Primary Signal | | ALT | Ball | Power Domain | Voltage Tolerance | See Page |
|---|---|---|---|---|---|---|---|
| | Name | Type | | | | | |
| GPIO_1_00 | HOST_INTn | I/O | – | AA4 | HOST | 1.8, 2.5, 3.3 | 33 |
| GPIO_1_01 | CF_INPACKn | I | – | W5 | HOST | 1.8, 2.5, 3.3 | 38 |
| GPIO_1_02 | CF_IORDn | O | – | Y5 | HOST | 1.8, 2.5, 3.3 | 38 |
| GPIO_1_03 | CF_IOWRn | O | – | AA5 | HOST | 1.8, 2.5, 3.3 | 38 |
| GPIO_1_04 | CF_BVD1 | I/O | – | AB5 | HOST | 1.8, 2.5, 3.3 | 38 |
| GPIO_1_05 | CF_BVD2 | I/O | – | Y6 | HOST | 1.8, 2.5, 3.3 | 38 |
| GPIO_1_06 | CF_CD1 | I | – | AA6 | HOST | 1.8, 2.5, 3.3 | 38 |
| GPIO_1_07 | CF_CD2 | I | – | AB6 | HOST | 1.8, 2.5, 3.3 | 38 |
| GPIO_1_08 | CF_RESET | O | – | Y7 | HOST | 1.8, 2.5, 3.3 | 38 |
| GPIO_1_09 | CF_WP | I | – | AA7 | HOST | 1.8, 2.5, 3.3 | 38 |
| GPIO_1_10 | CF_WAITn | I | – | AB7 | HOST | 1.8, 2.5, 3.3 | 38 |
| GPIO_1_11 | CF_REGn | O | – | Y8 | HOST | 1.8, 2.5, 3.3 | 38 |
| GPIO_1_12 | SDMMC_CDn | I | – | W13 | HOST | 1.8, 2.5, 3.3 | 44 |
| GPIO_1_13 | SDMMC_WP | I | – | Y13 | HOST | 1.8, 2.5, 3.3 | 44 |
| GPIO_1_14 | SDMMC_D3 | I/O | – | AA13 | HOST | 1.8, 2.5, 3.3 | 44 |
| GPIO_1_15 | SDMMC_D2 | I/O | – | AB13 | HOST | 1.8, 2.5, 3.3 | 44 |
| GPIO_1_16 | SDMMC_D1 | I/O | – | W14 | HOST | 1.8, 2.5, 3.3 | 44 |
| GPIO_1_17 | SDMMC_D0 | I/O | – | Y14 | HOST | 1.8, 2.5, 3.3 | 44 |
| GPIO_1_18 | SDMMC_CMD | I/O | – | AB14 | HOST | 1.8, 2.5, 3.3 | 44 |
| GPIO_1_19 | VID0_FIELD | O | – | A13 | VID01 | 1.8, 2.5, 3.3 | 28 |
| GPIO_1_20 | AUD0_ODAT2 | O | – | F20 | AUD | 1.8, 2.5, 3.3 | 28 |
| GPIO_1_21 | AUD0_SPDIF | O | – | F22 | AUD | 1.8, 2.5, 3.3 | 28 |
| GPIO_1_22 | AUD1_BCK | I/O | – | E20 | AUD | 1.8, 2.5, 3.3 | 28 |
| GPIO_1_23 | AUD1_LRCK | I/O | – | E21 | AUD | 1.8, 2.5, 3.3 | 28 |
| GPIO_1_24 | AUD1_IDAT | I | – | E22 | AUD | 1.8, 2.5, 3.3 | 28 |
| GPIO_1_25 | VID01_MSS | O | – | D19 | VID01 | 1.8, 2.5, 3.3 | 29 |
| GPIO_1_26 | VID01_MCLK | O | VID01_SCL | C20 | VID01 | 1.8, 2.5, 3.3 | 29 |
| GPIO_1_27 | VID01_MOSI | O | VID01_SDA | B20 | VID01 | 1.8, 2.5, 3.3 | 29 |
| GPIO_1_28 | VID01_MISO | I | – | A21 | VID01 | 1.8, 2.5, 3.3 | 29 |
| GPIO_1_29 | VID1_HSYNC | I | – | D17 | VID01 | 1.8, 2.5, 3.3 | 29 |
| GPIO_1_30 | VID1_VSYNC | I | – | C17 | VID01 | 1.8, 2.5, 3.3 | 29 |
| GPIO_1_31 | VID1_FIELD | I | – | D16 | VID01 | 1.8, 2.5, 3.3 | 29 |

The multiplexed signals associated with GPIO_2 are enabled using the associated bits in the GPIO 2 Sel register (see “Serial Registers” ). The GPIO_2_31 to GPIO_2_21 and GPIO_2_15 to GPIO_2_0 pins are disabled by default (the Primary/ALT function is active). GPIO_2_20 to GPIO_2_ are enabled by default, which forces the signals to be an input after reset.

When enabled, the I/O function has priority over both the Primary and the Alternate function (ALT). The I/O pins in the GPIO2 Signal Group have programmable 15 KOhm ±20% pull-up and pull-down resistors. The pull-up resistors are enabled by default, and can be disabled using the associated bit in the GPIO 2 Pull-up Enable register. The pull-down resistors are disabled by default, and can be enabled using the GPIO 2 Pull-down Enable register.

**Table 2-17 Additional GPIO Signals**

| GPIO | Primary Signal | | ALT | Ball | Power Domain | Voltage Tolerance | See Page |
|---|---|---|---|---|---|---|---|
| | Name | Type | | | | | |
| GPIO_2_00 | VID0_HSYNC | I | – | C14 | VID01 | 1.8, 2.5, 3.3 | 29 |
| GPIO_2_01 | VID0_VSYNC | I | – | B14 | VID01 | 1.8, 2.5, 3.3 | 29 |
| GPIO_2_02 | CFG_0 | I | – | E13 | VID01 | 1.8, 2.5, 3.3 | 29 |
| GPIO_2_03 | CFG_1 | I | – | E11 | VID01 | 1.8, 2.5, 3.3 | 50 |
| GPIO_2_04 | CFG_2 | I | – | D11 | VID01 | 1.8, 2.5, 3.3 | 50 |
| GPIO_2_05 | CFG_3 | I | – | C11 | VID01 | 1.8, 2.5, 3.3 | 50 |
| GPIO_2_06 | VID23_GPIO | I/O | – | D9 | VID23 | 1.8, 2.5, 3.3 | 31 |
| GPIO_2_07 | VID_DATA_17 | I/O | – | D8 | VID23 | 1.8, 2.5, 3.3 | 31 |
| GPIO_2_08 | VID_DATA_16 | I/O | – | C8 | VID23 | 1.8, 2.5, 3.3 | 31 |
| GPIO_2_09 | VID2_FIELD | I/O | – | B6 | VID23 | 1.8, 2.5, 3.3 | 31 |
| GPIO_2_10 | VID2_HSYNC | I/O | – | D5 | VID23 | 1.8, 2.5, 3.3 | 31 |
| GPIO_2_11 | VID2_VSYNC | I/O | – | C5 | VID23 | 1.8, 2.5, 3.3 | 31 |
| GPIO_2_12[1] | VID23_MISO | I | – | B2 | VID23 | 1.8, 2.5, 3.3 | 31 |
| GPIO_2_13[2] | VID23_MSS | O | – | D4 | VID23 | 1.8, 2.5, 3.3 | 31 |
| GPIO_2_14 | VID23_MCLK | O | VID23_SCL | A1 | VID23 | 1.8, 2.5, 3.3 | 31 |
| GPIO_2_15[3] | VID23_MOSI | O | VID23_SDA | C3 | VID23 | 1.8, 2.5, 3.3 | 31 |
| GPIO_2_16 | SPI_MISO | IO | – | J5 | HOST | 1.8, 2.5, 3.3 | 45 |
| GPIO_2_17 | SPI_MSS0 | IO | BS_ENABLE | K5 | HOST | 1.8, 2.5, 3.3 | 45 |
| GPIO_2_18 | SPI_MSS1 | IO | BS_REQ | K4 | HOST | 1.8, 2.5, 3.3 | 45 |
| GPIO_2_19 | SPI_MCLK | IO | BS_CLK | K3 | HOST | 1.8, 2.5, 3.3 | 45 |
| GPIO_2_20 | SPI_MOSI | IO | BS_DATA | K2 | HOST | 1.8, 2.5, 3.3 | 45 |
| GPIO_2_21 | TWI0_SCL | IOD | TWI1_SCL | K1 | HOST | 1.8, 2.5, 3.3 | 45 |
| GPIO_2_22 | TWI0_SDA | IOD | TWI1_SDA | M1 | HOST | 1.8, 2.5, 3.3 | 45 |
| GPIO_2_23 | UART0_TXD | O | – | M3 | HOST | 1.8, 2.5, 3.3 | 44 |
| GPIO_2_24 | UART0_RTS | O | – | M2 | HOST | 1.8, 2.5, 3.3 | 44 |
| GPIO_2_25 | UART0_RXD | I | – | N3 | HOST | 1.8, 2.5, 3.3 | 44 |
| GPIO_2_26 | UART0_CTS | I | – | N2 | HOST | 1.8, 2.5, 3.3 | 44 |
| GPIO_2_27 | UART1_RXD | I | – | N1 | HOST | 1.8, 2.5, 3.3 | 44 |
| GPIO_2_28 | UART1_TXD | O | – | P4 | HOST | 1.8, 2.5, 3.3 | 44 |
| GPIO_2_29 | PWM_0 | O | – | P3 | HOST | 1.8, 2.5, 3.3 | 46 |
| GPIO_2_30 | PWM_1 | O | – | P2 | HOST | 1.8, 2.5, 3.3 | 46 |
| GPIO_2_31 | PWM_2 | O | – | P1 | HOST | 1.8, 2.5, 3.3 | 46 |

1. This pin does not apply to MG2580. See Table 2-4 for more information about MG2580 pin descriptions.
2. The same as above.
3. The same as above.

### 2.3.15 JTAG Signal Group

The JTAG Signal Group has seven signals as shown in Table 2-18. These signals are all in the AUD power domain.

**Table 2-18 JTAG Signals**

| Primary Signal | | ALT | GPIO | Ball | Description |
|---|---|---|---|---|---|
| Name | Type | | | | |
| JTAG_TAP_SEL | I | – | – | B22 | This signal is used to select between the ARM tap controller and the test mode tap controller:<br>0:ARM Debugger<br>1: Boundary Scan |
| TEST | I | – | – | B21 | When set to 1, the chip is placed in test mode. |
| JTAG_TCK | I | – | – | D21 | JTAG test clock. |
| JTAG_TDI | I | – | – | C21 | JTAG test data input. |
| JTAG_TDO | O | – | – | D20 | JTAG test data output. |
| JTAG_TMS | I | – | – | D22 | JTAG test mode select. |
| JTAG_TRSTn | I | – | – | C22 | JTAG test reset active Low. |

### 2.3.16 Configuration

The Configuration Signal Group has six signals as shown in Table 2-19. These signals are all in the VID01 power domain. The configuration mode is determined at boot-up by the state of the CFG_[3:0] pins. See “Boot modes for the MMEs and the ARM” for more information. When the MG3500 SoC powers up in Serial Slave mode (CFG_HOST[1:0]=11), the CFG_[3:0] pins are not used and can be used as GPIO pins.

**Table 2-19 Configuration Signals**

| Primary Signal | | ALT | GPIO | Ball | Description |
|---|---|---|---|---|---|
| Name | Type | | | | |
| CFG_0 | I/O | – | GPIO_2_02 | E13 | General Purpose Configuration (GPC) input |
| CFG_1 | I/O | – | GPIO_2_03 | E11 | GPC input |
| CFG_2 | I/O | – | GPIO_2_04 | D11 | GPC input |
| CFG_3 | I/O | – | GPIO_2_05 | C11 | GPC input |
| CFG_HOST0 | I/O | – | – | D13 | 00: Parallel Slave<br>01: Master<br>10: Reserved<br>11: Serial Slave |
| CFG_HOST1 | I/O | – | – | C13 | |

### *2.3.17 Clock*

The Clock Signal Group has two signals as shown in Table 2-20. These signals are all in the HOST power domain. See “Clock and PLL inputs” on page 99.

**Table 2-20 Clock Signals**

| Primary Signal | | ALT | GPIO | Ball | Description |
|---|---|---|---|---|---|
| Name | Type | | | | |
| CLK_IN | I | – | – | W15 | Clock input |
| CLK_SEL | I | – | – | V14 | Selects the source clock for the PLLs to come from either the USB oscillator or CLK_IN.<br>0 = USB Oscillator<br>1 = External CLK_IN |

### *2.3.18 Reset*

The Reset Signal Group has one signal as shown in Table 2-21. This signal is in the AUD power domain.

**Table 2-21 Reset Signals**

| Primary Signal | | ALT | GPIO | Ball | Description |
|---|---|---|---|---|---|
| Name | Type | | | | |
| RESETn | I | – | – | A22 | Active Low chip reset |

## 2.4 Power and Ground Pins

**Table 2-22 Power Pins**

| Signal Name | Ball | Description | Voltage |
|---|---|---|---|
| AUD_VDD | F18 | Power for the Audio Circuitry | 1.8, 2.5, 3.3 Volts |
| AUD_VDD | G17 | | |
| AUD_GND | E18 | Ground for the Audio Circuitry | – |
| AUD_GND | F17 | | |
| CORE_VDD | E14 | Power for the Core Logic | 1.05 Volts |
| CORE_VDD | F13 | | |
| CORE_VDD | F16 | | |
| CORE_VDD | H17 | | |
| CORE_VDD | J6 | | |
| CORE_VDD | R6 | | |
| CORE_VDD | U13 | | |
| CORE_VDD | U15 | | |
| CORE_VDD | V12 | | |
| CORE_VDD | V15 | | |
| CORE_GND | E17 | Ground for the Core Logic | – |
| CORE_GND | G18 | | |
| CORE_GND | K6 | | |
| CORE_GND | L11 | | |
| CORE_GND | L12 | | |
| CORE_GND | M12 | | |
| CORE_GND | N12 | | |
| CORE_GND | N13 | | |
| CORE_GND | P6 | | |
| CORE_GND | V13 | | |

**Table 2-22 Power Pins**

| Signal Name | Ball | Description | Voltage |
|---|---|---|---|
| DDR_VDD | H18 | Power for the DDR Memory Controller | 1.8 Volts |
| DDR_VDD | J17 | | |
| DDR_VDD | K17 | | |
| DDR_VDD | L17 | | |
| DDR_VDD | N17 | | |
| DDR_VDD | P17 | | |
| DDR_VDD | R17 | | |
| DDR_VDD | R18 | | |
| DDR_VDD | T17 | | |
| DDR_VDD | U16 | | |
| DDR_VDD | U18 | | |
| DDR_VDD | V17 | | |
| DDR_GND | J18 | Ground for the DDR Memory Controller | – |
| DDR_GND | K13 | | |
| DDR_GND | K18 | | |
| DDR_GND | L13 | | |
| DDR_GND | L18 | | |
| DDR_GND | M13 | | |
| DDR_GND | N18 | | |
| DDR_GND | P18 | | |
| DDR_GND | T18 | | |
| DDR_GND | U17 | | |
| DDR_GND | V16 | | |
| DDR_GND | V18 | | |
| ETH_VDD | G6 | Power for the Ethernet circuitry | 3.3 Volts |
| ETH_VDD | H6 | | |
| ETH_GND | G5 | Ground for the Ethernet circuitry | – |
| ETH_GND | H5 | | |
| HOST_VDD | T6 | Power for the Host Processor | 1.8, 2.5, 3.3 Volts |
| HOST_VDD | U10 | | |
| HOST_VDD | U12 | | |
| HOST_VDD | U7 | | |
| HOST_VDD | U8 | | |
| HOST_VDD | U9 | | |
| HOST_VDD | N6 | | |

**Table 2-22 Power Pins**

| Signal Name | Ball | Description | Voltage |
|---|---|---|---|
| HOST_GND | M6 | Ground for the Host Processor | – |
| HOST_GND | M10 | | |
| HOST_GND | M11 | | |
| HOST_GND | N10 | | |
| HOST_GND | N11 | | |
| HOST_GND | U6 | | |
| HOST_GND | V5 | | |
| PLL_VDD | U14 | Power for the Phase Lock Loop | 1.05 Volts |
| USB_DVDD | F11 | Digital Power for the USB Port | 1.05 Volts |
| USB_AVDD | F10 | Power for the USB Port | 3.3 Volts |
| USB_AVDD | F9 | Power for the USB Port | |
| USB_ACVDD | F8 | Power for the USB Port | |
| USB_AGND | E8 | Analog Ground for the USB Port | – |
| USB_AGND | E9 | Analog Ground for the USB Port | |
| USB_AGND | E10 | Analog Ground for the USB Port | |
| USB_ACGND | E7 | Analog Ground for the USB Port | |
| USB_DGND | K10 | Digital Ground for the USB Port | |
| VID01_VDD | E15 | Power for Video Ports 0 and 1 | 1.8, 2.5, 3.3 Volts |
| VID01_VDD | F14 | | |
| VID01_VDD | F15 | | |
| VID01_GND | E16 | Ground for Video Ports 0 and 1 | – |
| VID01_GND | K11 | | |
| VID01_GND | K12 | | |
| VID23_VDD | E6 | Power for Video Ports 2 and 3 | 1.8, 2.5, 3.3 Volts |
| VID23_VDD | F5 | | |
| VID23_VDD | F7 | | |
| VID23_GND | E5 | Ground for Video Ports 2 and 3 | – |
| VID23_GND | F6 | | |
| VID23_GND | L10 | | |

## 2.5 Pin List by Power Group

Table 2-23 shows the signals associated with each of the power domains.

**Table 2-23 Signal Group Names**

| Power Domain | Voltage Requirement | Group Name | Signals |
|---|---|---|---|
| HOST | 1.8, 2.5, 3.3 | HOST | HOST_A[6:1], HOST_A[22:7], HOST_ALEn, HOST_CS[5:0]n, HOST_D[15:0], HOST_D_EN, HOST_DMARQ, HOST_INTn, HOST_REn, HOST_WAITn, HOST_WEn, HOST_WP |
| | | CF | CF_BVD1, CF_BVD2, CF_CD1, CF_CD2, CF_INPACKn, CF_IORDn, CF_IOWRn, CF_REGn, CF_RESET, CF_WAITn, CF_WP |
| | | SD | SDMMC_CD, SDMMC_CLK, SDMMC_CMD, SDMMC_D3, SDMMC_D2, SDMMC_D1, SDMMC_D0, SDMMC_WP |
| | | UART | UARTD_RXD, UARTD_TXD, UART0_CTS, UART0_RTS, UART0_RXD, UART0_TXD, UART1_RXD, UART1_TXD |
| | | SPI/BS | SPI_MCLK, SPI_MISO, SPI_MOSI, SPI_MSS0, SPI_MSS1 |
| | | TWI | TWI0_SCL, TWI0_SDA |
| | | PWM | PWM_0, PWM_1, PWM_2 |
| | | GPIO | GPIO_0_[0:7] [1]<br>GPIO_1_00, GPIO_1_01, GPIO_1_02, GPIO_1_03, GPIO_1_04, GPIO_1_05, GPIO_1_06, GPIO_1_07, GPIO_1_08, GPIO_1_09, GPIO_1_10, GPIO_1_11, GPIO_1_12, GPIO_1_13, GPIO_1_14, GPIO_1_15, - GPIO_1_16, GPIO_1_17, GPIO_1_18,<br>GPIO_2_16, GPIO_2_17, GPIO_2_18, GPIO_2_19, GPIO_2_20, GPIO_2_21, GPIO_2_22, GPIO_2_23, GPIO_2_24, GPIO_2_25, GPIO_2_26, GPIO_2_27, GPIO_2_28, GPIO_2_29, GPIO_2_30, GPIO_2_31 |
| | | CLK | CLK_IN, CLK_SEL |
| AUD | 1.8, 2.5, 3.3 | AUDx | AUD0_BCK, AUD0_IDAT, AUD0_LRCK, AUD0_MCLK, AUD0_ODAT0, AUD0_ODAT1, AUD0_ODAT2, AUD0_SPDIF, AUD1_BCK, AUD1_IDAT, AUD1_LRCK, AUD1_MCLK |
| | | RESET | RESETn |
| | | JTAG | JTAG_TAP_SEL, TEST, JTAG_TCK, JTAG_TDI, JTAG_TDO, JTAG_TMS, JTAG_TRSTn |
| | | GPIO | GPIO_1_20, GPIO_1_21, GPIO_1_22, GPIO_1_23, GPIO_1_24 |
| CORE | 1.05 | CORE | No core signals are brought out directly to the I/O pins. |
| DDR | 1.8 | DDR | DDR_PADHI, DDR_PADLO, DDR_A[12:0], DDR_BA0, DDR_BA1, DDR_CASn, DDR_CKE, DDR_CLK0, DDR_CLK0n, DDR_CLK1, DDR_CLK1n, DDR_CSn, DDR_DQ[31:0], DDR_DQM[3:0], DDR_DQS[3:0], DDR_DQS[3:0]n, DDR_RASn, DDR_VREF, DDR_WEn |

**Table 2-23 Signal Group Names**

| Power Domain | Voltage Requirement | Group Name | Signals |
|---|---|---|---|
| ETH | 3.3 | ETH | ETH_COL, ETH_CRS,<br>ETH_MDCLK, ETH_MDIO, ETH_RXCLK,<br>ETH_RXD[7:0], ETH_RXDV, ETH_RXER,<br>ETH_TXCLK, ETH_TXD[7:0], ETH_TXEN,<br>ETH_TXER |
| USB | 3.3 | USB | USB_ANA_TST, USB_DM, USB_DP, USB_D_VBUS, USB_ID, USB_REXT, USB_XIN, USB_XO |
| | 5.0 | | USB_VBUS |
| VID01 | 1.8, 2.5, 3.3 | VID01, VIDx | VID01_MCLK, VID01_MISO, VID01_MOSI, VID01_MSS<br>VID0_D[7:0], VID0_FIELD, VID0_HSYNC, VID0_OUTCLK, VID0_PIXCLK, VID0_VSYNC,<br>VID1_D[7:0], VID1_FIELD, VID1_HSYNC, VID1_OUTCLK, VID1_PIXCLK, VID1_VSYNC |
| | | CFG | CFG_0, CFG_1, CFG_2, CFG_3, CFG_HOST0, CFG_HOST1 |
| | | GPIO | GPIO_1_19, GPIO_1_25, GPIO_1_26, GPIO_1_27, GPIO_1_28, GPIO_1_29, GPIO_1_30, GPIO_1_31;<br>GPIO_2_0, GPIO_2_1, GPIO_2_2, GPIO_2_3, GPIO_2_4, GPIO_2_5 |
| VID23 | 1.8, 2.5, 3.3 | VID23, VIDx | VID23_MCLK, VID23_MISO, VID23_MOSI, VID23_MSS,<br>VID2_D[7:0], VID2_FIELD, VID2_HSYNC, VID2_PIXCLK, VID2_VSYNC,<br>VID3_D[7:0], VID23_D17, VID23_D16, VID3_GPIO |
| | | GPIO | GPIO_2_6, GPIO_2_7, GPIO_2_8, GPIO_2_9, GPIO_2_10, GPIO_2_11;<br>GPIO_2_12, GPIO_2_13, GPIO_2_14, GPIO_2_15 |

1. Only GPIO_0_[0:7] are the dedicated GPIOs; All other GPIO signals are multiplexed with other signals listed in Table 2-23. For example, the primary function of GPIO_1_00 is a “host interrupt.” See Table 2-16 and Table 2-17 for detailed description.

### 2.6 Hookup Recommendations when Interfaces Are Unused

Table 2-24 shows the hookup recommendations when some of the interfaces are unused. The pull-up/pull-down column indicates:

- UP: The pin has the internal pull-up enabled at power-on/reset.
- DOWN: The pin has the internal pull-down enabled at power-on/reset.
- DIS: The pin has pull-up/pull-down control, but they are disabled at power-on/reset.
- NONE: The pin has no control over pull-up/pull-down at all.

The Default column indicates the state the pin is in at reset:

- 0: The pin is driven to 0.
- 1: The pin is driven to 1.
- 0(p): The pin is pulled by a resistor to a 0 value.
- 1(p): The pin is pulled by resistor to a 1 value.
- Hi-Z: The pin is not driven.
- —: The pin is an Input Only, and must be driven.
- NC: The pin is a no connect (leave it unconnected).

**Note:** SDRAM and power pins are not included in this list since they must always be connected for the device to operate correctly. This also applies when the USB block is not used on MG3500.

Internal pull-up and pull-down values are 15 KOhm ± 20%.

**Table 2-24 Hookup Recommendations when Interfaces Are Unused**

| Pin Name | Dir | Pad | Type | Pull-up/ Pull-down | Default | Recommendation if the Interface is Not Used |
|---|---|---|---|---|---|---|
| **VIDEO_PORT 0/1** | | | | | | |
| VID01_MCLK | O | IO | GPIO | UP | 0 | NC |
| VID01_MISO | I | IO | GPIO | UP | 1(p) | NC, pulled up by default |
| VID01_MOSI | O | IO | GPIO | UP | 1(p) | NC, pulled up by default |
| VID01_MSS | O | IO | GPIO | UP | 1 | NC |
| VID0_D0 | I | I | input_only | NONE | — | GND |
| VID0_D1 | I | I | input_only | NONE | — | GND |
| VID0_D2 | I | I | input_only | NONE | — | GND |
| VID0_D3 | I | I | input_only | NONE | — | GND |
| VID0_D4 | I | I | input_only | NONE | — | GND |
| VID0_D5 | I | I | input_only | NONE | — | GND |
| VID0_D6 | I | I | input_only | NONE | — | GND |
| VID0_D7 | I | I | input_only | NONE | — | GND |
| VID0_PIXCLK | IO | IO | | NONE | Hi–Z | GND |
| VID0_FIELD | I | IO | GPIO | UP | 1(p) | NC, pulled up by default |

**Table 2-24 Hookup Recommendations when Interfaces Are Unused**

| Pin Name | Dir | Pad | Type | Pull-up/ Pull-down | Default | Recommendation if the Interface is Not Used |
|---|---|---|---|---|---|---|
| VID0_HSYNC | I | IO | GPIO | UP | 1(p) | NC, pulled up by default |
| VID0_VSYNC | I | IO | GPIO | UP | 1(p) | NC, pulled up by default |
| VID0_OUTCLK | O | O | output_only | NONE | Hi–Z | NC |
| VID1_D0 | I | I | input_only | NONE | — | GND |
| VID1_D1 | I | I | input_only | NONE | — | GND |
| VID1_D | I | I | input_only | NONE | — | GND |
| VID1_D3 | I | I | input_only | NONE | — | GND |
| VID1_D4 | I | I | input_only | NONE | — | GND |
| VID1_D5 | I | I | input_only | NONE | — | GND |
| VID1_D6 | I | I | input_only | NONE | — | GND |
| VID1_D7 | I | I | input_only | NONE | — | GND |
| VID1_PIXCLK | IO | IO | | NONE | Hi–Z | GND |
| VID1_FIELD | I | IO | GPIO | UP | 1(p) | NC, pulled up by default |
| VID1_HSYNC | I | IO | GPIO | UP | 1(p) | NC, pulled up by default |
| VID1_VSYNC | I | IO | GPIO | UP | 1(p) | NC, pulled up by default |
| VID1_OUTCLK | O | O | output_only | NONE | Hi–Z | NC |
| **VIDEO_PORT 2/3** | | | | | | |
| VID23_MCLK | O | IO | GPIO | UP | 0 | NC |
| VID23_MISO[1] | I | IO | GPIO | UP | 1(p) | NC, pulled up by default |
| VID23_MOSI[2] | O | IO | GPIO | UP | 1(p) | NC, pulled up by default |
| VID23_MSS[3] | O | IO | GPIO | UP | 1 | NC |
| VID2_D0 | IO | IO | | NONE | Hi–Z | NC, configure as output after reset |
| VID2_D1 | IO | IO | | NONE | Hi–Z | NC, configure as output after reset |
| VID2_D2 | IO | IO | | NONE | Hi–Z | NC, configure as output after reset |
| VID2_D3 | IO | IO | | NONE | Hi–Z | NC, configure as output after reset |
| VID2_D4 | IO | IO | | NONE | Hi–Z | NC, configure as output after reset |
| VID2_D5 | IO | IO | | NONE | Hi–Z | NC, configure as output after reset |
| VID2_D6 | IO | IO | | NONE | Hi–Z | NC, configure as output after reset |
| VID2_D7 | IO | IO | | NONE | Hi–Z | NC, Configure as output after reset |
| VID2_PIXCLK | IO | IO | | NONE | Hi–Z | NC, Configure as output after reset |
| VID2_FIELD | IO | IO | GPIO | UP | 1(p) | NC, pulled up by default |
| VID2_HSYNC | IO | IO | GPIO | UP | 1(p) | NC, pulled up by default |

**Table 2-24 Hookup Recommendations when Interfaces Are Unused**

| Pin Name | Dir | Pad | Type | Pull-up/ Pull-down | Default | Recommendation if the Interface is Not Used |
|---|---|---|---|---|---|---|
| VID2_VSYNC | IO | IO | GPIO | UP | 1(p) | NC, pulled up by default |
| VID3_D0 | IO | IO | | NONE | Hi–Z | NC, configure as output after reset |
| VID3_D1 | IO | IO | | NONE | Hi–Z | NC, configure as output after reset |
| VID3_D2 | IO | IO | | NONE | Hi–Z | NC, configure as output after reset |
| VID3_D3 | IO | IO | | NONE | Hi–Z | NC, configure as output after reset |
| VID3_D4 | IO | IO | | NONE | Hi–Z | NC, configure as output after reset |
| VID3_D5 | IO | IO | | NONE | Hi–Z | NC, configure as output after reset |
| VID3_D6 | IO | IO | | NONE | Hi–Z | NC, configure as output after reset |
| VID3_D7 | IO | IO | | NONE | Hi–Z | NC, configure as output after reset |
| VID23_D17 | IO | IO | GPIO | UP | 1(p) | NC, pulled up by default |
| VID23_D16 | IO | IO | GPIO | UP | 1(p) | NC, pulled up by default |
| VID23_GPIO | IO | IO | GPIO | UP | 1(p) | NC, pulled up by default |

**Table 2-24 Hookup Recommendations when Interfaces Are Unused**

| Pin Name | Dir | Pad | Type | Pull-up/ Pull-down | Default | Recommendation if the Interface is Not Used |
|---|---|---|---|---|---|---|
| **USB[4]** | | | | | | |
| USB_D_VBUS | A | IO | | | | NC |
| USB_VBUS | A | IO | | | | NC |
| USB_DP | A | IO | | | | NC |
| USB_DM | A | IO | | | | NC |
| USB_ID | A | IO | | | | NC |
| USB_ANA_TST | A | IO | | | | NC |
| USB_REXT | A | IO | | | | USB_AVDD |
| USB_XIN | A | IO | | | | NC |
| USB_XO | A | IO | | | | NC |
| **Ethernet** | | | | | | |
| ETH_MDIO | IO | IO | NONE | Hi–Z | 0 | NC |
| ETH_MDCLK | O | IO | NONE | 0 | 1 | NC |
| ETH_TXCLK | I | I | NONE | — | — | GND |
| ETH_TXD0 | O | IO | NONE | 0 | 1 | NC |
| ETH_TXD1 | O | IO | NONE | 0 | 1 | NC |
| ETH_TXD2 | O | IO | NONE | 0 | 1 | NC |
| ETH_TXD3 | O | IO | NONE | 0 | 1 | NC |
| ETH_TXD4 | O | IO | NONE | 0 | 1 | NC |
| ETH_TXD5 | O | IO | NONE | 0 | 1 | NC |
| ETH_TXD6 | O | IO | NONE | 0 | 1 | NC |
| ETH_TXD7 | O | IO | NONE | 0 | 1 | NC |
| ETH_TXEN | O | IO | NONE | 0 | 1 | NC |
| ETH_TXER | O | IO | NONE | 0 | 1 | NC |
| ETH_RXCLK | I | I | NONE | — | — | GND |
| ETH_RXD0 | I | I | NONE | — | — | GND |
| ETH_RXD1 | I | I | NONE | — | — | GND |
| ETH_RXD2 | I | I | NONE | — | — | GND |
| ETH_RXD3 | I | I | NONE | — | — | GND |
| ETH_RXD4 | I | I | NONE | — | — | GND |
| ETH_RXD5 | I | I | NONE | — | — | GND |
| ETH_RXD6 | I | I | NONE | — | — | GND |

**Table 2-24 Hookup Recommendations when Interfaces Are Unused**

| Pin Name | Dir | Pad | Type | Pull-up/ Pull-down | Default | Recommendation if the Interface is Not Used |
|---|---|---|---|---|---|---|
| ETH_RXD7 | I | I | NONE | — | — | GND |
| ETH_RXER | I | I | NONE | — | — | GND |
| ETH_RXDV | I | I | NONE | — | — | GND |
| ETH_COL | I | I | NONE | — | — | GND |
| ETH_CRS | I | I | NONE | — | — | GND |
| **AUDIO** | | | | | | |
| AUD0_BCK | IO | IO | | NONE | Hi–Z | NC, configure as output after reset |
| AUD0_IDAT | I | I | GPIO (input_only) | NONE | Hi–Z | GND |
| AUD0_LRCK | IO | IO | | NONE | Hi–Z | NC, configure as output after reset |
| AUD0_MCLK | IO | IO | | NONE | Hi–Z | NC, configure as output after reset |
| AUD0_ODAT0 | O | O | output_only | NONE | 0 | NC |
| AUD0_ODAT1 | O | O | output_only | NONE | 0 | NC |
| AUD0_ODAT2 | O | IO | GPIO | UP | 0 | NC |
| AUD0_SPDIF | O | IO | GPIO | UP | 0 | NC |
| AUD1_BCK | IO | IO | GPIO | UP | 1(p) | NC, pulled up by default |
| AUD1_IDAT | I | IO | GPIO | UP | 1(p) | NC, pulled up by default |
| AUD1_LRCK | IO | IO | GPIO | UP | 1(p) | NC, pulled up by default |
| AUD1_MCLK | IO | IO | | NONE | Hi–Z | NC, configure as output after reset |
| **PWM** | | | | | | |
| PWM_0 | O | IO | GPIO | UP | 1 | NC |
| PWM_1 | O | IO | GPIO | UP | 1 | NC |
| PWM_2 | O | IO | GPIO | UP | 1 | NC |
| **SDMMC** | | | | | | |
| SDMMC_CD | I | IO | GPIO | UP | 1(p) | NC, pulled up by default |
| SDMMC_CLK | O | O | output_only | NONE | 0 | NC |
| SDMMC_D0 | O | IO | GPIO | UP | 1(p) | NC, pulled up by default |
| SDMMC_D1 | O | IO | GPIO | UP | 1(p) | NC, pulled up by default |
| SDMMC_D2 | O | IO | GPIO | UP | 1(p) | NC, pulled up by default |
| SDMMC_D3 | O | IO | GPIO | UP | 1(p) | NC, pulled up by default |
| SDMMC_CMD | IO | IO | GPIO | UP | 1(p) | NC, pulled up by default |
| SDMMC_WP | I | IO | GPIO | UP | 1(p) | NC, pulled up by default |

**Table 2-24 Hookup Recommendations when Interfaces Are Unused**

| Pin Name | Dir | Pad | Type | Pull-up/ Pull-down | Default | Recommendation if the Interface is Not Used |
|---|---|---|---|---|---|---|
| **GPIO** | | | | | | |
| GPIO_0 | IO | IO | GPIO | UP | 1(p) | NC |
| GPIO_1 | IO | IO | GPIO | UP | 1(p) | NC |
| GPIO_2 | IO | IO | GPIO | UP | 1(p) | NC |
| GPIO_3 | IO | IO | GPIO | UP | 1(p) | NC |
| GPIO_4 | IO | IO | GPIO | UP | 1(p) | NC |
| GPIO_5 | IO | IO | GPIO | UP | 1(p) | NC |
| GPIO_6 | IO | IO | GPIO | UP | 1(p) | NC |
| GPIO_7 | IO | IO | GPIO | UP | 1(p) | NC |
| **TWI** | | | | | | |
| TWI0_SCL | IO | IO | GPIO | UP | 1(p) | NC, pulled up by default |
| TWI0_SDA | IO | IO | GPIO | UP | 1(p) | NC, pulled up by default |
| **SPI** | | | | | | |
| SPI_MCLK | IO | IO | GPIO | UP | 1(p) | NC, pulled up by default |
| SPI_MISO | IO | IO | GPIO | UP | 1(p) | NC, pulled up by default |
| SPI_MOSI | IO | IO | GPIO | UP | 1(p) | NC, pulled up by default |
| SPI_MSS0 | IO | IO | GPIO | UP | 1(p) | NC, pulled up by default |
| SPI_MSS1 | IO | IO | GPIO | UP | 1(p) | NC, pulled up by default |
| **UART** | | | | | | |
| UARTD_RXD | I | I | input_only | NONE | — | Hook up to DBG_TXD |
| UARTD_TXD | O | O | output_only | NONE | 1 | Hook up to DBG_RXD |
| UART0_CTS | I | IO | GPIO | UP | 1(p) | NC, pulled up by default |
| UART0_RTS | O | IO | GPIO | UP | 1 | NC |
| UART0_RXD | I | IO | GPIO | UP | 1(p) | NC, pulled up by default |
| UART0_TXD | O | IO | GPIO | UP | 1 | NC |
| UART1_RXD | I | IO | GPIO | UP | 1(p) | NC, pulled up by default |
| UART1_TXD | O | IO | GPIO | UP | 1 | NC |

**Table 2-24 Hookup Recommendations when Interfaces Are Unused**

| Pin Name | Dir | Pad | Type | Pull-up/ Pull-down | Default | Recommendation if the Interface is Not Used |
|---|---|---|---|---|---|---|
| **HOST** | | | | | | |
| HOST_A1 | O | IO | | NONE | Hi–Z | GND |
| HOST_A2 | O | IO | | NONE | Hi–Z | GND |
| HOST_A3 | O | IO | | NONE | Hi–Z | GND |
| HOST_A4 | O | IO | | NONE | Hi–Z | GND |
| HOST_A5 | O | IO | | NONE | Hi–Z | GND |
| HOST_A6 | O | IO | | NONE | Hi–Z | GND |
| HOST_A7 | O | O | output_only | NONE | 0 | NC |
| HOST_A8 | O | O | output_only | NONE | 0 | NC |
| HOST_A9 | O | O | output_only | NONE | 0 | NC |
| HOST_A10 | O | O | output_only | NONE | 0 | NC |
| HOST_A11 | O | O | output_only | NONE | 0 | NC |
| HOST_A12 | O | O | output_only | NONE | 0 | NC |
| HOST_A13 | O | O | output_only | NONE | 0 | NC |
| HOST_A14 | O | O | output_only | NONE | 0 | NC |
| HOST_A15 | O | O | output_only | NONE | 0 | NC |
| HOST_A16 | O | O | output_only | NONE | 0 | NC |
| HOST_A17 | O | O | output_only | NONE | 0 | NC |
| HOST_A18 | O | O | output_only | NONE | 0 | NC |
| HOST_A19 | O | O | output_only | NONE | 0 | NC |
| HOST_A20 | O | O | output_only | NONE | 0 | NC |
| HOST_A21 | O | O | output_only | NONE | 0 | NC |
| HOST_A22 | O | O | output_only | NONE | 0 | NC |
| HOST_ALEn | O | O | output_only | NONE | 1 | NC |
| HOST_CS_5n | O | O | output_only | NONE | 1 | NC |
| HOST_CS_4n | O | O | output_only | NONE | 1 | NC |
| HOST_CS_3n | O | O | output_only | NONE | 1 | NC |
| HOST_CS_2n | O | O | output_only | NONE | 1 | NC |
| HOST_CS_1n | O | IO | | NONE | 1 | NC |
| HOST_CS_0n | O | IO | | NONE | 1 | NC |
| HOST_D0 | IO | IO | | NONE | Hi–Z | GND |
| HOST_D1 | IO | IO | | NONE | Hi–Z | GND |

**Table 2-24 Hookup Recommendations when Interfaces Are Unused**

| Pin Name | Dir | Pad | Type | Pull-up/ Pull-down | Default | Recommendation if the Interface is Not Used |
|---|---|---|---|---|---|---|
| HOST_D2 | IO | IO | | NONE | Hi–Z | GND |
| HOST_D3 | IO | IO | | NONE | Hi–Z | GND |
| HOST_D4 | IO | IO | | NONE | Hi–Z | GND |
| HOST_D5 | IO | IO | | NONE | Hi–Z | GND |
| HOST_D6 | IO | IO | | NONE | Hi–Z | GND |
| HOST_D7 | IO | IO | | NONE | Hi–Z | GND |
| HOST_D8 | IO | IO | | NONE | Hi–Z | GND |
| HOST_D9 | IO | IO | | NONE | Hi–Z | GND |
| HOST_D10 | IO | IO | | NONE | Hi–Z | GND |
| HOST_D11 | IO | IO | | NONE | Hi–Z | GND |
| HOST_D12 | IO | IO | | NONE | Hi–Z | GND |
| HOST_D13 | IO | IO | | NONE | Hi–Z | GND |
| HOST_D14 | IO | IO | | NONE | Hi–Z | GND |
| HOST_D15 | IO | IO | | NONE | Hi–Z | GND |
| HOST_WPn | O | O | output_only | NONE | 0 | NC |
| HOST_WAITn | I | IO | | NONE | Hi–Z | GND |
| HOST_INTn | I | IO | | UP | 1(p) | GND |
| HOST_REn | O | IO | | NONE | 1 | NC |
| HOST_WEn | O | IO | | NONE | 1 | NC |
| HOST_DMARQ | I | IO | | NONE | 0 | GND |
| HOST_D_EN | O | O | output_only | NONE | 0 | NC |
| CFG_HOST0 | I | I | input_only | NONE | 0 | GND |
| CFG_HOST1 | I | I | input_only | NONE | 0 | GND |
| CFG_0 | I | IO | GPIO | UP | 1(p) | NC, pulled up by default |
| CFG_1 | I | IO | GPIO | UP | 1(p) | NC, pulled up by default |
| CFG_2 | I | IO | GPIO | UP | 1(p) | NC, pulled up by default |
| CFG_3 | I | IO | GPIO | UP | 1(p) | NC, pulled up by default |

**Table 2-24 Hookup Recommendations when Interfaces Are Unused**

| Pin Name | Dir | Pad | Type | Pull-up/ Pull-down | Default | Recommendation if the Interface is Not Used |
|---|---|---|---|---|---|---|
| **CF** | | | | | | |
| CF_WP | I | IO | GPIO | UP | 1(p) | NC, pulled up by default |
| CF_WAITn | I | IO | GPIO | UP | 1(p) | NC, pulled up by default |
| CF_IORDn | O | IO | GPIO | UP | 1 | NC |
| CF_IOWRn | O | IO | GPIO | UP | 1 | NC |
| CF_REGn | O | IO | GPIO | UP | 1 | NC |
| CF_RESET | O | IO | GPIO | UP | 1 | NC |
| CF_BVD1 | I | IO | GPIO | UP | 1(p) | NC, pulled up by default |
| CF_BVD2 | I | IO | GPIO | UP | 1(p) | NC, pulled up by default |
| CF_CD1 | I | IO | GPIO | UP | 1(p) | NC, pulled up by default |
| CF_CD2 | I | IO | GPIO | UP | 1(p) | NC, pulled up by default |
| CF_INPACKn | I | IO | GPIO | DOWN | 0(p) | NC, pulled down by default |
| **JTAG** | | | | | | |
| JTAG_TAP_SEL | I | I | input_only | NONE | 1 | GND |
| TEST | I | I | input_only | NONE | 0 | GND |
| JTAG_TCK | I | I | input_only | UP | 1(p) | GND |
| JTAG_TDI | I | I | input_only | UP | 1(p) | Hook up to TEST_TDO |
| JTAG_TDO | O | O | output-only | UP | 1(p) | Hook up to TEST_TDI |
| JTAG_TMS | I | I | input_only | UP | 1(p) | GND |
| JTAG_TRSTn | I | I | input_only | UP | 1(p) | GND |

1. This pin does not apply to MG2580. See Table 2-4 for more information about MG2580 pin descriptions.
2. The same as above.
3. The same as above.
4. When the USB block is not used, in addition to connecting the USB pins as recommended in Table 2-24, the USB VDD pins still must be connected to their standard supply levels, as shown below:
   - USB_DVDD 1.05 V
   - USB_AVDD 3.3 V
   - USB_ACVDD 3.3 V

# 3.0 Device Configuration

## 3.1 Reset

When the device is first powered on, the power supplies must be brought up in the order shown in "Power Supply Sequencing" on page 193. Once the power supplies are stable follow this procedure to reset the MG3500 SoC:

When the MG3500 SoC is being reset into Master (SOC) mode:

1. Set CLK_SEL pin to select the source clock for the PLLs. The clock can come from either the internal USB oscillator or the CLK_IN pin.
2: Set the CFG_HOST[1:0] pins to 01 to select the Master configuration mode.
3: Set the boot mode using the CFG_[3-0] pins. See section 3.2 for more information.
4: Assert the RESETn pin low for at least one microsecond and then release it.

At this point, the ARM boot ROM will then start the initialization process.

When the MG3500 SoC is being reset into Slave (Coprocessor) mode:

1. Set CLK_SEL pin to select the source clock for the PLLs. The clock can come from either the internal USB oscillator or the CLK_IN pin.
2: Set the CFG_HOST[1:0] pins to either 00 to select Parallel Slave configuration mode or 11 to select Serial Slave configuration mode.
3: Using the CFG_[3-0] pins, set the boot mode to 0xC (boot disabled).
4: Assert the RESETn pin low for at least one microsecond and then release it.

At this point, the Video Multi-Media Engine (MME) is ready to accept the firmware download. Configure the Configuration/Status Registers, download the firmware, and start the other clocks. Then wait for the MG3500 SoC to return a valid GPB (Global Pointer Block) before proceeding.

## 3.2 Boot modes for the MMEs and the ARM

At power up, an on-chip ROM that contains boot code starts executing. It will check for the clock source (CLK_SEL), boot mode (CFG[0:3]) and continue on to start copying the bootloader from the specified boot device.

Ensure that systems booting from NAND use the correct configuration. Each configuration will use address cycles appropriate for the selected type of NAND.

Figure 3-1 lists the boot modes supported by the MG3500 in SoC mode.

**Table 3-1 Boot Modes**

| CFG_3 | CFG_2 | CFG_1 | CFG_0 | Boot Mode |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | Load from large page 8-bit NAND <= 1 Gbits on CS1 [1] |
| 0 | 0 | 0 | 1 | Load from large page 8-bit NAND > 1 Gbits on CS1 [1.] |
| 0 | 0 | 1 | 0 | Load from large page 16-bit NAND <= 1 Gbits on CS1 |
| 0 | 0 | 1 | 1 | Load from large page 16-bit NAND > 1 Gbits on CS1 |
| 0 | 1 | 0 | 0 | Load from small page 8-bit NAND <= 256 Mbits on CS1 [1.] |

**Table 3-1 Boot Modes**

| CFG_3 | CFG_2 | CFG_1 | CFG_0 | Boot Mode |
|---|---|---|---|---|
| 0 | 1 | 0 | 1 | Load from small page 8-bit NAND > 256 Mbits on CS1 [1]. |
| 0 | 1 | 1 | 0 | Reserved |
| 0 | 1 | 1 | 1 | Reserved |
| 1 | 0 | 0 | 0 | Reserved. |
| 1 | 0 | 0 | 1 | Load from SPI EEPROM (SPI 0). |
| 1 | 0 | 1 | 0 | Reserved. |
| 1 | 0 | 1 | 1 | Load from UARTDBG using Xmodem. |
| 1 | 1 | 0 | 0 | Boot disabled. |
| 1 | 1 | 0 | 1 | Boot from DDR. |
| 1 | 1 | 1 | 0 | Load from NOR Flash on CS0 using last 16 kBytes. |
| 1 | 1 | 1 | 1 | Reserved. |

1. When 8-bit memories are specified, the data is taken from the upper 8-bits of the 16-bit data bus (HOST_D[15:8]).

### 3.3 Firmware Loader

**TBD:** A description of the firmware loader and how it loads the operating software into the device.

### 3.4 API Configuration

The API that is supplied initializes the internal registers as part of the configuration process. These registers include the:

- Configuration and Control registers
- Power control registers (core power, different I/O powers, etc.)
- Clock and PLL registers

The default configuration for the Clock and PLL registers assumes that you are using the 12 MHz USB crystal as the primary clock source. It is also possible to drive the device using an externally generated 24 or 27 MHz clock. If you plan on using one of these external clocks, contact Mobilygen Technical Support for a specialized version of the API.

### 3.5 Pin Multiplexing, GPIOs, etc.

All shared I/O pins come up in the primary interface mode, and must be programmed to be used in the Alternate interface mode or GPIO mode. Dedicated GPIO pins come up as input pins, and must be programmed in order to be used as output pins.

### 3.6 Debug Mode

The API supports communication between the ARM processor and the Debug port. Any messages seen on the Debug port come from the firmware. The Debug port is very useful in debugging the system and should always be connected.

### 3.7 JTAG ID Register

This section provides a description and listing of the JTAG ID Register.

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Version | | | | Manufacturers Part Number[15:4] | | | | | | | | | | | |
| **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| Part Number[3:0] | | | | Manufacturers ID | | | | | | | | | | | 1 |

Reserved fields should be ignored (masked) when read, and only 0's should be written to them.

| | |
|---|---|
| Version | 4-bit Version code of the device. Currently set to 0x0. |
| Part Number | 16-bit Manufacturers Part Number, assigned by Mobilygen. Currently set to 0x0300. |
| Manufacturers ID | 11-bit Manufacturers identity code (Mobilygen specific), assigned by JTAG. This is set to 0x2EB. |
| 1 | This bit is always set to 1. |

# 4.0 Device Operating Conditions

## 4.1 Absolute Maximum Ratings

Table 4-1 gives the absolute maximum ratings. Exposure to stresses beyond those listed in this table can result in device unreliability, permanent damage, or both.

**Table 4-1 Absolute Maximum Ratings**

| Parameter | Value | Units | Notes |
|---|---|---|---|
| CORE_VDD | 1.5 | V | — |
| DDR_VDD | 2.5 | V | — |
| VID01_VDD | 4.5 | V | — |
| VID23_VDD | 4.5 | V | — |
| AUD_VDD | 4.5 | V | — |
| HOST_VDD | 4.5 | V | — |
| USB_VDD | 4.5 | V | — |
| ETH_VDD | 4.5 | V | — |
| Maximum Input Voltage, DDR | 2.1 | V | DDR_VDD + 300 mV |
| Maximum Input Voltage, Other I/O | VDD_VREF + 700 mV | V | Referenced to associated IO VDD |
| Storage Temperature Range | -40 to 150 | °C | — |

## 4.2 Recommended Operation Conditions

Table 4-2 specifies the operating conditions.

**Table 4-2 Operating Conditions**

| Parameter | Minimum | Typical | Maximum | Units | Notes |
|---|---|---|---|---|---|
| CORE_VDD<br>PLL_VDD | 0.9975 | 1.05 | 1.1025 | V | 1.05V ± 5% |
| VID01_VDD<br>VID23_VDD<br>AUD_VDD<br>HOST_VDD | 1.62<br>2.25<br>2.97 | 1.8<br>2.5<br>3.3 | 1.98<br>2.75<br>3.63 | V | Programmable Voltage<br>1.8 / 2.5 / 3.3V ±10% |
| USB_VDD | 2.97 | 3.3 | 3.63 | V | 3.3V ±10% |
| ETH_VDD | 3.13 | 3.3 | 3.46 | V | 3.3V ±5% |
| DDR_VDD | 1.7 | 1.8 | 1.9 | V | 1.8V ±0.1V |
| DDR_VREF | - | 0.60 x DDR_VDD | - | V | This should be tuned for every design.<br>Refer to DDR design guideline, "MG3500/MG2580 DDR2 User's Guide."<br>Use 1% resistors |
| Operating Temperature Range (case) | 0 | 50 | 90 | °C | — |

### 4.3 Essential Characteristics

Table 4-3 defines the DC characteristics for all of the interfaces except the SDRAM interface.

**Table 4-3 DC Characteristics**

| Symbol | Parameters | Test Conditions | VID01_VDD, VID23_VDD, AUD_VDD, HOST_VDD 3.3V ±10% | | VID01_VDD, VID23_VDD, AUD_VDD, HOST_VDD 2.5V ±10% [1] | | VID01_VDD, VID23_VDD, AUD_VDD, HOST_VDD 1.8 V ±10% [1] | | Units |
|---|---|---|---|---|---|---|---|---|---|
| | | | Min | Max | Min | Max | Min | Max | |
| $V_{IH}$ | Input High Level | $V_{DD}$ = Maximum | 2.00 | — | See Note 1 | | See Note 1 | | V |
| $V_{IL}$ | Input Low-Level Voltage | $V_{DD}$ = Minimum | — | 0.40 | | | | | V |
| $V_{OH}$ | Output High-Level Voltage | $V_{DD}$ = Minimum, $I_{OH}$ = –4 mA | 2.70 | — | | | | | V |
| $V_{OL}$ | Output Low-Level Voltage | $V_{DD}$ = Minimum, $I_{OL}$ = –4 mA | — | 0.42 | | | | | V |
| $I_{IH}$ | Input Leakage | $V_{DD}$ = Maximum, $V_{IN}$ = $V_{DD}$ | –5 | –5 | | | | | µA |
| $I_{IL}$ | Input Leakage | $V_{DD}$ = Maximum, $V_{IN}$ = 0V | –2.55 | –2.55 | | | | | µA |
| $C_{PIN}$ | Capacitance [2] | — | — | 5 | | | | | pF |

1. The I/O pads are optimized for 3.3 Volt operation. The outputs should scale proportionately for 2.5 and 1.8 Volt operation, but the actual values may vary, depending on the individual device.
2. Not 100% tested.

Table 4-3 defines the DC and AC characteristics for the DDR SDRAM interface.

**Table 4-4 DC Characteristics**

| Symbol | Parameters | Test Conditions | DDR_VDD 1.8 V ±10% | | Units |
|---|---|---|---|---|---|
| | | | Min | Max | |
| $V_{DCIH}$ | Input DC High Level | $V_{DD}$ = Maximum | DDR_VREF + 125 mV | DDR_VDD + 300 mV | V |
| $V_{DCIL}$ | Input DC Low-Level | $V_{DD}$ = Minimum | 0 | DDR_VREF – 125 mV | V |
| $V_{ACIH}$ | Input AC High Level | $V_{DD}$ = Maximum | DDR_VREF + 250 mV | DDR_VDD + 300 mV | V |
| $V_{ACIL}$ | Input AC Low-Level | $V_{DD}$ = Minimum | 0 | DDR_VREF – 250 mV | V |
| $V_{DCOH}$ | Output DC High Level | $V_{DD}$ = Maximum | 1.4 | DDR_VDD | V |
| $V_{DCOL}$ | Output DC Low-Level | $V_{DD}$ = Minimum | 0 | DDR_VREF – 250 mV | V |
| $V_{ACOH}$ | Output AC High Level | $V_{DD}$ = Maximum | 1.3 | DDR_VDD | V |
| $V_{ACOL}$ | Output AC Low-Level | $V_{DD}$ = Minimum | 0 | 0.5 | V |
| $C_{PIN}$ | Capacitance[1] | — | — | 5 | pF |

1. Not 100% tested.

### 4.4 Power Supply Currents for the Different Power Domains

The power supply input currents vary for each power domain. Table 4-5 shows the input current ranges for each of the domains.

**Table 4-5 Typical Power Supply Currents for the Different Power Domains**

| Domain | Conditions | Minimum | Typical | Maximum | Units |
|---|---|---|---|---|---|
| CORE | 1.0 Volt Supply Voltage | | | 1000 | mA |
| AUD | 3.3 Volt Supply Voltage | | | 18 | mA |
| ETH | 3.3 Volt Supply Voltage | | | 42 | mA |
| HOST | 3.3 Volt Supply Voltage | | | 42 | mA |
| DDR | 1.8 Volt Supply Voltage | | | 166 | mA |
| USB | 3.3 Volt Supply Voltage | | | 18 | mA |
| VID01 | 3.3 Volt Supply Voltage | | | 45 | mA |
| VID23 | 3.3 Volt Supply Voltage | | | 45 | mA |

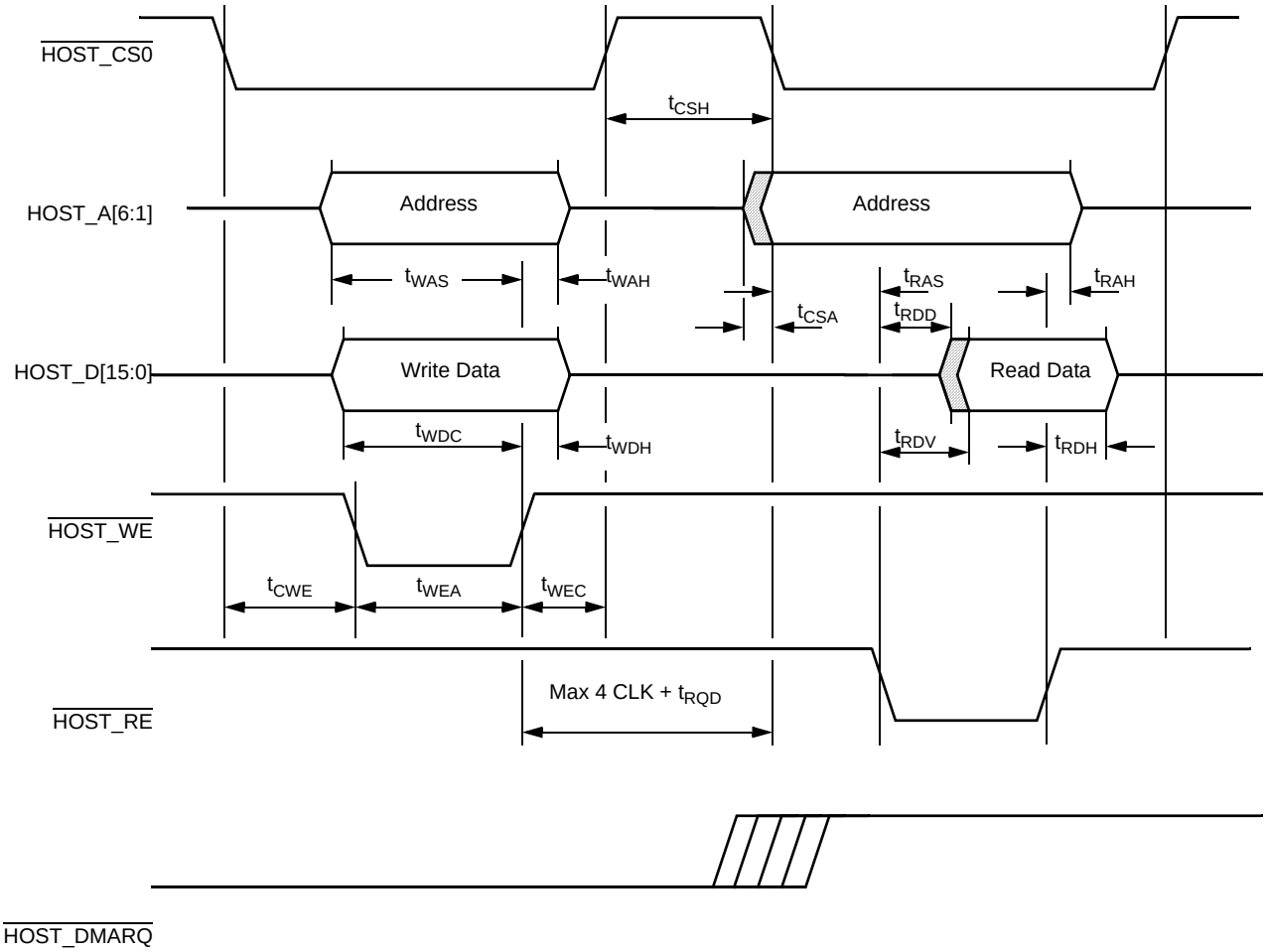
### 4.5 AC Timing

This section provides the AC timing for the MG3500 SoC's various interfaces. This section is divided into the following subsections:


- "MG3500 Parallel Slave Host Interface Timing" on page 74
- "Video Interface AC Timing" on page 78
- "Audio Interface AC Timing" on page 81
- "SDRAM Interface AC Timing" on page 88

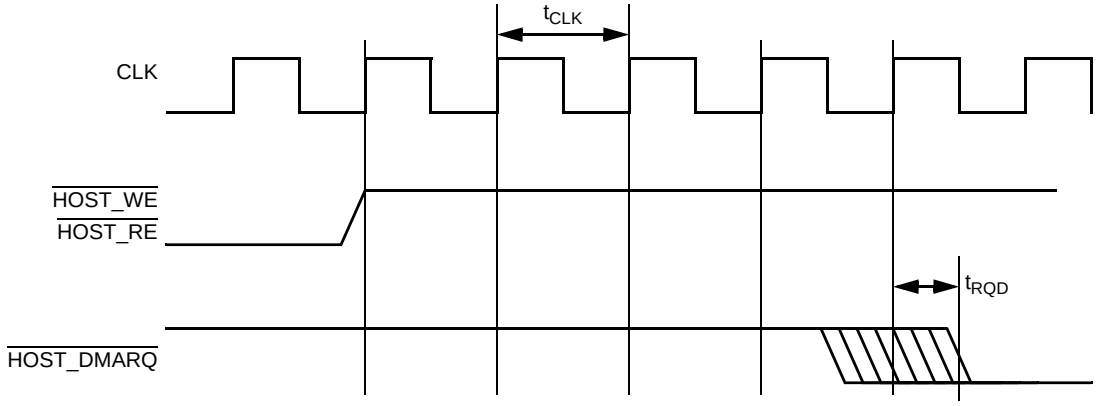
### 4.5.1 MG3500 Parallel Slave Host Interface Timing

Figure 4-1 shows the timing diagram for the MG3500 Parallel Slave Host Interface, Figure 4-2 shows the DMA Timing, Figure 4-3 shows the Wait timing, and Figure 4-4 shows the Interrupt Request timing. Table 4-6 lists the timing parameters for each of these diagrams.



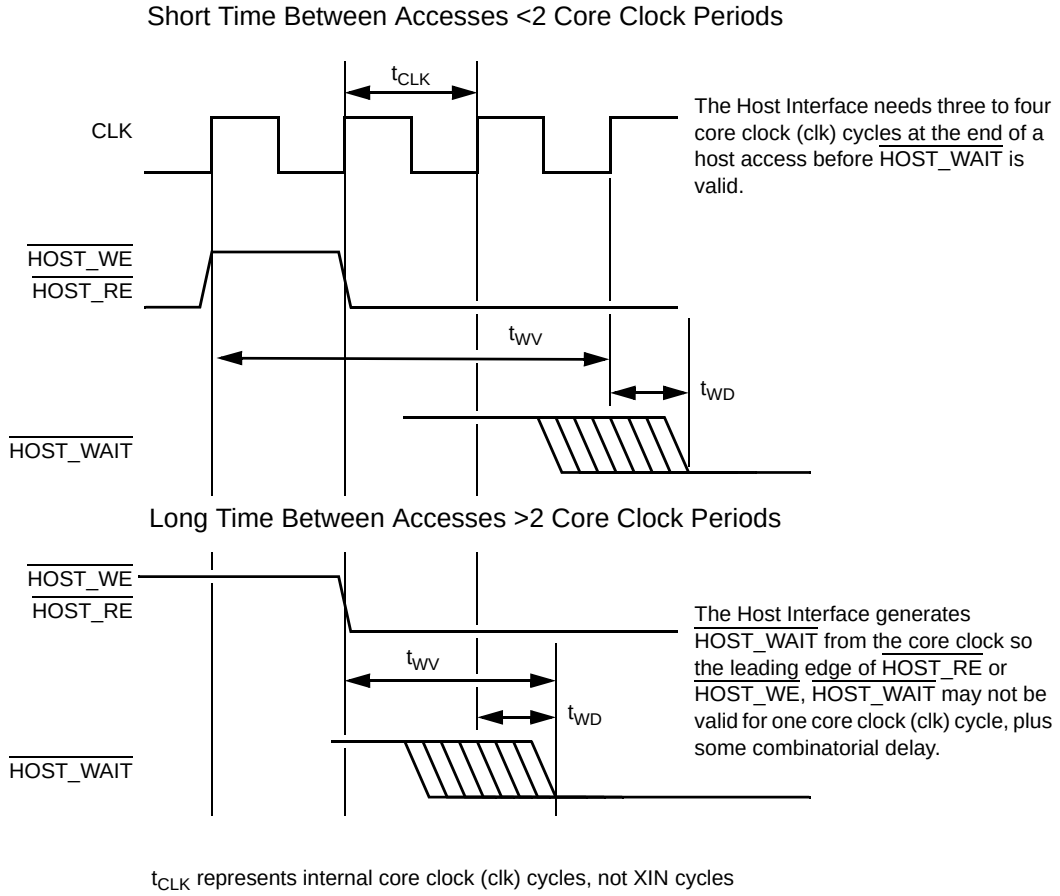

HOST_DMARQ takes three to four core clock (clk) periods before becoming valid

**Figure 4-1 Parallel Slave Host Slave Interface AC Timing Waveform**



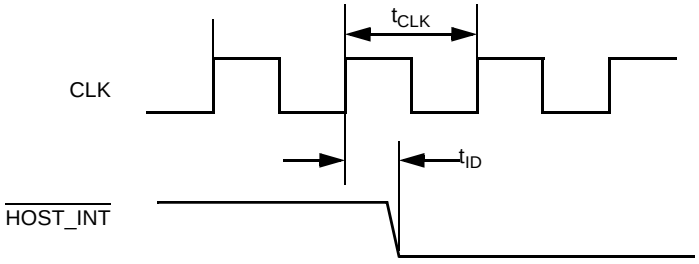

$t_{CLK}$ represents SDRAM clock cycles, not XIN cycles
$\overline{\text{HOST\_DMARQ}}$ takes three to four core clock (clk) periods before becoming valid

**Figure 4-2 $\overline{\text{HOST\_DMARQ}}$ Timing**




$t_{CLK}$ represents internal core clock (clk) cycles, not XIN cycles

**Figure 4-3 $\overline{\text{HOST\_WAIT}}$ Timing**




$t_{CLK}$ represents internal core clock (clk) cycles, not XIN cycles

**Figure 4-4 $\overline{HOST\_INT}$ Timing**

**Table 4-6 Slave Host Interface Timing**

| Signal | Parameter | Description | Min [1] | Max [1] | Units |
|---|---|---|---|---|---|
| Core Clock | $t_{CLK}$ | XIN x PLL Frequency | — | 180 | MHz |
| HOST_A[6:1] | $t_{WAS}$ | HOST_A[6:1] setup to trailing edge HOST_WE for write cycles | 20 | — | ns |
| | $t_{WAH}$ | HOST_A[6:1] hold from trailing edge HOST_WE for write cycles | 3 | — | ns |
| | $t_{RAS}$ | HOST_A[6:1] setup to leading edge HOST_RE for read cycles | 0 | — | ns |
| | $t_{RAH}$ | HOST_A[6:1] hold from trailing edge HOST_RE for read cycles | 0 | — | ns |
| | $t_{CSA}$ | HOST_A[6:1] setup to leading edge of HOST_CS | 0 | — | ns |
| HOST_D[15:0] | $t_{WDC}$ | HOST_D[15:0] setup to trailing edge HOST_WE for write cycles | 12 | — | ns |
| | $t_{WDH}$ | HOST_D[15:0] hold from trailing edge HOST_WE for write cycles | 3 | — | ns |
| | $t_{RDD}$ | HOST_D[15:0] driven from leading edge HOST_RE for read cycles | 0 | — | ns |
| | $t_{RDV}$ | HOST_D[15:0] valid from leading edge HOST_RE for read cycles | — | 17 | ns |
| | $t_{RDH}$ | HOST_D[15:0] hold from trailing edge HOST_RE for read cycles | 2 | 11 | ns |
| HOST_WE | $t_{CWE}$ | HOST_CS Active to HOST_WE Active | 0 | — | ns |
| | $t_{WEC}$ | HOST_WE Inactive to HOST_CS Inactive | 3 | — | ns |
| | $t_{WEA}$ | HOST_WE active time | 20 | — | ns |
| HOST_RE | $t_{CRE}$ | HOST_CS Active to HOST_RE Active | 0 | — | ns |
| | $t_{REC}$ | HOST_RE Inactive to HOST_CS Inactive | 0 | — | ns |
| | $t_{REA}$ | HOST_RE active time | 20 | — | ns |
| HOST_CS | $t_{CSH}$ | HOST_CS inactive time between accesses | 10 | — | ns |
| HOST_DMARQ | $t_{RQD}$ | HOST_DMARQ valid from internal clock | — | 8 | ns |
| HOST_IRQ | $T_{ID}$ | HOST_IRQ valid from internal clock | — | 8 | ns |
| HOST_WAIT | $t_{WD}$ | HOST_WAIT valid from internal clock | — | 8 | ns |
| | $t_{WV}$ | HOST_WAIT valid from HOST_RE/ HOST_WE | — | 12 | ns |

1. These numbers are based on simulation and will probably improve after characterization of the actual part.

### *4.5.2 Video Interface AC Timing*

Figure 4-5 and Table 4-7 show the AC timing parameters for the video interface.

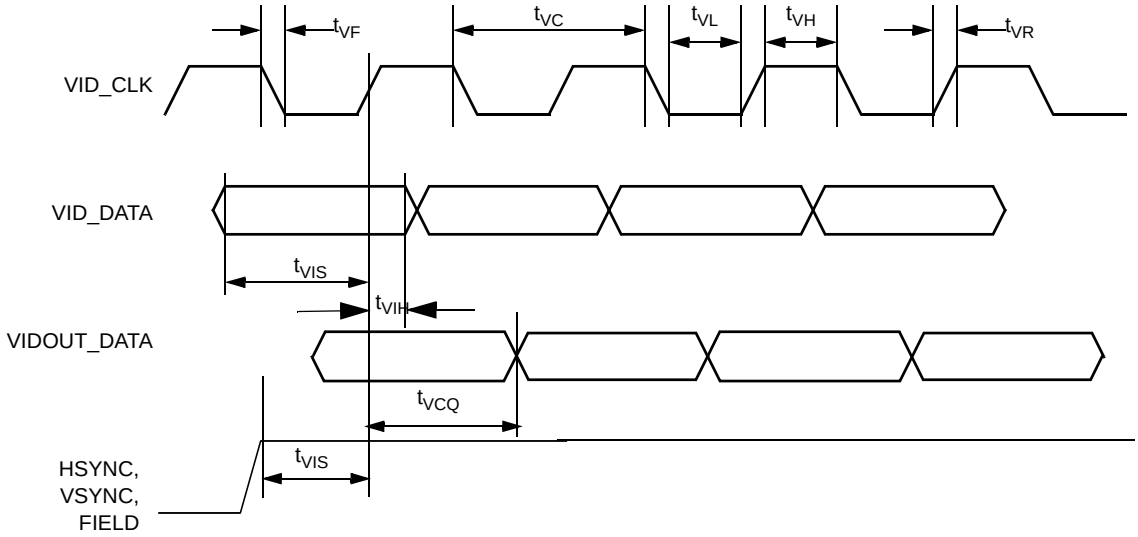



**Figure 4-5 Video Interface Timing Diagram**

**Table 4-7 Standard Definition Video Interface AC Timing Values**

| Signal | Parameter | Description | Timing Value (ns.) [1] | | | Units |
|---|---|---|---|---|---|---|
| | | | Min | Typ | Max | |
| VID[0:2]_CLK | $t_{VC}$ | VID_CLK Cycle Time (27 MHz) | — | 37.037 | | ns |
| | $t_{VH}$ | VID_CLK High Time | 16.67 | 18.5 | 20.37 | ns |
| | $t_{VL}$ | VID_CLK Low Time | $t_{VC} - t_{VH}$ | | | |
| | $t_{VR}$ | VID_CLK Slew (Rise Time) | Not Applicable | | | |
| | $t_{VF}$ | VID_CLK Slew (Fall Time) | Not Applicable | | | |
| VID[0:1]_DATA, HSYNC[0:1], VSYNC[0:1], FIELD[0:1] | $t_{VIS}$ | VID_DATA, HSYNC, VSYNC, FIELD Set-up Time to VID0_CLK or VID1_CLK | 3.5 | — | — | ns |
| | $t_{VIH}$ | VID_DATA, HSYNC, VSYNC, FIELD Hold Time from VID0_CLK or VID1_CLK | 2.8 | — | — | ns |
| VID[2:3]_DATA, HSYNC[2:3], VSYNC[2:3], FIELD[2:3] | $t_{VIS}$ | VID_DATA, HSYNC, VSYNC, FIELD Set-up Time to VID2_CLK | 3.5 | — | — | ns |
| | $t_{VIH}$ | VID_DATA, HSYNC, VSYNC, FIELD Hold Time from VID2_CLK | 2.8 | — | — | ns |
| VIDOUT[0:1] _DATA, HSYNC[0:1], VSYNC[0:1], FIELD[0:1 | $t_{VCQ}$ | VIDOUT_DATA, HSYNC, VSYNC, FIELD Delay from VID0_CLK or VID1_CLK | 4.0 | — | 13 | ns |
| VIDOUT[2:3] _DATA, HSYNC[2:3], VSYNC[2:3], FIELD[2:3] | $t_{VCQ}$ | VIDOUT_DATA, HSYNC, VSYNC, FIELD Delay from VID2_CLK | 4.0 | — | 13 | ns |

1. All timing values are in respect to rising edge on the VID_CLK pin. This clock can be supplied by either the external device or by the MG3500 SoC.

**Table 4-8 High Definition Video Interface AC Timing Values**

| Signal | Parameter | Description | Timing Value (ns.) [1] | | | Units |
|---|---|---|---|---|---|---|
| | | | Min | Typ | Max | |
| VID[0:2]_CLK | $t_{VC}$ | VID_CLK Cycle Time (74.25 MHz) | — | 13.468 | | ns |
| | $t_{VH}$ | VID_CLK High Time | 6.06 | 6.73 | 7.41 | ns |
| | $t_{VL}$ | VID_CLK Low Time | $t_{VC} - t_{VH}$ | | | |
| | $t_{VR}$ | VID_CLK Slew (Rise Time) | Not Applicable | | | |
| | $t_{VF}$ | VID_CLK Slew (Fall Time) | Not Applicable | | | |
| VID[0:1]_DATA[2], HSYNC[0:1], VSYNC[0:1], FIELD[0:1] | $t_{VIS}$ | VID_DATA, HSYNC, VSYNC, FIELD Set-up Time to VID0_CLK or VID1_CLK | 2.5 | — | — | ns |
| | $t_{VIH}$ | VID_DATA, HSYNC, VSYNC, FIELD Hold Time from VID0_CLK or VID1_CLK | 2.8 | — | — | ns |
| VID[2:3]_DATA[2], HSYNC[2:3], VSYNC[2:3], FIELD[2:3] | $t_{VIS}$ | VID_DATA, HSYNC, VSYNC, FIELD Set-up Time to VID2_CLK | 2.5 | — | — | ns |
| | $t_{VIH}$ | VID_DATA, HSYNC, VSYNC, FIELD Hold Time from VID2_CLK | 2.8 | — | — | ns |
| VIDOUT[0:1]_DATA, HSYNC[0:1], VSYNC[0:1], FIELD[0:1] | $t_{VCQ}$ | VIDOUT_DATA, HSYNC, VSYNC, FIELD Delay from VID0_CLK or VID1_CLK | 4.0 | — | 11.3 | ns |
| VIDOUT[2:3]_DATA, HSYNC[2:3], VSYNC[2:3], FIELD[2:3] | $t_{VCQ}$ | VIDOUT_DATA, HSYNC, VSYNC, FIELD Delay from VID2_CLK | 4.0 | — | 11.3 | ns |

1. All timing values are in respect to rising edge on the VID_CLK pin. This clock should be supplied either by the external device or by the the MG3500 SoC.
2. The external device should drive the data on the falling edge of VID_CLK to satisfy the input hold requirements.

**Table 4-9 High-Speed Video Interface AC Timing Values**

| Signal | Parameter | Description | Timing Value (ns.) [1] | | | Units |
|---|---|---|---|---|---|---|
| | | | Min | Typ | Max | |
| VID[0:2]_CLK | $t_{VC}$ | VID_CLK Cycle Time (125 MHz) | — | 8.0 | | ns |
| | $t_{VH}$ | VID_CLK High Time | 3.6 | 4 | 4.4 | ns |
| | $t_{VL}$ | VID_CLK Low Time | $t_{VC} - t_{VH}$ | | | |
| | $t_{VR}$ | VID_CLK Slew (Rise Time) | Not Applicable | | | |
| | $t_{VF}$ | VID_CLK Slew (Fall Time) | Not Applicable | | | |
| VID[0:1]_DATA[2], HSYNC[0:1], VSYNC[0:1], FIELD[0:1] | $t_{VIS}$ | VID_DATA, HSYNC, VSYNC, FIELD Set-up Time to VID0_CLK or VID1_CLK | 1.8 | — | — | ns |
| | $t_{VIH}$ | VID_DATA, HSYNC, VSYNC, FIELD Hold Time from VID0_CLK or VID1_CLK | 2.8 | — | — | ns |
| VID[2:3]_DATA[2], HSYNC[2:3], VSYNC[2:3], FIELD[2:3] | $t_{VIS}$ | VID_DATA, HSYNC, VSYNC, FIELD Set-up Time to VID2_CLK | 1.8 | — | — | ns |
| | $t_{VIH}$ | VID_DATA, HSYNC, VSYNC, FIELD Hold Time from VID2_CLK | 2.8 | — | — | ns |
| VIDOUT[0:1]_DATA HSYNC[0:1], VSYNC[0:1], FIELD[0:1] | $t_{VCQ}$ | VIDOUT_DATA, HSYNC, VSYNC, FIELD Delay from VID0_CLK or VID1_CLK | 1.8 | — | 6 | ns |
| VIDOUT[2:3]_DATA HSYNC[2:3], VSYNC[2:3], FIELD[2:3] | $t_{VCQ}$ | VIDOUT_DATA, HSYNC, VSYNC, FIELD Delay from VID2_CLK | 1.8 | — | 6 | ns |

1. All timing values are in respect to rising edge on the VID_CLK pin. This clock can be supplied by the MG3500 SoC.
2. The external device should drive the data on the falling edge of VID_CLK to satisfy the input hold requirements.
3. High-speed video interface is used when the video inputs are multiplexed.

### *4.5.3 Audio Interface AC Timing*

This section gives the AC timing parameters for the audio interface. Figure 4-6 shows the relationships between the three audio clocks. Figure 4-8 shows the left-justified audio timing waveforms. Table 4-10 lists the AC timing for Audio Operations.

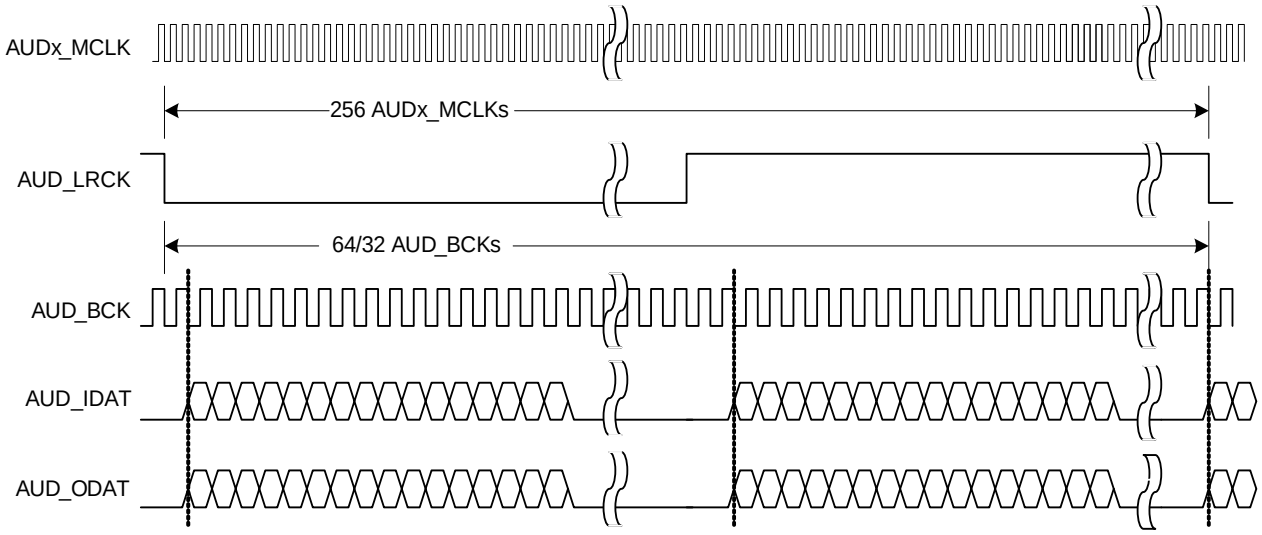



**Figure 4-6 Standard Audio Timing Diagram**

Figure 4-6 shows the $I^2S$ protocol, where the MSB bit is sent one AUD_BCK cycle after the AUD_LRCK signal has transitioned. In this mode, when LRCK is high the data is from the right channel, and when LRCK is low the data is from the left channel. This is opposite of left-justified audio.

Figure 4-7 shows sample waveforms for 16-, 20-, and 24-bit Left Justified audio. LRCK needs to be 64 BCKs in 20- and 24-bit modes. The MSB for each audio sample is aligned with LRCK's transition. The Audio Input Interface ignores the data bus after the LSB for each sample.

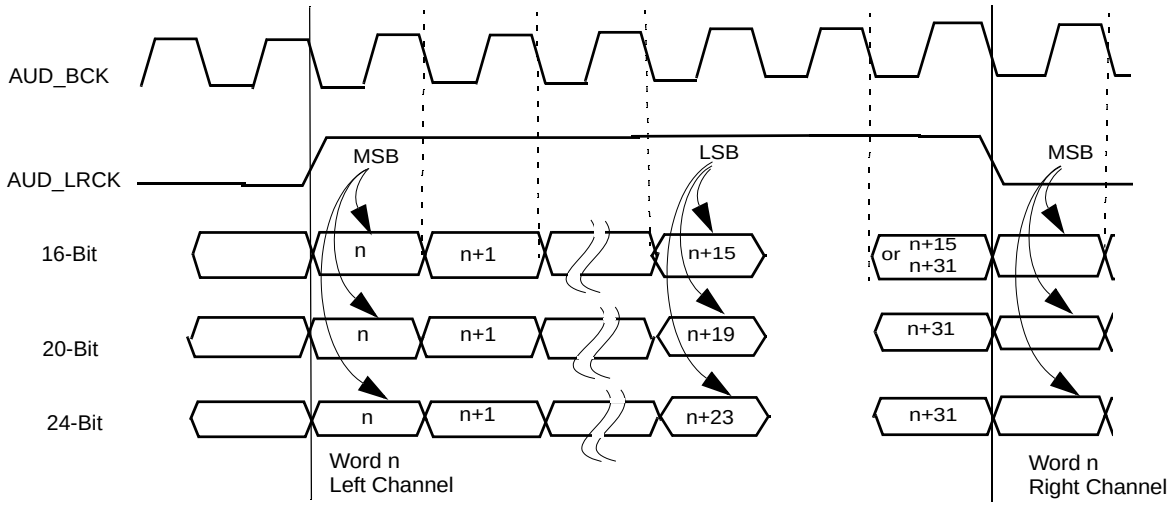



**Figure 4-7 16, 20, and 24-Bit Left Justified Audio Waveform**

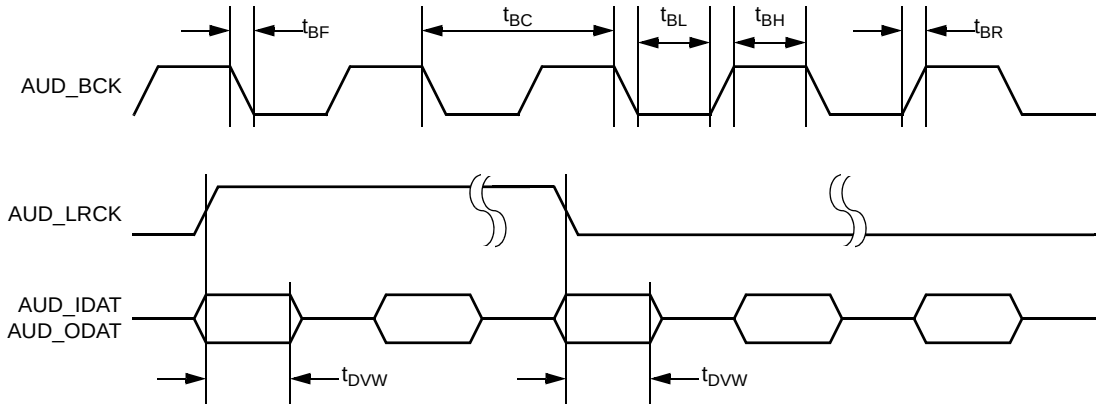



**Figure 4-8 Audio Interface Timing Diagram**

**Table 4-10 Audio Interface AC Timing Values**

| Signal | Parameter | Description | Timing Value (ns.) | | | Units |
|---|---|---|---|---|---|---|
| | | | Min | Typ | Max | |
| AUD_BCK | $t_{BC}$ | AUD_BCK Cycle Time (Fs = 48 kHz, 64 BCK/Sample) | — | 325.5 | — | ns |
| | $t_{BC}$ | AUD_BCK Cycle Time (Fs = 48 kHz, 32 BCK/Sample) | — | 651.04 | — | ns |
| | $t_{BC}$ | AUD_BCK Cycle Time (Fs = 32 kHz, 64 BCK/Sample) | — | 488.28 | — | ns |
| | $t_{BC}$ | AUD_BCK Cycle Time (Fs = 32 kHz, 32 BCK/Sample) | — | 976.56 | — | ns |
| | $t_{BH}$ | AUD_BCK High Time | $t_{BC}/2 * 0.8$ | $t_{BC}/2$ | $t_{BC}/2 * 1.2$ | ns |
| | $t_{BL}$ | AUD_BCK Low Time ($t_{BC}$ - $t_{BH}$) | $t_{BC} – t_{BH}$ | | | ns |
| | $t_{BR}$ | AUD_BCK Slew (Rise Time) | — | — | 3 | ns |
| | $t_{BF}$ | AUD_BCK Slew (Fall Time) | — | — | 3 | ns |
| AUD_LRCK AUD_ODAT AUD_IDAT | $t_{DVW}$ [1] | Data Valid Window for Slave Mode operation (Fs = 48 kHz or 32 kHz) | $t_{BC}/4 + 15$ | — | — | ns |
| | $t_{DVW}$ | Data Valid Window for Master Mode operation (Fs = 48 kHz or 32 kHz) | $t_{BC}/4 – 15$ | — | — | ns |

1. There is no restriction on the position of the Data Valid Window relative to BCK. The internal data sampling position is programmable and can be repositioned in $t_{BC}/4$ steps.

### *4.5.4 Ethernet Interface AC Timing*

This section shows the AC timing parameters for the Ethernet interface in each of the three operating modes:

- GMII
- MII
- RMII

Refer to the individual section for specific information.

## Gigabit Media Independent Interface (GMII) AC Timing

The Gigabit Media Independent Interface (GMII) defines speeds up to 1000 Mbit/s, implemented using an 8-bit data interface clocked at 125 MHz.

Figure 4-9 shows the AC timing parameters for the Ethernet interface in GMII Transmit mode, and Figure 4-10 shows the AC timing parameters for the Ethernet interface in GMII Receive mode.

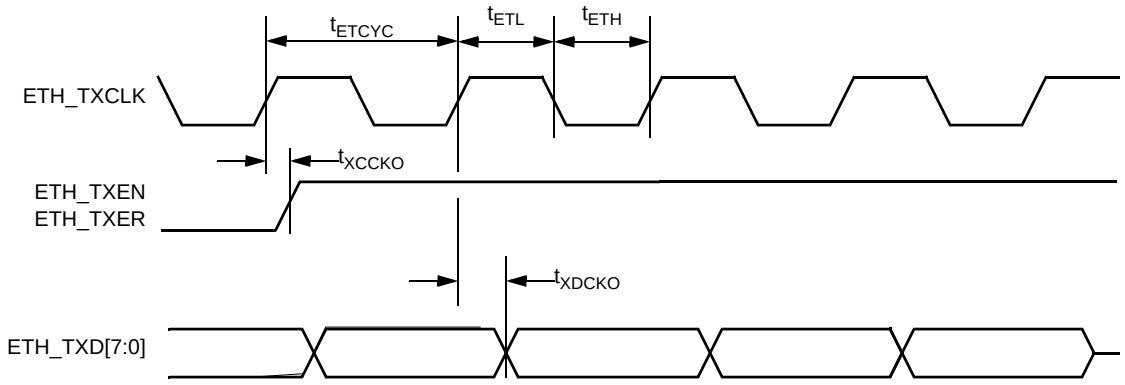



**Figure 4-9 Ethernet Interface GMII Transmit Timing Diagram**

**Table 4-11 Ethernet GMII Transmit Interface AC Timing Values**

| Signal | Parameter | Description | Timing Value (ns.) | | | Units |
|---|---|---|---|---|---|---|
| | | | Min | Typ | Max | |
| ETH_TXCLK | $t_{ECYC}$ | Ethernet GMII Transmit Clock Cycle Time | — | 8.0 | — | ns |
| | $t_{ETL}$ | Ethernet GMII Transmit Clock Low Time [1] | 3.2 | — | — | ns |
| | $t_{ETH}$ | Ethernet GMII Transmit Clock Low Time [1] | 3.2 | — | — | ns |
| ETH_TXEN ETH_TXER | $t_{XCCKO}$ | Ethernet GMII Transmit Control Signal Clock to Output Time | 2.0 | — | 5.9 | ns |
| ETH_TXD[7:0] | $t_{XDCKO}$ | Ethernet GMII Transmit Data Clock to Output Time | 2.0 | — | 5.9 | ns |

1. The minimum clock low and high times specify a worst case 60/40 duty cycle.

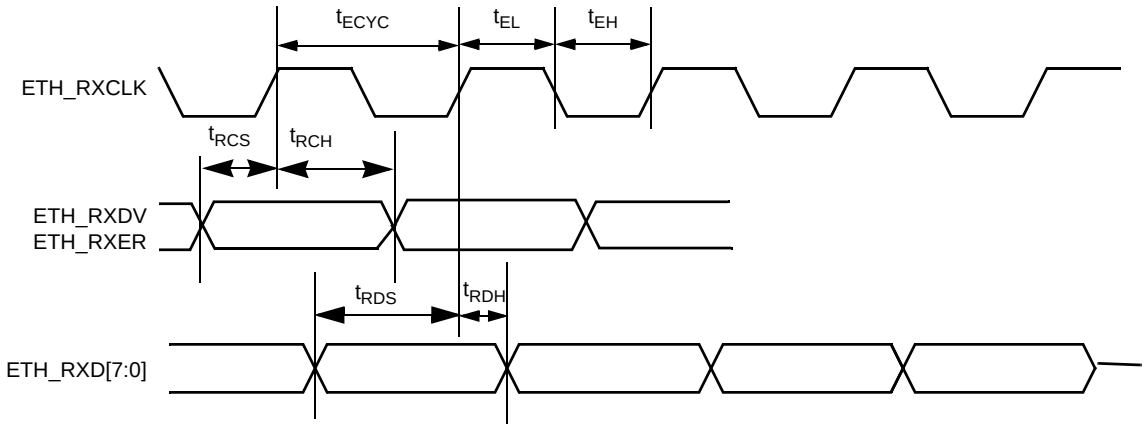



**Figure 4-10 Ethernet Interface GMII Receive Timing Diagram**

**Table 4-12 Ethernet GMII Receive Interface AC Timing Values**

| Signal | Parameter | Description | Timing Value (ns.) | | | Units |
|---|---|---|---|---|---|---|
| | | | Min | Typ | Max | |
| ETH_RXCLK | $t_{ECYC}$ | Ethernet GMII Receive Clock Cycle Time | — | 8.0 | — | ns |
| | $t_{ETL}$ | Ethernet GMII Receive Clock Low Time [1] | 3.2 | — | 4.8 | ns |
| | $t_{ETH}$ | Ethernet GMII Receive Clock High Time [1] | 3.2 | — | 4.8 | ns |
| ETH_RXDV ETH_RXER | $t_{RCS}$ | Ethernet GMII Receive Control Signal Setup Time | 2.2 | — | — | ns |
| | $t_{RCH}$ | Ethernet GMII Receive Control Signal Hold Time | 0 | — | — | ns |
| ETH_RXD[7:0] | $t_{RDS}$ | Ethernet GMII Receive Data Setup Time | 2.2 | — | — | ns |
| | $t_{RDH}$ | Ethernet GMII Receive Data Hold Time | 0 | — | — | ns |

1. The minimum clock low and high times specify a worst case 60/40 duty cycle.

### Media Independent Interface (MII) AC Timing

The Media Independent Interface (MII) defines speeds up to 100 Mbit/s, implemented using an 4-bit data interface clocked at either 25 MHz or 2.5 MHz.

Figure 4-11 shows the AC timing parameters for the Ethernet interface in MII Transmit mode, and Figure 4-12 shows the AC timing parameters for the Ethernet interface in MII Receive mode.

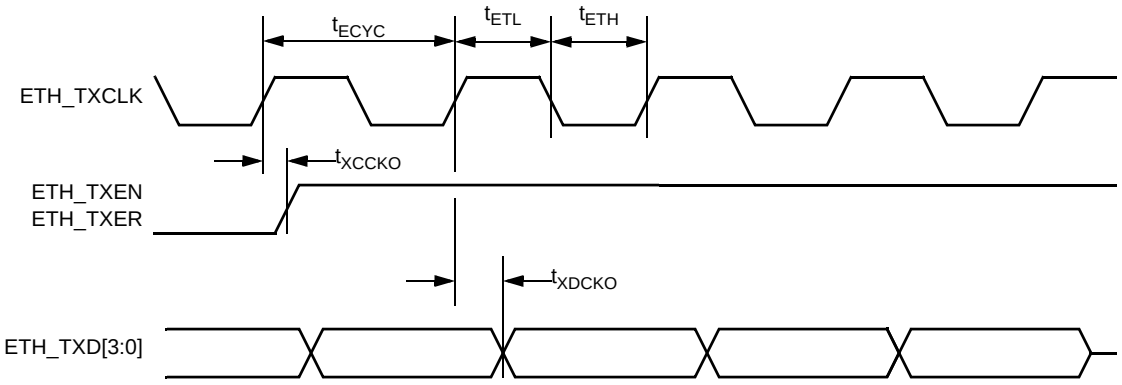



**Figure 4-11 Ethernet Interface MII Transmit Timing Diagram**

**Table 4-13 Ethernet MII Transmit Interface AC Timing Values**

| Signal | Parameter | Description | Timing Value (ns.) | | | Units |
|---|---|---|---|---|---|---|
| | | | Min | Typ | Max | |
| ETH_TXCLK | $t_{ECYC}$ | Ethernet MII Transmit Clock Cycle Time | — | 40.0 | — | ns |
| | $t_{ETL}$ | Ethernet MII Transmit Clock Low Time [1] | 16.0 | — | 24.0 | ns |
| | $t_{ETH}$ | Ethernet MII Transmit Clock High Time [1] | 16.0 | — | 24.0 | ns |
| ETH_TXEN ETH_TXER | $t_{XCCKO}$ | Ethernet MII Transmit Control Signal Clock to Output Time | 2.0 | — | 5.9 | ns |
| ETH_TXD[3:0] | $t_{XDCKO}$ | Ethernet MII Transmit Data Clock to Output Time | 2.0 | — | 5.9 | ns |

1. The minimum clock low and high times specify a worst case 60/40 duty cycle.

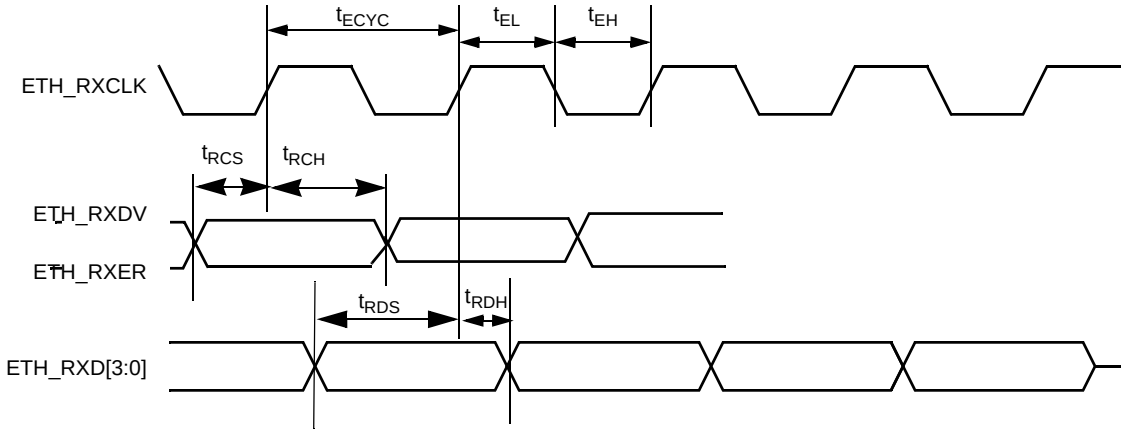



**Figure 4-12 Ethernet Interface MII Receive Timing Diagram**

**Table 4-14 Ethernet MII Receive Interface AC Timing Values**

| Signal | Parameter | Description | Timing Value (ns.) | | | Units |
|---|---|---|---|---|---|---|
| | | | Min | Typ | Max | |
| ETH_RXCLK | $t_{ECYC}$ | Ethernet MII Receive Clock Cycle Time | — | 40.0 | — | ns |
| | $t_{ETL}$ | Ethernet MII Receive Clock Low Time [1] | 16.0 | — | 24.0 | ns |
| | $t_{ETH}$ | Ethernet MII Receive Clock High Time [1] | 16.0 | — | 24.0 | ns |
| ETH_RXDV ETH_RXER | $t_{RCS}$ | Ethernet MII Receive Control Signal Setup Time | 2.2 | — | — | ns |
| | $t_{RCH}$ | Ethernet MII Receive Control Signal Hold Time | 0 | — | — | ns |
| ETH_RXD[3:0] | $t_{RDS}$ | Ethernet MII Receive Data Setup Time | 2.2 | — | — | ns |
| | $t_{RDH}$ | Ethernet MII Receive Data Hold Time | 0 | — | — | ns |

1. The minimum clock low and high times specify a worst case 60/40 duty cycle.

### Reduced Media Independent Interface (RMII) AC Timing

The Reduced Media Independent Interface (RMII) defines speeds up to 100 Mbit/s, implemented using a 2-bit data interface clocked at 50 MHz. For transmit data, the RMII interface only uses the ETH_TXEN and ETH_TXD[1:0] pins. For receive data, the RMII interface only uses the ETH_RXDV and ETH_RXD[1:0] pins.

Figure 4-13 shows the AC timing parameters for the Ethernet interface in RMII Transmit mode, and Figure 4-14 shows the AC timing parameters for the Ethernet interface in RMII Receive mode.

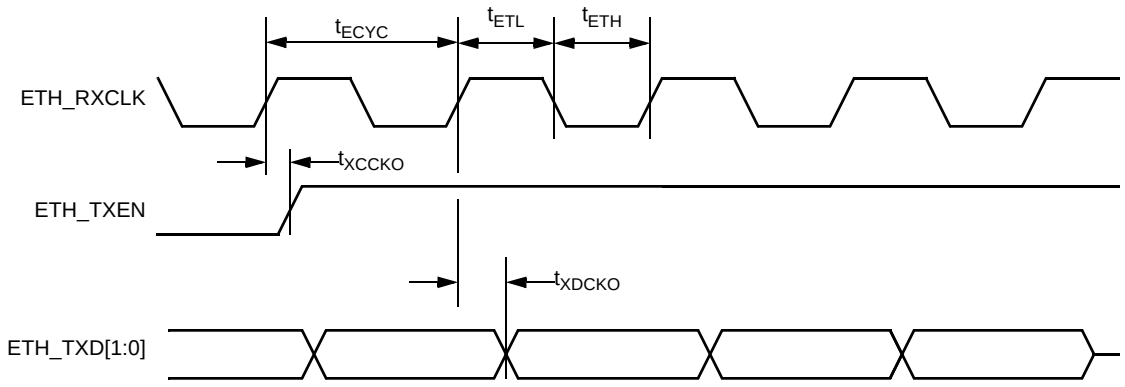



**Figure 4-13 Ethernet Interface RMII Transmit Timing Diagram**

**Table 4-15 Ethernet RMII Transmit Interface AC Timing Values**

| Signal | Parameter | Description | Timing Value (ns.) | | | Units |
|---|---|---|---|---|---|---|
| | | | Min | Typ | Max | |
| ETH_RXCLK | $t_{ECYC}$ | Ethernet RMII Clock Cycle Time | — | 20.0 | — | ns |
| | $t_{ETL}$ | Ethernet RMII Clock Low Time [1] | 8.0 | — | 12.0 | ns |
| | $t_{ETH}$ | Ethernet RMII Clock Low Time [1] | 8.0 | — | 12.0 | ns |
| ETH_TXEN | $t_{XCCKO}$ | Ethernet RMII Transmit Control Signal Clock to Output Time | 3.0 | — | 7.5 | ns |
| ETH_TXD[1:0] | $t_{XDCKO}$ | Ethernet RMII Transmit Data Clock to Output Time | 3.0 | — | 7.5 | ns |

1. The minimum clock low and high times specify a worst case 60/40 duty cycle.

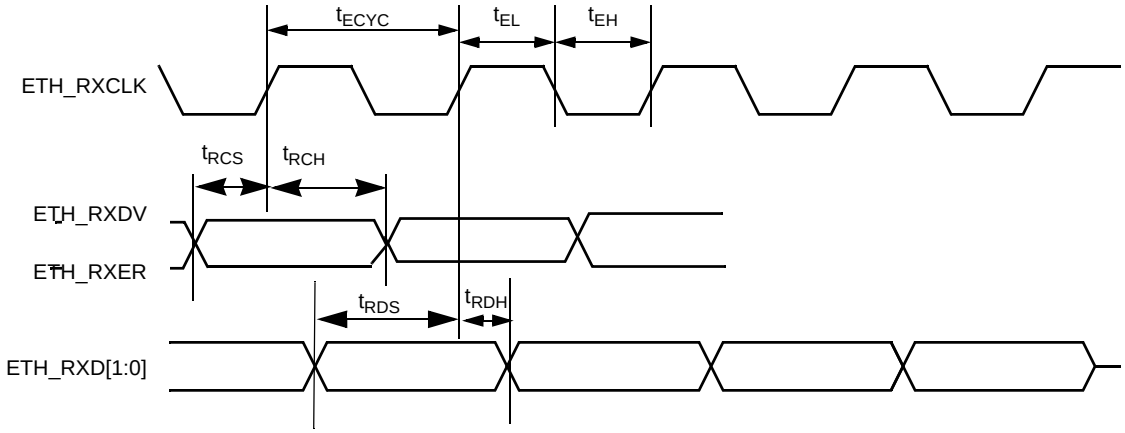



**Figure 4-14 Ethernet Interface RMII Receive Timing Diagram**

**Table 4-16 Ethernet RMII Receive Interface AC Timing Values**

| Signal | Parameter | Description | Timing Value (ns.) | | | Units |
|---|---|---|---|---|---|---|
| | | | Min | Typ | Max | |
| ETH_RXCLK | $t_{ECYC}$ | Ethernet RMII Clock Cycle Time | — | 20.0 | — | ns |
| | $t_{ETL}$ | Ethernet RMII Clock Low Time [1] | 8.0 | — | 12.0 | ns |
| | $t_{ETH}$ | Ethernet RMII Clock High Time [1] | 8.0 | — | 12.0 | ns |
| ETH_RXDV | $t_{RCS}$ | Ethernet RMII Receive Control Signal Setup Time | 2.7 | — | — | ns |
| | $t_{RCH}$ | Ethernet RMII Receive Control Signal Hold Time | 0.25 | — | — | ns |
| ETH_RXD[1:0] | $t_{RDS}$ | Ethernet RMII Receive Data Setup Time | 2.7 | — | — | ns |
| | $t_{RDH}$ | Ethernet RMII Receive Data Hold Time | 0.25 | — | — | ns |

1. The minimum clock low and high times specify a worst case 60/40 duty cycle.

### *4.5.5 SDRAM Interface AC Timing*

The MG3500 SoC adheres to the JEDEC definition of timing for SDRAMs. Refer to the appropriate specifications when designing the SDRAM Interface:

JEDEC Standard JESD789-2C DDR2 SDRAM Specification:
http://www.jedec.org/download/search/JESD79-2C.pdf

### *4.5.6 SPI/Bitstream Interface Timing*

This section shows the timing for the Serial Peripheral Interface and Bitstream Interface. The timing for the two interfaces is identical no matter which interface you are using. timing is shown for these four sets of conditions:

- BS_CLK driven from a source external to the MG3500 SoC and data mastered by a source external to the MG3500 SoC.
- BS_CLK driven from a source external to the MG3500 SoC and data mastered by the MG3500 SoC.
- BS_CLK mastered from the MG3500 SoC internal source and data mastered by a source external to the MG3500 SoC.
- BS_CLK mastered from the MG3500 SoC internal source and data mastered by the MG3500 SoC.

Refer to the specific sections that follow for the information that you need.

BS_CLK driven from a source external to the MG3500 SoC and Data mastered by source external to the MG3500 SoC.

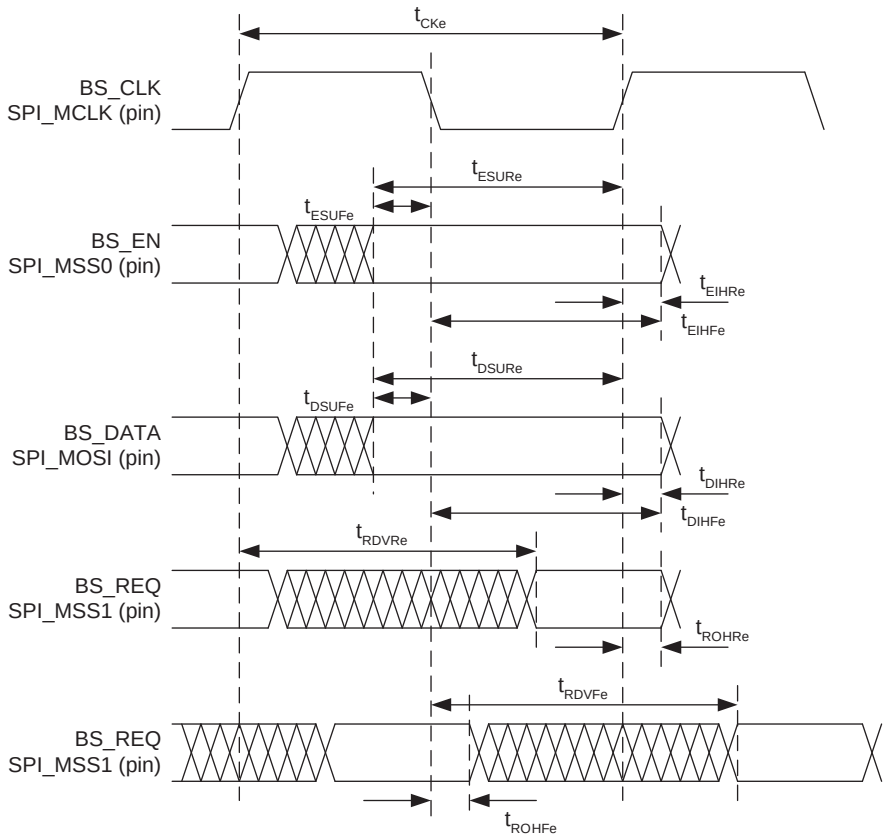



**Figure 4-15 Bitstream Timing with an External Clock and External Data Master**

**Table 4-17 Bitstream Timing AC Timing Values 1**

| Signal | Parameter | Description | Timing Value (ns.) | | |
|---|---|---|---|---|---|
| | | | Min | Typ | Max |
| BS_CLK<br>SPI_MCLK | $t_{CKe}$ | External Clock Period | 12.0 | | |
| BS_EN<br>SPI_MSS0 | $t_{ESUFe}$ | Bitstream Enable setup to falling edge of BS_CLK | 4.0 | | |
| | $t_{ESURe}$ | Bitstream Enable setup to rising edge of BS_CLK | 4.0 | | |
| | $t_{EIHFe}$ | Bitstream Enable input hold from falling edge of BS_CLK | 0.5 | | |
| | $t_{EIHRe}$ | Bitstream Enable input hold from rising edge of BS_CLK | 0.5 | | |
| BS_DATA<br>SPI_MOSI | $t_{DSUFe}$ | Bitstream Data setup to falling edge of BS_CLK | 3.5 | | |
| | $t_{DSURe}$ | Bitstream Data setup to rising edge of BS_CLK | 3.5 | | |
| | $t_{DIHFe}$ | Bitstream Data input hold from falling edge of BS_CLK | 0.5 | | |
| | $t_{DIHRe}$ | Bitstream Data input hold from rising edge of BS_CLK | 0.5 | | |
| BS_REQ<br>SPI_MSS1 | $t_{RDVFe}$ | Bitstream Request data valid from falling edge of BS_CLK | | | 12.5 |
| | $t_{RDVRe}$ | Bitstream Request data valid from rising edge of BS_CLK | | | 12.5 |
| | $t_{ROHFe}$ | Bitstream Request output hold from falling edge of BS_CLK | 2.0 | | |
| | $t_{ROHRe}$ | Bitstream Request output hold from rising edge of BS_CLK | 2.0 | | |

**BS_CLK driven from a source external to the MG3500 SoC and Data mastered by the MG3500 SoC.**

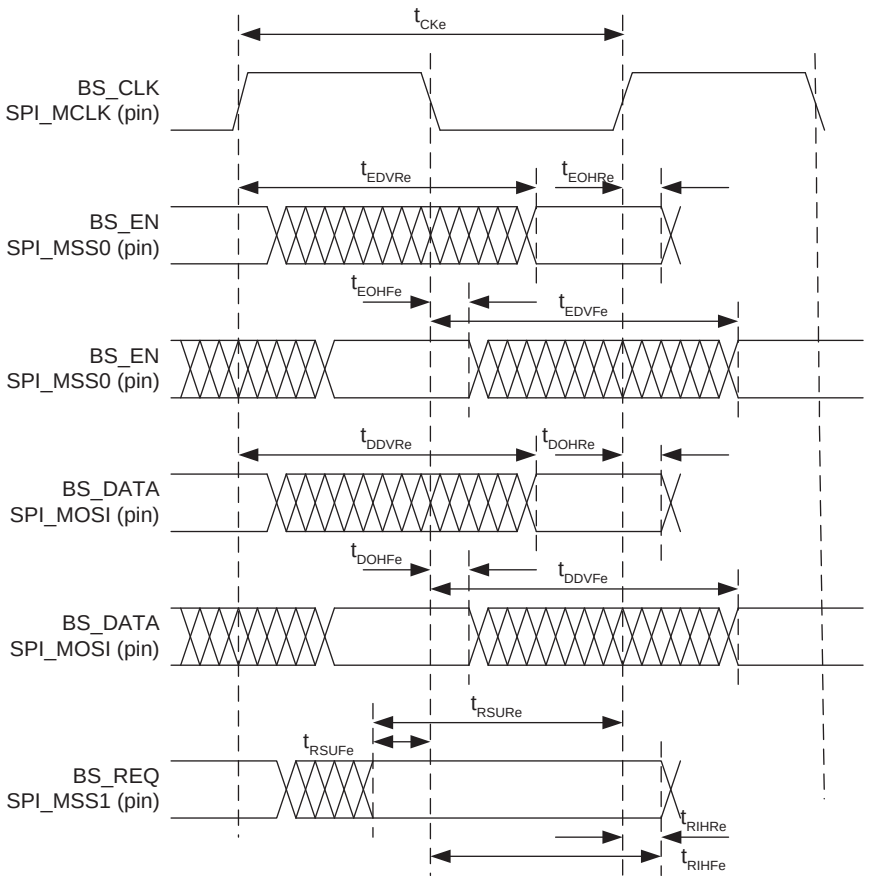



**Figure 4-16 Bitstream Timing with an External Clock and Internal Data Master**

**Table 4-18 Bitstream Timing AC Timing Values 2**

| Signal | Parameter | Description | Timing Value (ns.) | | |
|---|---|---|---|---|---|
| | | | Min | Typ | Max |
| BS_CLK<br>SPI_MCLK | $t_{CKe}$ | External Clock Period | 12.0 | | |
| BS_EN<br>SPI_MSS0 | $t_{EDVFe}$ | Bitstream Enable data valid from falling edge of BS_CLK | | | 12.5 |
| | $t_{EDVRe}$ | Bitstream Enable data valid from rising edge of BS_CLK | | | 12.5 |
| | $t_{EOHFe}$ | Bitstream Enable output hold from falling edge of BS_CLK | 2.0 | | |
| | $t_{EOHRe}$ | Bitstream Enable output hold from rising edge of BS_CLK | 2.0 | | |
| BS_DATA<br>SPI_MOSI | $t_{DDVFe}$ | Bitstream Data data valid from falling edge of BS_CLK | | | 12.0 |
| | $t_{DDVRe}$ | Bitstream Data data valid from rising edge of BS_CLK | | | 12.0 |
| | $t_{DOHFe}$ | Bitstream Data output hold from falling edge of BS_CLK | 2.0 | | |
| | $t_{DOHRe}$ | Bitstream Data output hold from rising edge of BS_CLK | 2.0 | | |
| BS_REQ<br>SPI_MSS1 | $t_{RSUFe}$ | Bitstream Request setup to falling edge of BS_CLK | 3.0 | | |
| | $t_{RSURe}$ | Bitstream Request setup to from rising edge of BS_CLK | 3.0 | | |
| | $t_{RIHFe}$ | Bitstream Request input hold from falling edge of BS_CLK | 0.5 | | |
| | $t_{RIHRe}$ | Bitstream Request input hold from rising edge of BS_CLK | 0.5 | | |

**BS_CLK mastered from the MG3500 SoC internal source and Data mastered by a source external to the MG3500 SoC.**

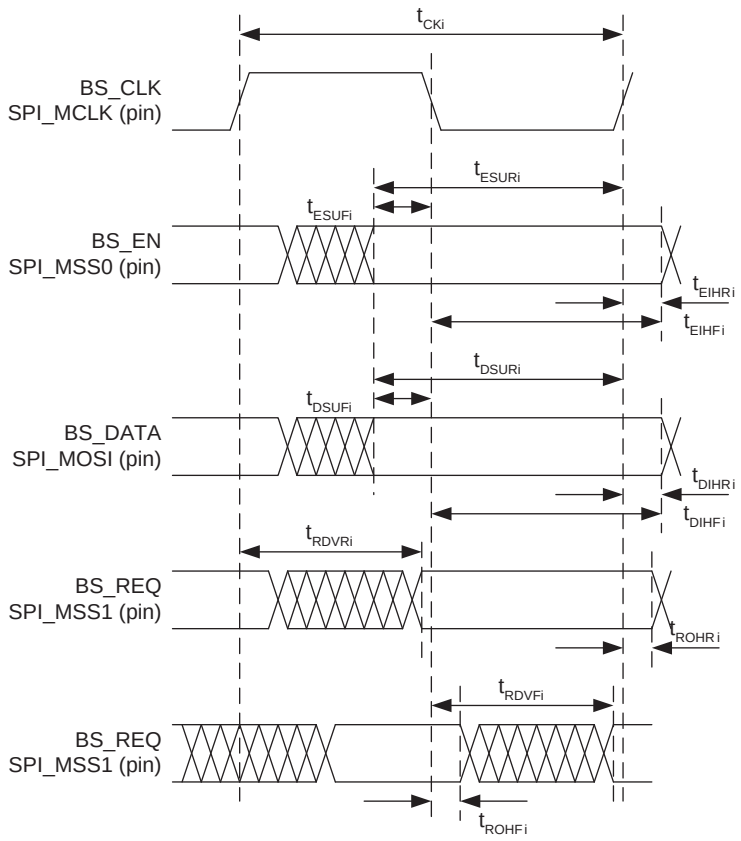



**Figure 4-17 Bitstream Timing with an Internal Clock and External Data Master**

**Table 4-19 Bitstream Timing AC Timing Values 3**

| Signal | Parameter | Description | Timing Value (ns.) | | |
|---|---|---|---|---|---|
| | | | Min | Typ | Max |
| BS_CLK<br>SPI_MCLK | $t_{CKi}$ | External Clock Period | 14.8 | | |
| BS_EN<br>SPI_MSS0 | $t_{ESUFi}$ | Bitstream Enable setup to falling edge of BS_CLK | 2.5 | | |
| | $t_{ESURi}$ | Bitstream Enable setup to rising edge of BS_CLK | 2.5 | | |
| | $t_{EIHFi}$ | Bitstream Enable input hold from falling edge of BS_CLK | 0.5 | | |
| | $t_{EIHRi}$ | Bitstream Enable input hold from rising edge of BS_CLK | 0.5 | | |
| BS_DATA<br>SPI_MOSI | $t_{DSUFi}$ | Bitstream Data setup to falling edge of BS_CLK | 2.0 | | |
| | $t_{DSURi}$ | Bitstream Data setup to rising edge of BS_CLK | 2.0 | | |
| | $t_{DIHFi}$ | Bitstream Data input hold from falling edge of BS_CLK | 0.5 | | |
| | $t_{DIHRi}$ | Bitstream Data input hold from rising edge of BS_CLK | 0.5 | | |
| BS_REQ<br>SPI_MSS1 | $t_{RDVFi}$ | Bitstream Request data valid from falling edge of BS_CLK | | | 5.0 |
| | $t_{RDVRi}$ | Bitstream Request data valid from rising edge of BS_CLK | | | 5.0 |
| | $t_{ROHFi}$ | Bitstream Request output hold from falling edge of BS_CLK | 2.0 | | |
| | $t_{ROHRi}$ | Bitstream Request output hold from rising edge of BS_CLK | 2.0 | | |

**BS_CLK mastered from the MG3500 SoC internal source and Data mastered by the MG3500 SoC.**

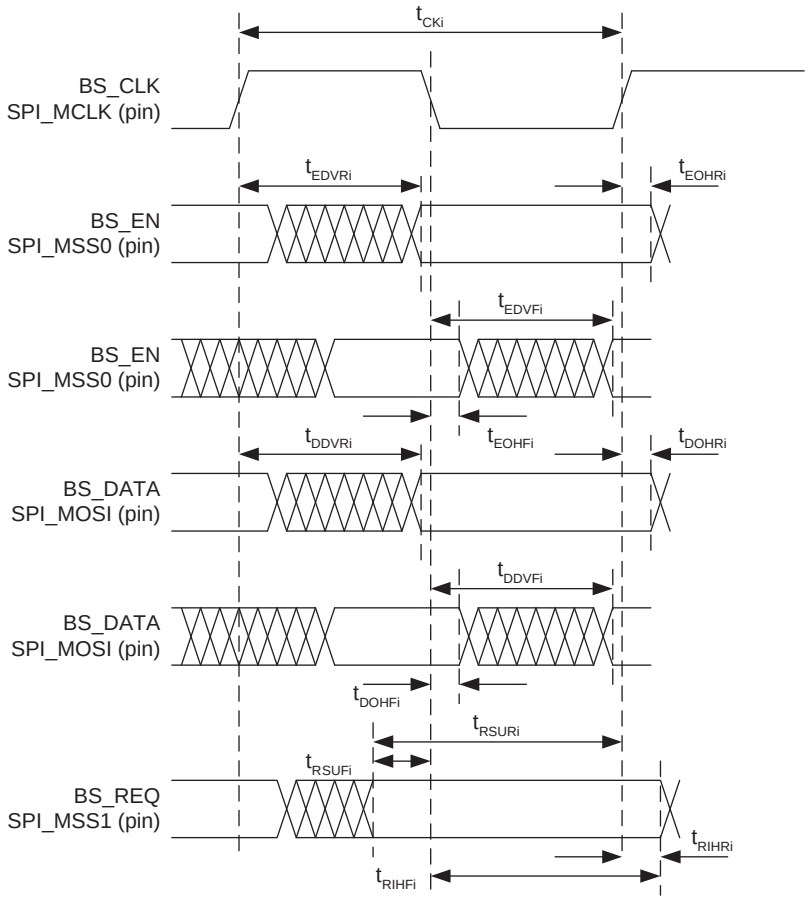



**Figure 4-18 Bitstream Timing with an Internal Clock and Internal Data Master**

**Table 4-20 Bitstream Timing AC Timing Values 4**

| Signal | Parameter | Description | Timing Value (ns.) | | |
|---|---|---|---|---|---|
| | | | Min | Typ | Max |
| BS_CLK<br>SPI_MCLK | $t_{CKi}$ | External Clock Period | 14.8 | | |
| BS_EN<br>SPI_MSS0 | $t_{EDVFi}$ | Bitstream Enable data valid from falling edge of BS_CLK | | | 5.0 |
| | $t_{EDVRi}$ | Bitstream Enable data valid from rising edge of BS_CLK | | | 5.0 |
| | $t_{EOHFi}$ | Bitstream Enable output hold from falling edge of BS_CLK | 2.0 | | |
| | $t_{EOHRi}$ | Bitstream Enable output hold from rising edge of BS_CLK | 2.0 | | |
| BS_DATA<br>SPI_MOSI | $t_{DDVFi}$ | Bitstream Data data valid from falling edge of BS_CLK | | | 4.5 |
| | $t_{DDVRi}$ | Bitstream Data data valid from rising edge of BS_CLK | | | 4.5 |
| | $t_{DOHFi}$ | Bitstream Data output hold from falling edge of BS_CLK | 2.0 | | |
| | $t_{DOHRi}$ | Bitstream Data output hold from rising edge of BS_CLK | 2.0 | | |
| BS_REQ<br>SPI_MSS1 | $t_{RSUFi}$ | Bitstream Request setup to falling edge of BS_CLK | 1.0 | | |
| | $t_{RSURi}$ | Bitstream Request setup to from rising edge of BS_CLK | 1.0 | | |
| | $t_{RIHFi}$ | Bitstream Request input hold from falling edge of BS_CLK | 0.5 | | |
| | $t_{RIHRi}$ | Bitstream Request input hold from rising edge of BS_CLK | 0.5 | | |

# 5.0 Block Level Operation

This section provides detailed block-level descriptions of each of the components, connection examples for each of the interfaces, and programming and register information as needed.

## 5.1 Detailed Block Diagram

Figure 5-1 shows a detailed block diagram of the MG3500 HD H.264 Codec SoC. Refer to it as you go through this section.

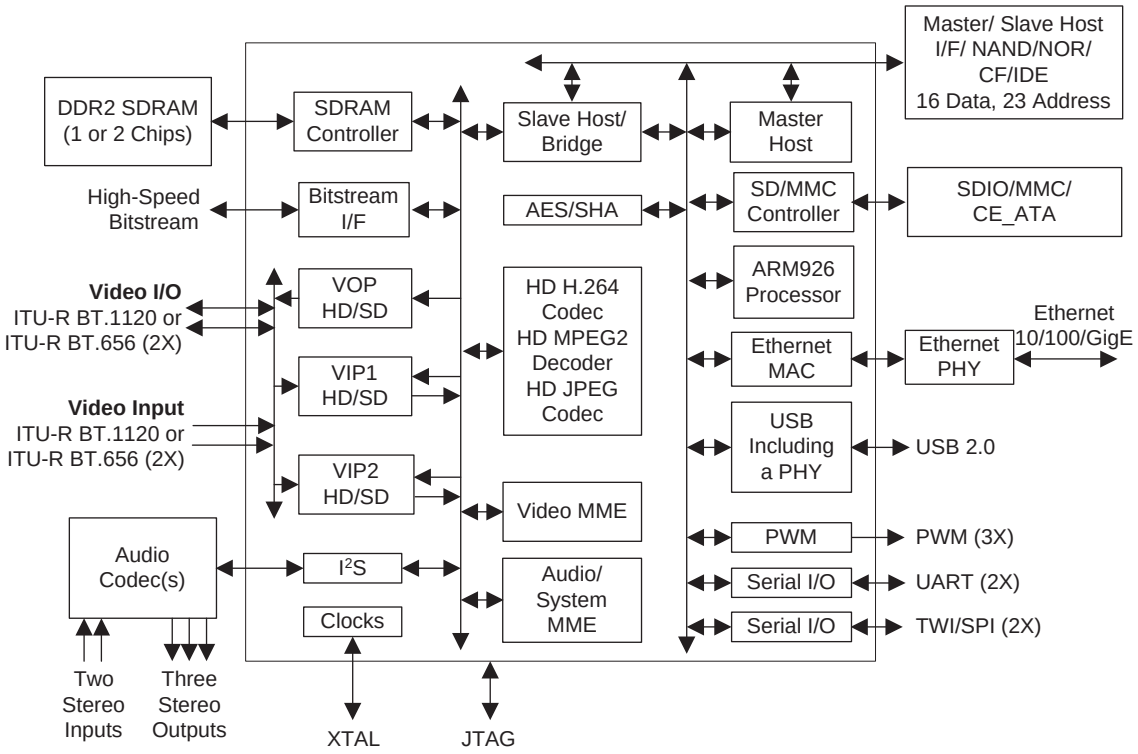



**Figure 5-1 Block Diagram of the MG3500 SoC**

## 5.2 Reset Logic

The Reset block within the MG3500 SoC is responsible for resetting the core logic as well as the SoC blocks that surround it.

The core reset signal consists of the power on reset signal from an external pin (RESETn), a watchdog reset, and a software controlled chip reset signal.

### *5.2.1 Power On Reset*

The power on reset signal comes in directly from the external RESETn pin and is asynchronous with respect to the clock. It is assumed that the clock is not running both at the time of the assertion and de-assertion of the power on reset signal.

### *5.2.2 WatchDog Reset*

The watchdog reset is asserted when the internal watchdog logic detects an internal error. The watchdog reset needs to be enabled before it can take effect. Resetting the watchdog timer will cause the watchdog reset to be de-asserted, so it is self-clearing.

### *5.2.3 Software Chip Reset*

The software chip reset is ORed with the watchdog reset. Resetting the software chip reset register cause the software chip reset signal to be de-asserted, so it provides a form of self-clearing mechanism.

All three of these resets get combined into a signal that resets the both the core and also the SoC blocks that surround it. In addition, there are reset registers that allow each of the blocks within the SoC logic to be reset independently. Since control of these reset signals is done using the software API, they are not discussed in this manual.

## 5.3 Clocks and PLLs

The MG3500 SoC internally creates over 20 clocks to minimize power consumption and maximize performance. All of the clocks are derived from a single 12 MHz crystal oscillator that is built into the USB Interface block. This oscillator can be used even when the USB interface is not used.

### *5.3.1 Clock and PLL inputs*

There are some cases where the MG3500 SoC requires a direct clock source, such as when the video must be synchronized with an another video source. Typical applications use a 27 MHz. clock. There are also some cases where the MG3500 SoC requires a direct clock source **and** the 12 MHz crystal input to maintain USB functionality. The CLK_IN pin is provided for these cases, and the selection is controlled by the configuration pin, CLK_SEL (see Figure 5-2).

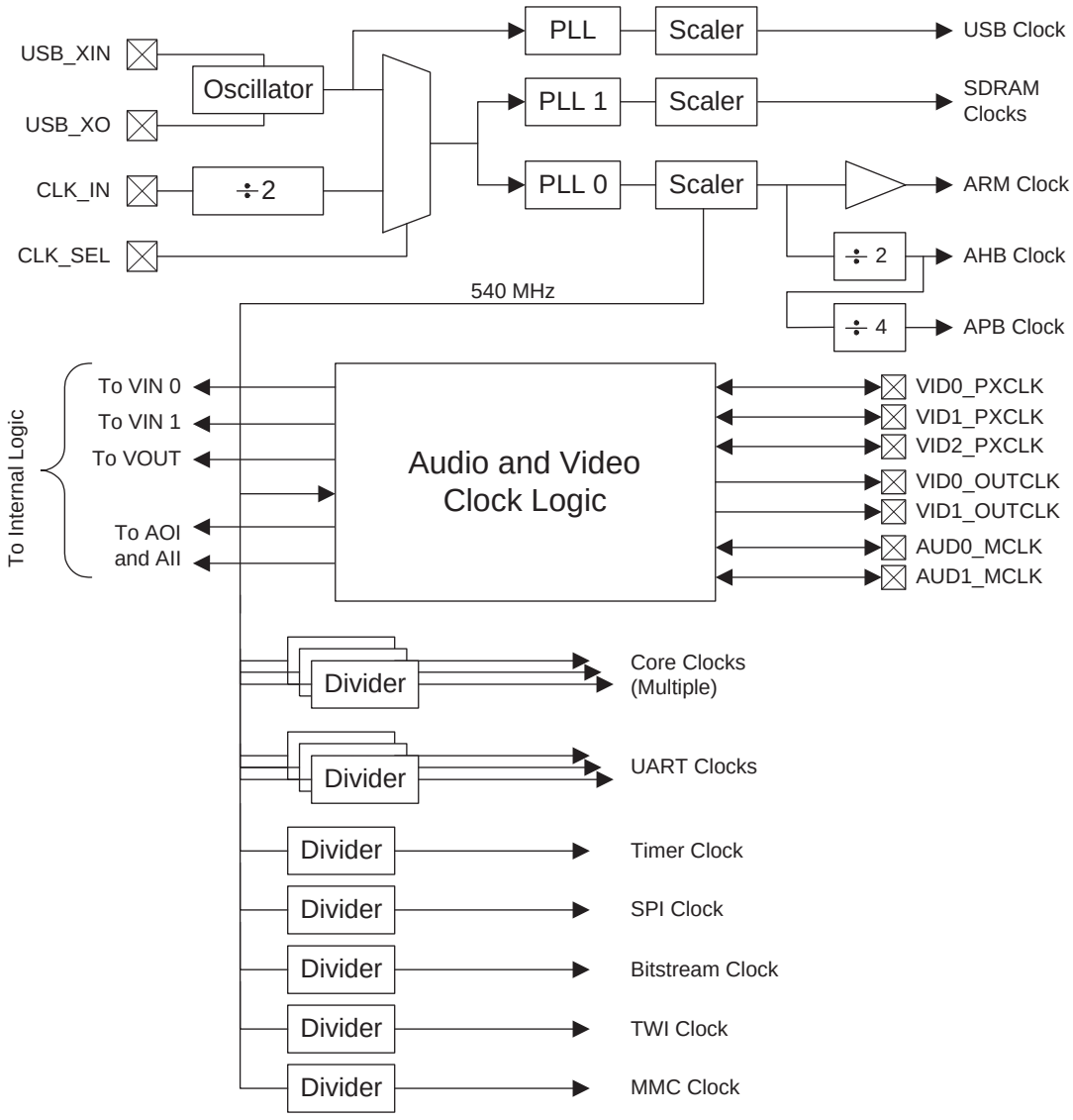



**Figure 5-2 Clocking Structure**

### 5.3.2 Phase Lock Loops

The MG3500 SoC has a total of five PLLs. One PLL is included as part of the USB PHY, and is used for USB PHY clocking and UTMI interfacing to the internal USB MAC. The remaining four PLLs are used to generate all the remaining required clock frequencies. PLL1 is used to generate the four-phase SDRAM clocking. PLL0 generates the codec core clocks, ARM processor, host bus clocks, and generates the input clocks for PLL2 and PLL3. Audio and video clocks can be generated from either PLL2 or PLL3, depending on the configuration of the multiplexers.

The remainder of the clocks are used for peripheral I/O circuitry, and are discussed in their individual sections. Refer to the *MG3500 SoC Programmers Guide* for information on programming the clocks.

### 5.3.3 Video and Audio Clocks

Figure 5-3 shows the circuitry used to generate the video input, video output, and audio clocks. Each is discussed in the sections that follow.

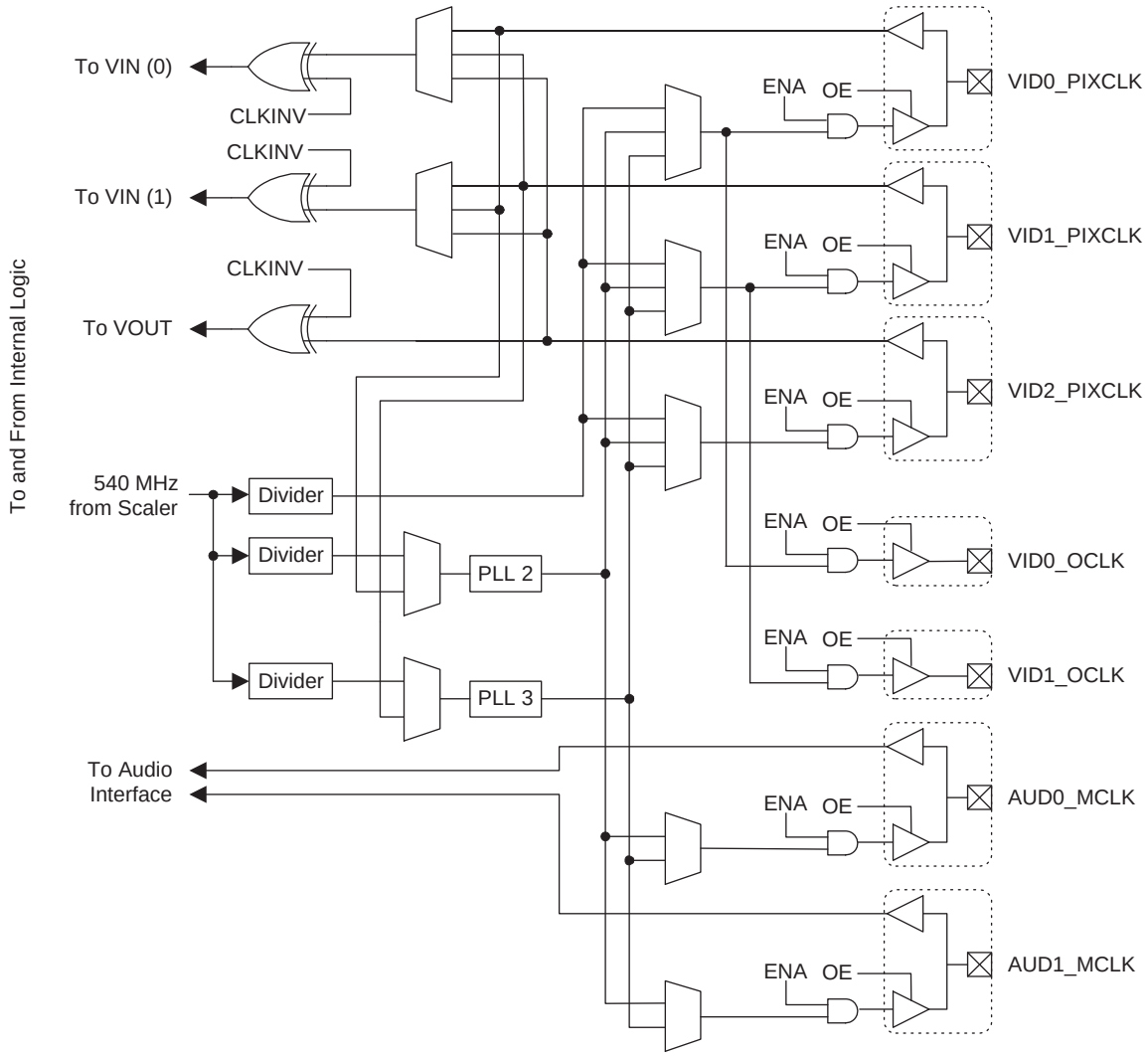



**Figure 5-3 Video and Audio Clocks**

## Video Input Clocks

There are two video input interfaces; VIN0 and VIN1. Each VIN has a V*_PIXCLK (V0_PIXCLK, V1_PIXCLK, V2_PIXCLK) signal, which is the clock to which the video data is timed. Alternately, an inverted version of V*_PIXCLK can be used to time the interface. The MG3500 SoC can configure the V*_PIXCLK pad to be either an input or an output.

In input mode, V*_PIXCLK is driven from an external source. In output mode, V*_PIXCLK is driven from an internal clock generator, and the clock is routed through the I/O cell to drive both the internal logic and external devices. In other words, the clock supplied to the video input circuitry is always the signal from the IO pad regardless of whether V*_PIXCLK is in input or output mode.

### Video Output Clocks

The MG3500 SoC also has the capability to generate the clock sources, V0_OUTCLK and V1_OUTCLK. This mode is especially useful in interfacing to sensors that require a clock input and send a clock output (typically a recovered, delayed version of the input clock) with the data.

The internal VOUT signal (see Figure 5-3) that goes to the internal video output circuitry can only be sourced by VID2_PXCLK. Alternately, an inverted version of VID2_PXCLK can be used to time the interface.

The V2_PIXCLK pad can be either an output or an input. In the output mode, V2_PIXCLK is driven from the internal clock generators. In the input mode, V2_PIXCLK is driven from an external source.

### Audio Clocks

The MG3500 SoC has two bidirectional audio clock pins; AUD0_MCLK and AUD1_MCLK. When these pins are configured as inputs, they supply the clock for the internal Audio Input Interface and Audio Output Interface blocks. Like in the input circuitry, in bi-directional mode, the audio clock is driven from internal clock generators, and the clock is routed through the I/O cell to drive the logic. PLL0 generates the main clock from which all other clocks in the system are derived.

## 5.4 Video Interfaces

### *5.4.1 Video Signal Groups*

The MG3500 SoC has four 8-bit video busses that are configurable in a variety of ways (see Figure 5-4). They are VID0_DATA, VID1_DATA, and VID2_DATA, and VID3_DATA. They can be used individually as separate standard video streams, or a pair of them can be combined to be used as a 16-bit data bus for HD video.

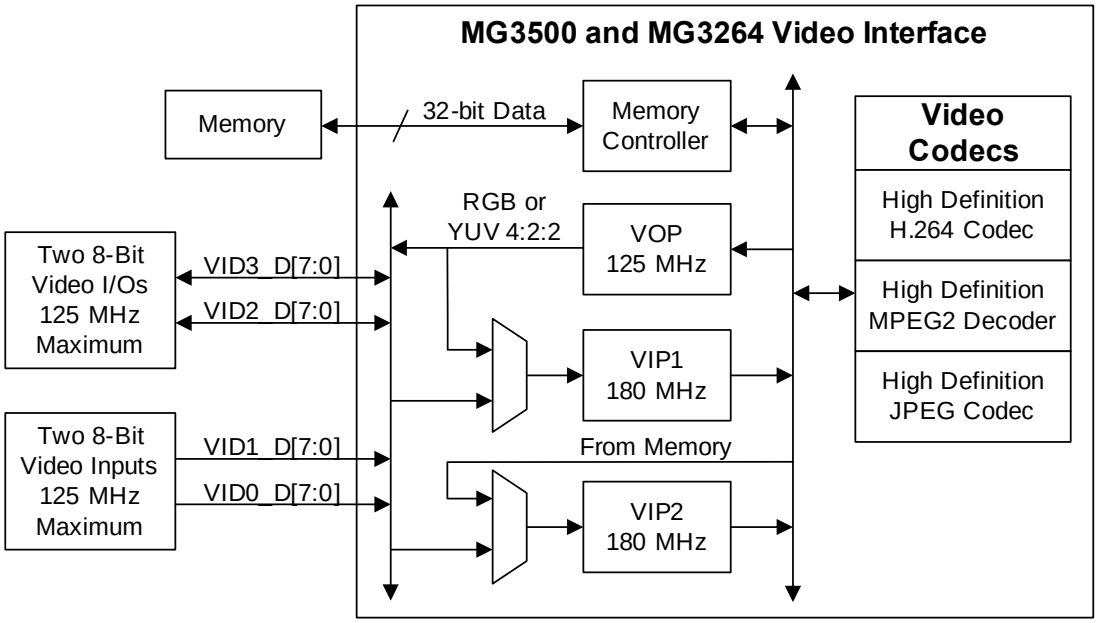



**Figure 5-4 MG3500 and MG3264 Video Path Overview**

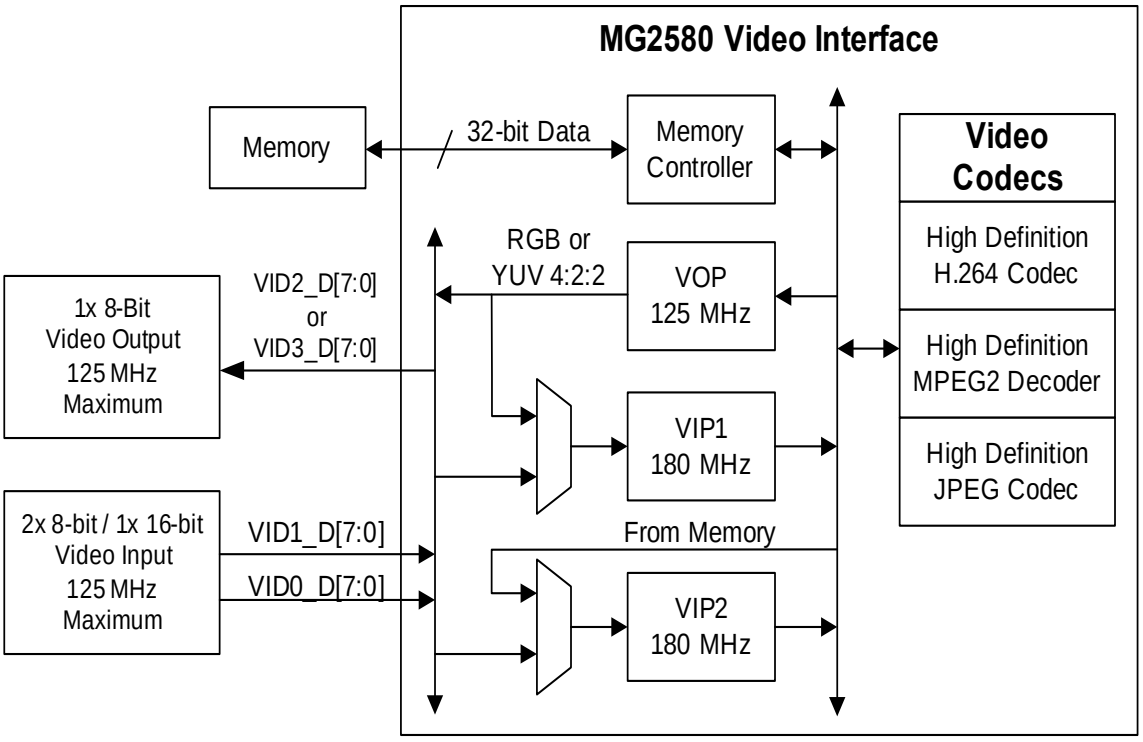



**Figure 5-5 MG2580 Video Path Overview**

VID0_D[7:0] and VID1_D[7:0] can only be configured as an input bus. VID2_D[7:0] and VID3_D[7:0] can be configured as either an input bus or an output bus. The data bus inputs to the two VIP's are connected to all of VID0_D[7:0], VID1_D[7:0], VID2_D[7:0], and VID3_D[7:0] through a multiplexer. The data bus output of the VOP is connected to VID2_D[7:0] and VID3_D[7:0].

The MG3500 SoC also has three sets of video control signals (VC0, VC1, VC2) consisting of HSYNC, VSYNC, and FIELD/VALID. The video control inputs to the two VIP's are connected to all of VC0, VC1, and VC2 through a multiplexer. The video control bus output of the VOP is connected to VC2 only.

There are only three video control signal groups because the MG3500 SoC can have a maximum of two inputs and one output. In general, any of the control groups can be assigned to any of the video ports, but there are these restrictions:

- Video control signal groups VC0 and VC1 are associated with the power plane for Video ports 0 and 1, and Video control signal groups VC2 is associated with the power plane for Video ports 2 and 3. If these power planes are connected to the same power source, then the statement above is true. If these power planes are connected to different power sources, then the Video control signal group must be assigned to a video port in the same power plane. See "Video IO Power Domains" on page 105.

- The VID2 control signals should not be used as the controls for a video input because they do not include the OUTCLK signal, which is sometimes used to synchronize an external video input device.

**Table 5-1 Video Control Signal Groups**

| Video Control Signal Group | Video Signals |
|---|---|
| VC0 | VID0_FIELD |
| | VID0_VSYNC |
| | VID0_HSYNC |
| VC1 | VID1_FIELD |
| | VID1_VSYNC |
| | VID1_HSYNC |
| VC2 | VID2_FIELD |
| | VID2_VSYNC |
| | VID2_HSYNC |

**Note:** Video Input Process (VIP) can only provide a standard live video streaming or perform non real-time scaling operation.

### *5.4.2 Video Clocks*

There are three Pixel Clock signals against which the data and control signals are timed. Two of these (VID0_PIXCLK, VID1_PIXCLK) are associated with video ports 0 and 1, and the other one (VID2_PIXCLK) is associated with video ports 2 and 3 (since only one video output path can be active at a time, only one Pixel Clock signal is needed). The pixel clocks can either drive the clock out or take the clock as an input. The two VIP's as well as the VOP can select any of the pixel clock inputs as the reference for the interface timing.

There are two output clock pins, VID0_OUTCLK and V1_OUTCLK, that drive out a video clock. These are simply clock sources and no video interface signals are timed against these pins. They are associated with video ports 0 and 1.

### *5.4.3 Video IO Power Domains*

The signals VID0_D[7:0], VID1_[7:0], Video control signal groups VC0 and VC1, VID0_PIXCLK, VID1_PIXCLK, VID0_OUTCLK, and VID1_OUTCLK are all in the VID01 Power Domain. The signals VID2_DATA, VID3_DATA, Video control signal group VC2, and VID2_PIXCLK are in the VID23 Power Domain, which is separate from the VID01 Power Domain.

The Video IO Power Domains can either be tied to the same voltage on the board or tied to different voltages on the board. If the Video IO Power Domains are tied to different voltages on the board, you must use the video control signal group associated with that power domain.

### *5.4.4 Video IO Scenarios*

This section shows some typical scenarios for video input and output. In the "two output" scenarios, there are NOT two independent concurrent video outputs processing paths. The same VOP output goes to both video output ports. Either port can selectively be turned off. One scenario where this is can occur is a camera that has both an LCD and a video output port (again, not concurrently). One port is

connected to the LCD and the other port is connected an NTSC/PAL encoder. When a port is active, the VOP is configured to output the correct video timing appropriate to the interface.

- One or two SD Inputs, one or two SD Outputs
- One or two SD Inputs, one HD Output
- One SD Input, one HD Input, one SD Output
- Two HD Inputs
- One HD Input, one HD Output
- One HD Input, one or two SD Outputs

Since VID2_D[7:0] and VID3_D[7:0] can be either inputs or outputs, there are also some scenarios in which the bus can be reconfigured in the system depending on the system use case that is currently active. For instance, VID2_D[7:0] and/or VID3_D[7:0] can be configured as inputs when in record or as outputs when in playback.

- Bi-directional SD input and output on VID2_D[7:0] or VID3_D[7:0]
- Bi-directional HD input and output on VID2_D[7:0] and VID3_D[7:0]

### *5.4.5 18-Bit LCD Interface*

The MG3500 HD H.264 Codec SoC provides for an 18-bit output for driving LCD displays. In this case, the extra two data signals are:

- VID23_D[16]
- VID23_D[17]

### *5.4.6 Video Connections*

This section shows how the video inputs and outputs should be connected in typical applications. Video ports 0 and 1 are used as a video inputs only, and their connections are shown in Table 5-2. Video ports 2 and 3 can be used as either a video input ports or as video output ports. Their connections are shown in Table 5-3 and Table 5-4.

## Video Input Connections, Ports 0 and 1

**Table 5-2 Video Input Connections, Ports 0 and 1**

| Pin | 8-bit Video Port 0 | 8-bit Video Port 1 | 16-bit Video Ports 0 and 1 |
|---|---|---|---|
| VID0_D0 | Video In Bit 0 | | Video In Bit 0 |
| VID0_D1 | Video In Bit 1 | | Video In Bit 1 |
| VID0_D2 | Video In Bit 2 | | Video In Bit 2 |
| VID0_D3 | Video In Bit 3 | | Video In Bit 3 |
| VID0_D4 | Video In Bit 4 | | Video In Bit 4 |
| VID0_D5 | Video In Bit 5 | | Video In Bit 5 |
| VID0_D6 | Video In Bit 6 | | Video In Bit 6 |
| VID0_D7 | Video In Bit 7 | | Video In Bit 7 |
| VID0_FIELD | Video In Field | | Video In Field |
| VID0_HSYNC | Video In Hsync | | Video In Hsync |
| VID0_VSYNC | Video In Vsync | | Video In Vsync |
| VID1_D0 | | Video In Bit 0 | Video In Bit 8 |
| VID1_D1 | | Video In Bit 1 | Video In Bit 9 |
| VID1_D2 | | Video In Bit 2 | Video In Bit 10 |
| VID1_D3 | | Video In Bit 3 | Video In Bit 11 |
| VID1_D4 | | Video In Bit 4 | Video In Bit 12 |
| VID1_D5 | | Video In Bit 5 | Video In Bit 13 |
| VID1_D6 | | Video In Bit 6 | Video In Bit 14 |
| VID1_D7 | | Video In Bit 7 | Video In Bit 15 |
| VID1_FIELD | | Video In Field | |
| VID1_HSYNC | | Video In Hsync | |
| VID1_VSYNC | | Video In Vsync | |

## Video Input Connections, Ports 2 and 3

**Table 5-3 Video Input Connections, Ports 2 and 3**

| Pin | 8-bit Video Port 2 | 8-bit Video Port 3 | 16-bit Video Ports 2 and 3 |
|---|---|---|---|
| VID2_D0 | Video In Bit 0 | | Video In Bit 0 |
| VID2_D1 | Video In Bit 1 | | Video In Bit 1 |
| VID2_D2 | Video In Bit 2 | | Video In Bit 2 |
| VID2_D3 | Video In Bit 3 | | Video In Bit 3 |
| VID2_D4 | Video In Bit 4 | | Video In Bit 4 |
| VID2_D5 | Video In Bit 5 | | Video In Bit 5 |
| VID2_D6 | Video In Bit 6 | | Video In Bit 6 |
| VID2_D7 | Video In Bit 7 | | Video In Bit 7 |
| VID2_FIELD | Video In Field | | Video In Field |
| VID2_HSYNC | Video In Hsync | | Video In Hsync |
| VID2_VSYNC | Video In Vsync | | Video In Vsync |
| VID3_D0 | | Video In Bit 0 | Video In Bit 8 |
| VID3_D1 | | Video In Bit 1 | Video In Bit 9 |
| VID3_D2 | | Video In Bit 2 | Video In Bit 10 |
| VID3_D3 | | Video In Bit 3 | Video In Bit 11 |
| VID3_D4 | | Video In Bit 4 | Video In Bit 12 |
| VID3_D5 | | Video In Bit 5 | Video In Bit 13 |
| VID3_D6 | | Video In Bit 6 | Video In Bit 14 |
| VID3_D7 | | Video In Bit 7 | Video In Bit 15 |

## Video Output Connections, Ports 2 and 3

**Table 5-4 Video Output Connections, Ports 2 and 3**

| Pin | 8-bit Video Port 2 | 8-bit Video Port 3 | 16-bit Video Ports 2 and 3 | 18-bit Video Ports 2 and 3 |
|---|---|---|---|---|
| VID2_D0 | Video Out Bit 0 | | Video Out Bit 0 | Video Out Bit 0 |
| VID2_D1 | Video Out Bit 1 | | Video Out Bit 1 | Video Out Bit 1 |
| VID2_D2 | Video Out Bit 2 | | Video Out Bit 2 | Video Out Bit 2 |
| VID2_D3 | Video Out Bit 3 | | Video Out Bit 3 | Video Out Bit 3 |
| VID2_D4 | Video Out Bit 4 | | Video Out Bit 4 | Video Out Bit 4 |
| VID2_D5 | Video Out Bit 5 | | Video Out Bit 5 | Video Out Bit 5 |
| VID2_D6 | Video Out Bit 6 | | Video Out Bit 6 | Video Out Bit 6 |
| VID2_D7 | Video Out Bit 7 | | Video Out Bit 7 | Video Out Bit 7 |
| VID2_FIELD | Video Out Field | | Video Out Field | Video Out Field |
| VID2_HSYNC | Video Out Hsync | | Video Out Hsync | Video Out Hsync |
| VID2_VSYNC | Video Out Vsync | | Video Out Vsync | Video Out Vsync |
| VID3_D0 | | Video Out Bit 0 | Video Out Bit 8 | Video Out Bit 8 |
| VID3_D1 | | Video Out Bit 1 | Video Out Bit 9 | Video Out Bit 9 |
| VID3_D2 | | Video Out Bit 2 | Video Out Bit 10 | Video Out Bit 10 |
| VID3_D3 | | Video Out Bit 3 | Video Out Bit 11 | Video Out Bit 11 |
| VID3_D4 | | Video Out Bit 4 | Video Out Bit 12 | Video Out Bit 12 |
| VID3_D5 | | Video Out Bit 5 | Video Out Bit 13 | Video Out Bit 13 |
| VID3_D6 | | Video Out Bit 6 | Video Out Bit 14 | Video Out Bit 14 |
| VID3_D7 | | Video Out Bit 7 | Video Out Bit 15 | Video Out Bit 15 |
| VID23_D16 | | | | Video Out Bit 16 |
| VID23_D17 | | | | Video Out Bit 17 |

### *5.4.7 VOUT Direct to VIN*

VOUT can be tied directly to VIN internally as shown in Figure 5-6.

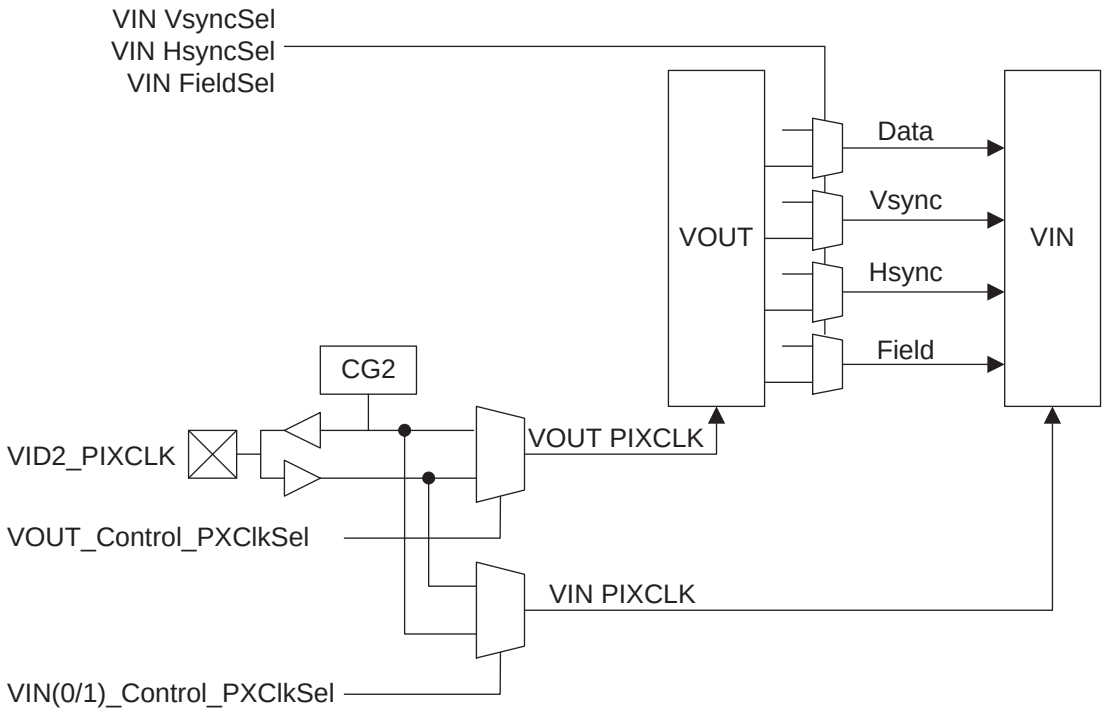



**Figure 5-6 VOUT Connected Directly to VIN**

## 5.5 Video Scaling

Video can be scaled on either the input or the output, and the scaling can occur in either the Video Input Processor (VIP) or the Video Output Processor (VOP)

### *5.5.1 Video Input Scaling*

The Video Input Processor can run with up to a 180 MHz clock, and data can be processed at any speed up to the VIP clock. Video Scaling includes:

- Cropping (can be used to reduce input prior to up-scaling)
- Luma processing
- Chroma Processing
- Horizontal Scaling
- Motion Adaptive De-Interlacing
- Vertical Scaling
- Motion Adaptive Temporal Filtering
- Pixel Processing extracts video statistics to guide compression

The data flows as shown in Figure 5-7.

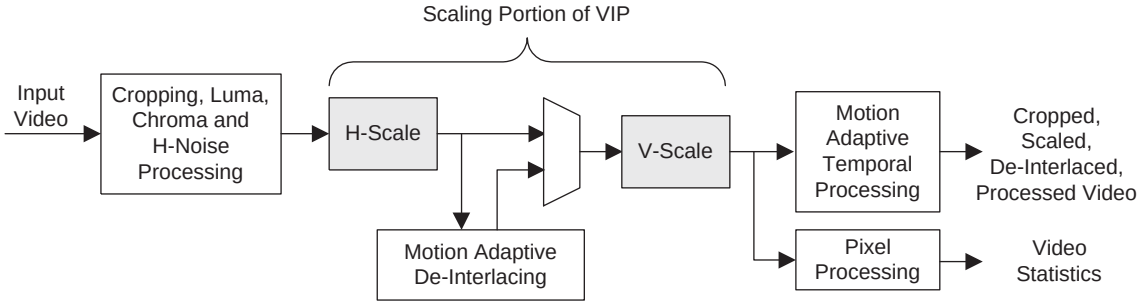



**Figure 5-7 Video Input Scaling Data Flow**

### *5.5.2 Horizontal Scaling*

Horizontal scaling uses an eight tap, eight phase programmable Finite Impulse Response (FIR) filter. Taps 0 and 7 are 7-bit 2's complement, taps 1 and 6 are 8-bit 2's complement, and taps 2 and 5 are 9-bit 2's complement. Taps 3 and 4 are 10-bit magnitude value. The result is normalized by 512.

The maximum down-sampling is 8:1. The maximum up-sampling limited by the 180 MHz VIP pixel clock.

- Example 1: The SD Input can be horizontally up-sampled by 1:13.0 (from 13.5 MHz input -> 175.5 MHz, no vertical up-sampling)
- Example 2: The SD Input can be horizontally up-sampled by 1:3.6 (from 13.5 MHz input -> 48.6 MHz)
- Example 3: The HD Input can be horizontally up-sampled 1:2.4 (from 74.25 MHz input -> 178.2 MHz, no vertical up-sampling)

### 5.5.3 De-Interlacing

De-Interlacing uses a Four-Field Motion Adaptive algorithm. The pixels per line must be <= 960 (after cropping and H-SCALE) with a maximum pixel rate of 90 MHz (2 pixels per clock)

### 5.5.4 Vertical Scaling

Vertical Scaling uses up to an eight tap, eight phase programmable FIR filter. The available taps depend on the input to V-SCALE (after H-SCALE):

- SD (<=960 pixels per line) 8 taps
- SD (<=960 pixels per line) De-Interlaced 6 taps
- HD 4 taps
- HD vertically up-scaled 3 taps

Taps 0 and 7 are 7-bit 2's complement, taps 1 and 6 are 8-bit 2's complement, and taps 2 and 5 are 9-bit 2's complement. Taps 3 and 4 are 10-bit magnitude value. If less than eight taps are available, the center most taps are used. The result is normalized by 512.

The maximum down-sampling is 8:1, and the quality is limited by the number of taps. The maximum up-sampling limited by the 180 MHz VIP pixel clock.

- Example 1: The SD Input can be vertically up-sampled by 1:13.0 (from 13.5 MHz input -> 175.5 MHz, no horizontal up-sampling)
- Example 2: The SD Input can be horizontally and vertically up-sampled by 1:3.6 (from 13.5 MHz input -> 48.6 MHz from H-SCALE -> 174.96 MHz)
- Example 3: The HD Input can be vertically up-sampled by 1:2.4 (from 74.25 MHz input -> 178.2 MHz, no horizontal up-sampling)

### 5.5.5 Video Output Scaling

Video output scaling is a data driven process that occurs at the 125 MHz VOP clock rate. The data can be processed at any speed up to the VOP clock. Video Output Scaling includes:

- Luma and Chroma Vertical and Horizontal Scaling
- Luma Edge Enhancement
- Video and Graphics Mixing
- Hardware Cursor
- Color Processing
- Formatting video for desired output: RGB or YUV 4:2:2

Figure 5-8 shows the flow of data during video output scaling.

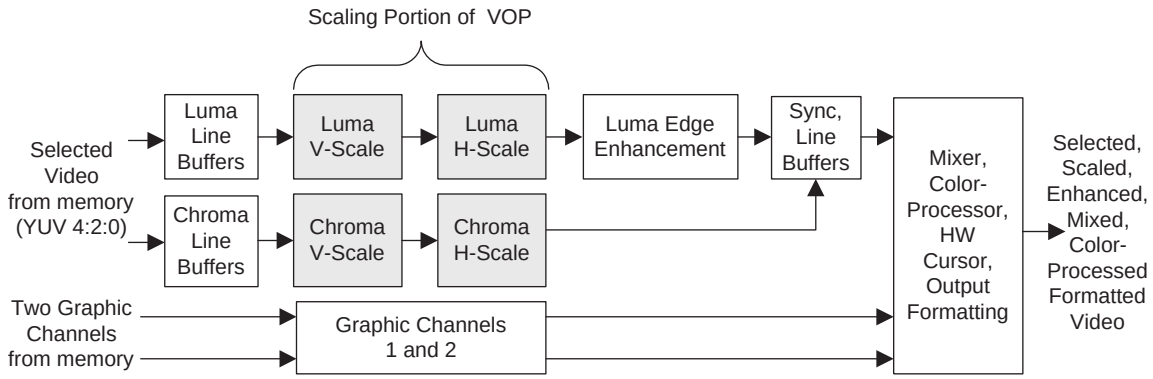



**Figure 5-8 Video Output Processing Data Flow**

### *5.5.6 Vertical Scaling*

Vertical scaling uses up to an eight-tap, eight-phase programmable FIR filter for Luma and Chroma. The number of available taps depends on whether up-scaling or down-scaling is occurring and the horizontal resolution input to V-SCALE:

- Up-sampling, <=960 pixels per line: 8 taps Luma, 4 taps Chroma
- Up-sampling, <=960 pixels per line: 6 taps Luma, 4 taps Chroma
- Up-sampling, > 960 pixels per line: 4 taps Luma, 2 taps Chroma
- Down-sampling: 2 taps Luma, 2 taps Chroma

Taps 0 and 7 and taps 1 and 6 are 6-bit 2's complement, and taps 2 and 5 are 7-bit 2's complement. Taps 3 and 4 are 8-bit magnitude value. If less than eight taps are available, the center most taps are used. The result is normalized by 256.

The down-sampling ratio limited by the quality achievable with two taps (and the 125 MHz VOP pixel clock). There is no limit on the up-sampling ratio other than limit imposed by video output clock.

### *5.5.7 Horizontal Scaling*

Taps 0 and 7 and taps 1 and 6 are 6-bit 2's complement, and taps 2 and 5 are 7-bit 2's complement. Taps 3 and 4 are 8-bit magnitude value The result is normalized by 256.

The maximum down-sampling is 8:1 and limited by the 125 MHz VOP pixel clock.

- Example 1: The video for SD Output can be horizontally down-sampled 8:1 (from 108.0 MHz -> 13.5 MHz SD output, no vertical down-sampling)
- Example 2: The video for SD Output can be vertically and horizontally down-sampled 3:1 (from 121.5 MHz -> 40.5 MHz from V-SCALE -> 13.5 SD MHz output)
- Example 3: The video for HD Output can be horizontally down-sampled 1.67:1 (from 124.2 MHz -> 74.25 MHz HD output, no vertical down-sampling)

There is no limit on up-sampling ratio other than that it is limited to eight taps and the limit imposed by video output clock).

### 5.5.8 Using the VIP for Output Scaling

The Video Output Processor (VOP) vertical down-scaling is limited to two taps, which restricts the output quality. Using one of the Video Input Processors (VIPs) for down-scaling can greatly improve the output quality. Refer to Figure 5-4 on page 103.

To use the Video Input Processor for down-scaling:

1. The desired video is read from memory into a VIP
2: The VIP implements High-Quality down-scaling
3: The output of the VIP is written back to memory
4: The Scaled Video is then output through the VOP as normal

### 5.5.9 Using the VIP for Picture in Picture

A Video Input Processor can be used to scale video and create composites. If a VIP is not used as an input processor it is available for scaling video. An example is Video Conferencing. In this application, the local live video is sent out to the remote site, but it is also combined with the video being received as a picture-in-Picture (PIP) so that the local participants can see the signal that is being sent out.

In this case:

1. The remote video signal is received and decompressed into memory
2: VIP1 processes the local live camera input for compression
3: That input is compressed and transmitted to the far end.
4: VIP2 scales the live camera input for PIP
5: VIP2 writes the PIP video over the decompressed video received from the remote site
6: The VOP outputs the combined image each frame to the local monitor

***VIN0 and VIN1 Control***

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | CLK En | MSBDataS-el | |
| **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| MSB-Data Sel | LSBDataSel | | | PXCLKSel | | | FieldSel | | | VsyncSel | | | HsyncSel | | |

Reserved fields should be ignored (masked) when read, and only 0's should be written to them.

| | |
|---|---|
| CLKEn | Clock Enable bit for the video input clock. |
| MSBDataSel | 000: Source from V0_DATA (default)<br>001: Source from V1_DATA<br>010: Source from V2_DATA<br>011: Source from V3_DATA<br>100: Source from VOUT internal signals<br>In HD mode, this field selects the video data bus to source the MSB, data[15:8]. |
| LSBDataSel | 000: Source from V0_DATA (default)<br>001: Source from V1_DATA<br>010: Source from V2_DATA<br>011: Source from V3_DATA<br>100: 100: Source from VOUT internal signals<br>In SD mode this field selects the video data bus to source the input data. In HD mode, this field selects the video data bus to source the LSB, data[7:0]. |
| PXCLKSel | 000: Source from V0_PXCLK pad input (default)<br>001: Source from V1_PXCLK pad input<br>010: Source from V2_PXCLK pad input<br>011: Reserved<br>100: Source from V0_PXCLK clock generator directly. (likely not used)<br>101: Source from V1_PXCLK clock generator directly. (likely not used)<br>110: Source from V2_PXCLK clock generator directly<br>111: Source from V3_PXCLK clock generator directly<br>When VIN sources a video stream from the VOP, it gets it from the internal signals, not from the signals that go through the pad. These signals should be timed internally directly from the clock generator. The VOP should be set to source its clock from the same internal signal from the clock generator |
| FieldSel | 000: Source FIELD from VC0 pad input (default)<br>001: Source FIELD from VC1 pad input<br>010: Source FIELD from VC2 pad input<br>011: Source FIELD from VC3 pad input<br>100: Source FIELD from the internal VOP signal<br>The ‘100’ option will probably not be used as the device can just operate in 656 mode. |
| VsyncSel | 000: Source VSYNC from VC0 pad input (default)<br>001: Source VSYNC from VC1 pad input<br>010: Source VSYNC from VC2 pad input<br>011: Source VSYNC from VC3 pad input<br>100: Source VSYNC from the internal VOP signal<br>The ‘100’ option will probably not be used as the device can just operate in 656 mode. |
| HsyncSel | 000: Source HSYNC from VC0 pad input (default)<br>001: Source HSYNC from VC1 pad input<br>010: Source HSYNC from VC2 pad input<br>011: Source HSYNC from VC3 pad input<br>100: Source HSYNC from the internal VOP signal<br>The ‘100’ option will probably not be used as the device can just operate in 656 mode. |

***VOUT Control***

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | ClkEn | VidData FmtSel | | PXCLKSel | |
| **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| VC2Data Src | | VC2Field Src | | VC2Vsync Src | | VC2Hsync Src | | VC3Data Src | | VC3Field Src | | VC3Vsync Src | | VC3Hsync Src | |

Reserved fields should be ignored (masked) when read, and only 0's should be written to them.

| | |
|---|---|
| ClkEn | 0: VOUT clock is disabled. (Default)<br>1: Enable VOUT clock. |
| VidDataFmtSel | 00: Defaults to VC2DataSrc/VC1DataSrc (Default)<br>01: 18 Bit RGB mode is enabled.<br>10: 16 Bit RGB mode is enabled.<br>11: Reserved |
| PXCLKSel | 00: Source from Video Clock Generator 2<br>01: Source from Video Clock Generator 3<br>10: Source from V2_PXCLK pad input<br>01: Reserved |
| VC2DataSrc | 00: VC2 DATA output is inactive. (Default)<br>01: VC2 DATA is actively driven to zero.<br>10: if (VidDataFmtSel==01)<br>VC2_DATA = {pixdata[21:18], pixdata [15:12]};<br>else if (VidDataFmtSel==10)<br>VC2_DATA = {pixdata [23:19], pixdata [15:13]};<br>else<br>VC2_DATA = pixdata 23:16];<br>11: if (VidDataFmtSel==01)<br>VC2_DATA = {pixdata [11:10], pixdata [7:2]};<br>else if (VidDataFmtSel==10)<br>VC2_DATA = {pixdata [12:10], pixdata [7:3]};<br>else<br>VC2_DATA = pixdata [15:8]; |
| VC2FieldSrc | 00: VID2_FIELD output is inactive. (Default)<br>01: VOUT sources FIELD from VID2_FIELD.<br>10: VID2_ FIELD output is actively driven from VOUT.<br>11: VID2_ FIELD is actively driven to zero |
| VC2VsyncSrc | 00: VID2_VSYNC output is inactive. (Default)<br>01: VOUT sources VSYNC from VID2_FIELD.<br>10: VID2_ VSYNC output is actively driven from VOUT.<br>11: VID2_ VSYNC is actively driven to zero. |
| VC2HsyncSrc | 00: VID2_HSYNC output is inactive. (Default)<br>01: VOUT sources Field from VID2_HSYNC.<br>10: VID2_ HSYNC output is actively driven from VOUT.<br>11: VID2_ HSYNC is actively driven to zero. |

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | ClkEn | VidData FmtSel | | PXCLKSel | |
| **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| VC2Data Src | | VC2Field Src | | VC2Vsync Src | | VC2Hsync Src | | VC3Data Src | | VC3Field Src | | VC3Vsync Src | | VC3Hsync Src | |

| | |
|---|---|
| VC3DataSrc | 00: VC3 DATA output is inactive. (Default)<br>01: VC3 DATA is actively driven to zero.<br>10: if (VidDataFmtSel==01)<br>VC3_DATA = {pixdata[21:18], pixdata [15:12]};<br>else if (VidDataFmtSel==10)<br>VC3_DATA = {pixdata [23:19], pixdata [15:13]};<br>else<br>VC3_DATA = pixdata 23:16];<br>11: if (VidDataFmtSel==01)<br>VC3_DATA = {pixdata [11:10], pixdata [7:2]};<br>else if (VidDataFmtSel==10)<br>VC3_DATA = {pixdata [12:10], pixdata [7:3]};<br>else<br>VC3_DATA = pixdata [15:8]; |
| VC3FieldSrc | 00: VID3_FIELD output is inactive. (Default)<br>01: VOUT sources FIELD from VID3_FIELD. If VC2FieldSrc is set to 01, VC2FieldSrc has priority.<br>10: VID3_ FIELD output is actively driven from VOUT.<br>11: VID3_ FIELD is actively driven to zero. |
| VC3VsyncSrc | 00: VID3_VSYNC output is inactive. (Default)<br>01: VOUT sources VSYNC from VID3_FIELD. If VC2VsyncSrc is set to 01, VC2VsyncSrc has priority.<br>10: VID3_ VSYNC output is actively driven from VOUT.<br>11: VID3_ VSYNC is actively driven to zero. |
| VC3HsyncSrc | 00: VID3_HSYNC output is inactive. (Default)<br>01: VOUT sources Field from VID3_HSYNC. If VC3HsyncSrc is set to 01, VC2HSyncSrc has priority.<br>10: VID3_ HSYNC output is actively driven from VOUT.<br>11: VID3_ HSYNC is actively driven to zero. |

## 5.6 Audio Interfaces

The Audio interface has two signal groups. Each group can operate at audio frequencies independent of each other. The two signals groups allow for two independent stereo inputs which matches the two independent video inputs. There are also three stereo outputs in Audio Group 0.

### *5.6.1 Audio Group Signals*

The following diagram illustrates signal paths and timing for the two audio interfaces.

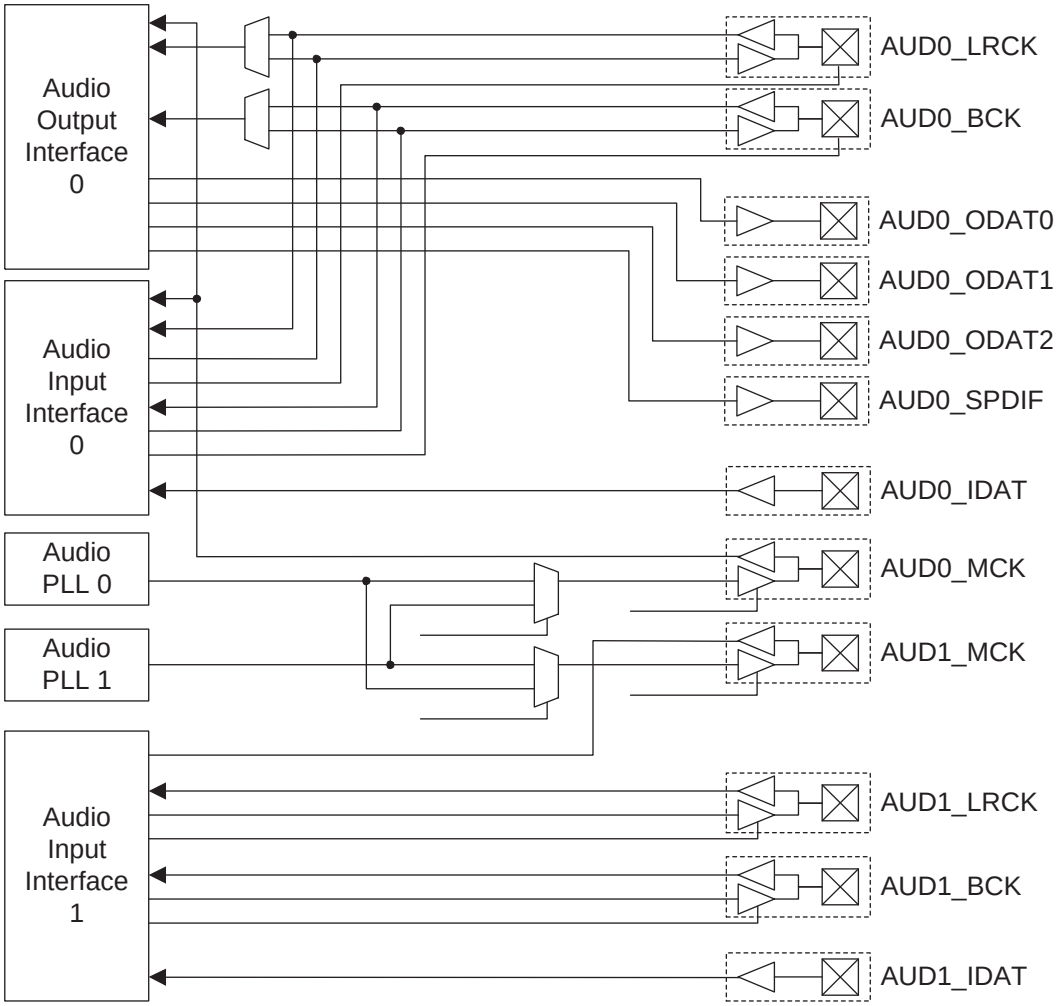



**Figure 5-9 Audio Group Signals**

### 5.6.2 Audio Clocking

Each signal group has an independent MCLK associated with it. MCLK runs at 256 * fs, where fs is the sampling frequency. The MG3500 HD H.264 Codec SoC is intended to operate at the following sampling frequencies.

- 48, 24, 12 kHz
- 44.1, 22.05, 11.025 kHz
- 32, 16, 8 kHz

MCLK can either be sourced from the internal PLL or from an external clock source.

### 5.6.3 Audio Registers

- Select AUD0_MCLK between PLL2, PLL3, and external clocks
- Select AUD1_MCLK between PLL2, PLL3, and external clocks

## 5.7 Host Interfaces

The MG3500 HD H.264 Codec SoC has three types of Host Interfaces: Parallel Master, Parallel Slave, and Serial.

The MG3500 SoC Parallel Host Interface is modeled on the commonly used generic asynchronous-style interface. It consists of a 16-bit data path (HOST_D[15:0]), 23 bits of address, and control signals. The 23 bits address are formed using HOST_A[22:7] and HOST_A[6:1] for the lower 22 bits, and one of the Chip Select pins HOST_CS[5:1] for A23 (see “HOSTChipSelect Register” on page 151). The parallel interface also can be used as a Multiplexed/Data address bus. When using devices with multiplexed address/data buses, the lower 16-bits of the address are sent on the data bus, and all 23-address bits are sent on the address bus. The external device needs to read the upper eight address bits from the address data.

The MG3500 SoC Serial Host Interface is a Serial Peripheral Interface Bus (SPI)-type interface with CPHA=1 and CPOL=1.

The Host Interface pins are shared between the different interfaces. The multiplexing scheme for the Host Interface pins is shown in “Host Interface Pin Multiplexing” on page 149.

### *5.7.1 MG3500 SoC Master Host Interface*

In Master mode, the ARM926-EJ microcontroller inside controls the operation of the internal Codec and peripherals and uses the Host Bus to communicate with external devices. Examples of external devices that are supported are:

- Seve1 and other Mobilygen Coprocessors
- TMS320DM642 (Texas Instruments DSP)
- ADSP2191 (Analog Devices DSP)
- 62256 (industry standard static RAM)

The external devices can have an 8-bit or 16-bit wide data bus, and up to a 24-bit wide address bus.

In Master mode, the host interface is dynamically shared between three different functions:

- Host Master Interface
- NAND/NOR Flash Memory Interface
- Compact Flash Memory Interface

and the Host interface and Compact Flash interface pins are shared between these three functions. The Chip Select signals for each of these interfaces is used to signal the start and end of the interface’s transaction. A single interface owns the common set of I/O pins for the duration of the transaction.

The remainder of this section discusses the Host Processor interface. The Compact Flash and IDE inter face is discussed in “Compact FLASH, IDE” on page 162, and the NAND/NOR Flash Interface is discussed in “NAND and NOR Flash Controller” on page 164.

#### Host Master Interface Connection Diagram

The MG3500 HD H.264 Codec SoC Host Interface connections in Master mode are shown in Figure 5-10.

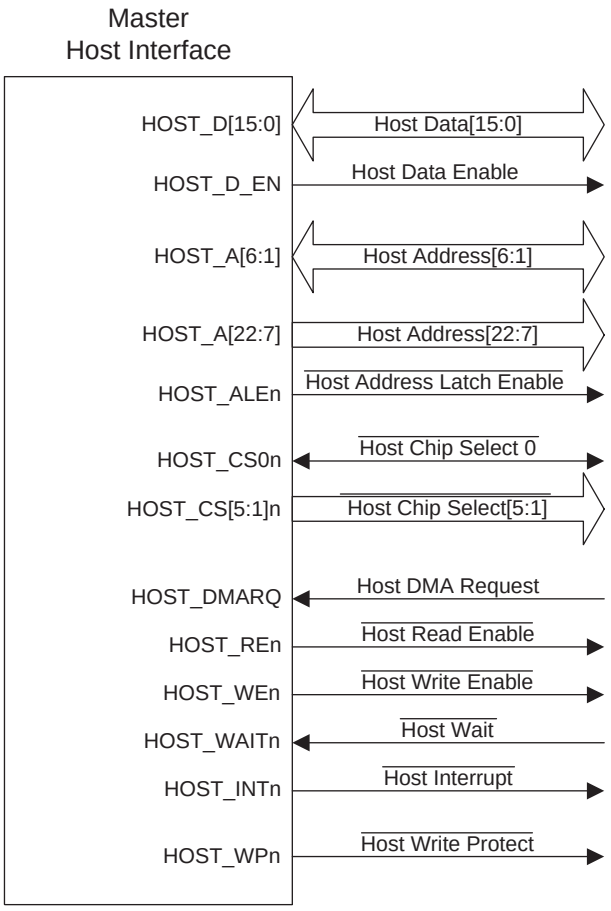



**Figure 5-10 Host Interface Master Mode Connections Diagram**

The signals that comprise the MG3500 SoC Master Host Interface are shown in Table 5-5.

**Table 5-5 MG3500 SoC Master Host Interface Pin Descriptions**

| Pin Name | Signal Name | Direction | Description |
|---|---|---|---|
| HOST_D[15:00] | Data [15:0] | Bidirectional | 16-bit bidirectional Host Data Bus |
| HOST_D_EN | Host Data Enable | O | Host Data Enable |
| HOST_A[6:1] | Address [6:1] | Bidirectional | In Master mode, these act as output pins, and are the six LSBs of the Host Address (along with HOST_A[22:7]. In Slave mode, these act as input pins, and are the complete Host Address. |
| HOST_A[22:7] | Address [22:7] | Outputs | In Master mode, these act as output pins, and are the MSBs of the Host Address (along with HOST_A[6:1]. These signals are not used in Slave Mode. |

**Table 5-5 MG3500 SoC Master Host Interface Pin Descriptions**

| Pin Name | Signal Name | Direction | Description |
|---|---|---|---|
| HOST_ALEn | Host Address Latch Enable | Output | In Master mode, this signal is used to latch the address. This signal is not used in Slave Mode. |
| HOST_CS0n | Host Chip Select 0 | Bidirectional | Active Low Host Chip Select. In Master mode, this signal acts as an output to select the external logic. In Slave mode, this signal acts as a chip select input, and is used to access the MG3500 SoC's Internal registers, External memory, bitstream read and write FIFO registers. |
| HOST_CS[5:1]n | Host Chip Select x | Output | Active Low Host Chip Selects. These pins are programmed using the "HOSTChipSelect Register" on page 151. |
| HOST_REn | Read Enable | Output (Master)<br>Input (Slave) | Active Low Read Enable. In Master mode, the MG3500 SoC asserts this output to indicate that the ARM processor wants to read data from a resource outside the MG3500 SoC. In Slave mode, the external host processor asserts this input to indicate that the it wants to read data from an address inside the MG3500. |
| HOST_WRn | Write Enable | Output (Master)<br>Input (Slave) | Active Low Write Enable. In Master mode, the MG3500 SoC asserts this output to indicate that the ARM processor wants to write data to a resource outside the MG3500 SoC. In Slave mode, the external host processor asserts this input to indicate that the it wants to write data to an address inside the MG3500. |
| HOST_INTn | Interrupt | Output | Active Low Host Interrupt Request. In Host Slave mode, this signal is an open-collector output and requires a 1 KOhm pull-up resistor. |
| HOST_DMARQ | Host DMA Request | Input | HOST DMA Request |
| HOST_WAITn | Wait | Input (Master)<br>Output (Slave) | In Master mode, the external device asserts this pin to extend the bus cycle until it is able to accept data (during a write cycle) or present data (during a read cycle). The polarity of HOST_WAIT is programmable on a per-Chip Select basis in Master mode.<br>In Slave mode, the MG3500 asserts this pin low to extend the bus cycle until it is able to accept data (during a write cycle) or present data (during a read cycle). In Host Slave mode, this signal is always an active low open-collector output and requires a 1 KOhm pull-up resistor. |

### Master Host Interface Read and Write Timing

This section provides information on the timing used when reading or writing the Master Host Interface (MHIF).

Note that in Figure 5-11 and Figure 5-12:

- Signals are sampled on the rising edge of CLK
- CLK is an internal signal and runs at half the ARM clock rate
- H_DMAREQi is an internal signal that is asserted one CLK cycle after the external signal H_DMAREQ

Figure 5-11 shows the timing used when reading the Master Host Interface and Figure 5-12 shows the timing used when writing the Master Host Interface.

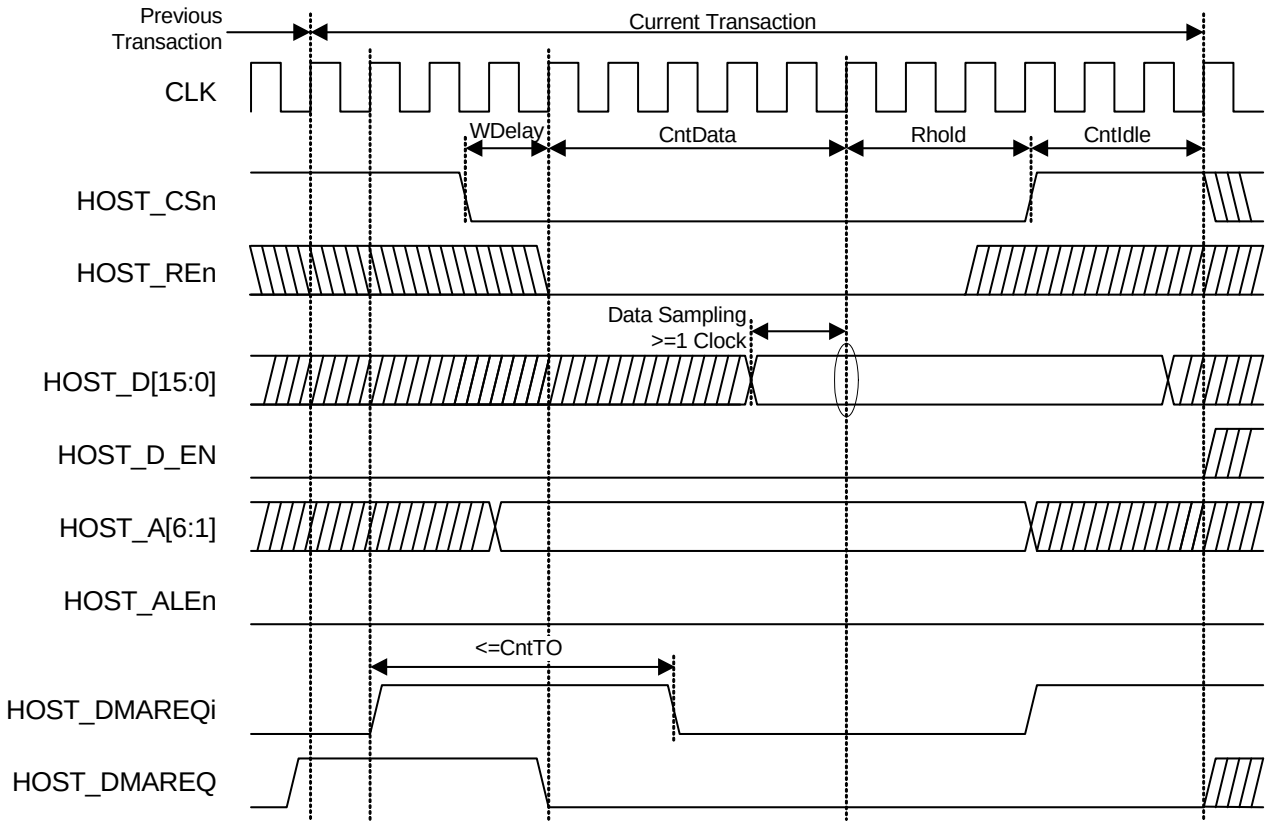



**Figure 5-11 Read Timing for the Master Host Interface**

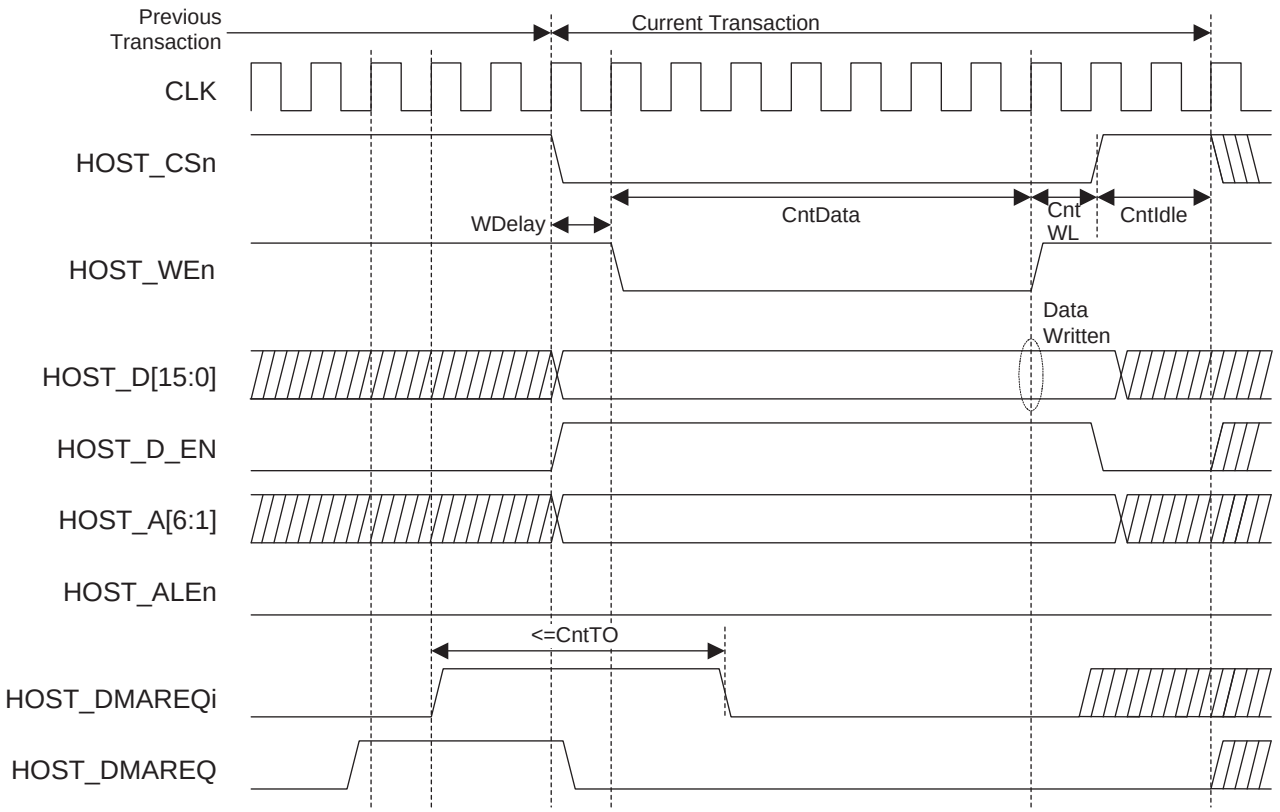



**Figure 5-12 Write Timing for the Master Host Interface**

## Master Host Interface Registers

This section provides a description of the Master Host Interface (MHIF) Registers.

### *DevConfigAn Register*

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Addr Inc | WEn | | DMA WEn | Bus Mux | DEndian | | DWB | AddrMask | | | | | | | |
| **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| AddrMask | | | | AddrBase | | | | | | | | | | | |

| | |
|---|---|
| Reserved fields should be ignored (masked) when read, and only 0's should be written to them. | |
| AddrInc | The device addresses will not be incremented when generating multiple device transactions out of an AMBA High-Speed Bus (AHB) transaction. Default = 0 |
| WEn | Wait for MHIF_WAIT from the device (during a device transaction).<br>The default (set to 0) means that the MHIF will not wait for MHIF_WAIT during a transaction.<br>If (set to 1) then the MHIF will wait for MHIF_ WAIT to be asserted during a transaction.<br>Bit 0: Wait during Data Phase.<br>Bit 1: Wait during Address Phase in muxed mode.<br>Attention: WEn's functionality overlaps with that of TOEn. See Note on page 126. |
| DMAWEn | Wait for MHIF_DMARQ from the device (before starting a device transaction).<br>The default (set to 0) means the MHIF will not wait for MHIF_DMARQ before starting a device transaction.<br>If (set to 1) then the MHIF will wait for MHIF_DMARQ to be asserted before starting a device transaction. |
| BusMux | Address and write data on separate buses versus a common bus. The default (set to 0) means there are separate buses for address and write data.<br>If (set to 1) then MHIF_ADDR is used as a combined bus to carry data and address, and MHIF_DATA_OUT is not used. |
| DEndian | Device Endianess. See Sub-chapter Data Endianess. |
| DWB | Device data Width in (Bytes-1)<br>If (set to 0) then the device data width is 8.<br>If (set to 1) then the device data width is 16. |
| AddrMask | Defines which bits in AddrBase should be compared.<br>The bits in AddrMask that are equal to "0" will be ignored during address comparison. |
| AddrBase | If the most significant 12 bits of Device address from the AHB (that is, bits 23…12) are equal to the AddrBase for the bits enabled with the corresponding AddrMask, and the Valid bit is set, then the corresponding CSn will be asserted. |

***DevConfigBn Register***

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| RHold | | DL | WaitL | CSL | ALEL | WDelay | | TOEn | CntPS | | CntIdle | | CntTO | | |
| **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| CntTO | | | | CntWH | | CntData | | | | CntAH | | CntAW | | CntAS | |

| | |
|---|---|
| Reserved fields should be ignored (masked) when read, and only 0's should be written to them. | |
| WDelay | Delay between CS and Write.<br>The default (set to 0) means CS and WR are asserted at the same time.<br>If (set to 1) then WR is asserted 1 clock after CS. |
| RHold | Hold time after sampling read data.<br>The default (set to 0) means CS, RD, ADDR are held 0 clocks after data sampling.<br>If (set to 1) then CS, RD, ADDR are held 1 clock after data sampling. |
| DL | MHIF_DMARQ is active low (default is active high). |
| WaitL | MHIF_WAIT is active low (default is active high). |
| CSL | Chip Select is active low (default is active high). |
| ALEL | Address Latch Enable strobe acts on the high-to-low transition (default is low-to-high). |
| TOEn | Timeout enable.<br>Default (set to 0) means the Time Out mechanism is disabled (ignore MHIF_WAIT and CntTO).<br>If (set to 1) then the Time Out mechanism is enabled.<br>Attention: TOEn's functionality overlaps with that of WEn. See Note on page 126. |
| CntPS | Pre-scaler for CntAleS, CntAleW, CntAleH, CntData, CntWH, CntIdle. |
| CntIdle | Inter-access Chip Select idle time in number of clocks. |
| CntTO | Device Wait Time Out delay in number of clocks. |
| CntWH | Write Hold Data delay in number of clocks. |
| CntData | Read Data out sampling delay in number of clocks (previously known as WaitStates). |
| CntAH | ALE Hold time in number of clocks. |
| CntAW | ALE Width in number of clocks minus 1. |
| CntAS | ALE Setup time in number of clocks. |

**Note:** Flags TOEn (timeout enable) and WEn (wait enable) are not independent. For example, if TOEn is set to 1 then WEn needs to be set to 1, and if WEn is set to 0 then TOEn needs to be set to 0. The situation TOEn set to 1 and WEn set to 0 is illegal (the user requests wait timeout but doesn't enable waiting). *Attention:* In this situation, the MHIF disregards WEn, and implements the timeout mechanism assuming WEn set to 1.

| WEn | TOEn | Command |
|---|---|---|
| 0 | 0 | No wait, no timeout |
| 0 | 1 | Wait, timeout (Attention) |
| 1 | 0 | Wait, no timeout |
| 1 | 1 | Wait, timeout |

***IntPend Register***

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

| Reserved fields should be ignored (masked) when read, and only 0's should be written to them. | |
|---|---|
| IntPendWTO | "MHIF_WAIT Time Out" interrupt pending |
| IntPendUM | "Trying to access address UnMapped to any device" interrupt pending |

***IntPend Register***

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

| Reserved fields should be ignored (masked) when read, and only 0's should be written to them. | |
|---|---|
| IntEnClrWTO | "MHIF_WAIT Time Out" interrupt enable clear |
| IntEnClrUM | "Trying to access address UnMapped to any device" interrupt enable clear |

Even though this address is seen by the Firmware as a register in the MHIF Register File, it's not a register. Physically it's a bus connected to control logic that eventually feeds/reads the Interrupt Enable register. The Interrupt Enable register itself is only visible to firmware through IntEnSet and IntEnClr.

***IntPend Register***

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

| Reserved fields should be ignored (masked) when read, and only 0's should be written to them. | |
|---|---|
| IntPendWTO | "MHIF_WAIT Time Out" interrupt pending |
| IntPendUM | "Trying to access address UnMapped to any device" interrupt pending |

Even though this address is seen by the Firmware as a register in the MHIF Register File, it's not a register. Physically it's a bus connected to control logic that eventually feeds/reads the Interrupt Enable register. The Interrupt Enable register itself is only visible to firmware through IntEnSet and IntEnClr.

### *5.7.2 Slave Host Interface*

In slave mode, an external System Host CPU controls the MG3500 through the Host Interface. The MG3500 Host Interface also serves as the compressed data interface. This interface allows for directly-addressable access to the external DRAM, the Bitstream Write FIFO, and the MG3500 SoC registers.

In slave mode, only the resources on the Codec side of the MG3500 are addressable through the Host Slave interface. Address lines HOST_A[6:1] are used to address the desired resource, and address lines HOST_A[22:7] are not used.

#### Slave Host Interface Connections

Figure 5-13 shows the connections when using the MG3500 SoC in Slave Host Interface mode.

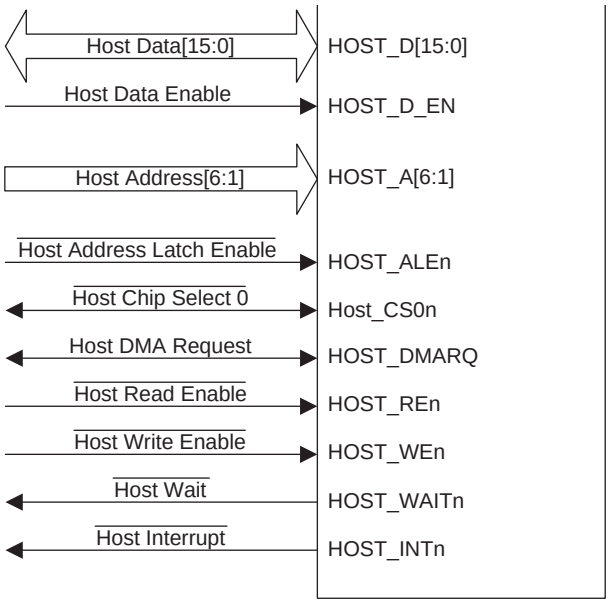



**Figure 5-13 MG3500 Slave Host Connections**

#### MG3500 SoC Slave Host Interface Signals

The signals that comprise the MG3500 Slave Host Interface are shown in Table 5-6.

**Table 5-6 Slave Host Interface Pin Descriptions**

| Pin Name | Signal Name | Direction | Description |
|---|---|---|---|
| HOST_D[15:0] | Data [15:0] | Bidirectional | 16-bit bidirectional Host Data Bus |
| HOST_A[6:1] | Address [6:1] | Bidirectional | In Master mode, these act as output pins, and are the six LSBs of the Host Address (along with HOST_A[22:7]. In the Slave mode, these input pins are the complete Host Address. |
| HOST_CS0n | $\overline{\text{Host Chip Select 0}}$ | Bidirectional | Active Low Host Chip Select. In Master mode, this signal acts as an output to select the external logic. In Slave mode, this signal acts as a chip select input, and is used to access the MG3500 SoC's internal registers, external memory, and bitstream read and write FIFO registers. |
| HOST_REn | $\overline{\text{Read Enable}}$ | Output (Master)<br>Input (Slave) | Active Low Read Enable. In Master mode, the MG3500 SoC asserts this output to indicate that the ARM processor wants to read data from a resource outside the MG3500 SoC. In Slave mode, the external host processor asserts this input to indicate that the it wants to read data from an address inside the MG3500. |
| HOST_WRn | $\overline{\text{Write Enable}}$ | Output (Master)<br>Input (Slave) | Active Low Write Enable. In Master mode, the MG3500 SoC asserts this output to indicate that the ARM processor wants to write data to a resource outside the MG3500 SoC. In Slave mode, the external host processor asserts this input to indicate that the it wants to write data to an address inside the MG3500. |
| HOST_INTn | $\overline{\text{Interrupt}}$ | Output | Active Low Host Interrupt Request. In Slave mode, this signal has an open-collector output and requires a 1 KOhm pull-up resistor. |
| HOST_DMARQ | Host DMA Request | Output | HOST DMA Request. |
| HOST_WAITn | Wait | Input (Master)<br>Output (Slave) | Host Wait pin. In Master mode, the external device asserts this pin to extend the bus cycle until it is able to accept data (during a write cycle) or present data (during a read cycle).<br>In Slave mode, the MG3500 asserts this pin to extend the bus cycle until it is able to accept data (during a write cycle) or present data (during a read cycle). In Slave mode, this signal has an open-collector output and requires a 1 KOhm pull-up resistor. |

### Slave Host Interface Programmer's Model

Because internal operations such as DRAM access and Configuration/Status Register access can incur a lot of latency, and because this latency may not be tolerable to an external host access transaction, the host interface on the MG3500 SoC uses an indirect access method to access internal device resources. In this mode of operation, read and write accesses are deterministic and no Host Ready (or Wait) signaling is needed. A simple SRAM access-type interface is sufficient.

There are only enough registers to require six bits of addressing. Based on this, the Asynchronous Parallel Interface supports a single chip-select mode consisting of one chip select (HOST_CS0n) and six bits of address (HOST_A[6:1]).

The Slave Interface uses a flat six-bit address space. Table 5-7 shows the address space.

**Table 5-7 Register Address Map**

| Register | Offset | Access | Description |
|---|---|---|---|
| EM1Cmd | 0x00 | R/W | External Memory DMA Command |
| EM1XferSize | 0x02 | R/W | External Memory DMA Transfer Size |
| EM1SrcAddrH | 0x04 | R/W | External Memory DMA Source Address High |
| EM1SrcAddrL | 0x06 | R/W | External Memory DMA Source Address Low |
| EM1DestAddrH | 0x08 | R/W | External Memory DMA Destination Address High |
| EM1DestAddrL | 0x0A | R/W | External Memory DMA Destination Address Low |
| EM1Status | 0x0C | Read | External Memory DMA Status |
| EM1RemCount | 0x0E | Read | External Memory DMA Transfer Remainder Count |
| EM1Config | 0x10 | R/W | External Memory DMA Configuration |
| EM1FifoRdPort | 0x12 | Read | External Memory DMA FIFO Read Port (from memory) |
| EM1FifoWrPort | 0x14 | R/W | External Memory DMA FIFO Write Port (to memory) |
| EM1FifoStatus | 0x16 | Read | External Memory DMA FIFO Status |
| CSRCmd | 0x20 | R/W | Configuration/Status Register Command |
| CSRAddr | 0x22 | R/W | Configuration/Status Register Address |
| CSRWrDataH | 0x24 | R/W | Configuration/Status Register Write Data High |
| CSRWrDataL | 0x26 | R/W | Configuration/Status Register Write Data Low |
| CSRRdDataH | 0x28 | Read | Configuration/Status Register Read Data High |
| CSRRdDataL | 0x2A | Read | Configuration/Status Register Read Data Low |
| CSRStat | 0x2C | R/W | Configuration/Status Register Status |
| PeriIntPend | 0x2E | R/W | Peripherals Interrupt Pending (including CSR) |
| PeriIntEnSet | 0x30 | R/W | Peripherals Interrupt Enable – Set |
| PeriIntEnClr | 0x32 | R/W | Peripherals Interrupt Enable – Clear |
| HS_PLL0Control1 | 0x34 | R/W | Slave Host PLL 0 Control |
| HS_PLL0Control2 | 0x36 | R/W | Slave Host PLL 0 Control |
| ChipID | 0x38 | Read | Chip ID |

**Table 5-7 Register Address Map**

| Register | Offset | Access | Description |
|---|---|---|---|
| EM2Cmd | 0x40 | R/W | External Memory DMA Command |
| EM2XferSize | 0x42 | R/W | External Memory DMA Transfer Size |
| EM2SrcAddrH | 0x44 | R/W | External Memory DMA Source Address High |
| EM2SrcAddrL | 0x46 | R/W | External Memory DMA Source Address Low |
| EM2DestAddrH | 0x48 | R/W | External Memory DMA Destination Address High |
| EM2DestAddrL | 0x4A | R/W | External Memory DMA Destination Address Low |
| EM2Status | 0x4C | Read | External Memory DMA Status |
| EM2RemCount | 0x4E | Read | External Memory DMA Transfer Remainder Count |
| EM2Config | 0x50 | R/W | External Memory DMA Configuration |
| EM2FifoRdPort | 0x52 | Read | External Memory DMA FIFO Read Port (from memory) |
| EM2FifoWrPort | 0x54 | R/W | External Memory DMA FIFO Write Port (to memory) |
| EM2FifoStatus | 0x56 | Read | External Memory DMA FIFO Status |
| BFifoWrPort | 0x60 | R/W | Bitstream FIFO Write Port (to Media Engine) |
| BFifoStatus | 0x62 | Read | Bitstream FIFO Status Register |
| BFifoConfig | 0x64 | R/W | Bitstream FIFO Configuration Register |

### 5.8 Configuration and Status Register (CSR) Definition

To access the configuration/status registers (CSRs) from an external host processor, an interface block is provided with a simple set of registers mapped to HOST_CS0. The offsets for each of the registers in this device are listed in Table 5-7 on page 130.

Note: A data “word” in this section refers to a 32-bit word.

This interface device also allows for interrupt status reporting and enabling. The detailed description of the registers is given starting on the next page.

The procedure to read a MG3500 SoC register is:

1. Set up the PeriIntEn registers to enable the CSR interrupt, if that is the preferred method for getting the “access done” message. This only needs to be done once for all CSR accesses.
2: Write the Address to the CSRAddr register.
3: Write the command bits (CSRAccess = 0) to the CSRCmd register.
4: Poll the CSRDone bit in the CSRStat register, or wait for the interrupt.
5: Read the return data from the CSRRdDataH and CSRRdDataL registers.
6: Read CSRStat and check that it has the expected value.
7: Clear the CSRInt bit in the PeriIntPend register if using interrupts, or clear CSRDone bit in the CSRStatus register if polling.

The procedure to write a MG3500 SoC register is:

1. Set up the PeriIntEn registers to enable the CSR interrupt, if that is the preferred method for getting the “access done” message. This only needs to be done once for all CSR accesses.
2: Write the return data to the CSRRdDataH and CSRRdDataL registers.
3: Write the Address to the CSRAddr register..
4: Write the command bits (CSRAccess = 1) to the CSRCmd register.
5: Poll the CSRDone bit in the CSRStat register, or wait for the interrupt.
6: Read the CSRStat register and check that it has the expected value. (In some cases, it may be necessary to read the CSRRdData registers to check a value returned by the Configuration/Status Register Block, if the operation is more complex than a simple register read or write.)
7: Clear the CSRInt bit in the PeriIntPend register if using interrupts, or clear CSRDone bit in the CSRStatus register if polling.

***CSRCmd*** ***Offset: 0x20***

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CAcc | CSRLen | | | Reserved | | | | CSRBlockID | | | | | | | |

| | |
|---|---|
| Reserved fields should be ignored (masked) when read and only 0's should be written to them. | |
| CSRAccess | A 0 written to this field initiates a CSR read from the address provided in CSRAddr.<br>A 1 written to this field, initiates a CSR write to the address provided in CSRAddr with the data provided in CSRWrData. |
| CSRLen | 000 = 4 byte (word) access<br>001 = 1 byte access<br>010 = 2 byte (halfword) access<br>Other codes are reserved and should not be used. |
| CSRBlockID | Block ID for a Configuration/Status Register access |

***CSRAddr*** ***Offset: 0x22***

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CSRAddr | | | | | | | | | | | | | | | |

| | |
|---|---|
| Reserved fields should be ignored (masked) when read and only 0's should be written to them. | |
| CSRAddr | Address (within a Configuration/Status Register Block) for a CSR access. Expected to be word-aligned (bits [1:0] are 0) for 4-byte access and half-word aligned (bit [0] is 0) for 2-byte access. |

***CSRWrDataH*** ***Offset: 0x24***

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CSRWrDataH | | | | | | | | | | | | | | | |

| | |
|---|---|
| CSRWrDataH | High-order 16-bit register from which the data for a CSR write is taken. Used with CSRWrDataL. |

***CSRWrDataL*** ***Offset: 0x26***

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CSRWrDataL | | | | | | | | | | | | | | | |

| | |
|---|---|
| CSRWrDataL | Low-order 16-bit register from which the data for a CSR write is taken. Used with CSRWrDataH. |

***CSRRdDataH*** ***Offset: 0x28***

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CSRWrDataH | | | | | | | | | | | | | | | |

| | |
|---|---|
| CSRRdDataH | High-order 16-bit register containing the data returned for a CSR read or the status information returned for a write. Used with CSRRdDataL. This register is read-only. |

***CSRRdDataL*** ***Offset: 0x2A***

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CSRWrDataL | | | | | | | | | | | | | | | |

| | |
|---|---|
| CSRRdDataL | Low-order 16-bit register containing the data returned for a CSR read or the status information returned for a write. Used with CSRRdDataH. This register is read-only. |

***CSRStat*** ***Offset: 0x2C***

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CSRRespID | | | | | | | | Rsvd | CSRRespLen | | | Reserved | | CS-RErr | CSR Done |

| | |
|---|---|
| Reserved fields should be ignored (masked) when read and only 0's should be written to them. | |
| CSRRespID | Block ID information from the I_obid port when a CSR access is completed (which block responded). If it doesn't match the CSRBlockID originally programmed, then something is wrong. This field is read-only. |
| CSRRespLen | Length of the access actually performed. For a write, it should be 1; for a read, it should match the CSRLen code originally programmed. If not, then something is wrong. This field is read-only. |
| CSRErr | If set to 1 when CSRDone is set, an error occurred in the access. This should never happen. This field is read-only. |
| CSRDone | This bit is set to 1 after each CSRAccess is completed. When the hardware sets this bit to 1, the read data (or write response status) is available in the CSRRdData register. It is not required to clear this bit before initiating a new access; however, software should clear CSRDone if it is polling this bit to determine when an access completes, instead of using the CSRInt interrupt. |

***PeriIntPend*** ***Offset: 0x2E***

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | HMB 1 Int | HMB 0 Int | Reserved | | |

Reserved fields should be ignored (masked) when read and only 0's should be written to them.

The bits in these registers are "sticky": if an interrupt event occurs and sets a bit, the bit stays set until it is cleared. A bit is cleared only by writing a 1 to it; writing a 0 to it has no effect (so the same value that was read from the register can be written back to clear only the interrupt bits that were previously set, not any new ones).

An interrupt on HOST_INTn is generated if any of these bits are set and the corresponding bit in PeriIntEn is also set. For example: HOST_INTn = | (PeriIntPend & PeriIntEn);

| | |
|---|---|
| HMBxInt | Host Mailbox Interrupt. When this interrupt is generated, the host should check the Mailbox module, because this bit is a sticky reflection of the interrupts generated there. The interrupt in the Mailbox module should be cleared before clearing this interrupt. |

***PeriIntEnSet*** ***Offset: 0x30***

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | PeriIntEnSet | | Reserved | | |

Reserved fields should be ignored (masked) when read and only 0's should be written to them.

The Peripheral Interrupt Enable function is implemented with separate "set" and "clear" register addresses, allowing each interrupt enable bit to be set or cleared independently of the other bits so that no read-modify-write cycles are required.

| | |
|---|---|
| PeriIntEnSet | Writing a 1 to a bit at the address for PeriIntEnSet sets the corresponding bit to 1 in PeriIntEn; writing a 0 has *no effect*. Reading the register at the address for PeriIntEnSet returns the current value for PeriIntEn. |

***PeriIntEnClr*** ***Offset: 0x32***

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | PeriIntEnClr | | Reserved | | |

Reserved fields should be ignored (masked) when read and only 0's should be written to them.

The Peripheral Interrupt Enable function is implemented with separate "set" and "clear" register addresses, allowing each interrupt enable bit to be set or cleared independently of the other bits so that no read-modify-write cycles are required.

| | |
|---|---|
| PeriIntEnClr | Writing a 1 to a bit at the address for PeriIntEnClr clears the corresponding bit in PeriIntEn; writing a 0 has *no effect*. Reading the register at the address for PeriIntEnClr returns the current value for PeriIntEn. |

### 5.9 DMA Engine Register Definition

To access the MG3500 Codec's external memory from the external host processor, an interface block is provided with a simple set of registers mapped to interface registers.

The base address and the offsets to access the Codec's host interface are listed in Figure 5-7.

Do the following steps to read a block of the MG3500 Codec's memory:

1. Verify that the EMBusy bit in the EMStatus register is set to 0; otherwise, you must wait.
2: If necessary, update the MG3500 Codec's DMA engine configuration in the EMConfig register.
3: Setup the address in the EMSrcAddrH and EMSrcAddrL registers.
4: Write the transfer length (in units of 32-bit words) to the EMXferSize register.
5: Write the 'read' command to the EMCmd register.
6: Set up the external host to DMA the data from the EMFifoRdPort register to a buffer in the external host's memory
   -or-
   Loop through enough loads from the EMFifoRdPort register to read the specified number of words (You must check the EMFifoStatus register in this case).
7: Optionally, check the EMBusy bit in the EMStatus register or use the EMInt bit in the PeriIntPend register to determine when the DMA engine is finished. (Though for a 'read' operation, the external host's DMA engine can generate an interrupt when the DMA is complete).

The procedure to write to a block of the MG3500 Codec's memory is:

1. Verify that the EMBusy bit in the EMStatus register is set to 0; otherwise, you must wait.
2: If necessary, update the MG3500 Codec's DMA engine configuration in the EMConfig register.
3: Setup the address in the EMDestAddrH and EMDestAddrL registers.
4: Write the transfer length (in units of 32-bit words) to EMXferSize.
5: Write the 'write' command to the EMCmd register.
6: Set up the External Host to DMA the data from a buffer in the External Host's memory to the EMFifoWrPort register
   -or-
   Loop through enough stores to the EMFifoWrPort register to write the specified number of words (You must check the EMFifoStatus register in this case).
7: Optionally, check the EMBusy bit in the EMStatus register or use the EMInt bit in the PeriIntPend register to determine when the DMA engine is finished. (For a 'write' operation, the external host's DMA engine can generate an interrupt when the DMA is complete from the external host's point of view, but the MG3500 Codec may still be working on it).
8: When the EMBusy bit goes from 1 to 0, indicating that a DMA operation has just completed, the EMInt bit in the PeriIntPend register is set. This can be used to generate an interrupt to the external host (the HOST_INTn pin is pulled low) by setting the corresponding bit in the PeriIntEn register.

The interface logic for this device asserts the DMA request to the external host (by asserting HOST_DMAREQ) when it has available at least EMDThresh 16-bit words of data in its Read FIFO or can accept at least EMDThresh 16-bit words of data into its Write FIFO, depending on the direction of

the transfer programmed in the EMCmd register. If the External Host's DMA engine is not used, individual words can be read (loaded) from or written (stored) to this port, but software must check the status of the FIFO after every EMDThresh words.

***EM1Cmd*** ***Offset: 0x00***
***EM2Cmd*** ***Offset: 0x40***

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| EMCmd | | EM-Marb Prior | Swap | Clk Md | Clk Edge | Reserved | | | | | | | | | |

| | |
|---|---|
| Reserved fields should be ignored (masked) when read and only 0's should be written to them.<br>This register should not be modified while EMBusy is set to 1. | |
| EMCommand | 00: Idle: No operation is performed<br>01: Read: Initiate transfer from internal memory, starting at EMSrcAddr, to the Memory Read FIFO, which can be read by the external host (Static Bus) via the EMFifoRdPort.<br>10: Write: Initiate transfer from the Memory Write FIFO to internal memory, starting at EM-DestAddr; the Memory Write FIFO is filled by the External Host (Static Bus) via the EMFifoWrPort.<br>11: Copy: Initiate transfer (copy) from memory starting at EMSrcAddr to memory starting at EMDestAddr. This mode is not expected to be used.<br>For all operations, the transfer length is given by EMXferSize. |
| EMMarbPrior | 0 = Don't queue up multiple memory transactions<br>1 = Queue up multiple memory transactions – this DMA engine has priority.<br>The Slave Host I/F has a single Memory Subsystem Read Port and a (separate) single Memory Subsystem Write Port, both shared between EM1 and EM2 DMA engines. If only one of these engines is doing Reads, then EMMarbPrior can be set to 1 for that engine, which will result in higher data throughput. Similar for writes. For example, if EM1 is dedicated to doing reads and EM2 is dedicated to doing writes, or if the Host only allows one DMA at a time (uses only one engine at a time), both engines' EMMarbPrior bits can be set to 1 for maximum throughput. |
| EMEndianSwap | 0: Don't swap bytes<br>1: Swap bytes in 16-bit word on the way into or out of memory |
| EMFifoRdClock-Mode | 0: use original clock-synchronized pulse for reading from DMA Read Data FIFO<br>1: fast read mode from DMA Read Data FIFO |
| EMFifoRd-ClockEdge | 0: Clock "fast read mode" on rising edge of HOST_REn<br>1: Clock "fast read mode" on falling edge of HOST_REn |

**Note:** While the External Memory 1 (EM1) and External Memory 2 (EM2) DMA engines are identical in design, the MG3500 SoC only has the DMA request signal from the devices on $\overline{\text{HCS1}}$ (External Memory Port 2, Bitstream Write FIFO Port) connected to the $\overline{\text{HDMAREQ}}$ pin. When the EM2Cmd register is written with an active value, the $\overline{\text{HDMAREQ}}$ signal represents the request generated from that logic; otherwise, it represents the request signal generated from the Bitstream FIFO logic.

***EM1XferSize*** ***Offset: 0x02***
***EM2XferSize*** ***Offset: 0x42***

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| EMXferSize | | | | | | | | | | | | | | | |

Reserved fields should be ignored (masked) when read and only 0's should be written to them.
This register should not be modified while EMBusy is set to 1.

| | |
|---|---|
| EMXferSize | Number of 32-bit data words to transfer. 0 means no words are to be transferred; EMBusy will not be set.<br>For Frame Mode, interpreted as:<br>EMYSize[5:0] = EMXferSize[15:10] - Vertical size of block to transfer (number of "rows")<br>EMXSize[9:0] = EMXferSize[9:0] - Horizontal size (in bytes) of block to transfer (size of "row") |

***EM1SrcAddrH*** ***Offset: 0x04***
***EM2SrcAddrH*** ***Offset: 0x44***

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| EMSrcAddrH | | | | | | | | | | | | | | | |

| | |
|---|---|
| EMSrcAddrH | High 16-bits of the source address for a "read" (external host — internal memory) or "copy" (internal memory — internal memory) operation. EMSrcAddrH and EMSrcAddrL are concatenated to form a 32-bit byte address going to the memory subsystem. This register should not be modified while EMBusy is set to 1. During the operation, the hardware will update this register as it progresses. |

For Frame Buffer Access (EMMode=00 or 01), this register is interpreted as follows:

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| EMSrcYAddr | | | | | | | | | | | | | | | |

| | |
|---|---|
| EMSrcYAddr | Starting Vertical/Y source address |

***EM1SrcAddrL*** ***Offset: 0x06***
***EM2SrcAddrL*** ***Offset: 0x46***

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| EMSrcAddrL | | | | | | | | | | | | | | | |

| | |
|---|---|
| EMSrcAddrL | Low 16-bits of the source address for a "read" (external host — internal memory) or "copy" (internal memory — internal memory) operation. EMSrcAddrH and EMSrcAddrL are concatenated to form a 32-bit byte address going to the memory subsystem. This register should not be modified while EMBusy is set to 1. During the operation, the hardware will update this register as it progresses. |

For Frame Buffer Access (EMMode=00 or 01), this register is interpreted as follows:

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| EMSrcXAddr | | | | | | | | | | | | | | | |

| | |
|---|---|
| EMSrcXAddr | Starting Horizontal/X source address |

***EM1DestAddrH*** ***Offset: 0x08***
***EM2DestAddrH*** ***Offset: 0x48***

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| EMDestAddrH | | | | | | | | | | | | | | | |

| | |
|---|---|
| EMDestAddrH | High 16-bits of the Destination address for a “write” (external host — internal memory) or “copy” (internal memory — internal memory) operation. EMDestAddrH and EMDestAddrL are concatenated to form a 32-bit byte address going to the memory subsystem. This register should not be modified while EMBusy is set to 1. During the operation, the hardware will update this register as it progresses. |

For Frame Buffer Access (EMMode=00 or 01), this register is interpreted as follows:

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| EMDestYAddr | | | | | | | | | | | | | | | |

| | |
|---|---|
| EMDestYAddr | Starting Vertical/Y Destination address |

***EM1DestAddrL*** ***Offset: 0x0A***
***EM2DestAddrL*** ***Offset: 0x4A***

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| EMDestAddrL | | | | | | | | | | | | | | | |

| | |
|---|---|
| EMDestAddrL | Low 16-bits of the Destination address for a “write” (external host — internal memory) or “copy” (internal memory — internal memory) operation. EMDestAddrH and EMDestAddrL are concatenated to form a 32-bit byte address going to the memory subsystem. This register should not be modified while EMBusy is set to 1. During the operation, the hardware will update this register as it progresses. |

For Frame Buffer Access (EMMode=00 or 01), this register is interpreted as follows:

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| EMDestXAddr | | | | | | | | | | | | | | | |

| | |
|---|---|
| EMDestXAddr | Starting Horizontal/X Destination address |

***EM1Status*** ***Offset: 0x0C***
***EM2Status*** ***Offset: 0x4C***

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Busy | | | | | | | | | | | | | | | |

| Reserved fields should be ignored (masked) when read. This register is read-only. | |
|---|---|
| Busy | 0: No operation is in progress; other registers may be changed.<br>1: A DMA operation is in progress; the EMCmdParams, EMSrcAddr, EMDestAddr, and EMConfig registers may *not* be changed. |

***EM1RemCount*** ***Offset: 0x0E***
***EM2RemCount*** ***Offset: 0x4E***

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| EMRemCount | | | | | | | | | | | | | | | |

| Reserved fields should be ignored (masked) when read. This register is read-only. | |
|---|---|
| EMRemCount | Number of 32-bit data words remaining to be transferred |

***EM1Config*** ***Offset: 0x10***
***EM2Config*** ***Offset: 0x50***

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| EMW | EMDThresh | | | | EMB | EMMode | | EMBaseId | | | | | | | |

| | |
|---|---|
| Reserved fields should be ignored (masked) when read and only 0's should be written to them. | |
| EMWait | **EM1Config:**<br>0: Reserved: Setting EMWait in the EM1Config register to zero can result in undesired behaviour.<br>1: Use HOST_WAIT to stall the parallel asynchronous host interface; don't use HOST_DMARQ.<br>**EM2Config:**<br>0: Use HOST_DMARQ for hardware flow control on the parallel asynchronous host interface instead of HOST_WAIT.<br>1: Use HOST_WAIT to stall the parallel asynchronous host interface; don't use HOST_DMARQ. |
| EMDThresh | **EM1Config**:<br>Reserved<br>**EM2Config:**<br>For writes: the DMA request signal or the EMFIFOStatus bits are deasserted when there are less than EMDThresh spaces available in the DMA FIFO for writing.<br>For reads: the DMA request signal or the EMFIFOStatus bits are de-asserted when there are less than or equal to EMDThresh words in the FIFO to be read. |
| EMBurst | Number of 16-bit words per internal memory burst access. A DMA operation is broken into sequential memory requests of the specified burst size. This parameter must be set to a value less than (usually half of) the MMU buffer for the external host.<br>Code:<br>0: Eight 16-bit words<br>1: Sixteen 16-bit words (default)<br>This field is not used if EMMode is set for Frame Buffer access. The entire DMA operation is sent as one internal memory operation (using EMYSize/EMXSize/EMY*Addr/EMX*Addr). The software must take care not to attempt a request larger than the memory subsystem can handle (must be no larger than the MMU buffer size allocated to the external host interface). |
| EMMode | Use EMMode as the MMU Transaction Mode<br>00: Frame Buffer — frame access<br>01: Frame Buffer — field access<br>10: Linear (default)<br>11: Reserved; don't use. |
| EMBaseId | EMSrcAddr and EMDestAddr specify addresses (offsets) relative to the memory subsystem partition identified by EMBaseId. (default: 0) |

**Note:** The EMWait and EMDThresh bits change functions between the EM1Config and EM2Config registers as shown above.

***EM1FifoRdPort*** ***Offset: 0x12***
***EM2FifoRdPort*** ***Offset: 0x52***

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved fields should be ignored (masked) when read. This register is read-only. | | | | | | | | | | | | | | | |
| EMFifoRdPort | A read from this port removes and returns a 16-bit data word from the Memory Read FIFO that was read from the internal memory.<br>***Warning: Do not write to this register! Data will be lost!*** | | | | | | | | | | | | | | |

***EM1FifoWrPort*** ***Offset: 0x14***
***EM2FifoWrPort*** ***Offset: 0x54***

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| EMFifoWrPort | | | | | | | | | | | | | | | |
| EMFifoWrPort | 16-bit data from the “Static Bus” written to this port’s address is placed into the Memory Write FIFO to be sent to internal memory. Reading from this address returns 0’s. | | | | | | | | | | | | | | |

***EM1FifoStatus*** ***Offset: 0x16***
***EM2FifoStatus*** ***Offset: 0x56***

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | ERdR | EWrR |
| Reserved fields should be ignored (masked) when read and only 0’s should be written to them. | | | | | | | | | | | | | | | |
| EMFifoRdReq | 0: No more words are available for reading beyond the current burst of eight<br>1: At least EMDThresh more 16-bit words are available in the Memory Read FIFO<br>If the External Host DMA engine is being used, then flow control is done by the DMA request line; in this case, it is not necessary for software to check this bit. | | | | | | | | | | | | | | |
| EMFifoWrReq | 0: No more words can be accepted beyond the current burst of eight<br>1: At least EMDThresh more 16-bit words can be accepted by the Memory Write FIFO<br>If the External Host DMA engine is being used, then flow control is done by the DMA request line; in this case, it is not necessary for software to check this bit. | | | | | | | | | | | | | | |

### 5.10 Bitstream Write Register Definition

To send a bitstream such as an MPEG transport or program stream from the external host to the MG3500 SoC, an interface block is provided with a simple set of registers. The offsets for these registers are shown in Table 5-7 on page 130.

***BiFifoWrPort*** ***Offset: 0x60***

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| BiFifoWrPort | | | | | | | | | | | | | | | |

| | |
|---|---|
| BiFifoWrPort | 16-bit data from the “Static Bus” written to this port’s address is placed into the Memory Write FIFO to be sent to internal memory. Reading from this address returns 0’s. |

***BiFifoStatus*** ***Offset: 0x62***

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | | | | | | BiFifoStatus |

| | |
|---|---|
| | 0: No more words can be accepted beyond the current burst of BThresh.<br>1: At least BThresh more 16-bit words can be accepted by the Bitstream FIFO<br>If the external host DMA engine is being used, then flow control is done by the HOST_DMAREQ; in this case, it is not necessary for software to check this bit. |

***BFifoConfig*** ***Offset: 0x64***

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | BEndS | BWait | BThresh | | | | Rsvd |

| | |
|---|---|
| Reserved fields should be ignored (masked) when read and only 0’s should be written to them. | |
| BEndianSwap | 0: Don’t swap bytes<br>1: Swap bytes in 16-bit word on the way into or out of memory |
| BWait | 0: Use HDMAREQ for hardware flow control, not HWAIT_, on the parallel asynchronous host interface.<br>1: Use HWAIT_ to stall the parallel asynchronous host interface; don’t use HDMAREQ. |
| BThresh | The DMA request signal or the BFIFOStatus bits are deasserted when there are less than BThresh spaces available in the DMA FIFO for writing. |

The interface logic for this device asserts the DMA request to the external Host (by driving HDMAREQ_ high) when it can accept at least BThresh of data into its FIFO. If the External Host’s DMA engine is not used, individual words can be written (stored) to this port, but software must check the status of the FIFO after every BThresh.

### 5.11 Special Registers

These registers are used for controlling or reading MG3500 SoC-specific features.

***ChipID*** ***Offset: 0x38***

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ProductID | | | | | | | | TapeoutRev | | | | MaskID | | | |

| | |
|---|---|
| Reserved fields should be ignored (masked) when read and only 0's should be written to them.<br>This register is read-only. | |
| ProductID | Chip Product ID: This field should read back 0x03. |
| TapeoutRev | Chip Tapeout Revision: Currently set to 0x0. |
| MaskID | Chip Mask ID: Currently set to 0x2. |

This register should read back 0x0302.

### *5.11.1 Serial Host Interface*

The MG3500 HD H.264 Codec SoC Serial Host Interface is a Serial Peripheral Interface Bus (SPI)-type interface with CPHA=1 and CPOL=1.

#### Serial Host Interface Connections

Figure 5-14 shows the connections when the Serial Host Interface is being used.

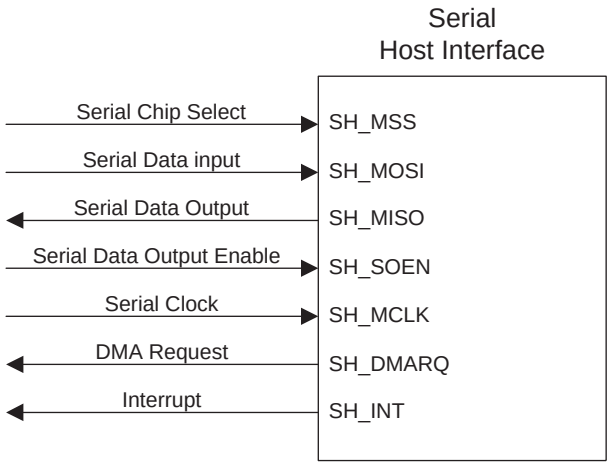



**Figure 5-14 Serial Host Interface Connections**

#### Serial Host Interface Signals

The signals that comprise the MG3500 SoC Serial Host Interface are shown in Table 5-8.

**Table 5-8 Serial Host Interface Signals**

| Pin | I/O | Description |
|---|---|---|
| SH_MSS | I | SH_MSS (Serial Host Chip Select). Assert this pin low to indicate that an operation is in progress. |
| SH_MOSI | I | Serial Host Serial Data Input- Data is serially shifted in on this pin. |
| SH_MISO | O | Serial Host Serial Data Output - Data is serially shifted out on this pin. |
| SH_SOEN | O | Serial Host Output Data Enable |
| SH_DMARQ | I/O | Serial Host DMA Request |
| SH_INT | O | Serial Host Interrupt |
| SH_MCLK | I | Serial Host MCLK Pin. Strobe this pin from high to low to generate the internal serial clock. During a read operation (when strobing data out), the serial data changes on the falling edge of SH_MCLK. The maximum frequency for this pin is 30 MHz. |

Figure 5-15 shows the functional timing for a Serial Host Write operation, and Figure 5-16 shows the functional timing for a Serial Host Read operation.

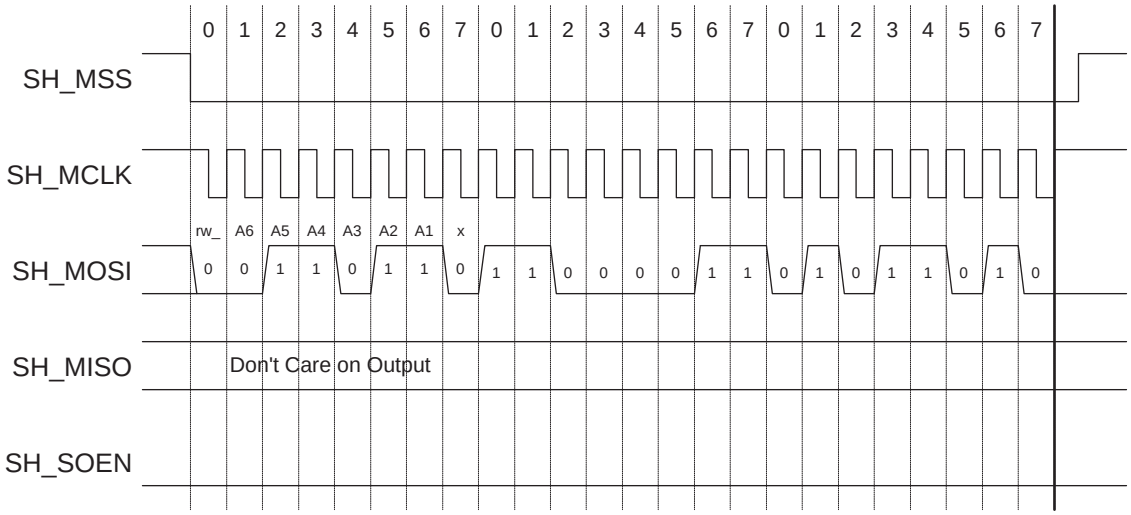



**Figure 5-15 Serial Host Write Timing**

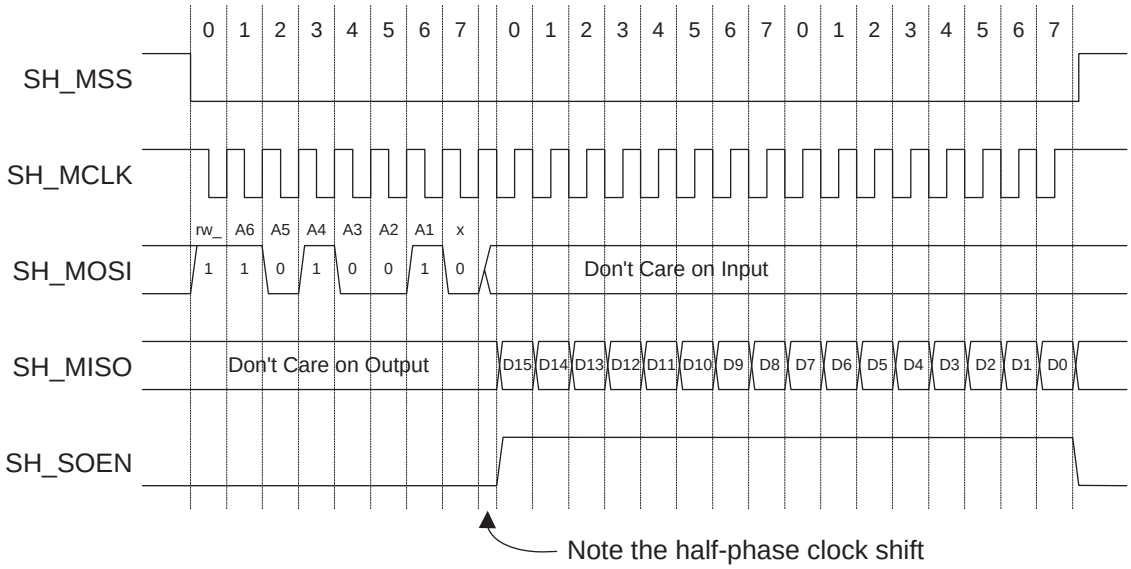



**Figure 5-16 Serial Host Read Timing**

## Protocol

The protocol that must be executed is as follows.

- The first access after any reset must always be a write operation.

- Both read and write operations consist of 24 bits or clocks. The first eight bits are control or transfer direction information and the register address. The following 16 bits or clocks shift in two bytes for a write operation (Figure 5-15) or shift out two bytes for a read operation (Figure 5-16).
- Before starting a serial operation, set the SH_MSS pin low (0). Leave the SH_MSS pin set to low until the read or write operation is complete, then set the SH_MSS pin high.
- All bits are shifted in serially on SH_MOSI pin. This pin is always an input in host slave serial mode. After driving the SH_MOSI bit to the appropriate binary value, shift the data bit in by strobing the SH_MCLK pin.
- The first bit shifted in indicates the direction of the upcoming operation. If the first bit is a 1, the operation is a serial read. If the first bit is a 0, the operation is a serial write.
- Shift in the six bit address corresponding to the host register address. Shift in the address MSB first to LSB last as shown in Table 5-9. The last bit of the first byte (bit 8) is a don't care. While a clock pulse must be generated, any value can be asserted on the SH_MOSI input pin.

**Table 5-9 Host Register Address Bit Shift Examples**

| Register | Address | Bit Shift Order (First to Last) | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | A6 | A5 | A4 | A3 | A2 | A1 | x |
| CSRAddr | 0x22 | 0 | 1 | 0 | 0 | 0 | 1 | x |
| PLL Dividers | 0x36 | 0 | 1 | 1 | 0 | 1 | 1 | x |
| EM1Cmd | 0x00 | 0 | 0 | 0 | 0 | 0 | 0 | x |
| EM2FifiStatus | 0x56 | 1 | 0 | 1 | 0 | 1 | 1 | x |

### *5.11.2 Host Interface Pin Multiplexing*

Table 5-10 shows the multiplexing for the Host Interface pins.

**Table 5-10 Host Interface Pin Multiplexing Signals**

| Pin | I/O | Host Master | | | Host Slave | |
|---|---|---|---|---|---|---|
| | | Parallel Master | NAND/NOR Flash | Compact Flash | Parallel Slave | Serial |
| HOST_CS0n | I/O | CS0 | CS0 | CS0 | HOST_CS0 | SH_MSS |
| HOST_CS1n | O | CS1 | CS1 | CS1 | | |
| HOST_CS2n | O | CS2 | CS2 | CS2 | | |
| HOST_CS3n | O | CS3 | CS3 | CS3 | | |
| HOST_CS4n | O | CS4 | CS4 | CS4 | | |
| HOST_CS5n | O | CS5 | CS5 | CS5 | | |
| HOST_A22 | O | A21 | A21 | | | |
| HOST_A21 | O | A20 | A20 | | | |
| HOST_A20 | O | A19 | A19 | | | |
| HOST_A19 | O | A18 | A18 | | | |
| HOST_A18 | O | A17 | A17 | | | |
| HOST_A17 | O | A16 | A16 | | | |
| HOST_A16 | O | A15 | A15 | | | |
| HOST_A15 | O | A14 | A14 | | | |
| HOST_A14 | O | A13 | A13 | | | |
| HOST_A13 | O | A12 | A12 | | | |
| HOST_A12 | O | A11 | A11 | | | |
| HOST_A11 | O | A10 | A10 | A10 | | |
| HOST_A10 | O | A9 | A9 | A9 | | |
| HOST_A9 | O | A8 | A8 | A8 | | |
| HOST_A8 | O | A7 | A7 | A7 | | |
| HOST_A7 | O | A6 | A6 | A6 | | |
| HOST_A6 | I/O | A5 | A5 | A5 | HOST_A6 | |
| HOST_A5 | I/O | A4 | A4 | A4 | HOST_A5 | |
| HOST_A4 | I/O | A2 | A2 | A2 | HOST_A4 | |
| HOST_A3 | I/O | A3 | A3 | A3 | HOST_A3 | |
| HOST_A2 | I/O | A1 | A1 | A1 | HOST_A2 | |
| HOST_A1 | I/O | A0 | A0 | A0 | HOST_A1 | |
| HOST_D15 | I/O | D15 | D15 | D15 | HOST_D15 | |
| HOST_D14 | I/O | D14 | D14 | D14 | HOST_D14 | |
| HOST_D13 | I/O | D13 | D13 | D13 | HOST_D13 | |
| HOST_D12 | I/O | D12 | D12 | D12 | HOST_D12 | |
| HOST_D11 | I/O | D11 | D11 | D11 | HOST_D11 | |

**Table 5-10 Host Interface Pin Multiplexing Signals**

| Pin | I/O | Host Master | | | Host Slave | |
|---|---|---|---|---|---|---|
| | | Parallel Master | NAND/NOR Flash | Compact Flash | Parallel Slave | Serial |
| HOST_D10 | I/O | D10 | D10 | D10 | HOST_D10 | |
| HOST_D9 | I/O | D9 | D9 | D9 | HOST_D9 | |
| HOST_D8 | I/O | D8 | D8 | D8 | HOST_D8 | |
| HOST_D7 | I/O | D7 | D7 | D7 | HOST_D7 | |
| HOST_D6 | I/O | D6 | D6 | D6 | HOST_D6 | |
| HOST_D5 | I/O | D5 | D5 | D5 | HOST_D5 | |
| HOST_D4 | I/O | D4 | D4 | D4 | HOST_D4 | |
| HOST_D3 | I/O | D3 | D3 | D3 | HOST_D3 | |
| HOST_D2 | I/O | D2 | D2 | D2 | HOST_D2 | |
| HOST_D1 | I/O | D1 | D1 | D1 | HOST_D1 | SH_MOSI |
| HOST_D0 | I/O | D0 | D0 | D0 | HOST_D0 | SH_MISO |
| HOST_D_EN | I/O | HOST_D_EN | FLASH_OEn | CD_DOEn | HOST_D_EN | SH_SOEN |
| HOST_REn | I/O | HOST_REn | FLASH_RE | CF_OE | HOST_REn | |
| HOST_WEn | I/O | HOST_WEn | FLASH_WE | CF_WE | HOST_WEn | SH_MCLK |
| HOST_WAITn | I/O | HOST_WAITn | FLASH_RB | | HOST_WAITn | |
| HOST_DMARQ | I/O | HOST_DMARQ | | | HOST_DMARQ | SH_DMARQ |
| HOST_ALEn | O | HOST_ALEn | FLASH_ALE | | | |
| HOST_INTn | O | HOST_INTn | | | HOST_INTn | SH_INT |
| HOST_WPn | O | | FLASH_WP | | | |
| CF_INPACKn | I | | | CF_INPACKn | | |
| CF_IORDn | O | | | CF_IORDn | | |
| CF_IOWRn | O | | | CF_IOWRn | | |
| CF_BVD1 | I | | | CF_BVD1 | | |
| CF_BVD2 | I | | | CF_BVD2 | | |
| CF_CD1 | I | | | CF_CD1 | | |
| CF_CD2 | I | | | CF_CD2 | | |
| CF_RESET | O | | | CF_RESET | | |
| CF_WP | I | | | CF_WP | | |
| CF_WAITn | I | | | CF_WAITn | | |
| CF_REGn | O | | | CF_REGn | | |

The signal that is output on each of the six Chip Select lines (HOST_CS0 and HOST_CS[5:1]) is selected using the HOSTChipSelect register. See “HOSTChipSelect Register” on page 151.

***HOSTChipSelect Register***

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | CycleDelay | | | | | HOST_CS5 | | | | HOST_CS4 | | | |
| **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| HOST_CS3 | | | HOST_CS2 | | | | | HOST_CS1 | | | | HOST_CS0 | | | |

Reserved fields should be ignored (masked) when read, and only 0's should be written to them.

| | |
|---|---|
| CycleDelay | At the end of each transaction, CycleDelay+1 idle cycles are generated to allow any external devices to stop driving the bus. Default=0 |
| HOST_CS5<br>HOST_CS4<br>HOST_CS3<br>HOST_CS2<br>HOST_CS1<br>HOST_CS0 | This field selects the source for each of the CSn pins from:<br>0000: nor_ce[0]<br>0001: nor_ce[1]<br>0010: nor_bt_ce[0]<br>0011: nor_bt_ce[1]<br>0100: nand_ce[0]<br>0101: nand_ce[1]<br>0110: nand_bt_ce[0]<br>0111: nand_bt_ce[1]<br>1000: cf_ce[0]<br>1001: cf_ce[1]<br>1010: MHIF_CS[0]<br>1011: Reserved<br>1100: If master HOST_A[22] else FLASH_A[22]<br>1101: If master HOST_A[23] else FLASH_A[23]<br>1110: Reserved<br>1111: Default (Sets output high) |

## 5.12 Memory Interfaces

### *5.12.1 SDRAM Requirements*

The MG3500 HD H.264 Codec SoC is intended for portable CE products such as DVD Camcorders and solid-state based digital cameras. It is designed to use a single external SDRAM memory wherever possible. The MG3500 HD H.264 Codec SoC requires one SDRAM for SD and HD or two for HD only.

While the MG3500 HD H.264 Codec SoC typically functions with a single SDRAM, there are cases in which two SDRAMs can lead to lower cost, better quality, and added features. This is because finding 32-bit wide SDRAMs that support greater that 166 Mhz can be extremely difficult. While DDR2 SDRAMs in 16-bit wide configurations are readily available in 200 MHz or greater, it is NOT expected that x32 DDR2 parts become available.

The specifications for the SDRAMs are:

- Types supported: DDR2
- Speed Grades: 133, 166, 200, 233, 248, and 264 MHz
- Bus widths: 16 (SD only) and 32 (SD and HD)
- Voltage levels DDR SDRAM: 1.8V (Supported by DDR2 parts)

Table 5-11 shows some of the SDRAM configurations that support HD video. Table 5-12 shows the amount of SDRAM memory needed for various applications.

**Table 5-11 HD SDRAM Configurations**

| Configuration | Notes |
|---|---|
| 32 bits, 166 MHz, One part | Minimum configuration for HD scenarios, most likely no MBAFF and two reference frames. |
| 32 bits, 200 MHz, One part | NO MBAFF and three reference frames. |
| 32 bits, 200 MHz, Two x16 parts | NO MBAFF and three reference frames, DDR2 |
| 32 bits, ≥233 MHz, Two x16 parts | MBAFF and up to three reference frames, DDR2 |
| 32 bits, 266 MHz, Two x16 parts | MBAFF, three reference frames, DDR2 |

**Table 5-12 SDRAM Requirements for Various Profiles**

| Configuration | Notes |
|---|---|
| 8 or 16 Mbytes | SD Coprocessor, 8 MBytes for Baseline Profile, 8 or 16 MBytes for Main Profile |
| 8 or 16 Mbytes | SD SoC, x16 or x32 DDR |
| 64 Mbytes | HD Coprocessor Applications |
| 64 Mbytes | HD SoC Applications [1] |

1. The MG3500 SoC requires approximately 50 MBytes for its internal operations. The remainder of the SDRAM is available for SoC operations. The actual amount could exceed 64 MBytes total, depending on the needs of your application.

### *5.12.2 SDRAM Connections*

This section shows the connections between the MG3500 HD H.264 Codec SoC and the SDRAMs in two configurations:

Figure 5-17 shows the SDRAM Connections for a single 16-bit SDRAM

Figure 5-18 shows the SDRAM Connections for two 16-bit wide SDRAMS

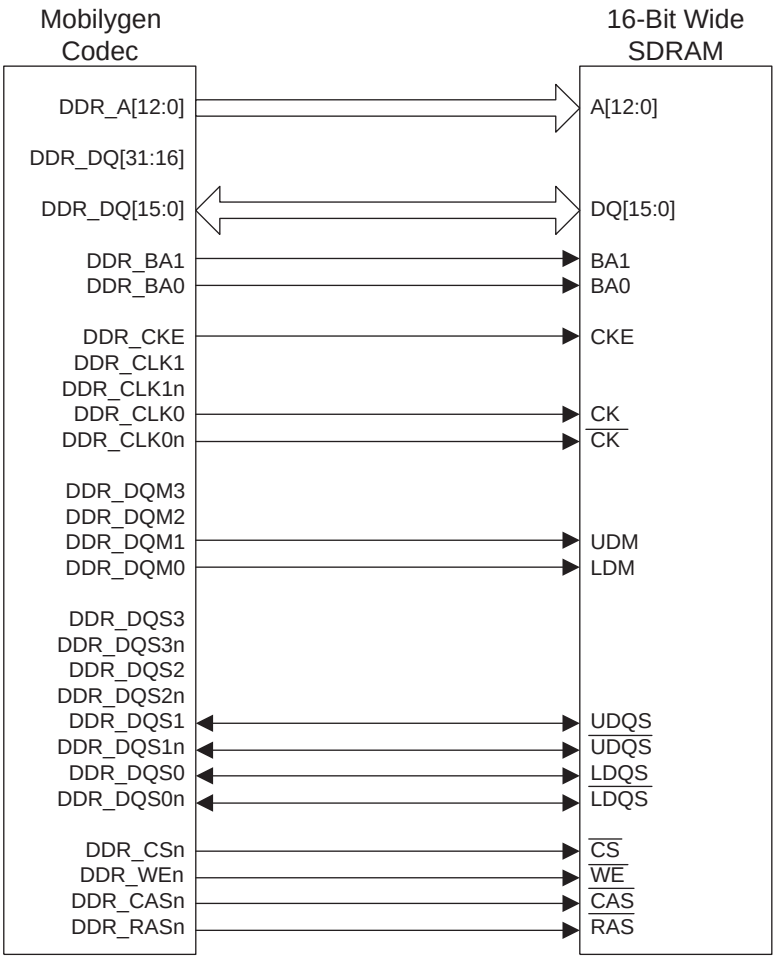



**Figure 5-17 SDRAM Connections for a Single 16-bit SDRAM**

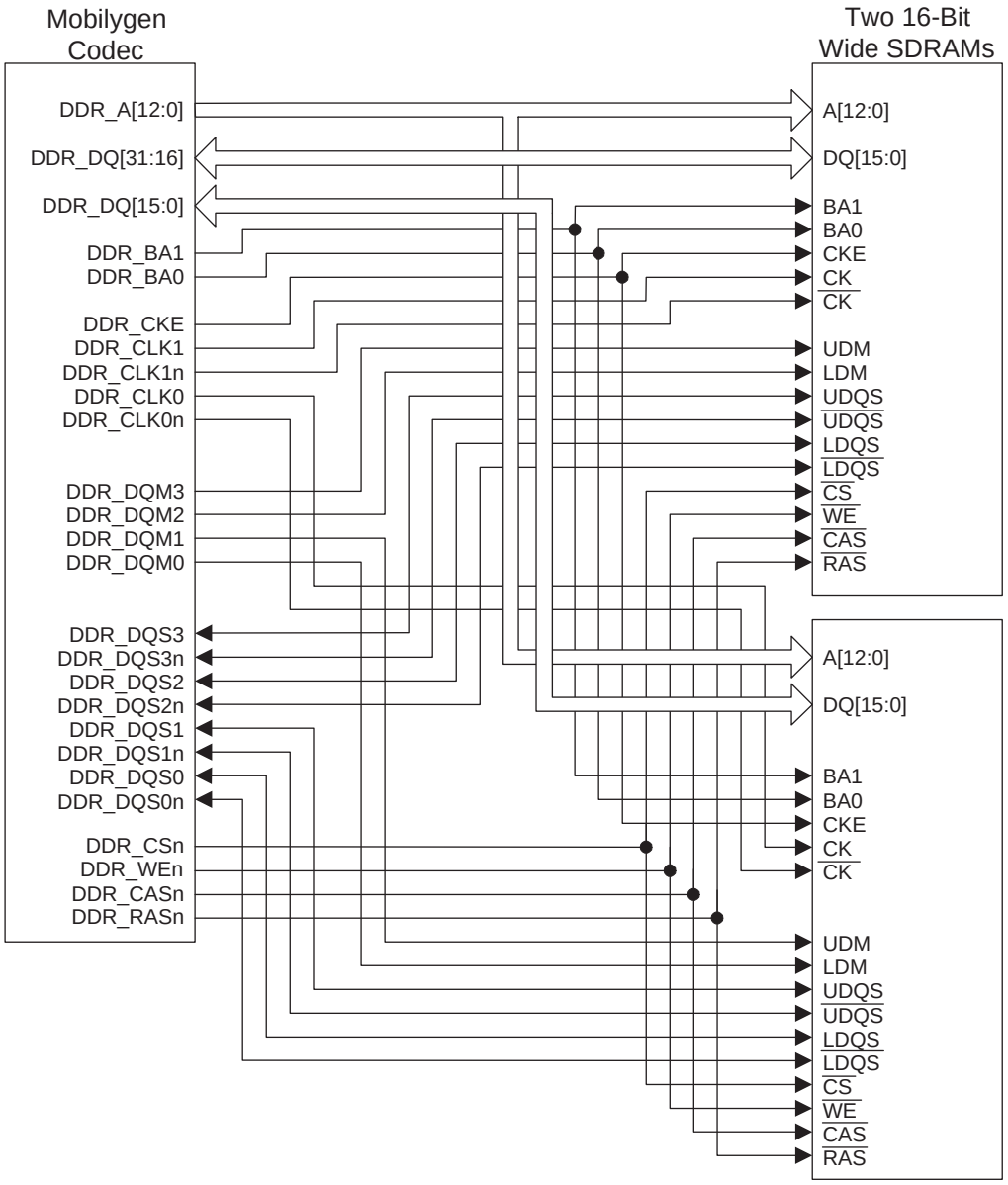



**Figure 5-18 SDRAM Connections for Two 16-bit Wide SDRAMS**

### *5.12.3 Mobile Storage Controller: SD, SDIO, MMC, CE-ATA*

The MG3500 SoC contains a 4-bit wide Mobile Storage controller that controls:

- Secure Digital memory (SD mem – version 1.10, DRAFT June 17, 2004)
- Secure Digital I/O (SDIO – version 1.10)
- Consumer Electronics Advanced Transport Architecture (CE-ATA – version 1.0)
- Multimedia Cards (MMC – version 4.0)

This section provides a basic overview of the Mobile Storage controller. It has the following features:

- Supports Secure Digital memory protocol commands
- Supports Secure Digital I/O protocol commands
- Supports Multimedia Card protocol commands
- Supports CE-ATA digital protocol commands
- Supports Command Completion signal and interrupt to host processor
- Command Completion Signal disable feature

#### Bus Interface Features

- Does not generate split, retry, or error responses
- 16-word deep FIFO and controller
- Supports FIFO over-run and under-run prevention by stopping card clock

#### Card Interface Features

- Can be configured as MMC-Ver3.3-only controller or SD_MMC controller
- Supports one card in MMC-Ver3.3-only mode, and one SD or MMC (3.3 or 4.0) or CE-ATA device in SD_MMC_CE-ATA mode
- Supports Command Completion Signal and interrupts to host
- Supports Command Completion Signal disable
- Supports CRC generation and error detection
- Supports programmable baud rate. Supports up to 4 clock dividers to support simultaneous operation of multiple cards with different clock speed requirements
- Provides individual clock control to selectively turn ON or OFF clock to a card
- Supports power management and power switch. Provides individual power control to selectively turn ON or OFF power to a car
- Supports host pull-up control
- Supports card detection and initialization
- Supports write protection
- Supports SDIO interrupts in 1-bit and 4-bit modes
- Supports SDIO suspend and resume operation
- Supports SDIO read wait
- Supports block sizes of 1 to 65,535 bytes

#### Standards Compliance

The Mobile Storage controller conforms to the AMBA Specification, Revision 2.0 from ARM. Readers are assumed to be familiar with this specification.

## Description

The Mobile Storage controller can be configured either as a Multimedia Card-only controller or as a Secure Digital Multimedia Card controller that simultaneously supports Secure Digital memory (SD Mem), Secure Digital I/O (SDIO), Multimedia Cards (MMC), and Consumer Electronics Advanced Transport Architecture (CE-ATA). One main difference between MMC-Ver3.3-only mode and SD_MMC_CE-ATA mode is the bus topology. In MMC-Ver3.3-only mode, the MMC cards are connected in a single shared-bus topology, illustrated in Figure 5-19.

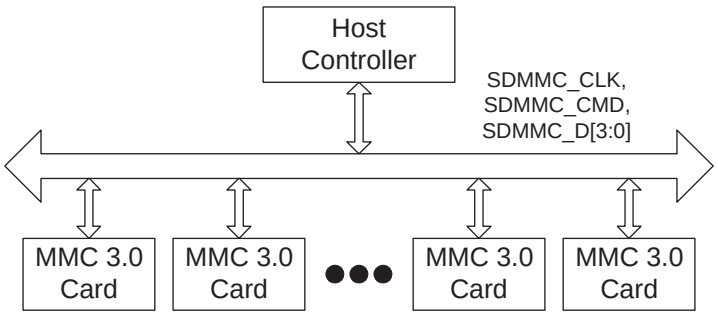



**Figure 5-19 Multimedia Ver3.3 Card System - Bus Topology**

In SD_MMC_CE-ATA mode, the memory cards are connected in a star topology, illustrated in Figure 5-20.

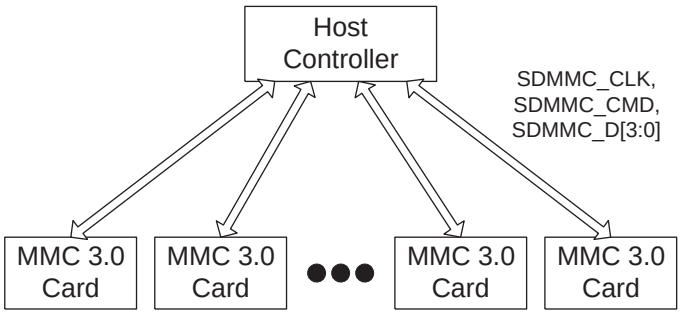



**Figure 5-20 SD_MMC and CE-ATA Card System Star Topology**

In MMC-Ver3.3-only mode, the Mobile Storage controller supports a 1-bit data bus width. In SD_MMC_CE-ATA mode, the Mobile Storage controller supports 1-bit, and 4-bit data bus widths, depending on the card types (SD/SDIO, HSMMC, CE-ATA).

**Note:** You must use an SD card connector that includes the SD_WP and SD_CD signals or you will be limited to 1-bit mode.

An SD_MMC memory card is typically a device for FLASH mass storage. The SDIO card usually functions in I/O applications that can also have optional FLASH memory. The CE-ATA card functions in mobile device applications. The SD_MMC/CE-ATA bus includes the following signals:

- SDMMC_CLK – Host-to-card clock signal
- SDMMC_CMD – Bidirectional command and response signal
- SDMMC_DATA – Bidirectional data signal (1-bit or 4-bit MMC Cards; 1-bit or 4-bit in SD cards)
- VDD, VSS1, VSS2 – Power and ground

Figure 5-21 shows a block diagram of a typical SD card and its and signals.

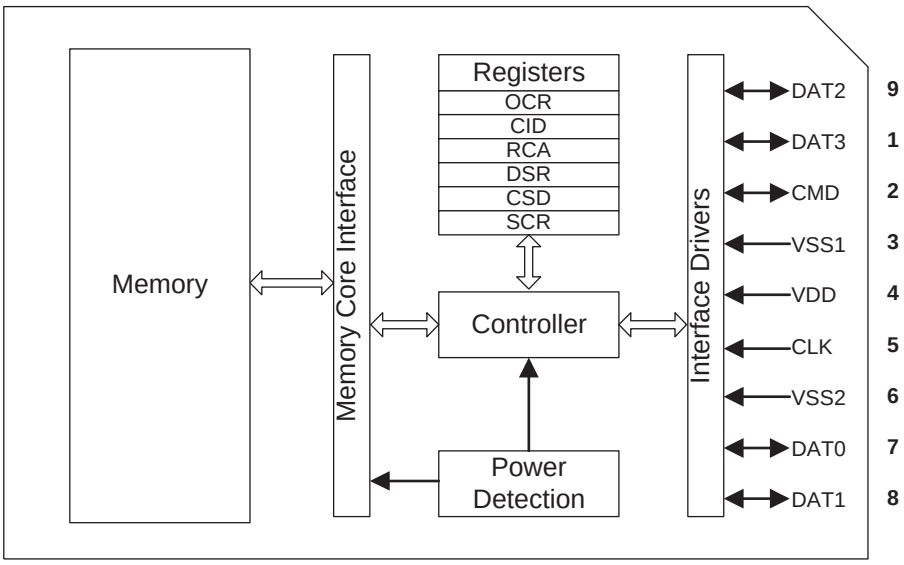



**Figure 5-21 Typical SD Memory Card**

Figure 5-22 shows a block diagram of the MMC 3.3 card and signals.

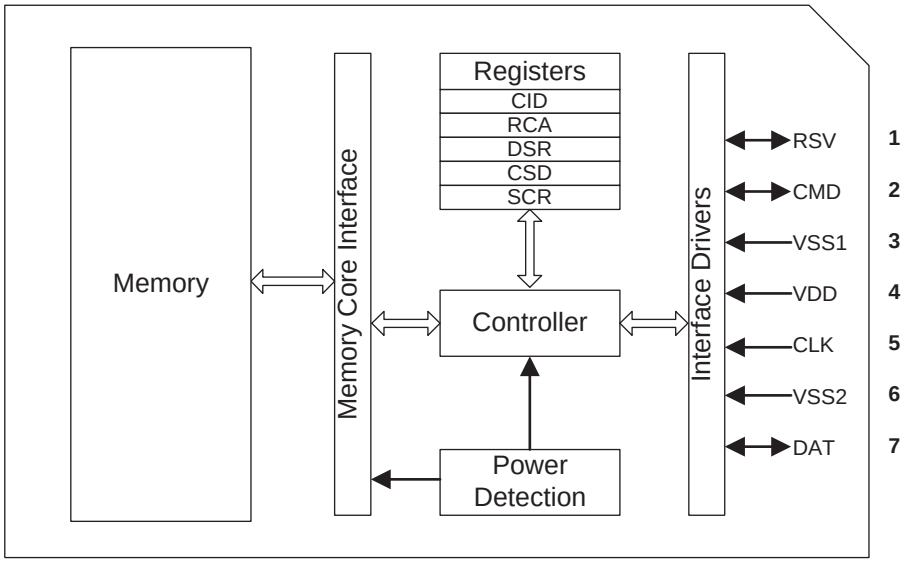



**Figure 5-22 Typical MMC 3.3 Card**

Figure 5-23 shows the block diagram of a high-speed MMC (HSMMC) card and signals.

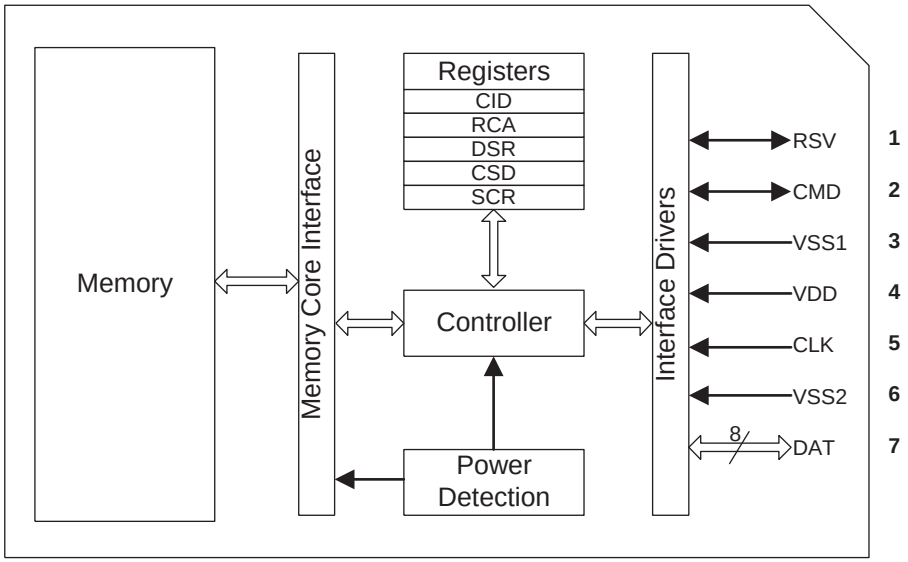



**Figure 5-23 Typical HSMMC Card**

The SD_MMC/CE-ATA protocol is based on command and data bit streams that are initiated by a start bit and terminated by a stop bit. Additionally, the controller provides a reference clock and is the only master that can initiate a transaction.

- Command – A token, sent serially on the CMD line, that starts an operation.
- Response – A token that is sent from an addressed card and serially on the CMD line; not all the commands expect a response from the cards.
- Data – Can be transferred from the host to the card or vice versa. Data is transferred serially on the data line; not all commands involve data transfer.

Figure 5-24 illustrates an example multiple-block read operation; the clock is representative only and does not show the exact the number of clock cycles.

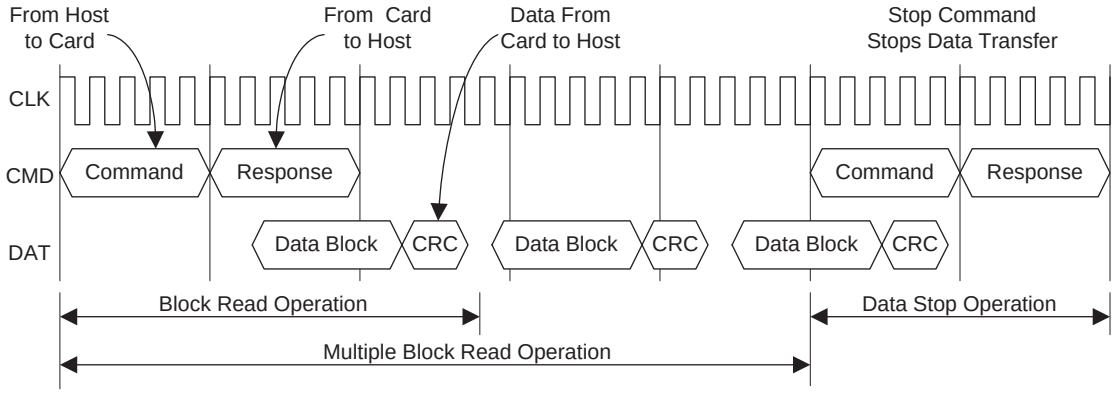



**Figure 5-24 Multiple-Block Read Operation**

Figure 7 illustrates an example multiple-block write operation; again, the clock is representative only and does not show the exact number of clock cycles.

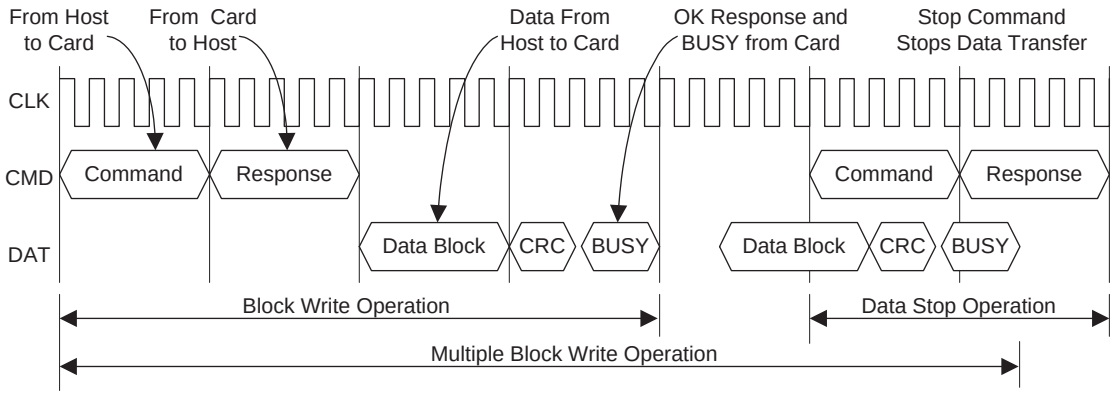



**Figure 5-25 Multiple-Block Write Operation**

Figure 5-26 shows an example command token sent by the host.

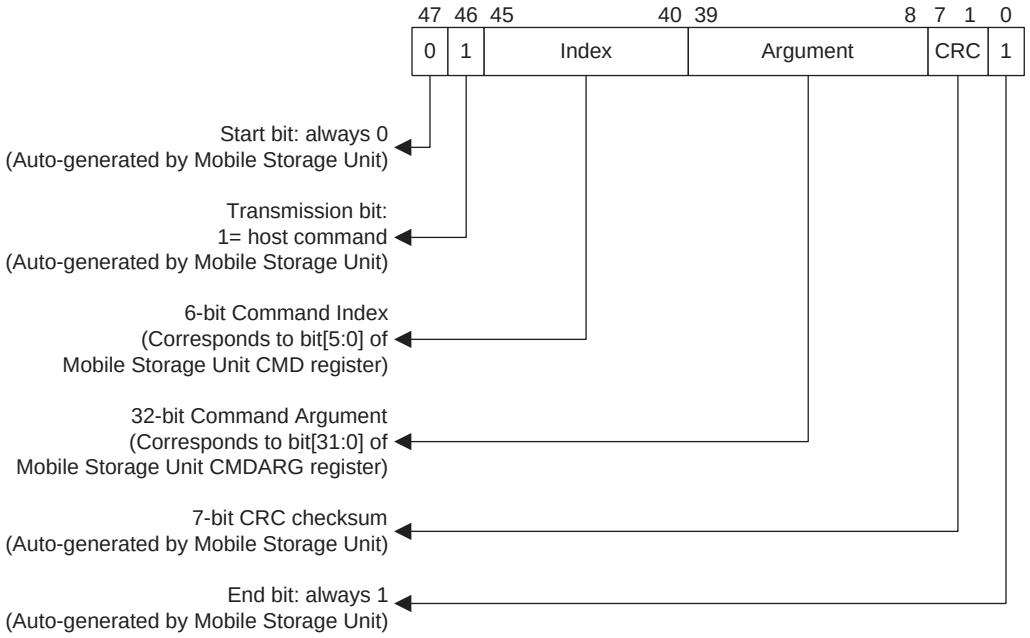



**Figure 5-26 Example Command Token**

Figure 5-27 shows an example Short Response from a card.

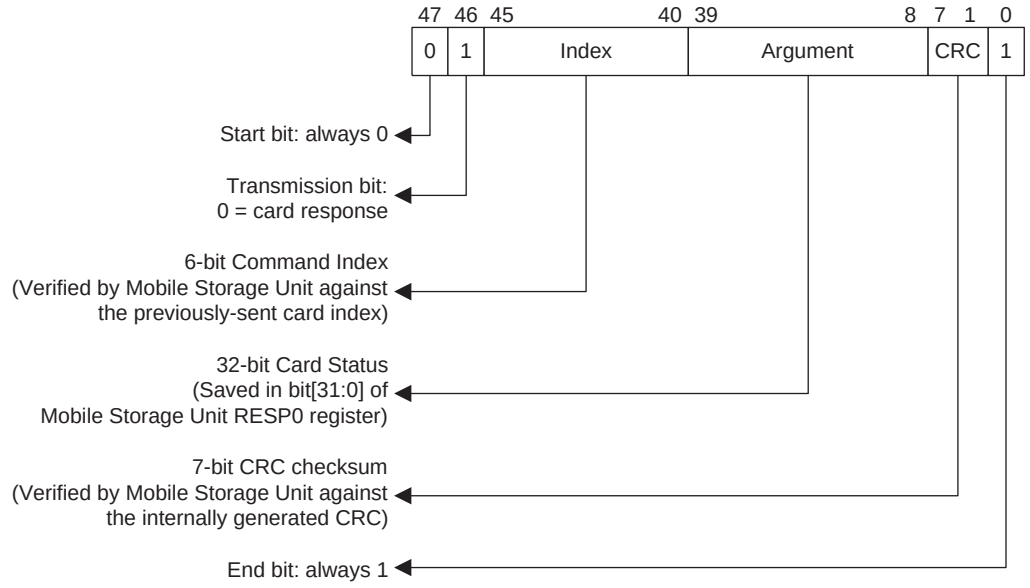



**Figure 5-27 Example Short Response from Card**

Figure 5-28 shows the block diagram for the Mobile Storage controller.

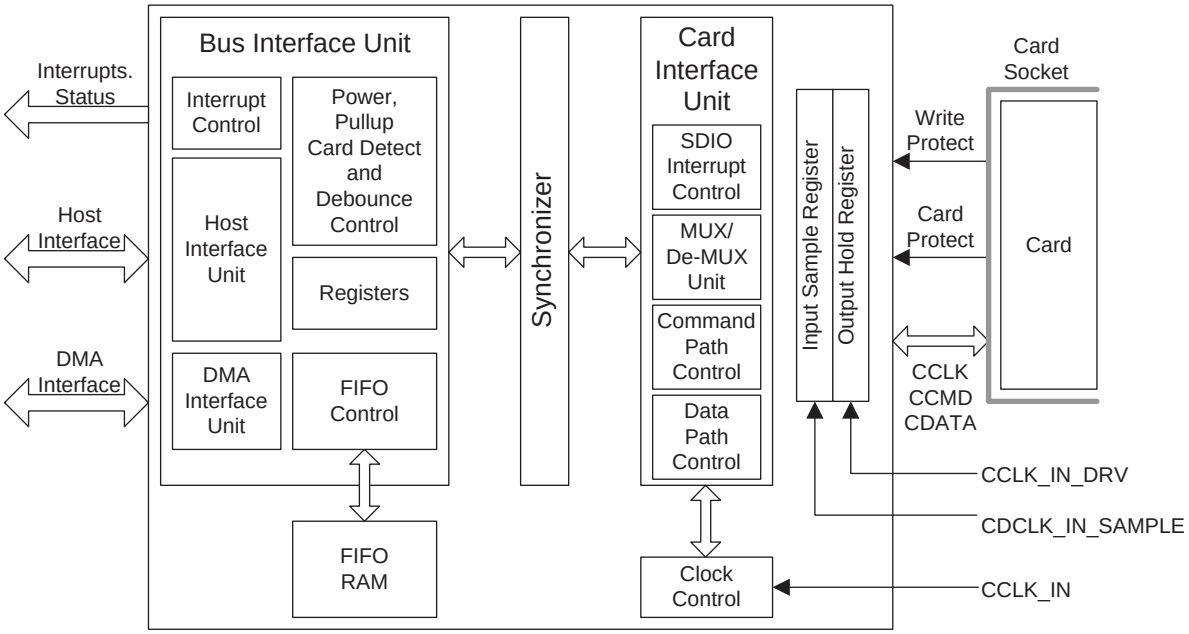



**Figure 5-28 Mobile Storage Controller Block Diagram**

### 5.12.4 Compact FLASH, IDE

The Compact Flash Controller includes a versatile Compact Flash card controller that supports CompactFlash and CF+ according to the *CF+ and CompactFlash Specification Revision 3.0*. It functions in PC card memory, PC card I/O, and true IDE modes. IDE multiword DMA and Ultra DMA are also included.

The Compact Flash Controller data bus width to the Compact Flash card, is user configurable to 8 or 16 bits. Compact Flash card timing parameters are software programmable to support a wide range of Compact Flash card speed grades and system clock frequencies.

The Compact Flash Controller supports all Compact Flash card commands listed in the *CF+ and CompactFlash Specification Revision 3.0*. In PC card modes, the host processor accesses the Compact Flash card common memory, attribute memory, and I/O space using the Compact Flash Controller. Additionally, the host processor can directly read and write Compact Flash card command block locations using the Compact Flash Controller.

The host processor initiates Compact Flash card operations by writing commands to the Compact Flash Controller. When a Compact Flash card operation completes, the Compact Flash Controller optionally signals a maskable interrupt to the host processor. The host processor may also poll Compact Flash Controller registers to determine when a Compact Flash card operation has completed.

Data is transferred to and from the Compact Flash card by way of a Write Data FIFO and a Read Data FIFO. For Compact Flash card commands that write data to the Compact Flash card, the write data is loaded into the Write Data FIFO. Data is read from the Read Data FIFO when executing Compact Flash card commands that read data from the Compact Flash card.

The Compact Flash Controller AMBA Subsystem features are:

- CompactFlash and CF+ version 3.0
- PC card memory, PC card I/O, and true IDE modes
- IDE multiword DMA and Ultra DMA modes
- 8-bit or 16-bit Compact Flash Controller data bus
- User configurable reset values and fully programmable Compact Flash card timing parameters
- Supports all Compact Flash card commands
- Supports direct command block accesses
- Supports common memory, attribute memory, and IO accesses (PC card modes)
- Interrupt or host processor polling for Compact Flash command completion
- Two FIFOs to transfer data to/from the Compact Flash card- user configurable sizes
  - Write Data FIFO
  - Read Data FIFO
- Read data prefetching
- Write data packing
- Same cycle device request/response is supported for highest throughput
- Handles all data packing/unpacking and data alignment for data transfer sizes that do not match the AMBA Bus width and/or Compact Flash data bus width

Figure 5-29 shows the connections between the MG3500 SoC and the Compact Flash socket.

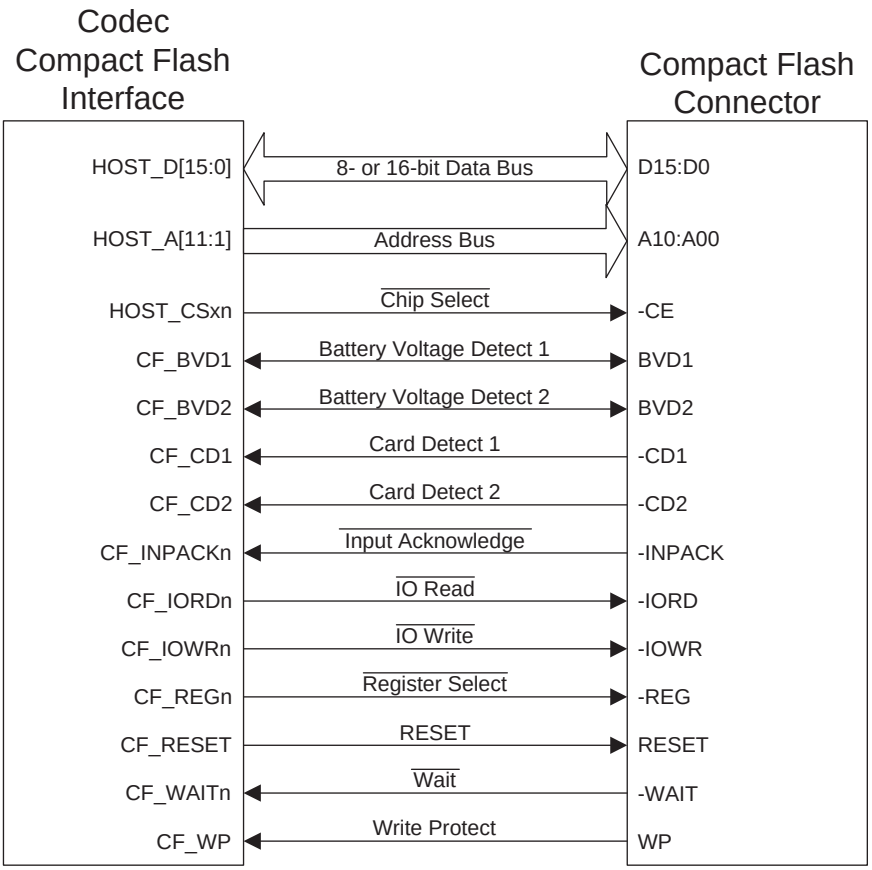



**Figure 5-29 Compact Flash Interface Connections**

### *5.12.5 NAND and NOR Flash Controller*

The Flash Controller Subsystem includes a versatile NAND/NOR Flash controller that supports NAND and NOR Flash chips from several manufacturers. The Flash Controller data bus width is user configurable to 8, 16, or 32 bits. The Flash Controller supports NAND Flash memory systems from 8 Mbytes to 16 GBytes, whereas NOR Flash memory systems from 512 kBytes to 1 GByte. Flash chip timing parameters are software programmable to support a wide range of Flash speed grades and system clock frequencies.

The Flash Controller supports the Read, Program, Erase, Read Status, Read ID, Copy Back, and Reset NAND Flash commands. ECC is provided for NAND Flash systems. NOR Flash commands that are supported include Read, Program, Erase, Read Status, Read ID, Read CFI, Clear Status, Buffered Write, Lock, Unlock, and Lock Down. Additionally, a Direct Read and Direct Write command are provided to access NOR Flash chips at arbitrary addresses and with arbitrary write data so that any other NOR Flash chip command sequences can be used.

In NAND flash mode, the maximum address size is 34 bits. This NAND flash mode address is a concatenation of the address field of the NAND Flash Control Register and either the AMBA bus address or the address field of the NAND Flash Command Register. Specifically:

**Table 5-13 Address Formation**

| Address | NAND Flash Operations |
|---|---|
| Address = A[7:0] of the NAND Flash Control Register,<br>AMBA bus address[25:0] | Read<br>Program (write) |
| Address = A[7:6] of the NAND Flash Control Register,<br>Address[27:0] of the NAND Flash Command Register, 4 x 0 | Read, Status Read ID,<br>Erase, Copy Back |

Large NAND flash systems require more address bits than are available from the bus address or the bits in a single 32 bit register. Therefore, the most significant address bits of these large systems are determined by the contents of the address field of the NAND Flash Control Register, as shown in Table 5-13. The address field of the NAND Flash Control Register is ignored in smaller NAND flash systems.

In NAND Flash systems, configurable features include page size, data bus width, Flash chip size, number of Flash banks, interrupt enable, ECC functionality, copy back functionality, command confirmation, number of address cycles, number of ID read cycles, spare area usage, and Address[33:32]. Configurable features for NOR Flash systems include, data bus width, Flash chip size, number of Flash banks, interrupt enable, block size, boot block configuration, lock feature, burst read feature, buffered write feature, and the CFI feature. Configurable features can be reconfigured by software.

Flash timing parameters are hardwired to default values at reset. After reset, they are software programmable. This allows the Flash Controller to be used for boot code at reset with a wide range of Flash speed grades and system cycle times. After booting, performance can be optimized by reconfiguring the Flash timing parameters for the specific Flash memories that are used and the system clock frequency.

The host processor initiates Flash chip operations that do not read or write data, by writing commands to the Flash Controller. When one of these non-data Flash operations completes, the Flash Controller optionally signals a maskable interrupt to the host processor. The host processor may also poll Flash Controller registers to determine when a non-data Flash chip operation has completed.

Internal to the MG3500 SoC, the Flash Controller Subsystem is a bus slave peripheral on the AMBA High-Speed Bus (AHB). Bus read and write transactions that target the Flash chips, are recognized by the AMBA Slave Interface of the Flash Controller Subsystem. The Slave Interface initiates Flash requests at the requester interface of the Flash Controller block. To complete the Bus transaction, the Slave Interface drives the appropriate response onto the AMBA High-Speed Bus (AHB).

The Flash Controller Subsystem features are:

- NAND and NOR Flash controller
- 1, 2, or 4 banks of Flash memories
- 8 bit or 16 bit Flash Controller data bus
- 8 Mbyte to 16 GByte NAND Flash memory systems
- 512 kByte to 1 GByte NOR Flash memory systems
- Default reset values with fully programmable Flash chip timing parameters
- NAND Flash
  - Read, Program, Erase, Read Status, Read ID, Copy Back, and Reset
  - 64 Mbit to 8 Gbit Flash chips- configurable
  - 8 bit or 16 bit Flash chip data bus- configurable
  - 2048 byte page size
  - ECC hardware
  - 2 or 4 cycle ID register read- configurable
- NOR Flash
  - Read, Program, Erase, Read Status, Read ID, Read CFI, Clear Status, Buffered Write, Lock, Unlock, Lock Down
  - Other NOR Flash commands using Direct Read and Direct Write
  - NOR Flash RP/RST/RESET (reset) assertion by the Flash Controller Subsystem reset input port
  - 4 Mbit to 512 Mbit Flash chips- configurable
  - 8 bit or 16 bit Flash chip data bus- configurable
  - 64 kByte or 128 kByte main block size- configurable
  - top, or no boot block- configurable
  - 8 kByte, 16 kByte, or 32 kByte boot block size- configurable
- Interrupt or host processor polling for non-data transfer Flash command completion
- Implements Bus timeout and RETRY response
- Read data prefetching
- Write data packing
- Same cycle device request/response is supported for highest throughput
- Handles all data packing/unpacking and data alignment for data transfer sizes that do not match the AMBA Bus width and/or Flash data bus width

Figure 5-30 shows the NAND/NOR Flash Interface connected to a NOR memory.

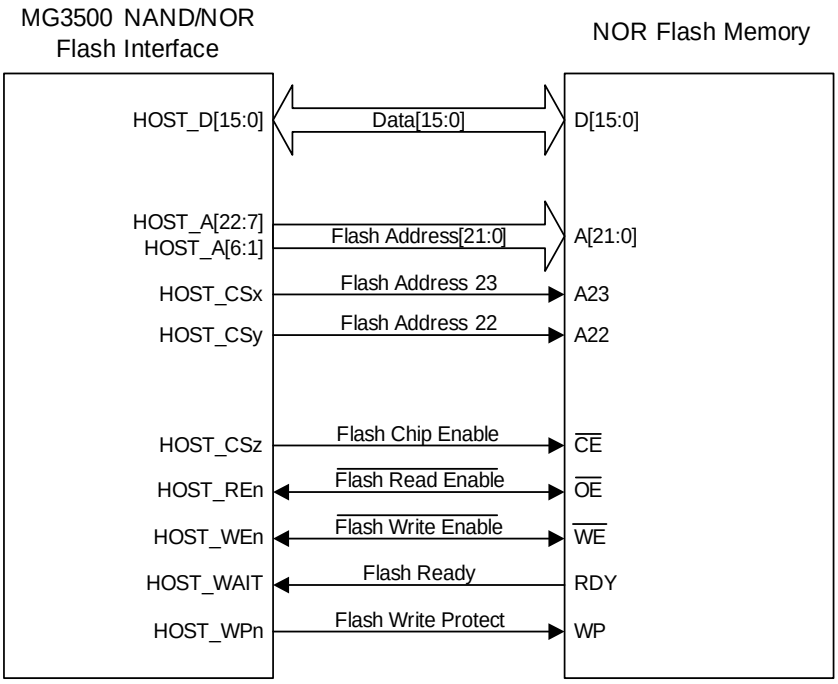



**Figure 5-30 NAND/NOR Flash Interface Connected to a NOR Flash Memory**

## 5.13 Serial Interfaces

The Serial I/O consists of UARTs, $I^2C$-compatible TWI Interfaces, and SPI interfaces.

**Table 5-14 Serial I/O Interfaces**

| Interface | Number of Internal Instances | Number of Interface Ports |
|---|---|---|
| UART | 3 ARM + 1 MME | 3 |
| TWI | 2 | 2 |
| SPI | 3 (1 Master with 2 SS, 1 Master with 1 SS, 1 Slave) | 2 |

### *5.13.1 UART*

The MG3500 SoC has three UART ports: Debug (UARTD), UART 0 (UART0), and UART 1 (UART1). The DBG UART port is shared by the MME and the ARM debugger ports, and consists of only the TXD and RXD signals (see Figure 5-31). The Debug port is very useful in debugging the system and should always be connected.

UART 0 has the TXD and RXD signals, but also includes the flow control signals, RTS and CTS. UART 1 only includes the TXD and RXD signals.

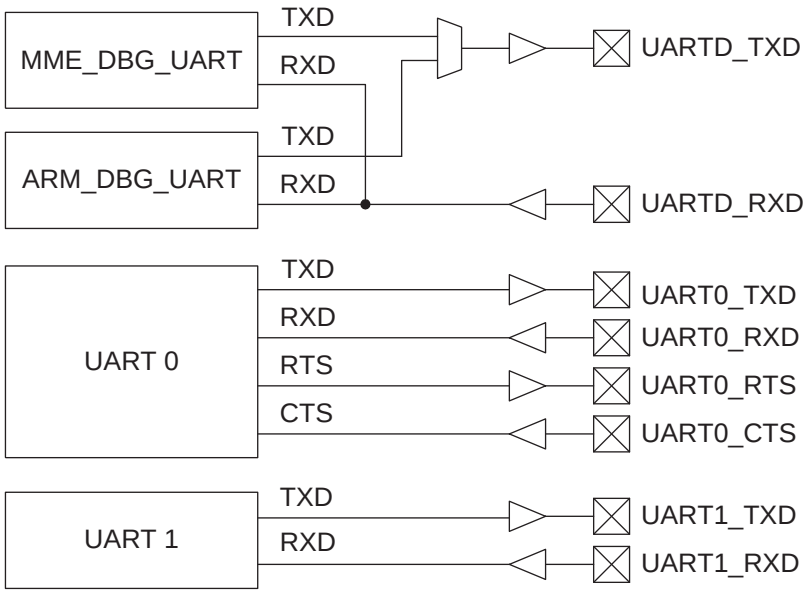



**Figure 5-31 UART Module to Interface Signal Mapping**

The baud rate for the UART can be set to any of the values listed in Table 5-6.

**Table 5-15 UART Baud Rate Frequencies**

| Baud Rate | Baud Rate | Baud Rate | Baud Rate |
|---|---|---|---|
| 300 | 19200 | 64000 | 250000 |
| 600 | 28800 | 76800 | 256000 |
| 1200 | 38400 | 115200 | 460800 |
| 2400 | 51200 | 128000 | 500000 |
| 4800 | 56000 | 153600 | 576000 |
| 9600 | 57600 | 230400 | 921600 |

### *5.13.2 MG3500 Two Wire Interface (TWI)*

There are two $I^2C$-compatible TWI modules and three TWI Interface ports on MG3500. Both TWI modules can be configured to hook up to any of the three TWI interfaces. The V01 and V23 TWI ports are shared with V01 and V23 SPI ports.

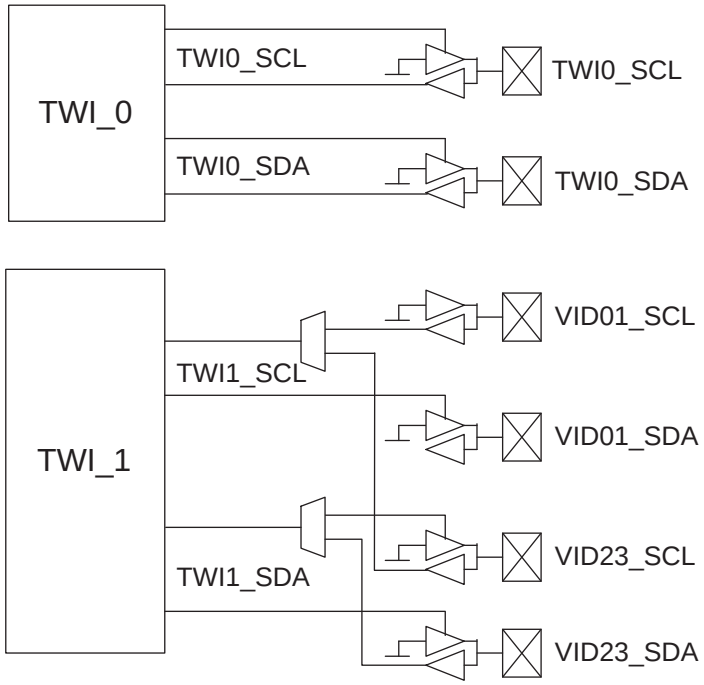



**Figure 5-32 TWI Module to Interface Signal Mapping on MG3500**

### *5.13.3 MG2580 Two Wire Interface (TWI)*

There are two $I^2C$-compatible TWI modules and two TWI Interface ports on MG2580.

**Note:** VID23_SCL and VID23_SDA are not available on MG2580. See Table 2-4 for more information about MG2580 pin descriptions.

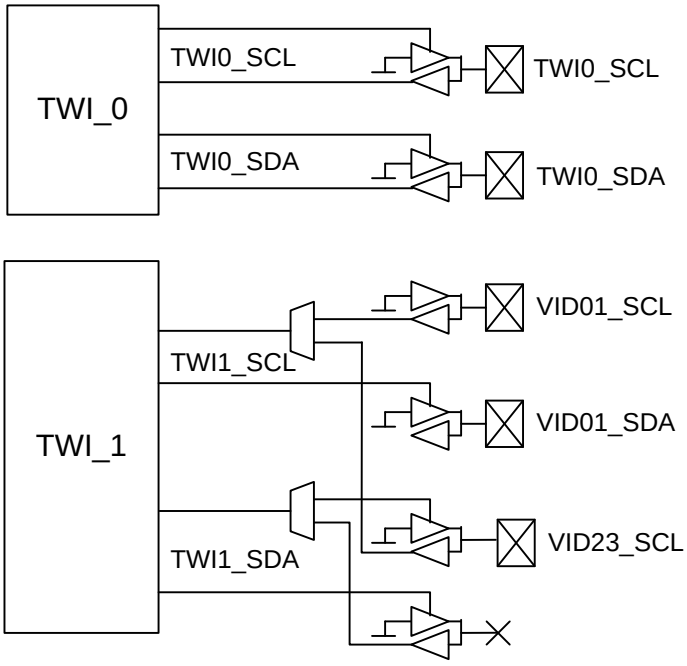



**Figure 5-33 TWI Module to Interface Signal Mapping on MG2580**

### *5.13.4 SPI on MG3500*

There are three SPI modules and three SPI Interface ports (see Figure 5-34). The following are SPI Interface ports:

- A V01 SPI port associated with the Video 0 and 1 ports
- A V23 SPI port associated with Video 2 and 3 ports
- An independent SPI port

**Note:** The V23 SPI port is not available on MG2580. See Table 2-4 for more information about MG2580 pin descriptions.

The SPI modules are the following:

- SPI_0 is configured to support two devices and therefore has two Slave Select (MSS) signals. It connects only to the primary SPI port.
- SPI_1 is configured with two Slave Selects. One Slave Select goes to the V01 port and the other goes to the V23 port. The CLK and MOSI output signals branch to both V01 and V23. The MISO line is multiplexed on the input and controlled by SS0. If SS0 is active (Low), then MISO is routed from V01_MISO; otherwise MISO is routed from V23_MISO.
- SPI_2 is configured to support one device and therefore has a single Slave Select (MSS) signal. It connects only to the primary SPI port.

SPI0 and SPI2 can be multiplexed because SPI_0 is a master mode SPI and SPI_2 is a slave mode SPI. After reset this interface comes up in GPIO mode so all signals are inputs.

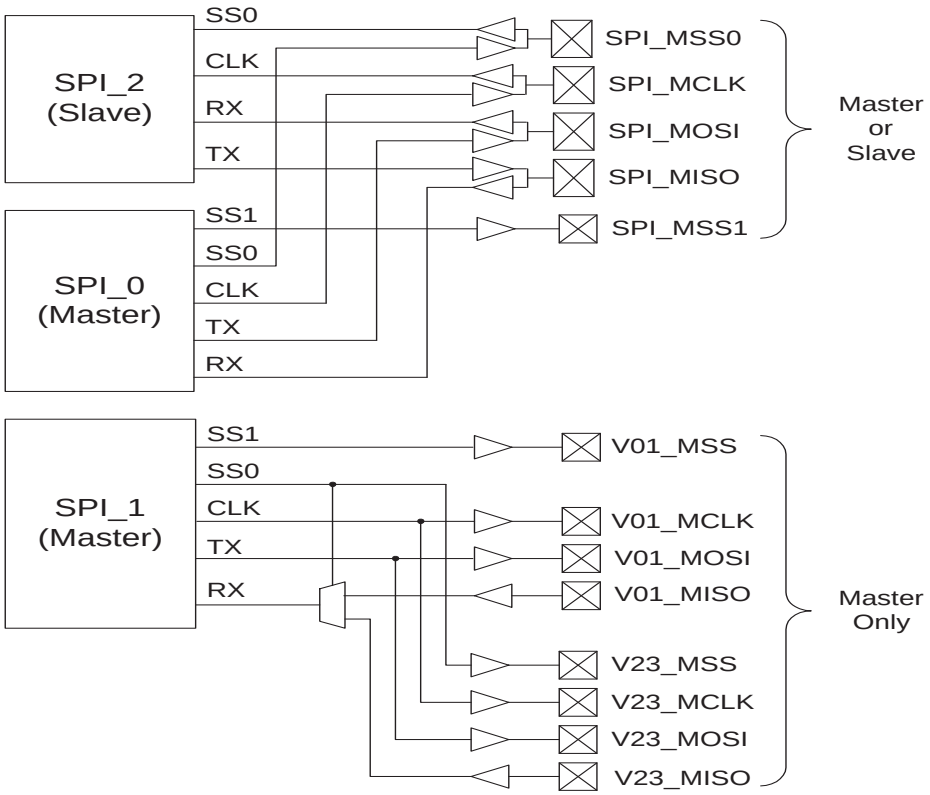



**Figure 5-34 SPI Module to Interface Signal Mapping**

### *5.13.5 SPI on MG2580*

There are three SPI modules and two SPI Interface ports (see Figure 5-36). The SPI Interface ports are the following:

- A V01 SPI port associated with the Video 0 and 1 ports
- An independent SPI port

**Note:** The V23 SPI port is not available on MG2580. See Table 2-4 for more information about MG2580 pin descriptions.

The SPI modules are the following:

- SPI_0 is configured to support two devices and therefore has two Slave Select (MSS) signals. It connects only to the primary SPI port.
- SPI_1 is configured with two Slave Selects. One Slave Select goes to the V01 port and the other is not connected. The MISO line is multiplexed on the input and controlled by SS0. SS0 should always be active (Low).
- SPI_2 is configured to support one device and therefore has a single Slave Select (MSS) signal. It connects only to the primary SPI port.

SPI0 and SPI2 can be multiplexed because SPI_0 is a master mode SPI and SPI_2 is a slave mode SPI. After reset this interface comes up in GPIO mode so all signals are inputs.

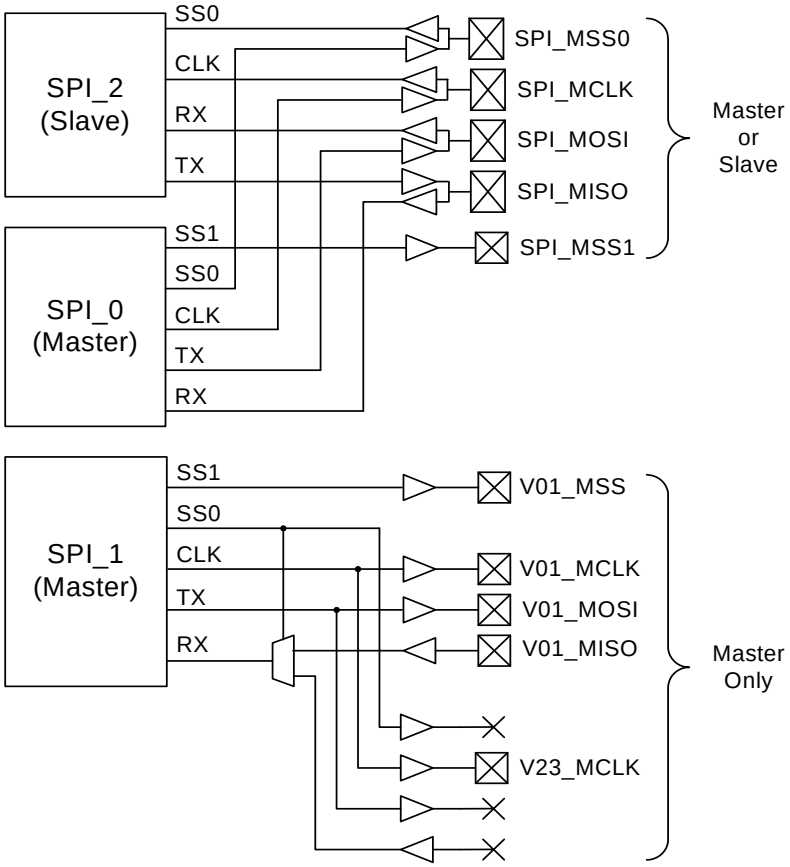



**Figure 5-35 SPI Module to Interface Signal Mapping on MG2580**

### 5.13.6 Pulse Width Modulators

The MG3500 SoC includes three PWM modules. The clock for these modules is 1 MHz, and you program a divider to set the PWM pulse width.

### 5.13.7 Serial IO Pad Programmable Features

In addition to defining a register to enable the GPIO functionality, the following features are programmable:

- Drive Strength (group controlled, see groups in “Drive Strength Encoding” on page 172)
- Slew Rate (group controlled)
- Pull Up (individually controlled)
- Pull Down (individually controlled)

The Drive Strength Groups in the MG3500 HD H.264 Codec SoC are as follows:

- DS0: Serial
- DS1: SPI
- DS2: V01 SPI
- DS3: V23 SPI
- DS4: Video 0
- DS5: Video 1
- DS6: Video 2
- DS7: Video 3
- DS8: Host
- DS9: Ethernet
- DS10: Audio
- DS11: SD/MMC

A three-bit encoding is used for the actual Drive Strength value. The three-bit encoding is re-encoded as shown in Figure 5-16 before being sent to the I/O cell.

**Table 5-16 Drive Strength Encoding**

| Register Value (dec) | Effective Drive Strength (mA) |
|---|---|
| 1 | 2 |
| 2 | 4 |
| 3 | 6 |
| 4 | 8 |
| 5 | 10 |
| 6 | 12 |

### *5.13.8 Serial Registers*

#### *Serial I/O Control*

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | |
| **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| Reserved | | | | | | SPI-MasterSel | SPI_MSS0_AltSel | SPI_MCLK_AltSel | SPI_MOSI_AltSel | V01_MCLK_AltSel | V01_MOSI_AltSel | V23_MCLK_AltSel | V23_MOSI_AltSel | TWI1Cfg | DB-GUARTSel |

Reserved fields should be ignored (masked) when read, and only 0's should be written to them.

| | |
|---|---|
| SPIMasterSel | 0: SPI interface is the Master (default)<br>1: SPI interface is the Slave. |
| SPI_MSS0_AltSel | 0: SPI_MSS0 signal is active (default)<br>1: BS_ENABLE signal is active<br>GPIOSel takes precedence. |
| SPI_MCLK_AltSel | 0: SPI_MCLK signal is active (default)<br>1: BS_CLK signal is active<br>GPIOSel takes precedence. |
| SPI_MOSI_AltSel | 0: SPI_MOSI signal is active (default)<br>1: BS_DATA signal is active<br>GPIOSel takes precedence. |
| V01_MCLK_AltSel | 0: V01_MCLK signal is active (default)<br>1: V01_SCL signal is active<br>GPIOSel takes precedence. |
| V01_MOSI_AltSel | 0: V01_MOSI signal is active (default)<br>1: V01_SDA signal is active<br>GPIOSel takes precedence. |
| V23_MCLK_AltSel[1] | 0: V23_MCLK signal is active (default)<br>1: V23_SCL signal is active<br>GPIOSel takes precedence. |
| V23_MOSI_AltSel[2] | 0: V23_MOSI signal is active (default)<br>1: V23_SDA signal is active<br>GPIOSel takes precedence. |
| TWI1Cfg | 0: TWI1 connects to the V01_* port<br>1: TWI1 connects to the V23_* port |
| DBGUARTSel | 0: Connects ARM_DBG_UART to the DBG UART port (default)<br>1: Connect MME_DBG_UART to the DBG UART port |

1. V23_MCLK_AltSeL is not available on MG2580.

2. V23_MOSI_AltSeL is not available on MG2580.

There is no GPIO 0 Sel as these are dedicated registers.

***GPIO 1 Sel***

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| S31 GPIO1 Sel | S30G PIO1 Sel | S29G PIO1 Sel | S28G PIO1 Sel | S27G PIO1 Sel | S26G PIO1 Sel | S25G PIO1 Sel | S24G PIO1 Sel | S23 GPIO 1 Sel | S22 GPIO 1 Sel | S21 GPIO 1 Sel | S20 GPIO 1 Sel | S19 GPIO 1 Sel | S18 GPIO 1 Sel | S17 GPIO 1 Sel | S16 GPIO 1 Sel |
| **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| S15G PIO1 Sel | S14G PIO1 Sel | S13G PIO1 Sel | S12G PIO1 Sel | S11G PIO1 Sel | S10G PIO1 Sel | S9 GPIO 1 Sel | S8 GPIO 1 Sel | S7 GPIO 1 Sel | S6 GPIO 1 Sel | S5 GPIO 1 Sel | S4 GPIO 1 Sel | S3 GPIO 1 Sel | S2 GPIO 1 Sel | S1 GPIO 1 Sel | S0 GPIO 1 Sel |

| | |
|---|---|
| Reserved fields should be ignored (masked) when read, and only 0's should be written to them. | |
| S31GPIO1Sel – S0GPIO1Sel | S31 - S0 signals<br>0: Select Primary/Alt Function (default)<br>1: Select GPIO Function |

***GPIO 2 Sel***

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| S31 GPIO2 Sel | S30 GPIO 2Sel | S29 GPIO 2Sel | S28 GPIO 2Sel | S27 GPIO 2Sel | S26 GPIO 2Sel | S25 GPIO 2Sel | S24 GPIO 2Sel | S23 GPIO 2Sel | S22 GPIO 2Sel | S21 GPIO 2Sel | S20 GPIO 2Sel | S19 GPIO 2Sel | S18 GPIO 2Sel | S17 GPIO 2Sel | S16 GPIO 2Sel |
| **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| S15[1] GPIO2 Sel | S14 GPIO 2Sel | S13[2] GPIO 2Sel | S12[3] GPIO 2Sel | S11 GPIO 2Sel | S10 GPIO 2Sel | S9 GPIO 2Sel | S8 GPIO 2Sel | S7 GPIO 2Sel | S6 GPIO 2Sel | S5 GPIO 2Sel | S4 GPIO 2Sel | S3 GPIO 2Sel | S2 GPIO 2Sel | S1 GPIO 2Sel | S0 GPIO 2Sel |

| | |
|---|---|
| Reserved fields should be ignored (masked) when read, and only 0's should be written to them. | |
| S0GPIO2Sel – S15GPIO2Sel<br><br>S21GPIO2Sel – S31GPIO2Se | S31 - S0 signals<br>0: Select Primary/Alt Function (default)<br>1: Select GPIO Function |
| S16GPIO2Sel – S20GPIO2Sel | S31 - S0 signals<br>0: Select Primary/Alt Function<br>1: Select GPIO Function (default)<br>Note: S16GPIO2Sel – S20GPIO2Sel default to GPIO selected which forces the signals to input after reset. |

1. The GPIO_2_15 field will have no effect on MG2580 since GPIO pins are not connected. See Table 2-4 for more information on MG2580 pin descriptions.
2. The GPIO_2_13 field will have no effect on MG2580 since GPIO pins are not connected. See Table 2-4 for more information on MG2580 pin descriptions.
3. The GPIO_2_12 field will have no effect on MG2580 since GPIO pins are not connected. See Table 2-4 for more information on MG2580 pin descriptions.

***Serial I/O Drive Strength ([n=0,3,6,9], 4 Registers Total***

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Rsvd | DS***[n+2]*** Drive Stength | | D***[n+2]***2SlewRate | | | DS***[n+1]*** Drive Stength | | DS***[n+1]*** SlewRate | | | DS***[n]*** SlewRate | | DS***[n]***Drive Stength | | |

Reserved fields should be ignored (masked) when read, and only 0's should be written to them.

| | |
|---|---|
| DS***[n]***SlewRate | 00: Slowest<br>01: (default)<br>10:<br>11: Fastest |
| DS***[n]***DriveStength | In 2 mA increments:<br>1: Slowest<br>2: (default)<br>3:<br>4:<br>5:<br>6: Fastest<br>Else: Invalid |

***GPIO 0 Pull-up Enable***

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | |
| **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| Reserved | | | | | | | | S7 PU En | S6 PU En | S5 PU En | S4 PU En | S3 PU En | S2 PU En | S1 PU En | S0 PU En |

| | |
|---|---|
| Reserved fields should be ignored (masked) when read, and only 0's should be written to them. | |
| S7PUEn – S0SPUEn | 0: Pull-up is NOT enabled<br>1: Pull-up is enabled (default) |

***GPIO 1 Pull-up Enable***

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| S31 PUEn | S30 PUEn | S29 PUEn | S28 PUEn | S27 PUEn | S26 PUEn | S25 PUEn | S24 PUEn | S23 PU En | S22 PU En | S21 PU En | S20 PU En | S19 PU En | S18 PU En | S17 PU En | S16 PU En |
| **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| S15 PU En | S14 PU En | S13 PU En | S12 PU En | S11 PU En | S10 PU En | S9 PU En | S8 PU En | S7 PU En | S6 PU En | S5 PU En | S4 PU En | S3 PU En | S2 PU En | S1 PU En | S0 PU En |

| | |
|---|---|
| Reserved fields should be ignored (masked) when read, and only 0's should be written to them. | |
| S31PUEn – S0SPUEn | 0: Pull-up is NOT enabled<br>1: Pull-up is enabled (default) |

***GPIO 2 Pull-up Enable***

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| S31 PUEn | S30 PUEn | S29 PUEn | S28 PUEn | S27 PUEn | S26 PUEn | S25 PUEn | S24 PUEn | S23 PU En | S22 PU En | S21 PU En | S20 PU En | S19 PU En | S18 PU En | S17 PU En | S16 PU En |
| **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| S15[1] PU En | S14 PU En | S13[2] PU En | S12[3] PU En | S11 PU En | S10 PU En | S9 PU En | S8 PU En | S7 PU En | S6 PU En | S5 PU En | S4 PU En | S3 PU En | S2 PU En | S1 PU En | S0 PU En |

| | |
|---|---|
| Reserved fields should be ignored (masked) when read, and only 0's should be written to them. | |
| S31PUEn – S0SPUEn | 0: Pull-up is NOT enabled<br>1: Pull-up is enabled (default) |

1. The GPIO_2_15 field will have no effect on MG2580 since GPIO pins are not connected. See Table 2-4 for more information on MG2580 pin descriptions.
2. The GPIO_2_13 field will have no effect on MG2580 since GPIO pins are not connected. See Table 2-4 for more information on MG2580 pin descriptions.
3. The GPIO_2_12 field will have no effect on MG2580 since GPIO pins are not connected. See Table 2-4 for more information on MG2580 pin descriptions.

***GPIO 0 Pull-down Enable***

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | |
| **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| Reserved | | | | | | | | S7PDEn | S6PDEn | S5PDEn | S4PDEn | S3PDEn | S2PDEn | S1PDEn | S0PDEn |

| | |
|---|---|
| Reserved fields should be ignored (masked) when read, and only 0's should be written to them. | |
| S7PDEn – S0SPDEn | 0: Pull-down is NOT enabled (default)<br>1: Pull-down is enabled |

***GPIO 1 Pull-down Enable***

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| S31 PDEn | S30 PDEn | S29 PDEn | S28 PDEn | S27 PDEn | S26 PDEn | S25 PDEn | S24 PDEn | S23 PD En | S22 PD En | S21 PD En | S20 PD En | S19 PD En | S18 PD En | S17 PD En | S16 PD En |
| **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| S15 PD En | S14 PD En | S13 PD En | S12 PD En | S11 PD En | S10 PD En | S9 PD En | S8 PD En | S7 PD En | S6 PD En | S5 PD En | S4 PD En | S3 PD En | S2 PD En | S1 PD En | S0 PD En |

| | |
|---|---|
| Reserved fields should be ignored (masked) when read, and only 0's should be written to them. | |
| S31PDEn – S0SPDEn | 0: Pull-down is NOT enabled (default)<br>1: Pull-down is enabled |

***GPIO 2 Pull-down Enable***

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| S31 PDEn | S30 PDEn | S29 PDEn | S28 PDEn | S27 PDEn | S26 PDEn | S25 PDEn | S24 PDEn | S23 PD En | S22 PD En | S21 PD En | S20 PD En | S19 PD En | S18 PD En | S17 PD En | S16 PD En |
| **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| S15[1] PD En | S14 PD En | S13[2] PD En | S12[3] PD En | S11 PD En | S10 PD En | S9 PD En | S8 PD En | S7 PD En | S6 PD En | S5 PD En | S4 PD En | S3 PD En | S2 PD En | S1 PD En | S0 PD En |

| | |
|---|---|
| Reserved fields should be ignored (masked) when read, and only 0's should be written to them. | |
| S31PDEn – S0SPDEn | 0: Pull-down is NOT enabled (default)<br>1: Pull-down is enabled |

1. The GPIO_2_15 field will have no effect on MG2580 since GPIO pins are not connected. See Table 2-4 for more information on MG2580 pin descriptions.
2. The GPIO_2_13 field will have no effect on MG2580 since GPIO pins are not connected. See Table 2-4 for more information on MG2580 pin descriptions.
3. The GPIO_2_12 field will have no effect on MG2580 since GPIO pins are not connected. See Table 2-4 for more information on MG2580 pin descriptions.

## 5.14 USB 2.0 On-the-Go Interface

The MG3500 SoC contains a High-Speed USB 2.0 interface with the ability to operate as Device, Host, or On-The-Go (OTG) at speeds of up to 480 MHz. The USB interface includes the Physical Layer on-chip. Figure 5-36 shows a high-level block diagram of the USB Interface.

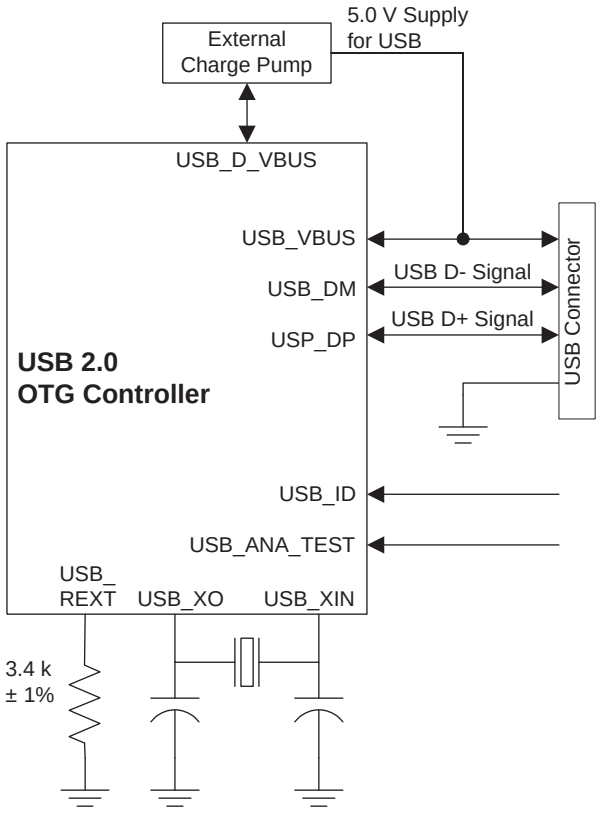



**Figure 5-36 USB Interface Block Diagram**

This section discusses the Physical Layer (PHY) first, followed by a discussion of the USB Controller.

### *5.14.1 Physical Layer (PHY)*

The USB 2.0 OTG PHY port has three distinct external interfaces:

- The USB Data Plus (D+) and Data Minus (D–) lines: These lines are USB 1.1 and 2.0 specification-compliant. The USB 2.0 OTG PHY supports high-speed, 480-Mbps transfers, as well as USB 1.1 full-speed and low-speed transfers.
- The active high signal USB_D_VBUS (USB VBUS Drive) is used to enable an external charge pump for USB_VBUS when operating as a host.
- USB 2.0 Transceiver Macrocell Interface (UTMI): The USB 2.0 OTG PHY supports the following modes through the UTMI:
  - High-Speed (HS)
  - Full-Speed (FS)

- Full-Speed-Only (FS-Only)
- Full-Speed Power-Save (FS Power-Save)
- Low-Speed Power-Save (LS Power-Save)
- Low-Speed Preamble (LS Preamble)
- Low-Speed Preamble Power-Save (LS Preamble Power-Save)

The UTMI contains a receive port, a transmit port, and associated control lines to interface with a USB host controller or device controller. The receive and transmit ports can be configured as 8/16-bit parallel ports for all modes of operation.

- Serial interface: This interface supports full-speed (FS-Serial mode) and low-speed (LS-Serial mode) data transmission rates to and from a controller.

The USB 2.0 OTG PHY handles low-level USB protocol and signaling. The USB 2.0 OTG PHY supports SYNC detection, data serialization and deserialization, and data recovery.

### Features

The USB 2.0 OTG PHY has the following features.

- General Features
  - Low power dissipation while active, idle, or on standby
  - Integrates high-, full-, and low-speed (Host mode only) termination and signal switching
  - Requires minimal external components: a single resistor and single crystal with two capacitors for best operation
  - Provides an on-chip PLL to reduce clock noise and eliminate the need for an external clock generator
  - Supports an off-chip charge pump regulator to generate 5 V for VBUS
  - Integrates short-to-5-V and short-to-ground protection for D+ and D– lines (requires only global electrostatic discharge (ESD) and 5 V-compliant dp/dm pads)
- USB 2.0 Features
  - Complies with *Universal Serial Bus Specification*, Revision 2.0
  - Complies with *On-The-Go Supplement to the USB 2.0 Specification*, Revision 1.0a
  - Complies with *UTMI+ Specification*, Revision 1.0 (Level 3)
  - Integrates 45-ohm termination, 1.5 KOhm pull-up and 15 KOhm pull-down resistors, with support for independent control of the pull-down resistors
  - Supports 480-Mbps high-speed, 12-Mbps full-speed, and 1.5-Mbps low-speed (Host mode only) data transmission rates
  - Supports 8/16-bit unidirectional parallel interfaces for HS, FS, and LS (Host mode only) modes of operation, in accordance with the UTMI specification
  - Provides dual (HS/FS) mode host/device support (LS operation is not supported for device applications)
  - Implements data recovery from serial data on the USB connector
  - Implements SYNC/End-of-Packet (EOP) generation and checking
  - Implements bit stuffing and unstuffing, and bit-stuffing error detection
  - Implements Non Return to Zero Invert (NRZI) encoding and decoding
  - Implements bit serialization and deserialization

- Implements holding registers for staging transmit and receive data
- Implements logic to support suspend, resume, and remote wakeup operations
- Implements VBUS pulsing and discharge Session Request Protocol (SRP) circuit
- Implements VBUS threshold comparators

### Crystal Specifications

To obtain optimal crystal performance, use a fundamental crystal with the following specifications:

- Resonance mode: parallel
- Load capacitance: 15–30 pF
- Shunt capacitance: 5–8 pF
- Series resistance: 20–60 ohms

Drive level: 50–500 µW

### Off-Chip Charge Pump

An off-chip charge pump is required to provide power to the PHY USB_VBUS pin. Figure 5-36 shows the charge pump connection to the USB 2.0 OTG PHY. The charge pump's output is connected directly to VBUS on the device board. The pin that connects to the USB 2.0 OTG PHY's USB_VBUS pin is also connected to VBUS.

The USB_VBUS pin can sink or source 8 mA of current. Therefore, to meet the design specification, the charge pump output must be able to source at least 10 mA at a voltage level of 5 V (± 4 percent). The USB_VBUS pin presents a worst-case load of 500 fF. This worst-case load is for the USB 2.0 OTG PHY only and does not account for capacitance due to routing, pads, package, or board traces.

The Charge Pump is only required for OTG operation. When the MG3500 SoC is a Device, power to the USB_VBUS pin must be supplied by an external host. When the MG3500 SoC is a Host, the design must provide power to the USB_VBUS pin. This can come from either a Charge Pump or an on-board supply.

### *5.14.2 USB Controller*

The MG3500 SoC USB Controller is a Dual-Role Device (DRD) controller that supports both device and host functions. It is fully compliant with the *On-The-Go Supplement to the USB 2.0 Specification*, Revision 1.0a. It can also be configured as a Host-only or Device-only controller, fully compliant with the *USB 2.0 Specification*. The USB 2.0 configurations support high-speed (HS, 480-Mbps), full-speed (FS, 12-Mbps), and low-speed (LS, 1.5-Mbps) transfers. Additionally, the USB Controller can be configured as a USB 1.1 full-speed/low-speed DRD.

The USB Controller is optimized for the following applications and systems:

- Portable electronic devices
- Point-to-point applications (no hub, direct connection to HS, FS, or LS device)
- Multi-point applications (as an embedded USB host) to devices (hub and split support)

Figure 5-37 shows a block diagram of the USB Controller.

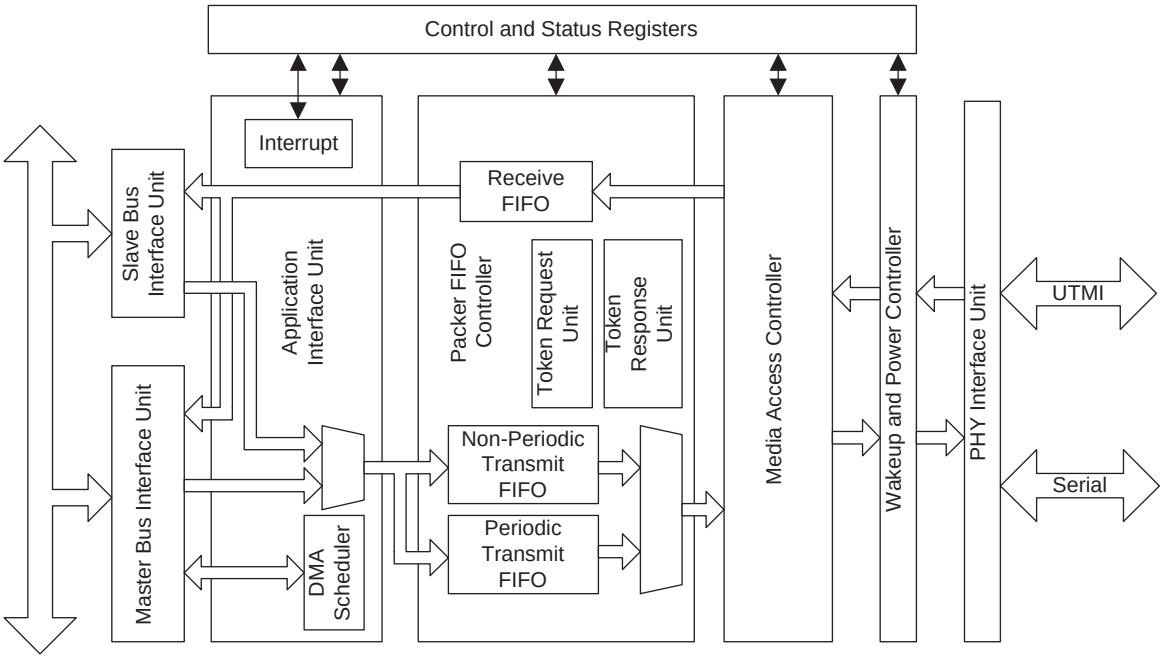



**Figure 5-37 USB Controller**

## General Features

- Includes USB power management features
- Includes clock gating to save power
- Supports packet-based, dynamic FIFO memory allocation for endpoints for small FIFOs and flexible, efficient use of RAM
- Supports the Keep-Alive in Low-Speed mode and SOFs in High/Full-Speed modes
- Power-optimized design

### USB 2.0 Supported Features

- Complies with the On-The-Go Supplement to the USB 2.0 Specification (Revision 1.0a)
- Operates in High-Speed (HS, 480-Mbps), Full-Speed (FS, 12-Mbps) and Low-Speed (LS, 1.5-Mbps) modes
- Supports Session Request Protocol (SRP)
- Supports Host Negotiation Protocol (HNP)
- $I^2C$ interface (for support of Mini USB Analog Carkit Interface Specification, CEA-936, revision 2, not intended for use with other devices)
- Supports a generic root hub
- Includes automatic ping capabilities

### Power Optimization Features

- PHY clock gating support during USB Suspend mode and Session-Off mode
- Partial power-off during USB Suspend mode and Session-Off mode
- Input signals to powered-off blocks driven to safe 0
- Data FIFO RAM chip-select deasserted when not active
- Data FIFO RAM clock-gating support

### Host Architecture

The host uses one transmit FIFO for all non-periodic OUT transactions and one transmit FIFO for all periodic OUT transactions. These transmit FIFOs are used as transmit buffers to hold the data (payload of the transmit packet) to be transmitted over USB. The host pipes the USB transactions through Request queues (one for periodic and one for non-periodic). Each entry in the request queue holds the IN or OUT channel number along with other information to perform a transaction on the USB. The order in which the requests are written into the queue determines the sequence of transactions on the USB. The host processes the periodic request queue first, followed by the non-periodic request queue, at the beginning of each (micro)frame.

The host uses one receive FIFO for all periodic and non-periodic transactions. The FIFO is used as a receive buffer to hold the received data (payload of the received packet) from the USB until it is transferred to the system memory. The status of each packet received also goes into the FIFO. The status entry holds the IN channel number along with other information, such as received byte count and validity status, to perform a transaction on the AMBA High-Speed Bus (AHB).

### Device Architecture

The OTG device uses a single transmit FIFO to store data for all non-periodic endpoints, and one transmit FIFO per periodic endpoint to store data to be transmitted in the next (micro)frame. The data is fetched by the DMA engine or is written by the application into the transmit FIFOs and is transmitted on the USB when the IN token is received. The request queue contains the number of the endpoint for which the data is written into the Data FIFO.

To improve performance, the application can use the learning queue to help predict the order in which the USB host will access the non-periodic endpoints and writes the data into the non-periodic FIFO accordingly. Since each periodic IN endpoint has its own FIFO, no order prediction is needed for periodic IN transfers.

The OTG device uses a single receive FIFO to receive the data and status for all OUT endpoints. The status of the packet includes the size of the received OUT data packet, data PID, and validity of the received data. The data in the receive FIFO is read by the DMA or the application when the data is received.

### 5.15 Ethernet Media Access Controller

The Ethernet MAC supports 10/100/1000 Mbps Ethernet interfaces. This is typically connected to an external Physical Layer (Phy) device but can also be connected directly to Ethernet switches that support Reverse MII interfaces.

> **Note:** When both 10/100 and GigE need to be enabled, an external switch must be installed to select the clock. Contact Mobilygen for more information.

The Ethernet MAC enables a host to transmit and receive data over Ethernet in compliance with the IEEE 802.3-2002 standard. It is compliant to the following standards:

- IEEE 802.3-2002 for Ethernet MAC and GMII
- AMBA 2.0 for AMBA High-Speed Bus (AHB) Master/Slave ports
- RMII specification from RMII consortium

Figure 5-38 shows the connections when the MG3500 SoC is connected to an external PHY.

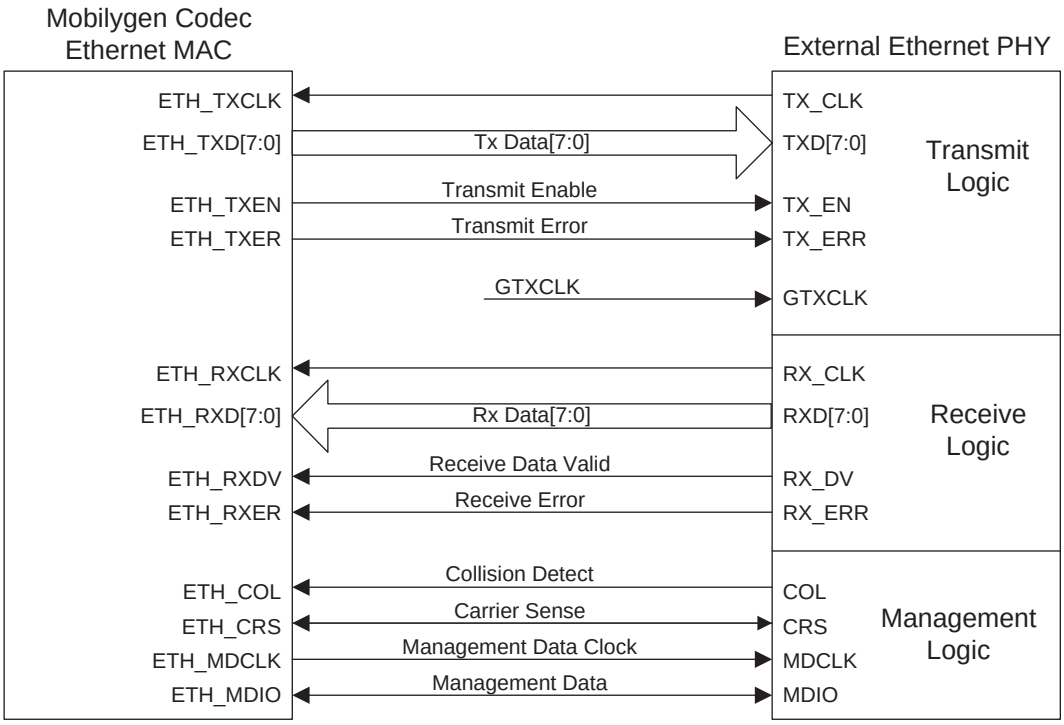



**Figure 5-38 Ethernet MAC to PHY Connections**

### *5.15.1 Overview*

Figure 5-39 shows a block diagram of the Ethernet MAC.

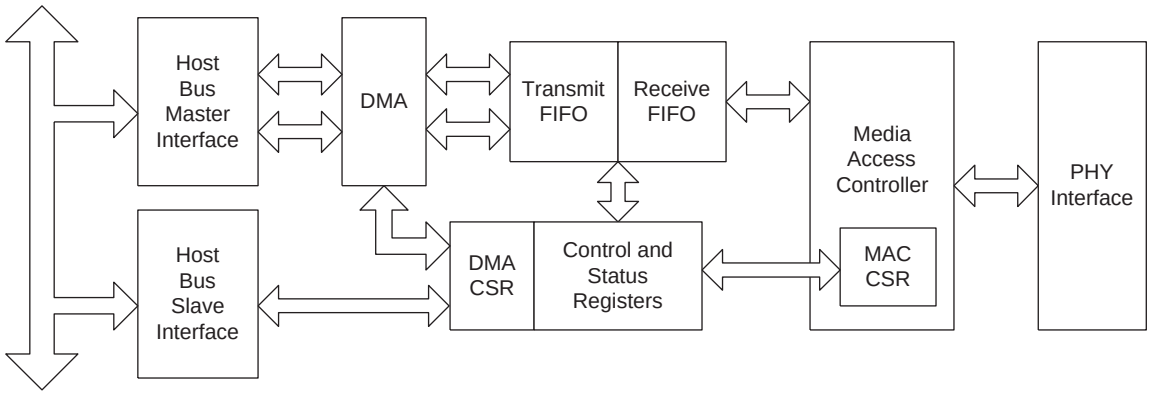



**Figure 5-39 Ethernet MAC Block Diagram**

The Ethernet MAC transfers data to system memory through the Host Bus Master interface. The host CPU uses the Host Bus Slave interface to access the MAC subsystem's Control and Status registers (CSRs).

#### Transmit and Receive FIFOs

The Transmit FIFO (Tx FIFO) buffers data read from system memory by the DMA before transmission by the MAC. Similarly, the Receive FIFO (Rx FIFO) stores the Ethernet frames received from the line until they are transferred to system memory by the DMA.

These are asynchronous FIFOs, as they also transfer the data between the application clock and the GMAC line clock domains. Both FIFOs are Dual-ported RAM.

### *5.15.2 Features List*

The Ethernet MAC has the following features, listed by category.

#### GMAC Core Features

- Supports 10/100/1000-Mbps data transfer rates with the following PHY interfaces
  - IEEE 802.3-compliant GMII/MII (default) interface to communicate with an external Gigabit/ Fast Ethernet PHY
- Supports both full-duplex and half-duplex operation
  - Supports CSMA/CD Protocol for half-duplex operation
  - Supports packet bursting and frame extension in 1000 Mbps half-duplex operation
  - Supports IEEE 802.3x flow control for full-duplex operation
  - Optional forwarding of received pause control frames to the user application in full duplex operation
  - Back-pressure support for half-duplex operation
  - Automatic transmission of zero-quanta pause frame on deassertion of flow control input in full-duplex operation

- Preamble and SFD insertion in Transmit, and deletion in Receive paths
- Automatic CRC and pad generation controllable on a per-frame basis
- Options for Automatic Pad/CRC Stripping on receive frames
- Programmable frame length to support Standard or Jumbo Ethernet frames with sizes up to 16 KB
- Programmable InterFrameGap (40-96 bit times in steps of 8)
- Supports a variety of flexible address filtering modes:
  - Up to 31 additional 48-bit perfect (DA) address filters with masks for each byte
  - Up to 31 48-bit SA address comparison check with masks for each byte
  - 64-bit Hash filter (optional) for multicast and uni-cast (DA) addresses
  - Option to pass all multicast addressed frames
  - Promiscuous mode support to pass all frames without any filtering for network monitoring
  - Passes all incoming packets (as per filter) with a status report
- Separate 32-bit status returned for transmission and reception packets
- Supports IEEE 802.1Q VLAN tag detection for reception frames
- Separate transmission, reception, and control interfaces to the Application
- Configurable big endian and little endian support for transmission and reception data paths
- Supports 32/64/128-bit data transfer interface on the system-side
- Complete network statistics (optional) with RMON/MIB Counters (RFC2819/RFC2665)
- MDIO Master interface (optional) for PHY device configuration and management

### DMA Block Features

The DMA block exchanges data between FIFOs and host memory. A set of registers (DMA CSR) to control DMA operation is accessible by the host. DMA features include:

- 32/64/128-bit data transfers
- Single-channel Transmit and Receive engines
- Fully synchronous design operating on a single system clock
- Optimization for packet-oriented DMA transfers with frame delimiters
- Byte-aligned addressing for data buffer support
- Dual-buffer (ring) or linked-list (chained) descriptor chaining
- Descriptor architecture, allowing large blocks of data transfer with minimum CPU intervention; each descriptor can transfer up to 2 KB of data
- Comprehensive status reporting for normal operation and transfers with errors
- Individual programmable burst size for Transmit and Receive DMA Engines for optimal host bus utilization
- Programmable interrupt options for different operational conditions
- Per-frame Transmit/Receive complete interrupt control
- Round-robin or fixed-priority arbitration between Receive and Transmit engines
- Start/Stop modes
- Separate ports for host CSR access and host data interface

### Transaction Layer (MTL) Features

The MTL block consists of two sets of FIFOs: a Transmit FIFO with programmable threshold capability, and a Receive FIFO with a configurable threshold (default of 64 bytes). MTL features include:

- 32/64/128-bit Transaction Layer block providing a bridge between the application and the GMAC-CORE
- Single-channel Transmit and Receive engines
- Data transfers executed using simple FIFO-protocol
- Synchronization for all clocks in the design (Transmit, Receive and system clocks)
- Optimization for packet-oriented transfers with frame delimiters
- Four Separate ports for system-side and MAC-side transmission and reception
- Two dual-port RAM-based asynchronous FIFOs with synchronous/asynchronous Read and Write operation with respect to the Read and Write clocks (one for transmission and one for reception)
- Supports 128/256/512/1K/2K/4K/8K/16K-byte Receive FIFO depth on reception.
- Programmable burst-length support for starting a burst up to half the size of the MTL Rx and Tx FIFO in the GMAC-MTL configuration
- Receive Status vectors inserted into the Receive FIFO after the EOF transfer enables multiple-frame storage in the Receive FIFO without requiring another FIFO to store those framesf Receive Status.
- Configurable Receive FIFO threshold (default fixed at 64 bytes)
- Option to filter all error frames on reception and not forward to application
- Option to forward under-sized good frames
- Supports statistics by generating pulses for frames dropped or corrupted (due to overflow) in the Receive FIFO
- Supports 256/512/1K/2K/4K/8K-byte FIFO depth on transmission
- Supports Store and Forward mechanism for transmission to the GMAC core
- Supports threshold control for transmit buffer management
- Supports configurable number of frames to be stored in FIFO at any time. The default is 2 frames (fixed) with internal DMA, and up to 8 frames in GMAC-MTL configuration.
- Automatic generation of PAUSE frame control or backpressure signal to the GMAC core based on Receive FIFO-fill (threshold configurable) level.
- Handles automatic retransmission of Collision frames for transmission
- Discards frames on late collision, excessive collisions, excessive deferral and underrun conditions
- Software control to flush Tx FIFO
- Data FIFO RAM chip-select disabled when inactive, to reduce power consumption

## 5.16 High-Speed Bitstreams

### *5.16.1 Bitstream Introduction*

The Bitstream Port provides a bidirectional serial data port for Input and Output of compressed bitstreams, with an associated valid signal. Performance supports a maximum compressed stream data rate of 74 Mbps (High Profile, Level 4.1_62.5 Mbps + audio_384 kbps + NAL), to match the maximum of the codec. A minimum data rate of 30 Mbps offers compliance with Level 4.0 (High Profile, Level 4.0_25 Mbps + audio_384 kbps + NAL).

The bitstream interface is designed to stream high-speed bitstream data directly to and from the MG3500 in Code or SoC mode. The Bitstream interface is multiplexed with the Serial Peripheral Interface (SPI) signals.

**Note:** The maximum frequency when running with the internal clock is 67.5 MHz. This is due to a limitation if the granularity (step size) of the PLL.

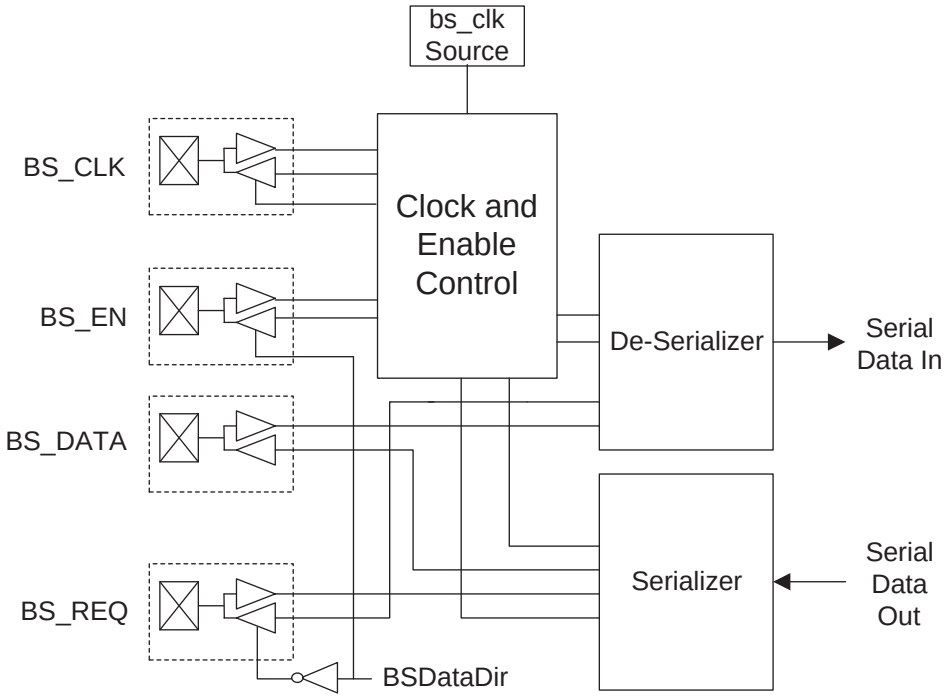



**Figure 5-40 High-Speed Bitstream Signals**

### *5.16.2 Bitstream Signals*

**Table 5-17 Bitstream Signals**

| Signal | Direction | | Description |
|---|---|---|---|
| | Transmitter | Receiver | |
| BS_CLK | Input or Output | Output or Input | Data gets latched by this signal. It can be provided either by the MG3500 SoC or externally. The Bitstream interface supports data transfers up to 74 MHz. |

**Table 5-17 Bitstream Signals**

| | | | |
|---|---|---|---|
| BS_DATA | Output | Input | The data itself. |
| BS_EN | Output | Input | This signal can be used to qualify BS_CLK. BS_EN is always the same direction to BS_DATA. |
| BS_REQ | Input | Output | A request signal used for flow control. BS_REQ is always the opposite direction to BS_DATA. |

### *5.16.3 Bitstream Modes*

In the following diagram, all signals are active high or active rising edge. BS_CLK and B_EN can be programmed to be active falling edge or active Low. The following section explains the primary way in which the control signals are used.

### *5.16.4 Clock Plus Enable Mode*

The Clock Plus Enable Mode is BS_CLK enabled, which is qualified by BS_EN. When BS_EN is active and a BS_CLK edge event occurs, the data on BS_DATA is latched.

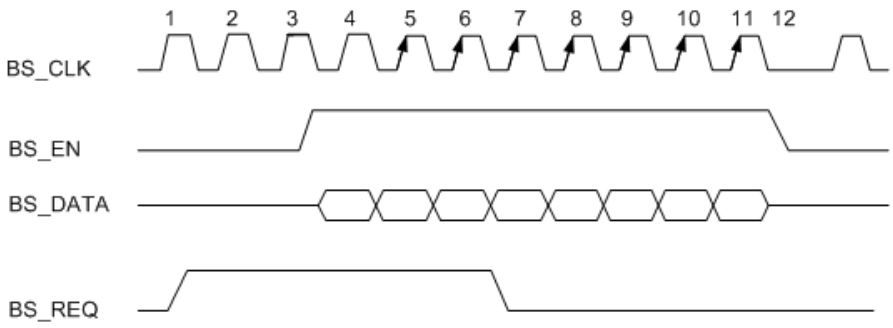



**Figure 5-41 Receive with BS_CLK and BS_ENABLE Timing**

Some points to note about the waveform diagram are as follows:

- Cycle 1: The receiver is not ready for data, so BS_REQ is not asserted.
- Cycle 2 and 3: The receiver is ready, but the transmitter is not ready because BS_EN is not asserted.
- Cycle 4-6: Both the transmitter and receiver are ready and data is being transferred.
- Cycle 7: The receiver has de-asserted BS_REQ indicating it cannot receive any more data. However, it is necessary that the receiver be designed to accept all bits in the current byte being transferred. When the MG is “txer,” it can stop transfers only at byte boundaries. In the rxer mode, MG expects the transmitter to stop only at a byte boundary.
- Cycle 8: The clock doesn’t necessarily need to be free-running.

### *5.16.5 Bitstream Registers*

The Bitstream Interface registers exist in the Configuration/Status Registers.

***Bitstream Interface Control 0***

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | BS Req Pol | BS Req En | BS Clk Drv Edge | BS ClkS amp Edge | BS Clk Dir | BS En Pol | BS Data-Dir | BSI-FEn |

| | |
|---|---|
| Reserved fields should be ignored (masked) when read, and only 0's should be written to them. | |
| BSReqPol | 0: Active High (default)<br>1: Active Low |
| BSReqEn | 0: BS_REQ is NOT generated or used (default)<br>1: BS_REQ is generated or used. |
| BSClkDrvEdge | 0: Rising edge (default)<br>1: Falling edge<br>Defines on which edge to drive signals. |
| BSClkSampEdge | 0: Rising edge (default)<br>1: Falling edge<br>Defines on which edge to sample signals. |
| BSClkDir | 0: Input (default)<br>1: Output |
| BSEnPol | 0: Active High (default)<br>1: Active Low |
| BSDataDir | 0: Input/Receive (default)<br>1: Output/Transmit |
| BSIFEn | 0: The Bitstream Interface is disabled (default).<br>1: The Bitstream Interface is enabled.<br>When this register is set to 0, the de-serializer pointer is reset. In the case where the bit-stream data becomes unaligned (bits within the bytes), disabling the interface and re-enabling the interface helps to re-sync. Also, this register allows the interface to be off while the other configuration registers are being set. |

**Note:** Attention should be paid as to how BSIFEn is implemented. Depending on the design, if the control signals are changed, they may cause a data latch event. If BSIFEn is enabled at exactly the same time, it may cause the data to get off by one bit. One way to get around this is to program the control registers first while programming the BSIFEn to 0, then programming the registers once again, this time programming BSIFEn to 1.

***Bitstream Interface Control 2***

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | BS Flash TxEn | BS Flash RxEn | BS Clk En Mode | BS Stop Cond | BS Strobe Mode En | MSB First | BSThresh | |

| | |
|---|---|
| Reserved fields should be ignored (masked) when read, and only 0's should be written to them. | |
| BSFlashTxEn | 0: Disable Flash FIFO serializer when transmitter is disabled (default).<br>1: Enable Flash FIFO serializer when transmitter is disabled |
| BSFlashRxEn | 0: Disable Flash FIFO serializer when receiver is disabled (default).<br>1: Enable Flash FIFO serializer when receiver is disabled |
| BSClkEnMode | 0: Clock enable align to serial data out (default).<br>1: Clock enable advance one clock to serial data out (not supported).<br>For transmit data out only. This is used to help the clock gate issue. |
| BSStopCond | 0: Upon BS_REQ being de-asserted, stop on the current bit (default).<br>1: Upon BS_REQ being de-asserted, stop at the end of the current byte (not supported).<br>For transmit (data output) only. |
| BSStrobeModeEn | 0: The interface works in 'clock plus enable' mode (default).<br>1: The interface works in 'strobed mode'.(not supported). |
| MSbFirst | 0: TheLSB of the byte is sent first.<br>1: The MSB of the byte is sent first (default). |
| BSThresh | When in bitstream input mode, this register sets the threshold (how many free bytes are left) at which the BS_REQ would be asserted.<br>00: 1<br>01: 2 (default)<br>10: 3<br>11: 4 |

***Bitstream Interface Control 4***

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | Threshold_low | | | | | | | |

| | |
|---|---|
| Reserved fields should be ignored (masked) when read and only zeros should be written to them. | |
| Threshold_low | Request Time Out Low |

***Bitstream Interface Control 6***

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | Threshold_high | | | | | | | |

| | |
|---|---|
| Reserved fields should be ignored (masked) when read and only zeros should be written to them. | |
| Threshold_high | Request Time Out High |

The Bitstream Interface Control 4 and Bitstream Interface Control 6 registers have a power-on default state of 0x00. These registers should be programmed with 0xFF when acting as a receiver. However for ease of implementation, it is safe to always load these registers with 0xFF.

# 6.0 System Design and Applications

## 6.1 Power Supply Design and Recommendations

**TBD:** An overview of the power supplies that the device needs for proper operation and information on how each of these supplies should be connected and filtered.

## 6.2 Power Supply Sequencing

Figure 6-1 provides the recommended power-up and power-down sequences. In an ideal design, all of the power supplies become stable at the same time to prevent any direct feed-through current. In real designs, however, there is typically a time delay between when the various power supplies stabilize. This section describes the restrictions on the time differences between the power supplies.

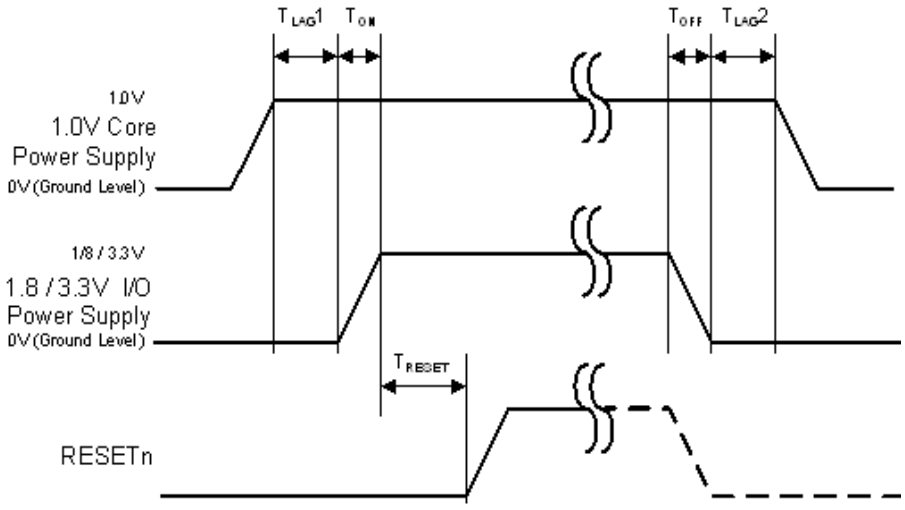



**Figure 6-1 Power Supply Sequencing**

The MG3500 uses three different power rails: 1.0V for the core, 1.8V for DDR I/Os, and 3.3V for all other I/Os. Both I/O voltages can be brought up at any time and in any order after the 1.0V core supply becomes stable. During power-down, the I/O voltages must be brought down before the core voltage.

The $T_{RESET}$ has to be greater than 1 micro seconds.

The restrictions are as follows:

$T_{LAG}1$, $T_{LAG}2$ >= 0 ms.
$T_{ON}$, $T_{OFF}$ >= 0 ms.

## 6.3 Reset timing Diagrams

Figure 6-1 also shows the timing for the active low reset signal RESETn. This signal must remain low for a minimum of 1µsecond after the power supplies stabilize.

## 6.4 Oscillator Connections, Values and Formulas

The USB 2.0 OTG PHY supports the following reference clock sources:

### *6.4.1 Crystal Connected to the USB_XIN and USB_XO Pins*

Figure 6-2 shows the clock configuration with an external crystal.

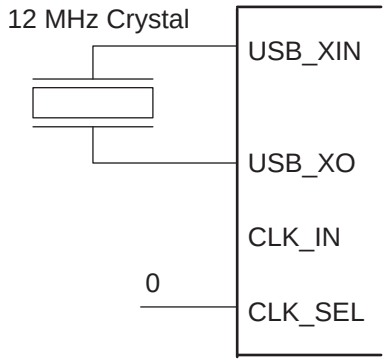



**Figure 6-2 Clock Configuration with an External Crystal**

The crystal must have a fundamental frequency of 12 MHz. and must meet the specifications shown in Table 6-1.

**Table 6-1 Crystal Specifications**

| Parameter | Value |
|---|---|
| Frequency Tolerance | ± 200 ppm |
| Peak Jitter | ± 100 ps |
| Output Differential Voltage | > 500 mV w.r.t Xi |
| Shunt Capacitance | 5 – 8 pF |
| Load Capacitance | 15-30 pF |
| Series Resistance | 20-60 Ohms |
| Drive Level | 50-500 µW |

### *6.4.2 Crystal and External Clock Connected to the External CLK_IN Pin*

Figure 6-3 shows the clock configuration when the external CLK_IN pin is used. In this mode, both an external crystal and the external CLK_IN pin must be connected.

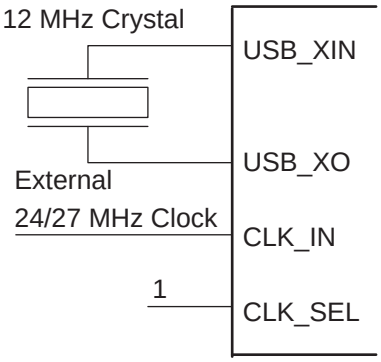



**Figure 6-3 Clock Configuration with an External Crystal**

The crystal must have a fundamental frequency of 12 MHz. and meet the specifications shown in Table 6-1. The external clock must have a fundamental frequency of 24 or 27 MHz, with a frequency tolerance of ±200 ppm, a peak jitter of ±100 ps., a duty cycle between 40/60 and 60/40, and a signal swing equal to the host power supply voltage.

### *6.4.3 Crystal and External Clock Connected to the External CLK_IN Pin*

Figure 6-3 shows the clock configuration when only the external CLK_IN pin is used.

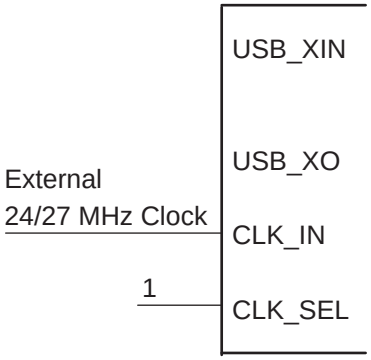



**Figure 6-4 Clock Configuration with an External Crystal**

In this mode the USB will not be operational because the USB only runs when using the crystal. This configuration is typically used in co-processor applications.

The external clock must have a fundamental frequency of 24 or 27 MHz, with a frequency tolerance of ±200 ppm, a peak jitter of ±100 ps., a duty cycle between 40/60 and 60/40, and a signal swing equal to the host power supply voltage.

# 7.0 Ordering Information

This section provides product ordering specifications, including soldering profile.

## 7.1 Product Information

- Family Products: MG3500, MG2580
- Package Type: B (BGA)

### *7.1.1 Product Part Number Format*

<xxxxxx><xx>–<xxx><x>

<Product Family><Product Revision>–<Number of Solder Connections><Package Type>

### *7.1.2 Product Part Number Examples*

<MG3500><A2>–<376><B>

<MG3500><A3>–<376><B>

<MG2580><A2>–<376><B>

<MG2580><A3>–<376><B>

### *7.1.3 Product Ordering Specifications*

The following table lists the ordering information for the MG3500 and MG2580 products:

| Part Order Number | Temperature Range | Pin Package | Body Size | Maximum Height | Lead (Pb) Free | JEDEC Standard | Package Method | Package Quality |
|---|---|---|---|---|---|---|---|---|
| MG3500A2-376B | 0°C to +90°C | 376 Bump FPBGA | 18mm x 18mm | 1.40 mm | Yes | Yes | Tray | 420 |
| MG3500A3-376B | 0°C to +90°C | 376 Bump FPBGA | 18mm x 18mm | 1.40 mm | Yes | Yes | Tray | 420 |
| MG2580A3-376B | 0°C to +90°C | 376 Bump FPBGA | 18mm x 18mm | 1.40 mm | Yes | Yes | Tray | 420 |

## 7.2 MG3500 Family Reflow Profile


- "Solder Profile" on page 198
- "Rework " on page 198
- "MG3500 Demount Guideline" on page 199


### *7.2.1 Solder Profile*

- Package MSL-3.
- Thermal profiling of the convection / IR reflow machine is required for each product design.
- Pb-free solder reflow temperature shall not exceed 260oC with time above liquidus temperature (217oC) of 60-150 seconds.
- All package solder joints must meet the solder paste manufacturer recommended reflow profile specification.
- Limit the PCB laminate temperature to 245oC.

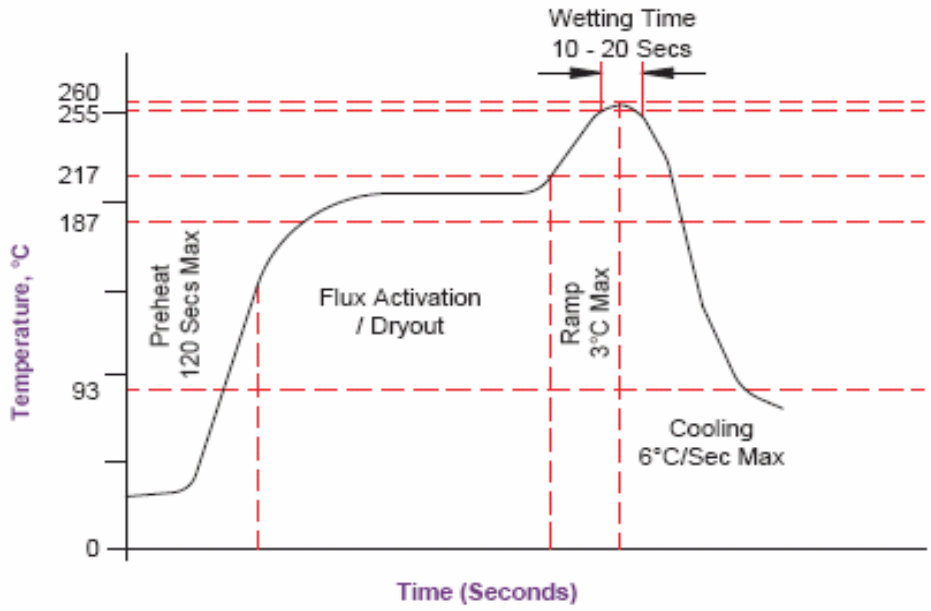



**Figure 7-1 Pb-Free Solder Reflow Profile**

### *7.2.2 Rework*

- Each MG3500 may be reworked only one time.
- Defective packages (electrical or visual rejects) shall be removed and replaced using the basic flow in the order:
  - Bake out the moisture of the whole printed circuit board assembly (refer to IPC/Jedec spec J-STD-033)
  - Thermal profiling
  - Package removal

- Package site conditioning
- Solder Replenishment
- Package Replacement
- Reflow (if the time of reflow exceed 168 hours after last bake, rebake the PCBA and MG3500 to drive out all the moisture)
- Visual Inspection

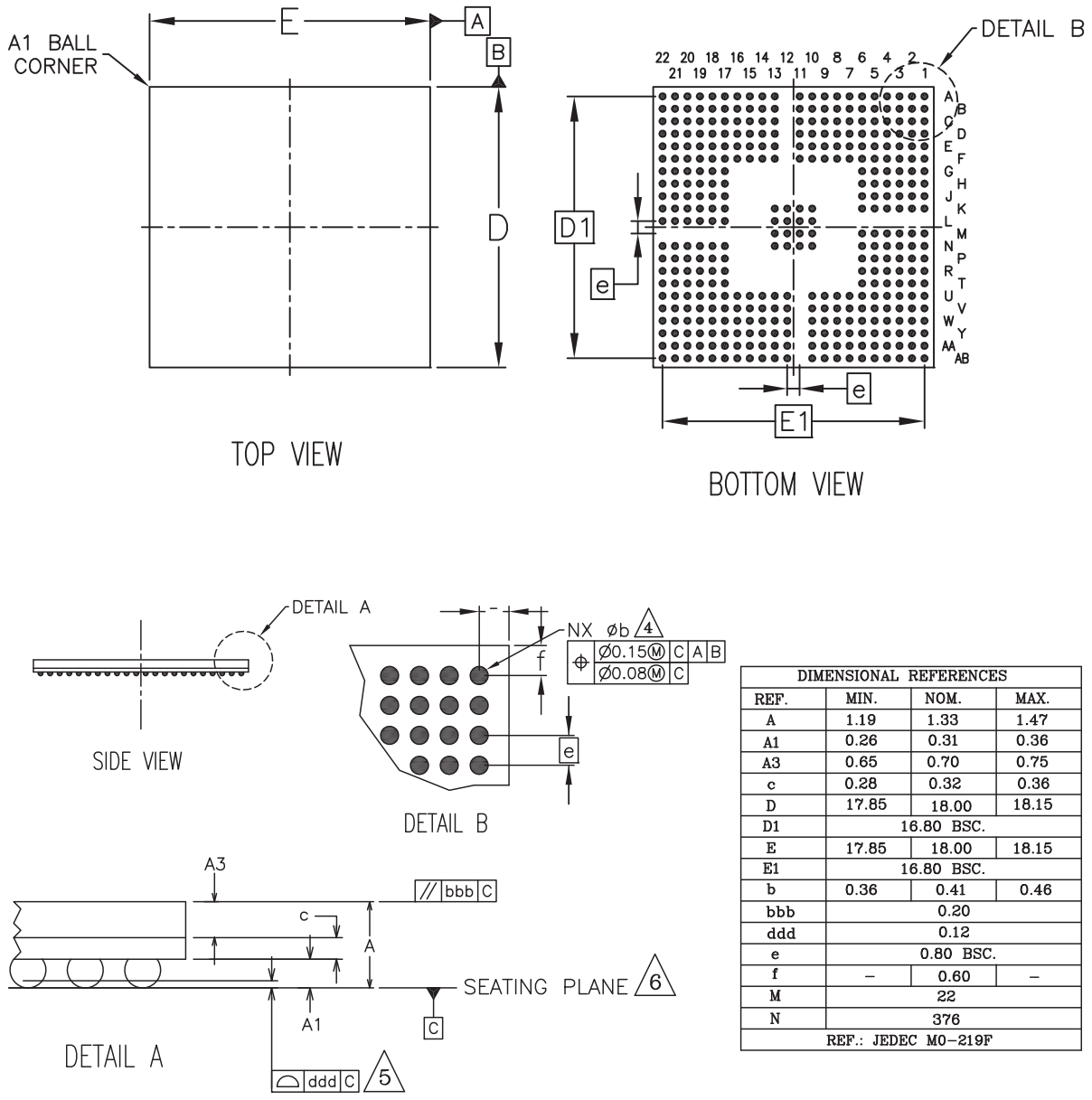
### *7.2.3 MG3500 Demount Guideline*

A Package rework tool with vacuum pick-up shall be used and a bottom PCB heater is required.

- The PCB needs to be dry. Recommend baking for 24 hours at 125°C. This is to prevent moisture-induced “pop-corn” damage resulting from any moisture in the PCB and/or package.
- Board rework site shall be supported from the bottom with non-thermally conductive material during module removal to minimize PCB warpage.
- The support block should be at least the same size as the rework nozzle and be centered with respect to the package size.
- Preheat the entire board assembly using a bottom PCB heater between 110°C to 120°C. Board temperature measured within 25 mm(1.0”) of the rework site.
- Limit adjacent component SMT joints to below solder reflow temperature.
- Components / PCB should be 100% inspected and any defects rejected such as:
  - Lifted pads on PCB
  - Thermal damage to PCB or adjacent components

# 8.0 Packaging Information

## 8.1 Package Diagram




| DIMENSIONAL REFERENCES | | | |
|---|---|---|---|
| REF. | MIN. | NOM. | MAX. |
| A | 1.19 | 1.33 | 1.47 |
| A1 | 0.26 | 0.31 | 0.36 |
| A3 | 0.65 | 0.70 | 0.75 |
| c | 0.28 | 0.32 | 0.36 |
| D | 17.85 | 18.00 | 18.15 |
| D1 | 16.80 BSC. | | |
| E | 17.85 | 18.00 | 18.15 |
| E1 | 16.80 BSC. | | |
| b | 0.36 | 0.41 | 0.46 |
| bbb | 0.20 | | |
| ddd | 0.12 | | |
| e | 0.80 BSC. | | |
| f | – | 0.60 | – |
| M | 22 | | |
| N | 376 | | |
| REF.: JEDEC MO-219F | | | |

**Figure 8-1 MG3500/MG2580 SoC 376-Ball FPBGA Package Physical Drawing**

### 8.2 Thermal Data

Table 8-1 shows the case thermal conductivity data for the MG3500/MG2580 SoC 376-Ball FPBGA package.

**Table 8-1 Case Thermal Conductivity Data**

| Symbol | Parameter | Value |
|---|---|---|
| $\Theta J_A$ | Junction to Ambient | 23°C/Watt |
| $\Theta J_C$ | Junction to Case | 8°C/Watt |

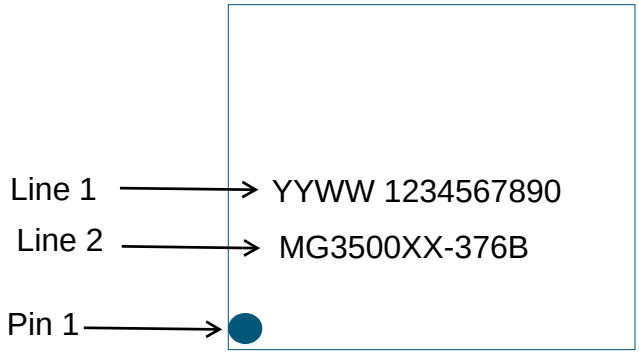
Temperature Range: 0°C — 90°C case temp.

## 9.0 Marking

| REVISIONS | | | |
|---|---|---|---|
| Rev | Description | Date | Approved |
| 0 | Initial Release | 7-July-08 | 7-July-08 |
| 1 | Change to Maxim | 2/18/2009 | 2/18/2009 |

Note:

1. Products:
   a. MG3500xx-376B
   b. MG2580xx-376B
   xx - revision of die
2: Height of Logo - 5mm
3: Height of Character - 1mm
4: Line Spacing - 0.5mm
5: Pin 1 location: lower left
6: Line 1: YYWW - datecode, YY-year, WW-week
7: Line 1: 1234567890 - lot number
8: Line 2: Full product name



# Mouser Electronics

Authorized Distributor

Click to View Pricing, Inventory, Delivery & Lifecycle Information:

Maxim Integrated:

MG3500EVKIT MG2580A2-376B