MSC8122/26ADS Overview
Board Configuration and Installation
ADS Power and Connectors
ADS Functional Description
ADS Memory Maps and Bus Mapping
ADS Control and Status
ADS Program Information
Contents

About This Book xi
Before Using This Manual—Important Note xi
Audience and Helpful Hints xi
Notational Conventions and Definitions xi
Organization xii
Related MSC8122/26ADS Documentation xiii
Abbreviations and Acronyms xiii

1 MSC8122/26ADS Overview
1.1 MSC8122/26ADS .................................................. 1-1
1.2 Board Specifications ............................................. 1-1
1.3 ADS Features .................................................... 1-2
1.3.1 MSC8122/26ADS Features .................................. 1-3
1.3.2 MSC8103 Features ............................................. 1-4
1.3.3 Board Capabilities ............................................. 1-4

2 Board Configuration and Installation
2.1 Hardware Preparation ............................................ 2-1
2.1.1 DIP Switches .................................................. 2-3
2.1.2 ADS Push Buttons ............................................ 2-5
2.1.3 ADS Jumpers .................................................. 2-6
2.1.3.1 JP1-EE0 for Slave .......................................... 2-6
2.1.3.2 JP2-CLKIN for Slave ...................................... 2-6
2.1.3.3 JP3-SET MII PHY ......................................... 2-7
2.1.3.4 JP4-CT-Bus Master Reset Enable ....................... 2-7
2.1.3.5 JP5-8102 Core Power ..................................... 2-7
2.1.3.6 JP6-Slave Clock Reference ............................... 2-8
2.1.3.7 JP7-DSI-Ethernet ......................................... 2-8
2.1.3.8 JP8-DLL-ON ............................................... 2-9
2.1.3.9 JP9-3V3-J3 .................................................. 2-9
2.1.3.10 JP11-MAC to MAC-enable .............................. 2-9
2.1.3.11 JP12-Board Slave Voltage Select ..................... 2-9
2.1.3.12 JP13-ADS Power-On-Reset Enable ................... 2-10
2.1.4 ADS Testing ................................................... 2-10
2.1.4.1 TP13-Slave ClockOut Test Point ....................... 2-10
2.1.4.2 JS1-JS4-Current Consumption Measurement ........ 2-11
2.1.4.3 JG1-JG5-GND Bridges ................................. 2-11
ADS LEDs. ................................................................. 2-12
2.2 Installation Options .............................................. 2-13
  2.2.1 Debug Connection Schemes ................................. 2-13
  2.2.2 Standalone Operation ................................. 2-14
  2.2.3 Connecting the ADS Board ................................. 2-17

3 ADS Power and Connectors
  3.1 Power ................................................................. 3-1
    3.1.1 Power Supply .............................................. 3-1
    3.1.2 Power Buses ............................................. 3-3
  3.2 Interconnect Signals .......................................... 3-4
    3.2.1 Stereo Phone Jack Connectors .............................. 3-5
    3.2.2 SMB Connector ............................................ 3-5
    3.2.3 Logic Analyzer Connectors ................................ 3-5
    3.2.4 Mezzanine Header Ethernet Connectors ...................... 3-6
    3.2.5 RJ45 E1/T1 Line Connector ................................ 3-6
    3.2.6 Slave UART Port Connector ............................... 3-6
    3.2.7 Altera’s In-System Programming (ISP) Connector .............. 3-7
    3.2.8 Host Debug EOnCE (SYS) Connector ....................... 3-7
    3.2.9 EE1 Connector ............................................ 3-9
    3.2.10 Ethernet Port Connectors ................................ 3-9
  3.3 ADS Expansion Connectors ..................................... 3-10
    3.3.1 Expansion Connector J1 .................................... 3-10
    3.3.2 Expansion Connector J2 .................................... 3-10
    3.3.3 Expansion Connector J3 .................................... 3-11
    3.3.4 Expansion Connector J4 .................................... 3-11
    3.3.5 Expansion Connector J5 .................................... 3-11

4 ADS Functional Description
  4.1 MSC8122/26ADS Block Diagram .............................. 4-1
  4.2 Reset and Reset-Configuration ................................ 4-3
    4.2.1 Power-On-Reset ............................................ 4-3
    4.2.2 Host Power-On-Reset Configuration ....................... 4-3
    4.2.3 Slave Power-On-Reset Configuration ....................... 4-4
    4.2.4 Hard/Soft Reset Capabilities .............................. 4-4
    4.2.5 Hard Reset Configuration ................................ 4-5
    4.2.6 Host Hard Reset Configuration ............................ 4-5
    4.2.7 Slave Reset Configuration ................................ 4-7
  4.3 Clock Source .................................................. 4-11
    4.3.1 Host Main Clock Scheme ................................ 4-11
Slave Main Clock Scheme .......................................................... 4-12

4.4 60x Bus Buffering and Multiplexing ........................................ 4-13
4.4.1 60x-Compatible System Bus User Instructions ......................... 4-15

4.5 Chip Select Designation ......................................................... 4-18

4.6 Interrupts .............................................................................. 4-19
4.6.1 Host-Side Interrupts ............................................................. 4-19
4.6.1.1 ABORT Interrupt to MSC8103 ......................................... 4-19
4.6.1.2 Slave Interrupt ............................................................... 4-19
4.6.1.3 Flash Ready Interrupt ...................................................... 4-20
4.6.1.4 Fast Ethernet PHY Interrupt ............................................ 4-20
4.6.2 Slave-Side Interrupts ............................................................. 4-20
4.6.2.1 ABORT Interrupt to MSC8122/26 ..................................... 4-21
4.6.2.2 TSI Interrupt ................................................................. 4-21
4.6.2.3 FALC56 Interrupt ........................................................... 4-21
4.6.2.4 Flash Ready Interrupt ...................................................... 4-21
4.6.2.5 MSC8122/26 RMII/MII Fast Ethernet PHY Interrupt .......... 4-21
4.6.2.6 MSC8122/26 Fast Ethernet Switch Interrupt ..................... 4-21

4.7 SDRAM .............................................................................. 4-22
4.7.1 SDRAM Programming .......................................................... 4-23
4.7.2 SDRAM Refresh ................................................................. 4-24

4.8 Flash ................................................................................... 4-24

4.9 TDM Port Peripherals ............................................................... 4-25
4.9.1 Time Slot Interchanger (TSI) ............................................... 4-26
4.9.2 E1/T1 Framer ................................................................... 4-27
4.9.3 CODEC .......................................................................... 4-27
4.9.3.1 CODEC Initialization ...................................................... 4-27

4.10 MSC8122/26 SMII/RMII/MII Fast Ethernet Switches ............. 4-29
4.10.1 Switching Ethernet Modes ................................................. 4-30
4.10.2 Fast Ethernet Clocking Modes ............................................ 4-32

4.11 MSC8122/26 I2C ................................................................ 4-33
4.12 RS-232 Transceivers ............................................................... 4-34

4.13 Slic-Slac Interface Support ..................................................... 4-36

4.14 Packet Peripherals (Host-Side) ............................................... 4-37
4.14.1 10/100 T-Base Ethernet PHY ............................................. 4-37

5 ADS Memory Maps and Bus Mapping

5.1 Memory Maps ....................................................................... 5-1
5.1.1 Host 60x Bus Mapping ....................................................... 5-1
5.1.2 Slave System Bus Mapping ................................................ 5-5

5.2 Host Memory Controller Registers Programming ................... 5-7
5.2.1 Host Machine A/B Mode Registers ..................................... 5-9
Slave Memory Controller Registers Programming .......................... 5-11
5.3.1 Slave Machine A/C Mode Registers Programming .................. 5-13
5.4 Expansion Connectors ...................................................... 5-15
5.4.1 60x Bus Expansion Connector J1 ....................................... 5-15
5.4.2 60x Bus Expansion Connector J2 ....................................... 5-19
5.4.3 MSC8122/26 Expansion Connector J3 .................................. 5-22
5.4.4 H.110 Bus Expansion Connector J4 .................................... 5-24
5.4.5 MSC8122/26 (Slic-Slac) Expansion Connector J5 .................... 5-27

6 ADS Control and Status
6.1 Board Control and Status Registers ........................................ 6-1
6.1.1 Board Control/Status Register 0 (BCSR0) .......................... 6-3
6.1.2 Board Control/Status Register 1 (BCSR1) .......................... 6-4
6.1.3 Board Control/Status Register 2 (BCSR2) .......................... 6-4
6.1.4 Board Control/Status Register 3 (BCSR3) .......................... 6-6
6.1.5 Board Control/Status Register 4 (BCSR4) .......................... 6-7
6.1.6 Board Control/Status Register Identification 5 (BCSR5) .......... 6-8
6.1.7 Board Control/Status Register Miscellaneous 6 (BCSR6) ......... 6-9
6.1.8 Board Control/Status Register Miscellaneous 7 (BCSR7) ........ 6-10
6.1.9 Board Control/Status Register Miscellaneous 8 (BCSR8) ......... 6-11
6.1.10 Board Control/Status Register Ethernet 9 (BCSR9) ............... 6-12
6.1.11 Board Control/Status Register Miscellaneous 10 (BCSR10) ...... 6-13

A Program Information
A.1 MSC8102/22/26ADS CPLD Code ......................................... A-1
A.2 MSC8122/26ADS CPLD Code ............................................. A-34
About This Book

This user’s guide describes engineering specifications for the MSC8122/26 application development system (ADS) board. The MSC8122/26ADS board targets either the StarCore® based MSC8122 or the MSC8126 processor. Each of these is a highly integrated system-on-a-chip device containing four StarCore SC140 digital signal processing (DSP) cores.

The MSC8122/26ADS board is intended to serve as a platform for software development for the Starcore processor environment. This manual lists and explains various features of the MSC8122/26ADS and describes how the board must be configured to develop and communicate code for each processor.

Before Using This Manual—Important Note

The information in this manual is subject to change without notice, as described in the disclaimers on the title page of this manual. Before using this manual, determine whether it is the latest revision and whether there are errata or addenda. To locate any published errata or updates associated with this manual or this product, refer to the world-wide web site listed on the back cover of this manual or call your local distributor or sales representative.

Audience and Helpful Hints

This manual is for software developers and applications programmers who are developing products using the Starcore MSC8122 or MSC8126 device.

Notational Conventions and Definitions

This manual uses the following notational conventions:

- **mnemonics**: Instruction mnemonics appear in lowercase bold.
- **italics**: Book titles in text are set in italics. In addition, italics are used for emphasis and to highlight the main items in bulleted lists. Variables and equations are also italicized.
- **0x**: Prefix to denote a hexadecimal number.
- **0b**: Prefix to denote a binary number.
Organization

Following is a summary and a brief description of the chapters of this manual:

- **Chapter 1, MSC8122/26ADS Overview.** Descriptive overview of main board specifications, MSC8103 and MSC8122/26ADS features.
- **Chapter 2, Board Configuration and Installation.** Lists and explains necessary dipswitch and jumper configurations, test points and bridges, functional LEDs, as well as describing installation options such as debug connection schema and standalone operation. Also gives brief instructions on how to connect the board.
- **Chapter 3, ADS Power and Connectors.** Device operating modes as determined by PLL settings, reset sources, PLL initialization example, and the system boot sequence.
- **Chapter 4, ADS Functional Description.** Describes the ADS memory map, memory mapping for host and slave buses, and how they may be controlled.
- **Chapter 5, ADS Memory Maps and Bus Mapping.** Describes board power connector, supply and buses. Describes all board peripheral connectors and expansion connectors. Explains the mapping of the expansion connector buses.
- **Chapter 6, ADS Control and Status.** Describes the ADS control and status registers and their programming.
- **Appendix A, Program Information.** Gives code listings for each version of the BCSR code contained in the ADS board CPLD.
Related MSC8122/26ADS Documentation

- MSC8122 Reference Manual
- MSC8126 Reference Manual
- MSC8122 Reference Manual
- MSC8126 Reference Manual
- MSC8102 Technical Data sheet
- MSC8103 Technical Data sheet
- MSC8122 Technical Data sheet
- MSC8126 Technical Data sheet
- SWITI Switching Device PEF24471 HTSI-XL Wired Communication Data sheet
- VIA Technologies VT6525A SMII Switch Data sheet
- VIA Technologies VT6510B RMII Switch Data sheet

Abbreviations and Acronyms

List of abbreviations and acronyms:

**ADS** - Application Development System

**ALU** - Arithmetic Logic Unit

**BCSR** - Board Control and Status Register

**CPM** - Communication Processor Module

**CW** - CodeWarrior® IDE for StarCore

**DIP** - Dual-In-Line Package

**DMA** - Direct Memory Access

**DSI** - Direct Slave Interface

**DSP** - Digital Signal Processor

**EOnCE** - Enhanced On-Chip Emulation

**GPCM** - Memory Controller General Purpose Chip-select Machine

**GPL** - General Purpose Line (associated with a UPM)

**HCW** - Hardware Configuration Word

**MII** - Media Independent Interface

**PHY** - Physical Layer
RMII - Reduced Media Independent Interface

SDRAM Machine - Memory Controller Synchronous Dynamic RAM Machine

SIU - System Interface Unit

SMII - Serial Media Independent Interface

TDM - Time Division Multiplex

UART - Universal Asynchronous Receiver Transmitter

UPM Machine - (Memory Controller) User Programmable Machine

ZD - Clock Zero Delay Buffer with internal PLL for skew elimination
MSC8122/26ADS Overview

The MSC8122/26ADS is based on the proven technology of the older MSC8102ADS. It functions just as its predecessor did and the software created using it is backwards compatible to the MSC8102, within chip limitations. The MSC8122/26ADS is identifiable by the sticker on the Altera BSCR chip, a customer programmable logic device (CPLD) in socket U38, which should indicate that the code contained is for the MSC8122/26ADS board. The processor in U15 can be identified as either MSC8122 or MSC8126 by a label affixed to cPCI connector J5, which is also the Slic-Slac connector.

1.1 MSC8122/26ADS

This manual describes engineering specifications for the MSC8122/26ADS. This applications development system (ADS) board targets the StarCore MSC8122 or MSC8126 chip. Each is a highly integrated system-on-a-chip device containing four StarCore SC140 digital signal processor (DSP) cores.

The MSC8122/26ADS board is intended to serve as a platform for software development for the MSC8122/26 processor environment. On-board resources and associated debugger enable developers to perform a variety of tasks: download and run code; set breakpoints; display memory and registers, and connect proprietary hardware via the expansion connectors. The MSC8122/26 processor enables incorporation of these features into selected systems.

In addition, the MSC8122/26ADS may be used as a demonstration tool. For example, application software may be burned into the flash memory of the ADS and run in exhibitions.

1.2 Board Specifications

Specifications for the MSC8122/26ADS board are provided in Table 1-1.

Table 1-1. MSC8122/26ADS Specifications

<table>
<thead>
<tr>
<th>Characteristics</th>
<th>Specifications</th>
</tr>
</thead>
<tbody>
<tr>
<td>Power requirements.</td>
<td>9-18V external DC power supply or RACK 12V supply for 12V max current 1.8A.</td>
</tr>
<tr>
<td>MSC8122/26ADS multicore (4xSC140 core) DSP.</td>
<td>Internal clock runs up to 500MHz @ 1.2V for the MSC8122/26.</td>
</tr>
<tr>
<td>MSC8103 (host-side) 60x bus (DSI).</td>
<td>MSC8103 60x bus running up to 100 MHz clock frequency.</td>
</tr>
</tbody>
</table>
1.3 ADS Features

The MSC8122/26ADS is based on 64-bit MSC8122 or MSC8126 chips. The system bus runs up to 166MHz for the MSC8122/26. The direct slave interface (DSI) bus runs up to 100MHz. The host controller MSC8103 and the 60x-bus interface with the slave MSC8122/26 and its DSI bus.
MSC8122/26ADS Features

The MSC8122/26ADS features are as follows:

- The DSI bus is a slave of the MSC8103 (with its 60x bus). This host controls the extended DSI address bus HA7-HA29 for MSC8122/26.
- The DSI is configurable to 32-bit when the system bus is 64-bit (default mode) or vice versa (DSI 64-bit/System bus 32-bit) or DSI 32-bit/System 32-bit/Ethernet.
- A Memory-Controller synchronous dynamic RAM (SDRAM) machine controls SDRAM memory size (32MB, 16MB, or 8MB on new boards) on the system bus. Memory size is dependant on the system bus configuration.
- The ADS features a 4MB @ 8-bit size Flash for configuration/boot/program storage.
- Four MSC8122/26 TDM ports connect to an Infineon TSI PEF24471 device. This device allows interconnecting of T1/E1 time-slots between Infineon FALC PEB2256 and Dual CODEC MT92303. It is possible to interface with H.110 TDM bus on the ADS J4 Compact PCI connector.
- An RS-232 Transceiver MAX3241 supports the universal asynchronous receiver/transmitter (UART) port operation of the MSC8122/26.
- A FETH PHY REALTEK RTL8208 supports SMII and enables MAC2PHY with the MSC8122/26 TDM/Ethernet multiplexed pins.
- A FETH PHY Davicom DM9161A supports RMII/MII and enables MAC2PHY with MSC8122/26 TDM/Ethernet multiplexed pins or DSI/Ethernet multiplexed pins.
- A FETH switch on the mezzanine board supports SMII, based on VIA VT6526A that can boot from on-board I²C EEPROM and enables MAC2MAC with MSC8122/26 TDM/Ethernet multiplexed pins. FETH MII-PHY populated.
- FETH switch on mezzanine board supports RMII based on VIA VT6510B that can boot from on-board I²C EEPROM. Enables MAC2MAC with MSC8122/26 TDM/Ethernet multiplexed pins or DSI/Ethernet multiplexed pins. FETH MII-PHY populated.
- MSC8122/26 Core power supply 0.9 - 2.0V set by potentiometer.
1.3.2 MSC8103 Features

The MSC8103 features are as follows:

- Acts as host for the MSC8122/26.
- On-board buffers disconnect MSC8103 from Slave-cPCI bus.
- SDRAM machine controls 16MB SDRAM on 60x bus.
- Features a 4MB @16-bit size Flash for configuration/boot/program storage.
- Communication processor module (CPM) ports are connected to 10/100 Base-T PHY Davicom (FCC2) and RS-232 Transceiver MAX3241(SCC1).
- An 8-bit Board Control and Status Register (BCSR) is required for ADS configuration.

1.3.3 Board Capabilities

The MSC8122/26ADS board capabilities are as follows:

- MSC8122/26 TDM/Ethernet or DSI/Ethernet pins connect MAC2MAC with FCC2 of the MSC8103.
- Programmable hard reset configuration for MSC8122/26 is executed from the Flash memory or the DSI bus; but, may also be forced from the BCSR.
- Boot MSC8122/26 from the host controller (via the DSI bus) or the system bus (Flash); but, can also be performed from the UART or TDM ports or I²C EEPROM.
- I²C bootable EEPROM for MSC8122/26 and/or MSC8103 in I²C multi-master configuration (Fairchild FM24C256FLEN - 2 x 32KB). Additional I²C EEPROM on the Mezzanine Ethernet switch found on the same I²C bus.
- High density logic analyzer connectors (MICTOR) facilitate measurement of MSC8122/26 signals. MSC8122/26 mictors reside on the buffered system bus enable measurements with special mode via buffers.
- As expansion connectors, PICMG2.16 CompactPCI® connector J3 transfers SMII/RMII/MII data from the MSC8122/26 Ethernet controller.
- MAX4372 Current-Sense Amplifier with voltage output supplies current measurement capability for MSC8122/26 PLL/IO/CORE power.
- Debugging performed via an external command converter connected to an enhanced on-chip emulation (EOnCE) 14-pin header. Debug through an external add-on board Command Converter via a parallel port connected to cPCI-J3 connector.
- Buffers disconnect MSC8103 from 60x bus thus enabling host from rack for the MSC8122/26.
• Via the backplane, the EOnCE debug chain allows the connection of additional ADS boards.
• Debug option is available in standalone mode (board on table) via the J3 connector. This enables EPP connection through an add-on connector with buffer to interface the PC parallel port.
• Select debug enable/disable and request options after reset. Processor EE pins enable and support noted options.
• Board identification and status can be read via the BCSR.
• Connect external pulse generator to MSC8122/26 clock input via the SMB-form RF-connector.
• Board configurations are available via the dual-in-line package (DIP) switch setting.
• Board features buttons for host and slave: Power-On-Reset, soft reset, hard reset, and abort.
• Board powered by a single 9-18V external DC supply with on-board reverse polarity protection.
• Board DC-DC SIP converters provided with input voltage. First converter parameters: 3.3V @ 16A 1%. Second converter parameters: 0.9-2.0V @ 16A or 10A or 5A (Depending on assembly) 1%.
• First DC-DC converter powers board’s 3.3V I/O and MSC8103 1.6V linear voltage regulator. Second converter powers MSC8122/26 core voltage. Potentiometer used to adjust voltage supplied by the converter to the board.
• Slic-Slac interface (via the rack) enables use of a 6-line communication board based on Voice-over-Broadband SLIC-SLAC™ chip set that implements a two-channel universal telephone line interface.
• SW option switch provides ten SW options controlled via BCSR.
• LEDs indicate power supply, peripheral enables, EE1-pin status and software signals.
Board Configuration and Installation

This chapter contains configuration and installation information for the MSC8122/26ADS board. The user can check the configuration of the ADS board against the factory settings shown in this manual to ensure proper board functionality.

**Caution:** Ensure that power is off or disconnected prior to reconfiguring an installed ADS board. Reconfiguring jumpers with the power on may damage system circuits.

### 2.1 Hardware Preparation

The first step in preparing the hardware is to unpack the ADS board. When unpacking the MSC8122/26ADS board from its shipping carton refer to the packing list to verify that all items are present and in good condition.

**Note:** If the ADS board arrives damaged, save all packing material and contact the carrier’s agent.

The location of MSC8122/26ADS board parts is shown in **Figure 2-1**.
Figure 2-1. MSC8122/26ADS Top-side Part Location Diagram
DIP Switches

ADS DIP switches are listed and described in Table 2-1, ADS Switches, on page 2-3. DIP switch locations are shown in Figure 2-1, MSC8122/26ADS Top-side Part Location Diagram, on page 2-2. The DIP switches discussed in the table are shown with their factory default positions.

Table 2-1. ADS Switches

<table>
<thead>
<tr>
<th>Designator and Purpose</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SW4</td>
<td>MODCK2</td>
<td>Switches SW4.1 and SW4.2 set the MODCK1,2 of the MSC8122/26 slave to PLL mode. When “ON”, the value is zero. If the clock-in frequency is 41.5MHz, the clock mode could be set to 5.</td>
</tr>
<tr>
<td></td>
<td>MODCK1</td>
<td>Switch SW4.3 (SYS &lt;-&gt; DSI) selects the power-up configuration source. When DSI is chosen then MSC8103 host is the configuration source. If SYS is selected then the configuration source is MSC8122/26 slave flash.</td>
</tr>
<tr>
<td></td>
<td>DSI</td>
<td>SW4.4 When in DBG, EE0 input is high at core reset allowing the core to enter debug mode immediately after negation of the slave HRESET. If not in DBG then after reset the core runs freely. Activate debug mode by toggling SW4/4.</td>
</tr>
<tr>
<td></td>
<td>DBG</td>
<td>Factory settings: SW4.1 - MODCK2 is “1” (OFF). SW4.2 - MODCK1 is “0” (ON). SW4.3 - is “0” (ON). Config source is from DSI. SW4.4 - is “0” (ON). Debug mode after slave hard reset.</td>
</tr>
<tr>
<td></td>
<td>CFG-S Ctrl</td>
<td></td>
</tr>
<tr>
<td>SW5</td>
<td>ChSel1</td>
<td>Switches SW5.1 and SW5.2, together with switch SW7.3, set the JTAG chain configuration as follows:&lt;br&gt;SW5.1 SW5.2 SW7.3&lt;br&gt;“ON” “ON” “ON” - Long System Slot in Rack (P16 + 8122 + 8103 + J1:cPCI)&lt;br&gt;“OFF” “ON” “OFF” - Rack Peripheral Slot (8122 + 8103 + J1:cPCI); not valid with P16&lt;br&gt;“OFF” “ON” “ON” - Local (P16 + 8122 + 8103)&lt;br&gt;“OFF” “OFF” “OFF” - J3 cPCI JTAG chain&lt;br&gt;Switch SW5.3 sets the slave system bus width. When “ON”. The MSC8122/26 system data bus is 64-bit and DSI bus is 32-bit. When “OFF”, system bus is 32-bit and DSI bus is 64-bit.</td>
</tr>
<tr>
<td></td>
<td>ChSel2</td>
<td>SW5.4 Reserved (Not Applicable).</td>
</tr>
<tr>
<td></td>
<td>SYS64</td>
<td>Factory settings: SW5.1.“OFF”, SW5.2 “ON” SW7.3 “ON” - P16 short JTAG chain. SW5.3 “ON” -MSC8122/26 system bus 64-bit.</td>
</tr>
<tr>
<td></td>
<td>JTAG_CHAIN</td>
<td>Note: The “Parallel-CC board” Rev PROTOTYPE plugs into the front of connector J3. The “Parallel-CC board Rev PILOT and REV-A connects with a flat cable to P16.</td>
</tr>
</tbody>
</table>
### SW6 Host Configuration Control (MSC8103)

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>ON</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
<td>ON</td>
</tr>
</tbody>
</table>

**MODCK**

- Switches SW6.1 through SW6.6 set the MSC8103 host MODCK1-6 to PLL mode. When "ON" the value is zero. Clock mode is set to 1 if the clock-in frequency is 50MHz.
- Switch SW6.7 sets the HCW1 source. When "ON" the source is host flash. If "OFF" the source is a BCSR programmed value.
- SW6.8 When in DBG ("ON"), EE0 input is high at core reset allowing the core to enter debug mode immediately after the negation of the host HRESET. If not in DBG ("OFF") then the core runs freely after reset. Activate debug mode by toggling SW6/8.

**Switches Setting**

- Factory settings:
  - SW6.1 "OFF" SW6.2 - 6.6 "ON" - Clock Mode is 1.
  - SW6.7 "OFF" - HCW1 originates from BCSR.
  - SW6.8 "ON" - Debug-enable after host hard reset.

### SW7 Software Option reading by MSC8103

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>ON</td>
<td>OFF</td>
<td>ON</td>
<td>ON</td>
</tr>
</tbody>
</table>

**ETH-ON**

- Switch SW7.1 sets ETH-ON function for 8122/26.
- Switch SW7.2 02: software option bit 1.
- SW7.3 Software option bit 2 is used for JTAG chain selection (see JTAG chain selection table).
- Switch SW7.4 is used for Host EE1 pin control after power-on-reset. When "ON", the value is zero.

**Switches Setting**

- Factory settings:
  - SW7.1 - ETH-ON is OFF.
  - SW7.2 - SW option 1 is ON.
  - SW7.3 - SW option 2 is ON.
  - SW7.4 - HEE1 is set to ON.

### SW8 Power Switch

- SW8 When the switch is "ON", the ADS is powered from an external 12V power supply via a P23 connector. When the switch is "OFF" the external power supply is disconnected.
- When the ADS is on the cPCI backplane and SW8 is OFF, the ADS is powered by an internal power supply via cPCI connectors.
## ADS Push Buttons

ADS push buttons are listed and described in Table. Push button locations are shown in Figure 2-1, MSC8122/26ADS Top-side Part Location Diagram, on page 2-2.

### Table 2-2. ADS Push Buttons

<table>
<thead>
<tr>
<th>Push Button</th>
<th>Appearance</th>
<th>Action Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>SW1 Slave Hard Reset</td>
<td><img src="image" alt="HRESET" /></td>
<td>Press SW1 for a hard reset of the MSC8122/26 (HRESET is asserted).</td>
</tr>
<tr>
<td>SW2 Slave Soft Reset</td>
<td><img src="image" alt="SRESET" /></td>
<td>Press SW2 for a soft reset of MSC8122/26. Clock and chip-select data as well as SDRAM contents are retained. Soft reset causes slave cores to run freely regardless of SW4/8 position.</td>
</tr>
<tr>
<td>SW3 Slave NMI (IRQ0)</td>
<td><img src="image" alt="ABORT" /></td>
<td>Press and release SW3 for MSC8122/26 non-maskable interrupt.</td>
</tr>
<tr>
<td>SW9 Power-On-Reset (PORESET)</td>
<td><img src="image" alt="PRESET" /></td>
<td>Press SW9 for the main power-on-reset for both processors-MSC8103 and MSC8122/26.</td>
</tr>
<tr>
<td>SW10 Host Hard Reset</td>
<td><img src="image" alt="HRESET" /></td>
<td>Press SW10 for an MSC8103 hard reset. This host hard reset produces a power-on-reset for the MSC8122/26.</td>
</tr>
<tr>
<td>SW11 Host Soft Reset</td>
<td><img src="image" alt="SRESET" /></td>
<td>Press SW11 for an MSC8103 soft reset. Clock and chip-select data, as well as SDRAM contents, are retained. Soft reset causes the host core to run freely regardless of the SW6/8 position.</td>
</tr>
<tr>
<td>SW12 Host NMI (IRQ0)</td>
<td><img src="image" alt="ABORT" /></td>
<td>Press and release SW12 for an MSC8103 non-maskable interrupt.</td>
</tr>
</tbody>
</table>
ADS Jumpers

The MSC8122/26ADS configuration jumpers and their settings are explained in the following sub-sections. Jumper locations are shown in Figure 2-1, MSC8122/26ADS Top-side Part Location Diagram, on page 2-2. The jumpers are shown with their factory default positions.

2.1.3.1 JP1-EE0 for Slave

For debug purposes, you can use jumper JP1 to measure or set the MSC8122/26 (slave) EE0 signal logic levels. If you wish to close JP1, place a jumper on pins 1 and 2, as shown below. The factory setting is “Open”.

![JP1 Open and Closed Diagram](image)

**Figure 2-2.** JP1-EE0s Control

2.1.3.2 JP2-CLKIN for Slave

The recommended clock mode for new designs is CLKIN mode: JP2 Jumper on IN and 2 = DRV - buffer output towards the MSC8122/26. The MSC8122 ADS is a development board and thus the default operation on it is CLKOUT mode as described below.

You can use jumper JP2 to measure or set the MSC8122/26 clock input signal source. Jumpering the various pins of JP2 produce the following effect:

- Jumper on IN and 2 = DRV - buffer output towards the MSC8122/26
- Jumper on IN and 4 = OSC - clock from the oscillator
- Jumper on IN and 6 = PLG - clock from Plug.
arious configurations of JP2 are shown below. Factory setting of JP2 places a jumper on the IN pin and pin 4 “OSC”.

![JP2-CLKIN Control](image)

**Figure 2-3.** JP2-CLKIN Control

### 2.1.3.3 JP3-SET MII PHY

When jumper JP3 is closed the MSC8122 RMII/MII PHY is in MII MODE. When the JP3 is open the PHY is in RMII Mode. The factory setting of JP3 is closed.

![JP3-Set MII PHY](image)

**Figure 2-4.** JP3-Set MII PHY

### 2.1.3.4 JP4-CT-Bus Master Reset Enable

If jumper JP4 is closed, then the MSC8122/26 can apply Master Reset to the CT-bus. You can close JP4 by jumpering pins 1 and 2. The factory setting for JP4 is “Open”.

![JP4-CT-Bus Master Reset Enable](image)

**Figure 2-5.** JP4-CT-Bus Master Reset Enable

### 2.1.3.5 JP5- 8102 Core Power

JP5 supplies core voltage and ground to the processor in socket U15 ONLY if it is an MSC8102. Failure to place jumpers on pins 1-2, 2-3, and 4-5, may create a malfunction for the MSC8102.
Remove jumpers from pins 1-2, 2-3, and 4-5, when processor in socket U15 is an MSC8122 or MSC8126. In all cases, the factory setting of jumper JP5 is all pins open.

**Figure 2-6.** JP5- 8102 Core Power

### 2.1.3.6 JP6-Slave Clock Reference

JP6 selects a phase lock loop (PLL) clock driver source from either the slave CLKOUT or the EXCLK-Oscillator. The latter can also supply the slave CLKIN. The factory setting of JP6 is closed pins 2 and 3. The states of JP6 pins are illustrated below in **Figure 2-7**.

**Figure 2-7.** JP6-Slave Clock Reference

### 2.1.3.7 JP7-DSI-Ethernet

Jumper JP7 selects the ADS DSI-Ethernet mode. Assembling the jumpers from the JP7 pins connects the slave’s DSI-Ethernet pins to the board’s Ethernet scheme. The factory setting for JP7 is all pins open.

**Figure 2-8.** JP7-DSI-Ethernet
2.1.3.8 JP8-DLL-ON

If jumper JP8 is closed, the MSC8122/26 should be configured to DLL-ON and thus the PLL-Buffer on the ADS is in Zero-Delay-Buffer mode. The factory setting of JP8 is open.

![JP8-DLL-ON](image)

**Figure 2-9.** JP8-DLL-ON

2.1.3.9 JP9-3V3-J3

If jumper JP9 is closed, the J3 connector receives 3.3V for external boards that are PICMG2.16 compliant. For non-PICMG2.16 rack the jumper JP9 should be open. The factory setting of JP9 is closed.

![JP9-3V3-J3](image)

**Figure 2-10.** JP9-3V3-J3

2.1.3.10 JP11-MAC to MAC-enable

If jumper JP11 is closed, the MSC8122/26 Ethernet port (for the CRS signal) connects to the MSC8103 TX_EN for running the MAC2MAC model. The factory setting of JP11 is open.

![JP11-MAC to MAC-enable](image)

**Figure 2-11.** JP11-MAC to MAC-Enable

2.1.3.11 JP12-Board Slave Voltage Select

Jumper JP12 settings indicate the voltage to be supplied for the processor in socket U15. Slave-core power settings are connected accordingly:

<table>
<thead>
<tr>
<th>JP12 Pins Closed</th>
<th>Setting</th>
<th>Voltage</th>
</tr>
</thead>
<tbody>
<tr>
<td>IN and 6</td>
<td>8102</td>
<td>1.22V-1.8V</td>
</tr>
<tr>
<td>IN and 4</td>
<td>8126/8122-2</td>
<td>1.15V-1.35V</td>
</tr>
<tr>
<td>IN and 2</td>
<td>8122-1</td>
<td>0.91V-1.21V</td>
</tr>
</tbody>
</table>
The factory setting for the MSC8122/26ADS, as shown in Figure 2-12, *JP12-Board Slave Voltage Select*, on page 2-10, is 8122-1, with the IN and 2 pins jumpered.

### 2.1.3.12 JP13-ADS Power-On-Reset Enable

When JP13 is closed the ADS exists in a power-on-reset state used solely for debugging purposes. The factory setting of JP13 is open.

#### Figure 2-13. JP10-ADS PON Reset Enable

### 2.1.4 ADS Testing

MSC8122/26ADS testing is detailed in the following sub-sections. The location of the various ADS test points and bridges is shown in Figure 2-1, *MSC8122/26ADS Top-side Part Location Diagram*, on page 2-2.

#### 2.1.4.1 TP13-Slave ClockOut Test Point

TP13 allows scope usage for measuring the MSC8122/26ADS system bus clock.

#### Figure 2-14. TP13-MSC8122/26 ClockOut
2.1.4.2 JS1-JS4-Current Consumption Measurement

Test points JS1-JS4 reside, respectively, on the following: PLL (JS1); core (JS2-JS3) and I/O-pin main flow (JS4). To measure current consumption remove the relevant JS using a soldering tool then connect a current meter (or shunt) using the shortest, thickest available wires. Alternatively, use a DVM/Oscilloscope to measure TP amplitude (TP8-PLL and TP91-Core and TP9-IO) and make current calculations.

*Caution:* The delicate task of removing JS1-JS4 should only be performed by a skilled technician. The MSC8122/26ADS may suffer permanent damage if an unskilled hand makes more than three attempts at the change.

2.1.4.3 JG1-JG5-GND Bridges

The MSC8122/26ADS has five bridges designated as GND. The bridges assist in general measurements and act as oscilloscope connections.

*Caution:* Only use INSULATED GND clips when connecting to a GND bridge—otherwise permanent damage may occur to the MSC8122/26ADS. Non-insulated clips that come into contact with surrounding “HOT” points may cause short-circuits.
### ADS LEDs

The MSC8122/26ADS operational LEDs are described in the following sub-sections. The LED locations are shown in **Figure 2-1, MSC8122/26ADS Top-side Part Location Diagram**, on page 2-2.

#### Table 2-3. ADS LEDs Functions

<table>
<thead>
<tr>
<th>LED</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>LD1 and LD2</td>
<td>RMII - (MSC8122/26) Ethernet Indicators</td>
</tr>
<tr>
<td></td>
<td>Yellow LD1 indicates a Full-Duplex or Collision. Yellow LD2 indicates either cable or link connection status. Two additional Ethernet LEDs are mounted inside the face of connector P8 RJ45: a green LED indicating an existing “LINK”, and a yellow LED serving as a 100Mbps indicator.</td>
</tr>
<tr>
<td>LD3</td>
<td>Slave Is Running</td>
</tr>
<tr>
<td></td>
<td>Green LD3 indicates that the MSC8122/26 is accessing the external system bus by reading/writing to the bus devices.</td>
</tr>
<tr>
<td>LD4</td>
<td>Host Is Running</td>
</tr>
<tr>
<td></td>
<td>Green LD4 indicates that the MSC8103 is accessing the external 60-x bus by reading/writing to the bus devices.</td>
</tr>
<tr>
<td>LD5 and LD6</td>
<td>Core Power Indicator</td>
</tr>
<tr>
<td></td>
<td>Green LD6 indicates the presence of host (1.5V) core power supply. A green LD5 indicates that a 8122/26 core power supply is present.</td>
</tr>
<tr>
<td>LD7 and LD8</td>
<td>Host Signaling LEDs</td>
</tr>
<tr>
<td></td>
<td>Orange LD7 and green LD8 are program-controlled and provide extra visibility on the host-running utility. The LEDs are lighted by setting bits BCSR0.6-7.</td>
</tr>
<tr>
<td>LD9</td>
<td>CODEC</td>
</tr>
<tr>
<td></td>
<td>Yellow LD9 indicates that the CODEC device is ready for programming. Bit BCSR0.3 controls the CODEC.</td>
</tr>
<tr>
<td>LD10, LD11 and LD15</td>
<td>Slave Core Power Supply Indicator</td>
</tr>
<tr>
<td></td>
<td>Green LED’s LD10, LD11 and LD15 indicate, respectively, core power supply modes for the slave cores and PLLs of MSC8122 or MSC8102 or MSC8122/26. LD15 is viable for both MSC8122 and MSC8126.</td>
</tr>
<tr>
<td>LD12 and LD13</td>
<td>Slave Signaling LEDs</td>
</tr>
<tr>
<td></td>
<td>Both orange LD12 and green LD13 are MSC8103 program-controlled. They provide extra visibility on the slave MSC8122/26 running utility. LED’s are lighted by setting bits BCSR0.4-5.</td>
</tr>
<tr>
<td>LD14</td>
<td>RS-232-1 Enable</td>
</tr>
<tr>
<td></td>
<td>Yellow LD14 indicates whether the host RS-232 Transceiver has been enabled (lighted) or disabled. RS-232 is controlled by bit BCSR1.6.</td>
</tr>
<tr>
<td>LD16 and LD21</td>
<td>GPIO Indicator</td>
</tr>
<tr>
<td></td>
<td>Green LD16 and LD21 indicate general purpose I/O usage (reserved).</td>
</tr>
<tr>
<td>LD17</td>
<td>SMII - (MSC8122/26) Ethernet Indicator</td>
</tr>
<tr>
<td></td>
<td>Green LD17 indicates SMII Mode for MSC8122/26.</td>
</tr>
<tr>
<td>LD18</td>
<td>ETH-ON Indicator</td>
</tr>
<tr>
<td></td>
<td>Green LD18 indicates Ethernet mode for MSC8122/26. It is set by writing a bit to BCSR7.5 and with dipswitch SW7/1.</td>
</tr>
<tr>
<td>LD19</td>
<td>RS-232-2 Enable</td>
</tr>
<tr>
<td></td>
<td>Activated yellow LD19 indicates that the slave RS-232 Transceiver is enabled. When it is disabled, MSC8122/26 UART pins may be used for off-board applications via the J5 expansion connector. RS-232 is controlled by bit BCSR1.7.</td>
</tr>
<tr>
<td>LD20</td>
<td>Slave in Debug Mode</td>
</tr>
<tr>
<td></td>
<td>Green LD20 indicates MSC8122/26 EE1 pin-debug acknowledge function. When it is lighted, the MSC8122/26 is either in debug mode or there is a high-level of EED input (debug request).</td>
</tr>
</tbody>
</table>
### 2.2 Installation Options

Configure the ADS board according to its environment.

#### 2.2.1 Debug Connection Schemes

On the ADS, host debug access is possible with any type of external Command Converter connected to the 14-pin JTAG/EOnCE connector, as illustrated in **Figure 2-15**.

![Figure 2-15. Host System Parallel-CC Debug Scheme](image_url)

<table>
<thead>
<tr>
<th>LED</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>LD22 External Power Indicator</td>
<td>When you are inserting the ADS into the cPCI rack, a lighted green LD22 indicates a 5V presence from the backplane power supply. The latter powers the ADS when the SW8 power switch is “OFF” (down position).</td>
</tr>
<tr>
<td>LD23 3.3V Power Indicator</td>
<td>A 12V power supply plugs into the P24 power connector on the board’s front side and powers the ADS when the SW8 power switch is “ON” (up position). A lighted green LD23 indicates a 3.3V power level (from the available 12V power supply).</td>
</tr>
<tr>
<td>LD24 and LD25 Host Ethernet Indicators</td>
<td>When it is burning, a yellow LD24 indicates a Full-Duplex or Collision. A lighted yellow LD25 indicates cable or link connection status. Two additional Ethernet LEDs are mounted inside connector P22: RJ45-a green LED indicating an existing “LINK”, and a yellow LED that serves as a 100Mbps indicator.</td>
</tr>
</tbody>
</table>

#### Table 2-3. ADS LEDs Functions (Continued)

<table>
<thead>
<tr>
<th>LED Function</th>
<th>14 Wire Flat Cable</th>
<th>Command Converter</th>
<th>P4</th>
<th>P5</th>
<th>P6</th>
<th>P16</th>
</tr>
</thead>
<tbody>
<tr>
<td>12V Power Supply</td>
<td>12V Power Supply</td>
<td>14 Wire Flat Cable</td>
<td>JTAG/EOnCE</td>
<td>MSC8122/26</td>
<td>MSC8103</td>
<td>MSC8122/26ADS</td>
</tr>
</tbody>
</table>

**Figure 2-15. Host System Parallel-CC Debug Scheme**
Alternatively, when the MSC8122/26ADS board, together with the MSC8122/26 are part of a multi-board system (DSP farm emulation) the JTAG signals are driven from the DSI host board via a cPCI expansion connector, as illustrated in Figure 2-17.

2.2.2 Standalone Operation

In standalone mode the host doesn’t control the ADS via the EOnCE port. Rather, the ADS may be connected to the host via alternate ports RS-232 (either from host or slave), Fast Ethernet, etc. Operating in standalone mode necessitates burning the application program into the board’s Flash.
memory (either the host’s or slave’s). For single board configuration the DSI bus connects with the host.

An MSC8103 60-x bus can be used as the host bus for a single board (ADS in system slot) standalone configuration, as illustrated in **Figure 2-18**.

![Standalone Configuration (System Slot)](image)

**Figure 2-18.** Standalone Configuration (System Slot)

Through J1/J2 connectors the ADS can provide a direct connection to the MSC8122/26 DSI bus on another ADS board in peripheral slot. To this purpose use a general 6U CompactPCI® rack
With a single system board (such as the ADS) and three ADS boards in peripheral slots, as illustrated in the figure below.

**Figure 2-19.** Standalone Slave Configuration (Peripheral Slot)
Connecting the ADS Board

To facilitate connection to the host computer and external power supply, place the MSC8122/26ADS board on a suitable work surface. The connector and switch locations are shown in Figure 2-1.

Connect the MSC8122/26ADS as follows:

1. Connect the PILOT / REV-A Parallel-CC or External Command Converter to the front EOnCE (P16) connector (for the host system debug scheme). The PROTOTYPE Parallel-CC board plugs into the front of connector J3. The Debug schemes are shown in Figure 2-15 and Figure 2-17.

2. For host system configurations, push DIP switches SW5-2 and SW7-3 to the “ON” position. For the JTAG-chain on backplane configuration ensure that DIP switches SW5-2 and SW7-3 are in the “OFF” position. The SW5 DIP switch settings are explained in DIP Switches on page 2-3.

3. Connect an external power supply to the MSC8122/26ADS P24 power jack. Alternatively, to apply power when the ADS is placed into backplane, insert a power supply plug into the P5 Power Jack on the Parallel-CC board. Insert the attached power cable to Parallel-CC board plug P6 on one side and to 8122ADS board P24 on the other.

4. Plug the external power supply into a 110-240V AC wall outlet.

5. On the MSC8122/26ADS board, turn the SW8 power switch to the “ON” position. LED LD23 “PWR” will light up.
ADS Power and Connectors

The location of the MSC8122/26ADS power, communication and expansion connectors is shown in Figure 2-1.

3.1 Power

Power is supplied to the MSC8122/26ADS via ADS connector P24. P24 is a 2mm Power Jack RAPC722 providing connection to an external +12VDC@1.8A power supply.

Voltage on the MSC8122/26 is regulated by the settings of ADS jumpers JP5 and JP12, as discussed in Chapter 2 of this manual.

3.1.1 Power Supply

The MSC8122/26ADS requires a power supply of +12V DC @ 1.8A max. To apply power directly to the ADS board, insert enclosed power supply plug into the board’s P24 power jack as shown in Figure 3-1.

Note: It is permissible to connect user hardware applications to the ADS board using cPCI Expansion Connectors J1-J5. Take additional power requirements into consideration when using an enclosed power supply.
the MSC8122/26ADS is within the cPCI rack and ADS switch SW8 is in the “OFF” (down) position, then power may be applied from a built-in power supply, as shown in Figure 3-2. A burning “PWR” LED on the lower front panel indicates that backplane power is “ON”.

![Figure 3-2. Power on Backplane](image-url)
3.1.2 Power Buses

There are three power buses serving the MSC8122/26ADS:

- 3.3V for on-board logic and I/O of two processors.
- 1.6V for the host MSC8103 core and PLL.
- 0.9-1.7V for slave MSC8122/26 core and PLL.

Core power supplies provide filtering voltage to both processor PLLs. MAX4372 Current-Sense Amplifier with voltage output supplies slave power bus with current measurement capability for the MSC8122/26 PLL/IO/CORE power.

Main 3.3V supply is a high-performance DC-to-DC SIP converter providing up to a 16A current for on and off-board loads. A second DC-to-DC SIP converter resides on the 12V rail and supplies a variant core voltage to the MSC8122/26 (0.9-2.0V). Additionally, a 1.5V linear voltage regulator produces quiet voltage for the MSC8103 processor. A suppressor diode protects the board from reverse voltage currents.

Slave core-power supply enables four variant modes; modes set by a four-phase jumper (JP12) that also sets the slave-chip (MSC8102 or MSC8122/26):

- 0.91V-1.21V (8122)
- 1.15V-1.35V (8122/26)
- 1.22V-1.8V (8102)
3.3V power bus connects to expansion connectors J1/J2 to support off-board application development. Thus external logic, mounted on a prototype board, is powered directly from the main ADS.

### 3.2 Interconnect Signals

MSC8122/26ADS interconnects with external devices via the following connectors:

- P1, P5 - 3.5mm stereo phone jack
- P2 - RF SMB Socket
- P3, P4, P7, P9, P11, P12, P15, P18, P19, P21 - 10 Logic Analyzer MICTOR connectors
• P6 - RJ45 for E1/T1 port
• P10 - D-type9 90° female RS232 port
• P13, P20 - Ethernet Mezzanine board
• P14 - 3-pin header for Slave’s UART port
• P13 - PLD, Altera In-System Programming (ISP)
• P16 - JTAG/EOnCE
• P17 - 2-pin header for EE1 MSC8122/26 debug request chain
• P8, P22 - RJ45 100/10 Base-T Ethernet port
• J1-J5 - expansion connectors

3.2.1 Stereo Phone Jack Connectors

Stereo phone jack connectors (P1 and P5) are 5-pin connectors whose pinout is shown in Table 3-1.

Table 3-1. Stereo Phone Connectors (P1 and P5) Interconnect Signals

<table>
<thead>
<tr>
<th>Pin No.</th>
<th>Signal Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>COMMON</td>
<td>Analog Ground. Connect to AGND1 plane.</td>
</tr>
<tr>
<td>2</td>
<td>LEFT</td>
<td>Left channel</td>
</tr>
<tr>
<td>3</td>
<td>RIGHT</td>
<td>Right channel</td>
</tr>
<tr>
<td>10</td>
<td>SPEAKER LEFT</td>
<td>Not connected</td>
</tr>
<tr>
<td>11</td>
<td>SPEAKER RIGHT</td>
<td>Not connected</td>
</tr>
</tbody>
</table>

3.2.2 SMB Connector

RF Subminiature Coaxial Connector P2 acts as an MSC8102 external clock source when jumper JP2 is closed IN-PLG. Connector P2 is not used when the slave is an MSC8122/26.

3.2.3 Logic Analyzer Connectors

Connectors P3, P4, P7, P9, P11, P12, P15, P18, P19, P21 are 38-pin, SMT, high density, matched impedance connectors (mictors) made by AMP (2-767004-2) and used for Logic Analyzer measurement. The connectors contain unbuffered MSC8103 signals, MSC8122/26 Buffered Bus signals and other control signals on the ADS. Connector pinouts are shown in the ADS schematics.
Mezzanine Header Ethernet Connectors

Two Samtec-QSE (P13 and P20) standard headers carry the Ethernet signals for the Mezzanine board.

3.2.5 RJ45 E1/T1 Line Connector

The E1/T1 line connector (P6), a twisted-pair compatible connector, is implemented with a 90°, 8-pin, RJ45 connector. Connector P6 signals are described in Table 3-2.

Table 3-2. E1/T1 Line Connector Interconnect Signals

<table>
<thead>
<tr>
<th>Pin No.</th>
<th>Signal Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>RX1+</td>
<td>Twisted-Pair Receive Data positive input from the MSC8122/26ADS.</td>
</tr>
<tr>
<td>2</td>
<td>RX1-</td>
<td>Twisted-Pair Transmit Data positive input from the MSC8122/26ADS.</td>
</tr>
<tr>
<td>3</td>
<td>GND</td>
<td>Digital Ground plane.</td>
</tr>
<tr>
<td>4</td>
<td>TX1+</td>
<td>Twisted-Pair Transmit Data positive output from the MSC8122/26ADS.</td>
</tr>
<tr>
<td>5</td>
<td>TX1-</td>
<td>Twisted-Pair Transmit Data negative output from the MSC8122/26ADS.</td>
</tr>
<tr>
<td>6</td>
<td>GND</td>
<td>Digital Ground plane.</td>
</tr>
<tr>
<td>7</td>
<td>N.C.</td>
<td>Not Connected.</td>
</tr>
<tr>
<td>8</td>
<td>N.C.</td>
<td>Not Connected.</td>
</tr>
</tbody>
</table>

3.2.6 Slave UART Port Connector

The slave UART RS-232 port connector (P10) is a 9-pin, 90°, female D-type shielded connector. Its signals are outlined in Table 3-3.

Table 3-3. UART RS-232 Interconnect Signals

<table>
<thead>
<tr>
<th>Pin No.</th>
<th>Signal Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1,6</td>
<td>DTR</td>
<td>Data Terminal Ready output from the MSC8122/26ADS shorted to pin 1.</td>
</tr>
<tr>
<td>2</td>
<td>TXD</td>
<td>Transmit Data output from the MSC8122/26ADS.</td>
</tr>
<tr>
<td>3</td>
<td>RXD</td>
<td>Receive Data input to the MSC8122/26ADS.</td>
</tr>
<tr>
<td>4</td>
<td>DSR</td>
<td>Data Set Ready input to the MSC8122/26ADS.</td>
</tr>
<tr>
<td>5</td>
<td>GND</td>
<td>Ground signal of the MSC8122/26ADS.</td>
</tr>
<tr>
<td>7</td>
<td>CTS</td>
<td>Clear To Send input to the MSC8122/26ADS.</td>
</tr>
<tr>
<td>8</td>
<td>RTS</td>
<td>Ready To Send output from the MSC8122/26ADS.</td>
</tr>
<tr>
<td>9</td>
<td>N.C.</td>
<td>Not connected</td>
</tr>
</tbody>
</table>
### Altera’s In-System Programming (ISP) Connector

Connector P23 is a 10-pin generic 0.050" pitch header connector providing in-system programming capability for Altera CPLD devices-for board programmable logic. P23 pinout is seen in Table 3-4.

**Table 3-4. ISP Connector-Interconnect Signals**

<table>
<thead>
<tr>
<th>Pin No.</th>
<th>Signal Name</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>TCK</td>
<td>I</td>
<td>ISP test port clock. This clock shifts in/out data to/from the programmable logic JTAG chain.</td>
</tr>
<tr>
<td>2</td>
<td>GND</td>
<td>P</td>
<td>Digital GND. Main GND plane.</td>
</tr>
<tr>
<td>3</td>
<td>TDO</td>
<td>O</td>
<td>ISP Transmit Data Output. Programmable logic of the JTAG serial data output driven by falling edge of TCK.</td>
</tr>
<tr>
<td>4</td>
<td>VCC</td>
<td>P</td>
<td>Connect to 3.3V power supply bus for feeding an external programmer logic.</td>
</tr>
<tr>
<td>5</td>
<td>TMS</td>
<td>I</td>
<td>ISP Test Mode Select. This signal is qualified with TCK - changes the state of the programmable logic JTAG machine.</td>
</tr>
<tr>
<td>6</td>
<td>N.C.</td>
<td>-</td>
<td>Not Connected.</td>
</tr>
<tr>
<td>7</td>
<td>N.C.</td>
<td>-</td>
<td>Not Connected.</td>
</tr>
<tr>
<td>8</td>
<td>N.C.</td>
<td>-</td>
<td>Not Connected.</td>
</tr>
<tr>
<td>9</td>
<td>TDI</td>
<td>I</td>
<td>ISP Transmit Data In. Programmable logic of the JTAG serial data input.</td>
</tr>
<tr>
<td>10</td>
<td>GND</td>
<td>P</td>
<td>Digital GND. Main GND plane.</td>
</tr>
</tbody>
</table>

### 3.2.8 Host Debug EOnCE (SYS) Connector

Connector P16 is a Motorola standard DSP JTAG/EOnCE connector featuring a 14-pin, 90° 2-row header connector with key. Host debug accesses all processors joined by the JTAG chain via connector P16. Pinout seen in Table 3-5, *Main EOnCE Connector-Interconnect Signals*, on page 3-7.

**Warning:** When driven by an external tool, pin 9 (HRESET<sub>hb</sub>) MUST be driven with an Open Drain gate. Failure to do so may result in permanent damage to the MSC8103 and/or to MSC8122/26ADS logic.

**Table 3-5. Main EOnCE Connector-Interconnect Signals**

<table>
<thead>
<tr>
<th>Pin No.</th>
<th>Signal Name</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>TDI&lt;sub&gt;h&lt;/sub&gt;</td>
<td>I</td>
<td>Transmit Data In. MSC8103’s JTAG serial data input - sampled on the rising edge of TCK.</td>
</tr>
<tr>
<td>2</td>
<td>GND</td>
<td>P</td>
<td>Digital GND. Main GND plane.</td>
</tr>
<tr>
<td>3</td>
<td>TDO&lt;sub&gt;hc&lt;/sub&gt;</td>
<td>O</td>
<td>Transmit Data Output. DSP JTAG serial data output driven by falling edge of TCK.</td>
</tr>
</tbody>
</table>
### Table 3-5. Main EOnCE Connector-Interconnect Signals (Continued)

<table>
<thead>
<tr>
<th>Pin No.</th>
<th>Signal Name</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>GND</td>
<td>P</td>
<td>Digital GND. Main GND plane.</td>
</tr>
<tr>
<td>5</td>
<td>TCKhc</td>
<td>I</td>
<td>Test port clock. The clock shifts in/out data to/from JTAG logic. Data is driven on the falling edge of TCK and sampled both internally and externally on its rising edge.</td>
</tr>
<tr>
<td>6</td>
<td>GND</td>
<td>P</td>
<td>Digital GND. Main GND plane.</td>
</tr>
<tr>
<td>7</td>
<td>N.C.</td>
<td>-</td>
<td>Not Connected.</td>
</tr>
<tr>
<td>8</td>
<td>KEY</td>
<td>-</td>
<td>No pin in connector. Serves for correct plug insertion.</td>
</tr>
<tr>
<td>9</td>
<td>HRESEThb</td>
<td>I/O,P.U.</td>
<td>When asserted by external H/W a MSC8103 Hard-Reset sequence is generated. During that sequence the MSC8103 asserts for 512 system clocks. Pulled-up on the ADS using a 1KW resistor. <strong>When driven by an external tool it MUST be driven with an Open Drain gate. Failure to do so may result in permanent damage to the MSC8103 and/or to ADS logic.</strong></td>
</tr>
<tr>
<td>10</td>
<td>TMSSh</td>
<td>I</td>
<td>Test Mode Select. Signal is qualified with TCK in a same manner as TDI - changes the state of the JTAG machines. The line is internally pulled-up by the MSC8103.</td>
</tr>
<tr>
<td>11</td>
<td>VDD</td>
<td>P</td>
<td>Connect to 3.3V power supply bus via protection resistor. May be used for Command Convertor power.</td>
</tr>
<tr>
<td>12</td>
<td>N.C.</td>
<td>-</td>
<td>Not Connected.</td>
</tr>
<tr>
<td>13</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>TRSThb</td>
<td>I</td>
<td>Test port reset. When signal is active (low) it resets the JTAG logic. The line is pulled-down on the ADS with a 2.2KW resistor in order to provide continuous JTAG logic reset when the connector is unplugged.</td>
</tr>
</tbody>
</table>

A front view of Connector P16 is shown in **Figure 3-4**.

![Figure 3-4. EOnCE Connector, Front View](image-url)
EE1 Connector

The EE1 connector (P17), a 2-pin header on the front side of the ADS, provides MSC8122/26 debug request connectivity on other ADS boards. P17 pinout is seen in Table 3-6.

Table 3-6. EE1 Chain Connector

<table>
<thead>
<tr>
<th>Pin No.</th>
<th>Signal Name</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>EE1in</td>
<td>I</td>
<td>Debug Request to the on-board MSC8122/26.</td>
</tr>
<tr>
<td>2</td>
<td>EEout</td>
<td>O</td>
<td>Debug Request to MSC8102/22/26 mounted on another board (ADSs).</td>
</tr>
</tbody>
</table>

3.2.10 Ethernet Port Connectors

Facilitating MSC8122/26ADS to/from external MSC8102/22/26 communication, two Ethernet connectors, P8 and P22, are twisted-pair (10-Base-T) compatible connectors implemented via a 90°, 8-pin, RJ45 connector. Signals are described in Table 3-7.

Table 3-7. P6 and P22: Ethernet Port Interconnect Signals

<table>
<thead>
<tr>
<th>Pin No.</th>
<th>Signal Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>TPTX(GRAY)</td>
<td>Twisted-Pair Transmit Data positive <strong>output</strong> from the MSC8122/26ADS.</td>
</tr>
<tr>
<td>2</td>
<td>TPTX~(BROWN)</td>
<td>Twisted-Pair Transmit Data negative <strong>output</strong> from the MSC8122/26ADS.</td>
</tr>
<tr>
<td>3</td>
<td>TPRX(YELLOW)</td>
<td>Twisted-Pair Receive Data positive <strong>input</strong> to the MSC8122/26ADS.</td>
</tr>
<tr>
<td>4, 5</td>
<td>(RED,GREEN)</td>
<td>Bob Smith terminated on the MSC8122/26ADS.</td>
</tr>
<tr>
<td>6</td>
<td>TPRX~(BLACK)</td>
<td>Twisted-Pair Receive Data negative <strong>input</strong> to the MSC8122/26ADS.</td>
</tr>
<tr>
<td>7, 8</td>
<td>(BLUE,ORANGE)</td>
<td>Bob Smith terminated on the MSC8122/26ADS.</td>
</tr>
</tbody>
</table>
ADS Expansion Connectors

Board expansion cPCI connectors (J1-J5) carry MSC8103 60x bus signals and MSC8122/26 peripheral signals for off-board connection. Expansion Connector interfaces are shown in Figure 3-5 and Figure 3-6, below.

Figure 3-5. cPCI J1 and J4: Connector View

Figure 3-6. cPCI J2 and J5: Connector View

3.3.1 Expansion Connector J1

The J1 cPCI expansion connector links the MSC8122/26ADS to the 60x Bus Expansion. This connector’s 60x bus expansion pins and signals are listed in Table 5-9, 60x Bus System Expansion, on page 5-15.

3.3.2 Expansion Connector J2

The J2 cPCI expansion connector links the MSC8122/26ADS to the 60x Bus Expansion. This connector’s 60x bus expansion pins and signals are listed in Table 5-10, 60x Bus System Expansion, on page 5-19.
Expansion Connector J3

The J3 cPCI expansion connector expands the MSC8122/26ADS signals. This connector’s pins signals and attributes are listed in Table 5-11, MSC8122/26 Signal Expansion Connector, on page 5-22.

3.3.4 Expansion Connector J4

The J4 cPCI connector links the MSC8122/26ADS to the H110 signal bus. The pins, signals, and attributes for the J4 H110 Signal bus are listed in Table 5-12, H.110 Bus, on page 5-24.

3.3.5 Expansion Connector J5

The J5 signal expansion connector is the Slic-Slac connector. The pins, signals, and attributes for J5 are listed in Table 5-13, MSC8122/26 Slic-Slac Connector, on page 5-27.
This chapter discusses the MSC8122/26ADS block diagram and the functionality it describes.

### 4.1 MSC8122/26ADS Block Diagram

The MSC8122/26ADS consists of two sides: the host based on MSC8103 (as host controller), and the slave based on the MSC8122 or MSC8126 chip, as shown in Figure 4-1.

![Figure 4-1. ADS Block Diagram](image-url)
The MSC8122/26 processor interfaces with the host via a DSI bus that connects asynchronously to the 60x bus of the MSC8103. Certain board connectors enhance board external communication, connectivity, expansion, and debug as follows:

- External tools access the DSI bus via J1/J2 expansion connectors.
- Connector J3 carries parallel port signals for JTAG debug via PC and PICMG2.16 analog signals from SMII/PHY on the ADS.
- Connector J4 interfaces with the standard H.110 bus (ECTF H.110 Specification).
- Connector J5 provides connectivity to four MSC8122/26ADS TDM ports: UART, Timers, Interrupts and GPIO (including the I²C). This is the Slic-Slac connector.

The DSI 32-bit data bus (default) is reconfigurable during power-on-reset to 64-bit; simultaneously the 64-bit system data bus becomes 32-bit. Another configuration is DSI 32-bit/system data 32-bit/Fast Ethernet.

The address bus switch for the SDRAM is located on the MSC8122/26ADS system bus. The switch provides correct connections for both the 64-bit and 32-bit data bus configurations.
Reset and Reset-Configuration

ADS reset signals are produced by either the MSC8103 or the MSC8122/26 controller. Reset circuitry details are shown in Figure 4-2:

![Figure 4-2. ADS Reset Diagram](image)

### 4.2.1 Power-On-Reset

After power-up, the ADS main power-on-reset button initializes both host processor and slave state. The initialization involves dedicated logic using MAX6827. This dual uP supervisor, responsible for 3.3V, 1.6V and 1.2V power rails, ensures that processors are supplied with nominal power. Its open-drain output circuitry allows off-board RESET sources such as the one-shot. The MSC8122/26 PORESET impulse is driven by BCSR logic and is either asserted during MSC8103 HRESET, or activated by programming the BCSR register. See Table 6-3, BCSR2 Register Description, on page 6-5 for details.

Main power-on-reset is asserted to MSC8103 for 350msec. This provides sample time to cover voltage regulator stabilization. Power-on-reset is either generated manually or by a dedicated push-button connected to the manual reset input of the supervisor device.

### 4.2.2 Host Power-On-Reset Configuration

After the power-on-reset sequence, the MSC8103 samples MODCK(1:3) together with three additional clock configuration bits. Other MSC8103 system clock modes (dsp core, cpm, 60x...
Following power-on-reset is a hard reset sequence that includes the configuring of multiple options. MODCK bits are sampled during each hard reset sequence and configuration. MODCK bits exert influence only once, directly after power-on-reset. Thereafter, if a hard reset sequence is entered then the MODCK bits, though sampled, have no impact.

### 4.2.3 Slave Power-On-Reset Configuration

After the power-on-reset sequence, the MSC8122/26 samples MODCK(1:2) together with three additional clock configuration bits. Other MSC8122/26 system (dsp cores, 60x bus) clock modes are also set. MODCK(1:2) combinations are controlled by DIP Switches SW4/1-2 and buffered via BCSR logic.

At power-on-reset some pins are sampled. Either MSC8103 or MSC8122/26 controllers produce ADS reset signals. Reset circuitry details are shown in Figure 4-2. The MSC8122/26 is also configurable via host MSC8103. The user should write the desired power-on-reset configuration to BCSR and set bit BCSR1.0 (RECONF) to 0. This invokes secondary power-on-reset for the MSC8122/26.

Following power-on-reset is a hard reset sequence that includes the configuring of multiple options. MODCK bits are sampled during hard reset configuration and after the first hard reset sequence following a power-on-reset. During subsequent hard reset sequences, MODCK bits are sampled but deemed irrelevant.

### 4.2.4 Hard/Soft Reset Capabilities

Using a command from the host debugger system, CodeWarrior (CW) IDE, and debug-mode host-processor, hard reset is available via JTAG/OnCE. Asserting host HRESET automatically generates a MSC8122/26 PORESET impulse via BCSR logic. This occurs despite host and slave processor reset systems being electrically isolated.

Dedicated push buttons facilitate manual hard and soft reset for processors MSC8103 and MSC8122/26. These buttons enable run-time reset when the command converter is disconnected from the ADS.

HRESET lines may be internally driven by the MSC8103 and MSC8122/26, but must be driven to the MSC8103 and MSC8122/26 with an open-drain gate.

Generating hard reset completely resets the registers of both processors. For example, hard reset configuration is re-sampled and all registers, including memory controllers but excepting PLLs, are reset. The reset results in a loss of dynamic memory content.
A maximum of four off-board-slave DSP hard-reset signals are present in inputs from the BCSR register. This allows for the separate monitoring and assertion of MSC8122/26 slave resets through programming.

### 4.2.5 Hard Reset Configuration

Hard reset configuration is performed for both the host (MSC8103) and slave (MSC8122/26) processors.

#### 4.2.6 Host Hard Reset Configuration

Hard reset samples the HCW (Hardware Configuration Word) when applied both externally and internally to MSC8103. The HCW might originate from an internal default if the RSTCONF signal is negated during an HRESET assertion. If the RSTCONF signal is asserted along with the HRESET then the HCW might be taken from the MSC8103 Flash memory (MS 8-bits of the data bus) or from the BCSR board register\(^1\).

If the MSC8103 Flash has been tampered with, then the default HCW will be taken from either the Flash or the BCSR. A dedicated DIP-SW determines if the source of the default HCW will be the Flash or the BCSR.

---

\(^1\) In general, reading from any device residing on CS0.
The hard reset sequence the configuration master reads, one byte at a time, the Flash (or BCSR) memory at addresses 0x00, 0x08, 0x18, 0x20. This is required to assemble the 32-bit configuration word.

The effect of the host pins on the power-on-reset configuration is shown in **Table 4-1**.

**Table 4-1. Effect of Host (MSC8103) Pins On the Power-On-Reset Configuration**

<table>
<thead>
<tr>
<th>Name of Signal</th>
<th>Value</th>
<th>Mode</th>
<th>Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>RSTCONF</td>
<td>0</td>
<td>Boot master from 60x bus.</td>
<td>Pulled-down permanently.</td>
</tr>
<tr>
<td>EE0</td>
<td>0</td>
<td>Core running free after hard reset.</td>
<td>Dipswitch SW6/8 setting (debug enable/disable) via BCSR buffer.</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>Core enters debug mode after hard reset.</td>
<td></td>
</tr>
<tr>
<td>EE1(HPE)</td>
<td>0</td>
<td>Host port disable - 60x bus configured for 64-bit.</td>
<td>Pulled-down permanently.</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>Core enters debug mode after hard reset.</td>
<td></td>
</tr>
<tr>
<td>EE<a href="">4:5</a></td>
<td>‘00’</td>
<td>Boot source resides at 60x bus.</td>
<td>Pulled-down permanently.</td>
</tr>
</tbody>
</table>

DIP switch SW6/8 controls the EE0 pin. This provides a post-reset manual debug capability (debug request).

**Table 4-2** describes the field values of the hard reset configuration word.

**Table 4-2. MSC8103 Hard Reset Configuration Word (HCW1)**

<table>
<thead>
<tr>
<th>Field</th>
<th>Data Bus Bits</th>
<th>Prog Value [Bin]</th>
<th>Implication</th>
<th>Offset In HCW source [Hex]</th>
<th>Value [Hex]</th>
</tr>
</thead>
<tbody>
<tr>
<td>EARB</td>
<td>0</td>
<td>‘0’</td>
<td>Internal arbitration selected.</td>
<td>0</td>
<td>28</td>
</tr>
<tr>
<td>EXMC</td>
<td>1</td>
<td>‘0’</td>
<td>Internal memory controller. CS0 active at system boot.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>IRQ7INT</td>
<td>2</td>
<td>‘1’</td>
<td>INT_OUT function is active.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EBM</td>
<td>3</td>
<td>‘0’</td>
<td>Single-chip bus mode is assumed.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BPS</td>
<td>4:5</td>
<td>‘10’</td>
<td>16-bit boot port size for both Flash memory and BCSR.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SCDIS</td>
<td>6</td>
<td>‘0’</td>
<td>SC140 enabled.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ISPS</td>
<td>7</td>
<td>‘0’</td>
<td>Internal space port-size for external master access is 64-bit. Currently not of concern as the present board configuration doesn’t support this feature.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
4.2.7 Slave Reset Configuration

The primary hardware setting, using two DIP switches, configures the MSC8122/26 after power-up. The following modes are available:

- **System Mode**: the configuration word is fetched and boot is achieved via the system bus in master mode (DIP switch SW4/3 is in “SYS” position).
- **DSI Mode**: configuration word is written by the host; boot is completed via the system bus in slave mode (DIP switch SW4/3 in “DSI” position).
- **DIP switch SW7/3** selects a DSI bus width of either 64- or 32-bits.

<table>
<thead>
<tr>
<th>Field</th>
<th>Data Bus Bits</th>
<th>Prog Value [Bin]</th>
<th>Implication</th>
<th>Offset In HCW source [Hex]</th>
<th>Value [Hex]</th>
</tr>
</thead>
<tbody>
<tr>
<td>IRPC</td>
<td>8:9</td>
<td>'00'</td>
<td>Interrupt pin configuration. RES/BADDR(29)/IRQ2, RES/BADDR(30)/IRQ3, RES/BADDR(31)/IRQ5 are selected as RES (not connect).</td>
<td>8</td>
<td>00</td>
</tr>
<tr>
<td>DPPC</td>
<td>10:11</td>
<td>'00'</td>
<td>Data parity pin configuration as IRQ[1:7].</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NMIOUT</td>
<td>12</td>
<td>'0'</td>
<td>Core services NMI interrupt.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ISB</td>
<td>13:15</td>
<td>'000'</td>
<td>I MMR initial value 0xF000_0000, i.e., initially the internal space resides at this address.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Reserved</td>
<td>16</td>
<td>'0'</td>
<td>Reserved. Non-functional cleared bit.</td>
<td>10</td>
<td>32</td>
</tr>
<tr>
<td>BBD</td>
<td>17</td>
<td>'0'</td>
<td>Bus busy pins set as the following: ABB/IRQ2 pin is ABB and DBB/IRQ3 pin is DBB.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MMR</td>
<td>18:19</td>
<td>'11'</td>
<td>External bus request masked.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Reserved</td>
<td>20:21</td>
<td>'00'</td>
<td>Reserved. Must be cleared.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TCPC</td>
<td>22:23</td>
<td>'10'</td>
<td>Transfer code pins are configured following PORESET: MODCK1/BKSEL(0)/TC(0) as BKSEL0 MODCK2/BKSEL(1)/TC(1) as BKSEL1 MODCK3/BKSEL(2)/TC(2) as BKSEL2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BC1PC</td>
<td>24:25</td>
<td>'00'</td>
<td>Buffer control 1-pin configuration BCTL1/DBG_DIS functions as BCTL1.</td>
<td>18</td>
<td>1E</td>
</tr>
<tr>
<td>Reserved</td>
<td>26</td>
<td>'0'</td>
<td>Reserved. Should be cleared.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DLLDIS</td>
<td>27</td>
<td>'1'</td>
<td>DLL off.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MODCK_H</td>
<td>28:30</td>
<td>'111'</td>
<td>High-order MODCK bits. Clock configuration scheme 57 for MSC8103.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Reserved</td>
<td>31</td>
<td>'0'</td>
<td>Reserved. Should be cleared.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Notes**: 1. Applied only once—after power-on-reset.

---

Table 4-2. MSC8103 Hard Reset Configuration Word (HCW1) (Continued)
• Ethernet Mode: DIP switch SW7-1 selects Ethernet Mode together with ETHSEL bit in MSC8122/26 configuration word.

For further details see Table 4-3, Pins Dependant Upon Power-On-Reset Configuration, on page 4-9.

The MSC8122/26 is reconfigured on demand when the host processor programs the BCSR register with the desired configuration setting and invokes a power-on-reset or hard reset sequence for the MSC8122/26. This setting is called a secondary reset configuration. The MSC8122/26 hard reset configuration is manually achieved by pressing the hard reset push button (HRESETs). Secondary reset configuration is shown in Figure 4-3.

![Figure 4-3. Secondary Reset Configuration Details](image-url)
The MSC8122/26 reset configuration setting, shown in Table below, can be loaded in either of two ways. It can be loaded by the host; or, depending upon the DSI or SYS mode, it can be taken from the device connected to the CS0 of the MSC8122/26ADS system bus (slave-side Flash).

**Table 4-3. Pins Dependant Upon Power-On-Reset Configuration**

<table>
<thead>
<tr>
<th>Name of signal</th>
<th>Value sampled at pins</th>
<th>Primary Mode</th>
<th>Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>MODCK[1–2]</td>
<td>‘00’ or ‘01’</td>
<td>Clock mode 0 or clock mode 1.</td>
<td>DIP switch SW4/1-2 primary setting or driven by the BCSR for secondary setting.</td>
</tr>
<tr>
<td>RSTCONF</td>
<td>‘0’</td>
<td>Configuration master from the system or DSI bus.</td>
<td>DIP switch setting.</td>
</tr>
<tr>
<td>CNFGS</td>
<td>‘0’</td>
<td>Configuration master from the system bus.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>‘1’</td>
<td>Configuration master from the DSI bus.</td>
<td></td>
</tr>
<tr>
<td>SWTE</td>
<td>‘0’</td>
<td>Software Watchdog Timer Disable.</td>
<td>Set logically if DSI is configuration master.</td>
</tr>
<tr>
<td></td>
<td>‘1’</td>
<td>Software Watchdog Timer Enable.</td>
<td>Set logically if system bus is configuration master.</td>
</tr>
<tr>
<td>BTM[0:2]</td>
<td>‘000’</td>
<td>Boot from the 60x bus</td>
<td>Configuration modes controlled by DIP switch SW4/3 setting.</td>
</tr>
<tr>
<td></td>
<td>‘001’</td>
<td>Boot from the DSI.</td>
<td></td>
</tr>
<tr>
<td>DSI64</td>
<td>‘0’ or ‘1’</td>
<td>32-bit DSI data bus with 32-bit system data bus and Ethernet. This input, combined with the ETHSEL hard reset configuration bit, defines pin multiplexing for the low part of DSI/60x data bus with Ethernet (for MSC8122/6).</td>
<td>Controlled by DIP switch SW7/3.</td>
</tr>
<tr>
<td>DSISYNC</td>
<td>‘0’</td>
<td>DSI set in asynchronous mode.</td>
<td>Pulled-down permanently.</td>
</tr>
<tr>
<td>CHIP_ID[0–3]</td>
<td>‘1111’</td>
<td>Chip ID encoded to ‘0xF’.</td>
<td>Pulled-up permanently. Prepared for backplane application. For multi-chip (multi-board) system, ID will be set according to slot’s GA (geographic address).</td>
</tr>
</tbody>
</table>
ard reset configuration word is shown in Table 4-4.

### Table 4-4. Hard Reset Configuration Word (HCW2)

<table>
<thead>
<tr>
<th>Field</th>
<th>Data Bus Bits</th>
<th>Prog Value [Bin]</th>
<th>Implication</th>
<th>Offset In HCW for system source</th>
<th>Value [Hex]</th>
</tr>
</thead>
<tbody>
<tr>
<td>EARB</td>
<td>0</td>
<td>'0'</td>
<td>Internal arbitration selected.</td>
<td>0</td>
<td>24</td>
</tr>
<tr>
<td>EXMC</td>
<td>1</td>
<td>'0'</td>
<td>Internal memory controller. CS0 active at system boot.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>INTOUT</td>
<td>2</td>
<td>'1'</td>
<td>INT_OUT function is active.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EBM</td>
<td>3</td>
<td>'0'</td>
<td>Single-chip bus mode is assumed.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BPS</td>
<td>4:5</td>
<td>'01'</td>
<td>8-bit boot port size for slave Flash memory.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SCDIS</td>
<td>6</td>
<td>'0'</td>
<td>SC140 enabled.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ISPS</td>
<td>7</td>
<td>'0'</td>
<td>64-bit internal space port size for external master access. Currently not an issue as this feature isn’t supported in the present board configuration.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>IRPC</td>
<td>8</td>
<td>'0'</td>
<td>Interrupt pin configuration. BADDR(29)/IRQ5, BADDR(30)/IRQ2, BADDR(31)/IRQ3 are selected as interrupt pins.</td>
<td>8</td>
<td>00</td>
</tr>
<tr>
<td>Reserved</td>
<td>9</td>
<td>'0'</td>
<td>Reserved. Must be cleared.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NMIOUT</td>
<td>12</td>
<td>'0'</td>
<td>Core services NMI interrupt.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ISBSEL</td>
<td>13:15</td>
<td>'000'</td>
<td>IMMR initial value 0xF000_0000. i.e., the internal space initially resides at this address.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Reserved</td>
<td>16</td>
<td>'0'</td>
<td>Reserved. Non-functional cleared bit.</td>
<td>10</td>
<td>26</td>
</tr>
<tr>
<td>BBD</td>
<td>17</td>
<td>'0'</td>
<td>Bus busy pins set as the following: ABB/IRQ4 pin is ABB and DBB/IRQ5 pin is DBB.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MMR</td>
<td>18</td>
<td>'1'</td>
<td>External bus request masked.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ETHSEL - 8122/26</td>
<td>19</td>
<td>'0'</td>
<td>Ethernet select. Activates the Ethernet.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TTPC</td>
<td>20</td>
<td>'0'</td>
<td>Transfer type pin configuration. TT[0]/NC, TT[2]/CS5, TT[3]/CS6, TT[4]/CS7 are chosen as transfer type pins: TT[0], TT[2], TT[3], TT[4].</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
4.3 Clock Source

The following paragraphs describe both the host and slave clock schemes.

4.3.1 Host Main Clock Scheme

MSC8103 requires a single clock-source for the main clock oscillator. For ease of replacement, a 55MHz @ 3.3V clock oscillator is mounted on the 8-pin DIP socket (half-size form factor). All MSC8103 60x bus timings reference to the DSP output clock. To split the load between board clock consumers, CLKOUT is driven to a zero delay buffer. To eliminate wire propagation delay to the SDRAM devices, one channel is relegated to MSC8103 DLL input.

When the internal PLL is disabled, an optional pull-down resistor can transform the clock Zero Delay (ZD) buffer into a regular clock buffer. This might cause the typical propagation delay of the clock buffer to reach 7ns.

If the ZD buffer is the ADS default then MSC8103 must be configured with DLL-off. (See HCW1 in Table 4-2, MSC8103 Hard Reset Configuration Word (HCW1), on page 4-6.)
Alternatively, if the ZD buffer is disabled (clock buffer mode) then MSC8103 must be configured with DLL-on in order to function properly with SDRAM, as shown in Figure 4-4.

**Figure 4-4. Host Clock Distribution Scheme**

### 4.3.2 Slave Main Clock Scheme

The MSC8122/26 requires a single clock-source for the main clock oscillator. For ease of replacement, the 41.5MHz @ 3.3V clock oscillator is mounted on the 8-pin DIP socket (half-size form factor). All MSC8122/26 system bus timings are dependant on the slave board clock operation mode of the ADS:

- **MODE A:** All timings are synchronized for DSP clockout. To split the load between the board’s clock consumers, CLKOUT is driven to a ZD buffer. One channel reaches the MSC8122/26 DLL input to eliminate wire propagation delay to the SDRAM device. An optional pull-down resistor can transform the ZD buffer into a regular clock buffer when the internal PLL is disabled. In such cases the typical propagation delay might reach up to 7ns. If the ZD buffer is the ADS default then the MSC8122/26 must be configured with DLL-off. (See Table 4-4, Hard Reset Configuration Word (HCW2), on page 4-10.) Alternatively, if the ZD buffer is disabled (clock buffer mode) then the MSC8122/26 is configured with DLL-on in order to function properly with the SDRAM.

- **MODE B:** To reach equal delay on all clocks, the on-board oscillator distributes all clocks via the clock buffer including the DLL input of the MSC8122/26, as shown in Figure 4-5.
4.4 60x Bus Buffering and Multiplexing

Achieve peak performance by reducing, as much as possible, the capacitive load of the 60x bus. As a consequence, slower bus devices (i.e. Flash, Framer M/P interface, BCSR and external tool bus) are buffered while the SDRAM devices and MSC8122/26 DSI-side are not buffered.

Buffers are provided over address lines and, where necessary, strobe lines. These buffers, from the 74LVC family, operate on 3.3V although are 5V tolerant\(^1\). All the expansion bus lines (64-bit data @29 address lines + strobes) are also buffered. To reduce noise and reflections, serial damping resistors are added to selected MSC8103 and MSC8122/26 strobe lines.

Transceivers are provided for data and are only open under two conditions: when access to a valid\(^2\) buffered board address exists, or during a hard reset configuration sequence\(^3\). For example, data conflicts are avoided when unbuffered or off-board memory is read; provided it is not mapped to a valid board address. Users can avoid these errors through correct programming of the memory controller.

---

1. Required for Flash and BCSR
2. An address covered by a chip-select region that controls a device buffered by BCSR logic.
3. To activate a configuration word stored in the Flash memory or BCSR.
The MSC8122/26 DSI bus provides additional configuration options in the form of different sized data buses: 64-bit or 32-bit. The first configuration, DIP switch SW7/3, configures the ADS by setting the system bus to 64-bit (SYS64 position) and the DSI bus to 32-bit. The second configuration, also using the DIP Switch SW7/3 setting, pertains to data bus width distribution: system bus at 32-bit and DSI bus at 64-bit.

Figure 4-6 shows a typical multiplexing (MUX1) schema.

A mux on DQM signals enables two modes:

- Slave controlling its own SDRAM in DSI-32 mode.

A third configuration is set using both the SW7/3 and SW7/11 DIP switches. The switches allow the MSC8122/26, via BCSR, to enable Ethernet by setting the system bus to 64-bit and the DSI bus to 32-bit. To achieve this connectivity a bus switch device is utilized. (See Table 4-7, Host Memory Controller Assignment, on page 4-18.)

After power-on-reset the DSI functions as it did for the MSC8122/26. This means that for any host access (except broadcast) including RCW write, the DSI utilizes all DCIR[CHIPID] register...
determine whether access is being attempted. This is true until the DCR[ADREN] bits are set to a value of 4'b0011 or higher.

### 4.4.1 60x-Compatible System Bus User Instructions

At RCW write and then at first DSI access, ensure that the host can access the DSI while placing the same values on HCID[3], the DSI-sampled value at power-on-reset.

First host access must be to the DCR register; the host will set the proper DCR[ADREN] bits. At this point, if HCID[3] becomes functionally HA[8] then the DCIR[CHIPID[3]] is ignored.

At power-on-reset the BCSR sets the HA8/HCID (input to slave) default to “0”, thus ensuring that the value to be sampled into DCIR[CHIPID[3]] will be logic "0". In general, the address MS bits must be “0” when accessing DSI registers.

DSI switches that are individually controlled by BCSR enable two different modes:

- MSC8122/26ADS-compatible mode wherein HCID[0-2] are connected to A7-A9; the latter also reside on the cPCI connector.
- MSC8122/26 DSI-Extended mode wherein the DSI address is extended to four different modes as per Table 4-5.

#### Table 4-5. DSI Extended Address Modes

<table>
<thead>
<tr>
<th>Slave</th>
<th>Extended Address</th>
<th>Slave Address</th>
<th>Host Address (And Chip-ID)</th>
<th>Slave HCID</th>
<th>Rack HCID</th>
<th>Clients</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSC8102</td>
<td>-</td>
<td>HA11-HA29</td>
<td>A7-A29</td>
<td>HCID0 = A7</td>
<td>A7-A10</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>HCID1 = A8</td>
<td>A7-A9</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>HCID2 = A9</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>HCID3 = A10</td>
<td>A7-A8</td>
<td>4</td>
</tr>
<tr>
<td>MSC8122/26</td>
<td>1 bit</td>
<td>HA10-HA29</td>
<td>A7-A29</td>
<td>HCID0 = A7</td>
<td>A7-A8</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>HCID1 = A8</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>HCID2 = A9</td>
<td>A7</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>HCID3 = GND</td>
<td>A7-A9</td>
<td></td>
</tr>
<tr>
<td>MSC8122/26</td>
<td>2 bits</td>
<td>HA9-HA29</td>
<td>A7-A29</td>
<td>HCID0 = A7</td>
<td>A7-A9</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>HCID1 = A8</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>HCID2 = GND</td>
<td>A7</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>HCID3 = GND</td>
<td>A7</td>
<td>1</td>
</tr>
<tr>
<td>MSC8122/26</td>
<td>3 bits</td>
<td>HA8-HA29</td>
<td>A5-A29</td>
<td>HCID0 = A7</td>
<td>A7-A9</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>HCID1 = A8</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>HCID2 = GND</td>
<td>A7</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>HCID3 = GND</td>
<td>A7</td>
<td></td>
</tr>
<tr>
<td>MSC8122/26</td>
<td>4 bits</td>
<td>HA7-HA29</td>
<td>A5-A29</td>
<td>HCID0 = A5</td>
<td>A7-A9</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>HCID1 = A6</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>HCID2 = GND</td>
<td>A7</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>HCID3 = GND</td>
<td>A7</td>
<td></td>
</tr>
</tbody>
</table>
DSI extension connectivity is shown in Table 4-6, where blue shading indicates where cells turn to address.

Table 4-6. DSI Extension Connectivity

<table>
<thead>
<tr>
<th>MCS8103 (Host) Pins</th>
<th>8122 pins</th>
<th>8102 mode</th>
<th>+1 address</th>
<th>+2 address</th>
<th>+3 address</th>
<th>+4 address</th>
</tr>
</thead>
<tbody>
<tr>
<td>TT [0]</td>
<td>TT0</td>
<td>TT0</td>
<td>TT0</td>
<td>TT0</td>
<td>A [7]</td>
<td></td>
</tr>
<tr>
<td># of HCID</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>

The DSI bus to 60-x bus connection is shown in Figure 4-7.
Figure 4-7. DSI Bus to 60-x Bus Connection
Chip Select Designation

The MSC8103 and MSC8122/26 memory controller is used as a chip-select generator to access on-board memory. Though the same functionality applies, off-board memory is accessed via J1/J2 expansion connectors. The applications result in reduced board area and costs, better power consumption and increased flexibility.

CS (Chip Select) regions, assigned to a buffered\textsuperscript{1} memory, are disabled via the BCSR. This disables local data transceivers accessing a given CS region; consequently avoiding potential\textsuperscript{2} contention over data lines. CompactPCI\textsuperscript{®} expansion connectors J1/J2 carry lines of memory controller used for accessing ADSs boards mounted on the peripheral slots of a CompactPCI\textsuperscript{®} 6U backplane; as opposed to the ADS board present on the CompactPCI\textsuperscript{®} 6U backplane system slot. These system configurations, incorporating ADS and ADSs board connections, illustrate the MSC8122/26 DSP-farm application when the MSC8103 acts as a host microcontroller.

MSC8103 CS assignments for various ADS memory/registers are outlined in Figure 4-7.

<table>
<thead>
<tr>
<th>MSC8103 Chip Select</th>
<th>Location</th>
<th>Assignment</th>
<th>Bus</th>
<th>Timing Machine</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS0</td>
<td>External</td>
<td>Flash, BCSR configuration word.</td>
<td>60x (buffered)</td>
<td>GPCM</td>
</tr>
<tr>
<td>CS1</td>
<td></td>
<td>BCSR.</td>
<td>60x (buffered)</td>
<td>GPCM</td>
</tr>
<tr>
<td>CS2</td>
<td></td>
<td>SDRAM1 bank.</td>
<td>60x (unbuffered)</td>
<td>SDRAM Machine</td>
</tr>
<tr>
<td>CS3</td>
<td></td>
<td>HBCS (DSI broadcast).</td>
<td>60x (buffered+Xbuffered)</td>
<td>GPCM</td>
</tr>
<tr>
<td>CS4</td>
<td></td>
<td>HCS (DSI select from 0 to 15).</td>
<td>60x (buffered+Xbuffered)</td>
<td>UPMA</td>
</tr>
<tr>
<td>CS5</td>
<td></td>
<td>Not supported on the ADS.</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>CS6</td>
<td></td>
<td>For off-board application.</td>
<td>60x (Xbuffered)</td>
<td>GPCM/UPMx(A or B)</td>
</tr>
<tr>
<td>CS7</td>
<td></td>
<td>Not supported on the ADS.</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>CS10</td>
<td>Internal</td>
<td>DSPRAM.</td>
<td>Local PPC</td>
<td>UPMC</td>
</tr>
<tr>
<td>CS11</td>
<td></td>
<td>DSP peripherals.</td>
<td>Local PPC</td>
<td>GPCM</td>
</tr>
</tbody>
</table>

1. Buffers don’t open when accessing an unbuffered CS region.
2. During read cycles.
4.6 Interrupts

Both host and slave side interrupts are discussed in the following paragraphs.

4.6.1 Host-Side Interrupts

External interrupts applied to the MSC8103:

- A specific-usage push button is the source of the ABORT (NMI) signal.
- Slave MSC8122/26 (INT_OUT) interrupt is received over line IRQ1.
- Available ADSs off-board interrupt sources include lines IRQ2 to IRQ4.
- Ready/busy Flash status is received over line IRQ7.
- Fast Ethernet PHY interrupt is received over line IRQ6.

4.6.1.1 ABORT Interrupt to MSC8103

Generate the ABORT (NMI) manually. Press the push button and IRQ0 input is asserted to MSC8103. This interrupt supports any resident debugger usage made available to the ADS.

4.6.1.2 Slave Interrupt

Host interrupt IRQ1 is on-board slave interrupt controller output. MSC8103 processor interrupt lines, IRQ2 to IRQ4, can be used to support up to three optional ADS expansion boards.

MSC8122/26 CS assignments for various ADS memory/registers are outlined in Table 4-8.

<table>
<thead>
<tr>
<th>MSC8122/26 Chip Select</th>
<th>Location</th>
<th>Assignment</th>
<th>Bus</th>
<th>Timing Machine</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS0</td>
<td>External</td>
<td>Flash.</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>CS1</td>
<td>Internal</td>
<td>SMII/RMII FETH switch.</td>
<td>60x (buffered)</td>
<td>GPCM</td>
</tr>
<tr>
<td>CS2</td>
<td>External</td>
<td>SDRAM2 bank.</td>
<td>60x (unbuffered)</td>
<td>SDRAM machine</td>
</tr>
<tr>
<td>CS3</td>
<td>Internal</td>
<td>TSI.</td>
<td>60x (buffered)</td>
<td>UPMA</td>
</tr>
<tr>
<td>CS4</td>
<td>Internal</td>
<td>FALC T1/E1.</td>
<td>60x (buffered)</td>
<td></td>
</tr>
<tr>
<td>CS5 - CS7</td>
<td>External</td>
<td>Not supported on the ADS.</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>CS9</td>
<td>Internal</td>
<td>IP bus.</td>
<td>Local PPC</td>
<td>GPCM</td>
</tr>
<tr>
<td>CS10</td>
<td>Internal</td>
<td>8126 TCOP/VCOP only.</td>
<td>Local PPC</td>
<td>-</td>
</tr>
<tr>
<td>CS11</td>
<td>Internal</td>
<td>L1 and L2 memory.</td>
<td>Local PPC</td>
<td>UPMC</td>
</tr>
</tbody>
</table>
4.6.1.3 Flash Ready Interrupt

Ready/busy Flash open-drain pin output indicates if a program/erase embedded algorithm is in progress or has been completed. Tying this output to interrupt line IRQ7 simplifies the burning program that services Flash.

4.6.1.4 Fast Ethernet PHY Interrupt

Connect the Davicom interrupt output (MDINTR) to the IRQ6 line of MSC8103 to support, by means of interrupt, Fast-Ethernet-transceiver event reports. A 2.2K pull-up connects to the high impedance Davicom PHY output.

4.6.2 Slave-Side Interrupts

External interrupts applied to the MSC8122/26 are:

- A specific-usage push button is the source of the ABORT (NMI) signal.
- TSI interrupt is received over line IRQ2 and muxed with line BADDR31.
- FALC56 interrupt is received over line IRQ3 and muxed with line BADDR30.
- Ready/busy Flash status is received over line IRQ1.

Figure 4-8. Host Interrupt Utilization Diagram
• cPCI interrupt from rack is received over line IRQ5 and muxed with line BADDR29.
• Fast Ethernet switch interrupt from the mezzanine board is received over line IRQ4.
• RMII/MII FETH PHY interrupt is received over line IRQ6.

4.6.2.1 **ABORT Interrupt to MSC8122/26**

Generate ABORT (NMI) manually. Press the button and IRQ0 input is asserted to the MSC8122/26. This interrupt supports any resident debugger usage made available to the ADS.

4.6.2.2 **TSI Interrupt**

The TSI interrupt request originates from the IREQ output of Infineon’s SWITI device. This device is configured as an open-drain and driven over line IRQ2.

4.6.2.3 **FALC56 Interrupt**

IRQ3 represents E1/T1 Framer (FALC56) interrupts. The FALC56 is configured as an open-drain output.

4.6.2.4 **Flash Ready Interrupt**

Ready/busy Flash open-drain pin output indicates if a program/erase embedded algorithm is in progress or has been completed. Tying this output to interrupt line IRQ1 simplifies the burning program that services Flash.

4.6.2.5 **MSC8122/26 RMII/MII Fast Ethernet PHY Interrupt**

Connect the Davicom interrupt output (FDS/MDINT) to the IRQ6 line of the MSC8122/26 to support, by means of interrupt, Fast-Ethernet-transceiver event reports.

FDS/MDINT is a dual functionality signal. The FDS (Full-Duplex Status) function indicates that the Davicom is configured to Full/Half Duplex mode. Alternatively, it operates as the transceiver’s MDINT active-low output.

To achieve dual functionality set bit 1 in register 17 (17.1) directly after the board comes out of hard reset. Set the bit through the Davicom MDIO port. The GPIO9/HD59 of the MSC8122/26 drives/samples data, while GPIO13/HD58 drives the clock. Failure of this condition results in constant assertion (low) of the MSC8122/26 IRQ6 pin.

4.6.2.6 **MSC8122/26 Fast Ethernet Switch Interrupt**

Mezzanine Fast Ethernet switch output enables the internal interrupt of the Fast Ethernet switch to address the MSC8122/26ADS. Fast Ethernet switch interrupt from the mezzanine board is received over line IRQ4.
4.7 SDRAM

Featured on the host-side of the ADS is an unbuffered SDRAM bank that includes two Micron devices with a total size of 16MB@64-bit.

A SDRAM bank with two SDRAM devices powers the MSC8122/26 system bus. For a full-width system bus configure the SDRAM bank at 16MB@64-bit with two SDRAM parts; or at 8MB@32-bit with one SDRAM part for a half-size system bus. BCSR logic, which controls an address bus multiplexing device, supports the SDRAM configuration feature. The bus multiplexing device provides address line shifting as shown in Figure 4-10.
4.7.1 SDRAM Programming

After power-up, initialize the SDRAM mode of operation by means of programming. Issue a mode register set command that passes command data to the mode register via SDRAM address lines. The MSC8103 and MSC8122/26 SDRAM machine fully supports the mode register set command.

The mode register programming value for Micron’s SDRAM MT48LC2M32B2-5 is found in Table 4-9.

<table>
<thead>
<tr>
<th>SDRAM Address Line</th>
<th>SDRAM Mode Reg Field</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A10</td>
<td>Reserved.</td>
<td>‘0’</td>
<td>Must program to zero.</td>
</tr>
<tr>
<td>A9</td>
<td>WB (Write Burst Mode).</td>
<td>‘0’</td>
<td>Write burst enabled.</td>
</tr>
<tr>
<td>A8, A7</td>
<td>Operating mode.</td>
<td>‘00’</td>
<td>Standard operation (default).</td>
</tr>
</tbody>
</table>
4.7.2 SDRAM Refresh

SDRAM has an auto-refresh mode. For example, the first SDRAM’s periodic timer issues an auto-refresh command every 15 msec. This refreshes all 4096\(^1\) SDRAM rows within the specification limits of 61.44 msec. There is a 2.56 msec interval of refresh redundancy within the specification limits window. For the refresh controller, this acts as a safety measure covering possible delays in bus availability.

4.8 Flash

The MSC8122/26ADS supports the AMD Flash device, Am29LV320 4MB volume. On the host-side the Flash interfaces to a 16-bit buffered 60x bus. On the slave-side, the HW configures the Flash to an 8-bit mode. BCSR logic drives the HW write-protect pin; a pin found on both the host- and slave-side Flash. The pin provides improved protection against accidental erasing of the HCW and the boot sector’s base boot code.

---

1. Each SDRAM component includes four internal banks of 4096 rows each; they are refreshed in parallel.

---

<table>
<thead>
<tr>
<th>SDRAM Address Line(^1)</th>
<th>SDRAM Mode Reg Field</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A6 - A4</td>
<td>CAS latency.</td>
<td>010</td>
<td>Program 2 or 3 CAS latency depending on bus frequency.</td>
</tr>
<tr>
<td>A3</td>
<td>Burst type.</td>
<td>0</td>
<td>Sequential burst access.</td>
</tr>
<tr>
<td>A2 - A0 (LSB)</td>
<td>Burst length.</td>
<td>010/011(^2)</td>
<td>4/8 word burst length (beat count)</td>
</tr>
</tbody>
</table>

Notes: 1. The SDRAM 2-1 A0 connects to the MSC8122/26ADS A29/A28 address line (32- and 64-bit system bus width mode).
2. An 8-beat burst is programmed for the MSC8122/26ADS 32-bit system bus width.
TDM Port Peripherals

Four MSC8122/26 Time Division Multiplexed (TDM) ports interface with the Infineon PEF24471 Time-Slot Interchanger (TSI). All TDM lines lead to the CompactPCI® J5 expansion connector in order to make the testability and flexibility features accessible to varying modes. Note the general interconnection diagram shown in Figure 4-11.

\[\text{Figure 4-11. TDM Connection Diagram}\]
Figure 4-12 depicts variant TDM clocking configurations.

Clock/sync has several modes: an internal clock master based on the 16MHz clock oscillator source for all four TDM ports, the T1/E1 network master mode provided by the FALC56 E1/T1 Framer for the third TDM port, and a slave to the CT-bus master clock synchronization of the TSI device.

4.9.1 Time Slot Interchanger (TSI)

The PEF24471 TSI device utilized on the ADS is a member of the SWITI family. The device provides a complete time slot switch and interface for the H.110 TDM (Time Division Multiplexed) buses. It switches between local input and output buses as well as between the H.110 bus and the local bus. PEF24471 has a capacity of up to 2048 connections.

PEF24471 is backward compatible to bus standards MVIP-90 and Dialogic's SC - bus and supports the newer H-MVIP and ECTF H. 1x0 standards. Local streams can operate on 16 physical inputs and outputs. Data rates are programmable on each of the 32 physical streams; the streams are selected in groups of four. Data rates are on a per stream basis: 2.048 Mbits/s and 4.096 Mbits/s or 8.192 Mbits/s and 16.384 Mbits/s.

After buffering, the PEF24471 interfaces at 8-bit to the MSC8122/26 system bus.
4.9.2 E1/T1 Framer

Infineon’s FALC®56 Framer PEB 2256 supports one network line and contains analog and digital function blocks configured and controlled by the MSC8122/26 slave-side processor. The FALC®56 Framer PEB 2256 has a multitude of implemented functions; it’s suited to a wide range of networking applications and fulfills all required interfacing between analog E1/T1/J1 lines and the digital PCM system highway. (See the connection schemes found in Figure 4-11, TDM Connection Diagram, on page 4-25.)

4.9.3 CODEC

The MSC8122/26ADS uses TI TLV320 Dual CODEC to provide complete audio to the PCM interfaces; including filtering and optional data companding as required by the ITU-T G.711 and G.712 recommendations. Two built-in amplifiers allow direct connection to a handset, headset, auxiliary channel and microphone/speaker.

Three PCM data formats are available: 8-bit or 16-bit linear, companded ITU-T A-law, and m-law. PCM data is transferred via a serial interface operating in ST-bus mode. Though serial I/F is programmed to an I²C-bus mode, it is available for allocating data to any of the 32 available channels and is easily interfaced to the local channel of the TSI device.

CODEC control and programming occur over a serial interface implemented by two MSC8122/26 GPIOs and a CS pin. CS control is realized via the BCSR register bit. One GPIO functions as a clock, the other as a bidirectional data path.

4.9.3.1 CODEC Initialization

The following code initializes the AIC22 codec via I²C:

```c
void InitCodec(void)
{
    unsigned int readData = 0;
    readData = ReadCodecRegister(CODEC_DEVICE_ADDR, 13);
    if (readData != 1)
        printf("Error reading from Codec: Value in Register 13 should be 0x01, read %X\n", readData);
    // TLV320AIC22C Codec initialization:
    //-----------------------------
```

Freescale Semiconductor
CODEC_DEVICE_ADDR = 0xE0 // I2C ADDR

// set Fsync as 8 kHz for MCLK 24.576 MHz
WriteCodecRegister(CODECDEVICE_ADDR, 12, 0x00);

// disable analog outputs for configuration
WriteCodecRegister(CODECDEVICE_ADDR, 13, 0x9F);
WriteCodecRegister(CODECDEVICE_ADDR, 14, 0x9D);
WriteCodecRegister(CODECDEVICE_ADDR, 5, 0x84); // 0x84 sets HSIN and HSOUT for Codec 1 (handset, right ear)
WriteCodecRegister(CODECDEVICE_ADDR, 3, 0x00);
WriteCodecRegister(CODECDEVICE_ADDR, 10, 0x42); // 0x42 sets HDIN and HDOUT for Codec 2 (headset, left ear)
WriteCodecRegister(CODECDEVICE_ADDR, 8, 0x00);

// for testing only: set both Codecs for digital loopback
//WriteCodecRegister(CODECDEVICE_ADDR, 2, 0x54); // 0x54 for digital loopback
//WriteCodecRegister(CODECDEVICE_ADDR, 7, 0x54); // 0x54 for digital loopback

// for testing only: set both Codecs for analog loopback
//WriteCodecRegister(CODECDEVICE_ADDR, 2, 0x58); // 0x58 for analog loopback
//WriteCodecRegister(CODECDEVICE_ADDR, 7, 0x58); // 0x58 for analog loopback

// set Codec 1 ADC input gain:
WriteCodecRegister(CODECDEVICE_ADDR, 3, 0x42); // 0x42 for 12 dB

// set Codec 1 DAC output gain:
WriteCodecRegister(CODECDEVICE_ADDR, 4, 0x21); // 0x21 for 12 dB

// set Codec 2 ADC input gain:
WriteCodecRegister(CODECDEVICE_ADDR, 8, 0x42); // 0x42 for 12 dB

// set Codec 2 DAC output gain:
WriteCodecRegister(CODECDEVICE_ADDR, 9, 0x21); // 0x21 for 12 dB

// enable analog outputs:
// mute headset echo gain (left ear)
WriteCodecRegister(CODECDEVICE_ADDR, 14, 0x1D);

// mute handset echo gain (right ear)
WriteCodecRegister(CODECDEVICE_ADDR, 13, 0x1F); // important: disable data valid flag!
should be 0x1F!!!

// initialization of conversion mode
WriteCodecRegister(CODECDEVICE_ADDR, 15, 0x08); // 0x08 for linear mode, enable zero-crossing, bypass HPF
WriteCodecRegister(CODEC_DEVICE_ADDR, 16, 0x08);// 0x08 for linear mode, enable zero
crossing, bypass HPF
return;}

}/**-----------------------------------------------*/

4.10 MSC8122/26 SMII/RMII/MII Fast Ethernet Switches

Serial media independent interface/media independent interface (SMII/MII) or reduced media
independent interface/media independent interface (RMII/MII) are add-on populating mezzanine
switches that enable testing of the MSC8122/26 SMII/RMII/MII Ethernet. The switch for an
SMII mezzanine is a VIA VT6526A SMII/MII. The switch for an RMII mezzanine is a VIA
VT6510B RMII/MII. The mezzanines populate an I²C EEPROM for default boot mode. The
MSC8122/26 can be programmed by the EEPROM via the selected switch. The
MSC8122/26ADS contains a 2.5V DC-to-DC switch regulator and clock oscillator. Figure 4-13
describes the mezzanine blocks.

Figure 4-13. SMII/RMII/MII Fast Ethernet Switch
1 Switching Ethernet Modes

Ethernet mode begins with two operations: an ADS SW7 set to ETH mode, and the setting of a protective bit in the BCSR by the MSC8103. Table 4-10 shows the Ethernet modes.

Table 4-10. Ethernet Modes

<table>
<thead>
<tr>
<th>MODE</th>
<th>Ethernet Mode</th>
<th>Source</th>
<th>through</th>
<th>through</th>
<th>Destination</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>MAC2MAC</td>
<td>RMII</td>
<td>MSC8122 - DSI</td>
<td>Switch</td>
<td>-</td>
</tr>
<tr>
<td>2</td>
<td>MAC2MAC</td>
<td>RMII</td>
<td>MSC8122 - TDM</td>
<td>Switch</td>
<td>-</td>
</tr>
<tr>
<td>3</td>
<td>MAC2MAC</td>
<td>SMII</td>
<td>MSC8122 - TDM</td>
<td>Switch</td>
<td>-</td>
</tr>
<tr>
<td>4</td>
<td>MAC2MAC</td>
<td>RMII</td>
<td>MSC8122 - TDM</td>
<td>Switch</td>
<td>8122 - TDM</td>
</tr>
<tr>
<td>5</td>
<td>MAC2MAC</td>
<td>RMII</td>
<td>MSC8122 - DSI</td>
<td>Switch</td>
<td>8122 - DSI</td>
</tr>
<tr>
<td>6</td>
<td>MAC2MAC</td>
<td>SMII</td>
<td>MSC8122 - TDM</td>
<td>Switch</td>
<td>8122 - TDM</td>
</tr>
<tr>
<td>7</td>
<td>MAC2PHY</td>
<td>SMII</td>
<td>MSC8122 - TDM</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>8</td>
<td>MAC2PHY</td>
<td>RMII</td>
<td>MSC8122 - TDM</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>9</td>
<td>MAC2PHY</td>
<td>RMII</td>
<td>MSC8122 - DSI</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>10</td>
<td>MAC2MAC</td>
<td>MII</td>
<td>MSC8122 - DSI</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>11</td>
<td>MAC2MAC</td>
<td>MII</td>
<td>MSC8122 - TDM</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 4-11 shows the DSI extension connectivity.

Table 4-11. DSI Extension Connectivity

<table>
<thead>
<tr>
<th>MCS8103 (Host) Pins</th>
</tr>
</thead>
<tbody>
<tr>
<td>8122 pins</td>
</tr>
<tr>
<td>TT [0]</td>
</tr>
<tr>
<td># of HCID</td>
</tr>
</tbody>
</table>
**Figure 4-14.** Ethernet Connection Diagram
4.10.2 Fast Ethernet Clocking Modes

Fast Ethernet clock connecting modes are described in Figure 4-16, *SMII/RMII/MII CLOCKING*, on page 4-33.
The MSC8122/26 I\textsuperscript{2}C module enables the reading of sequential bytes from two I\textsuperscript{2}C-slave memory devices implemented as I\textsuperscript{2}C-Serial-EEPROM in on-device memory.

The MSC8122/26 boots from address ‘111’ and then changes the address to ‘110’ to access the second EEPROM. This action requires changing the EEPROM address from ‘111’ to ‘110’ via a BCSR-controlled switch. The switch changes the main EEPROM address to ‘110’ and connects the I\textsuperscript{2}C bus between MSC8122/26 and MSC8103. EEPROM is based on the ST-M24256-BVDW6T 32 KB. The MSC8122/26-supported memory is 40 KB.
vitch initialization is done at power-on. If a change is required then BCSR7 is written into the software and power-on-reset is undertaken without turning off the power. The switches are prepared for correct boot prior to power-on-reset. MSC8103 boots and operates with address ‘111’. It can function as the I²C EEPROM slave for the MSC8122/26 I²C boot.

A third EEPROM, on the Ethernet mezzanine board, operates as boot memory for the Ethernet switch. Its address is ‘001’; it’s connected to 8122/26 for programming but is programmable via the switch itself. For test purposes, a BCSR-controlled switch can disconnect the I²C bus between 8103 and 8122/6. Figure 4-17 describes the diagram.

![Figure 4-17. MSC8122/26 I²C](image)

### 4.12 RS-232 Transceivers

Two ADS RS-232 ports connect to the MSC8103 host-side processor SCC1 port and the slave-side UART port. The port connections assist user applications and provide convenient communication channels for both the terminal and host computer.

Using a lone 3.3V single power supply with disable mode, the MAXIM MAX3241 transceiver internally generates RS-232 levels used to tri-state receive buffers. When asserting (low) the RS-232_EN1 or RS-232_EN2 bits in BCSR1/6-7, the corresponding transceiver is enabled. When negated, receiver outputs are tri-stated and the corresponding transceiver is disabled. For off-board applications it’s possible to use slave-side UART port signals via the J5 expansion connector.

9-pin female D-type RS-232 connectors are configured for direct connection (via a flat cable) to a standard IBM-PC-like RS-232 connector. Figure 4-18 and Figure 4-19 illustrate RS-232 connector pinouts. Directions “I” and “O” are relative to the ADS board. For example, “I” means input to the ADS.
host-side RS-232 port provides expanded serial I/F, including HW flow control support. The slave-side UART port is intended for null-modem connectivity.

**Figure 4-18.** Host RS-232 Serial Port Connector

- DTR - Data Terminal Ready; this line is always asserted by MSC8122/26ADS. (O)
- TXD - Transmit Data (O)
- RXD - Receive Data (I)
- DSR - Data Set Ready (I)
- CTS - Clear To Send (I)
- RTS - Request To Send (O)

**Figure 4-19.** Slave UART Serial Port Connector

- RxDUART - Receive Data (I)
- TxDUART - Transmit Data (O)
Slic-Slac Interface Support

A Slic-Slac Interface via the rack (Expansion connector J5) enables use of a six-line communication board. The board is based on a *Voice over Broadband SLIC/SLAC™* chip set (Legerty Le77D11) that is able to implement a two-channel universal telephone line interface. This makes possible the design of a single, low cost, high performance, fully software-programmable line interface for worldwide applications, as shown in Figure 4-20.

Figure 4-20. Slic-Slac VOIP Interface
Packet Peripherals (Host-Side)

The following paragraphs describe the host-side packet peripherals.

4.14.1 10/100 T-Base Ethernet PHY

The ADS provides a Fast Ethernet port with T.P.(100-Base-TX) I/F. The port supports 10 Mbps Ethernet (10-Base-T) via the DAVICOM DM9161A transceiver.

DM9161 operates from a single 3.3V power supply and connects directly to the FCC2 port of the MSC8103 via the MII interface. The interface is used for device control and data path. Set the MII control port to address “0” to configure the initial DM9161A.

The MSC8103 HRESET signal drives DM9161A reset input. The transceiver is reset after each hard reset sequence. The PHY interrupts the MSC8103 via the IRQ6 line.
ADS Memory Maps and Bus Mapping

The MSC8122/26ADS memory controllers (MSC8103 and MSC8122/26) support a glueless interface to external memory and peripheral devices on the system bus and enable interfacing with the IPBus peripherals and internal memories through the internal local bus.

The MSC8122/26 memory controller controls up to eight external memory banks that are located on the external system bus and shared by a high-performance SDRAM machine, a general purpose chip select machine (GPCM), and three user programmable machines (UPMs). It supports a seamless interface to synchronous dynamic RAM (SDRAM), SRAM, EPROM, Flash EPROM, burstable RAM, regular DRAM devices, extended data output RAM devices, and other peripherals.

5.1 Memory Maps

The MSC8103 (host) and MSC8122/26 (slave) memory maps are discussed in the following sections.

5.1.1 Host 60x Bus Mapping

Since the MSC8103 memory controller governs all access to MSC8103 memory slaves, the memory map is reprogrammable as per user needs. After hard reset the debug host initializes the memory controller via connector P14 at the JTAG/OnCE port; this allows additional access to bus addressable peripherals. SDRAM and FLASH memory respond to all memory access types-program/data and direct memory access (DMA). The MSC8103 memory map is shown in Table 5-1.
### Table 5-1. Host MSC8103 Memory Map

<table>
<thead>
<tr>
<th>ADDRESS RANGE</th>
<th>Memory Type</th>
<th>Device Name</th>
<th>Volume in Byte</th>
<th>Port Size in Bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>00000000 - 0007FFFF</td>
<td>Internal SDRAM(^1)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00080000 - 00EFFDFF</td>
<td>Empty Space</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00EFFE00 - 00EFFEFF</td>
<td>EOnCE Registers(^1)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00EFFF00 - 00EFFFFF</td>
<td>Empty Space</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00F00000 - 00F0FFFF</td>
<td>DSP Peripherals (Qbus Bank0)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00F10000 - 00F7FFFF</td>
<td>Empty Space</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00F80000 - 00F807FF</td>
<td>Boot ROM (Qbus Bank1)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00F80800 - 01EFFFF</td>
<td>Empty Space</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>01F00000 - 01F0FFFF</td>
<td>DSP Peripherals (CST1)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>01F10000 - 01FFFFFE</td>
<td>Empty Space</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>02000000 - 0207FFFF</td>
<td>Internal SDRAM (CST0)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>02080000 - 144FFFFF</td>
<td>Empty Space</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14500000 - 14507FFF</td>
<td>BCSR(0:5):(^1)</td>
<td>ALTERA EPM3256 resides on 60-x Bus</td>
<td>32K(^2)</td>
<td>8</td>
</tr>
<tr>
<td>14500000</td>
<td>BCSR0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14500004</td>
<td>BCSR1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14500008</td>
<td>BCSR2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1450000C</td>
<td>BCSR3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14500010</td>
<td>BCSR4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14500014</td>
<td>BCSR5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14500018</td>
<td>BCSR6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1450001C</td>
<td>BCSR7</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14500020</td>
<td>BCSR8</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14500024</td>
<td>BCSR9</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14500028</td>
<td>BCSR10</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14508000 - 146FFFFF</td>
<td>Empty Space</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14700000(^1) - 1483FFFFF</td>
<td>MSC8103 60x Bus Memory and CPM(^4)</td>
<td>MSC8103 Internal Space</td>
<td>128K</td>
<td>32</td>
</tr>
<tr>
<td>14840000 - 1FFFFFFF</td>
<td>Empty Space</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20000000 - 20FFFFFE</td>
<td>SDRAM</td>
<td>MT48LC2M32B2 x 2</td>
<td>16M</td>
<td>64</td>
</tr>
<tr>
<td>21000000 - 23FFFFFF</td>
<td>Empty Space</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Table 5-1. Host MSC8103 Memory Map (Continued)

<table>
<thead>
<tr>
<th>ADDRESS RANGE</th>
<th>Memory Type</th>
<th>Device Name</th>
<th>Volume in Byte</th>
<th>Port Size in Bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>24000000 - 241FFFFF</td>
<td>Slave0 (off-board) DSI Bus</td>
<td>MSC8122/26 ID 0</td>
<td>2M</td>
<td>32/64</td>
</tr>
<tr>
<td>24200000 - 243FFFFF</td>
<td>Slave1 (off-board) DSI Bus</td>
<td>MSC8122/26 ID 1</td>
<td>2M</td>
<td>32/64</td>
</tr>
<tr>
<td>24400000 - 245FFFFF</td>
<td>Slave2 (off-board) DSI Bus</td>
<td>MSC8122/26 ID 2</td>
<td>2M</td>
<td>32/64</td>
</tr>
<tr>
<td>24600000 - 247FFFFF</td>
<td>Slave3 (off-board) DSI Bus</td>
<td>MSC8122/26 ID 3</td>
<td>2M</td>
<td>32/64</td>
</tr>
<tr>
<td>24800000 - 249FFFFF</td>
<td>Slave4 (off-board) DSI Bus</td>
<td>MSC8122/26 ID 4</td>
<td>2M</td>
<td>32/64</td>
</tr>
<tr>
<td>24A00000 - 24BFFFFF</td>
<td>Slave5 (off-board) DSI Bus</td>
<td>MSC8122/26 ID 5</td>
<td>2M</td>
<td>32/64</td>
</tr>
<tr>
<td>24C00000 - 24DFFFFF</td>
<td>Slave6 (off-board) DSI Bus</td>
<td>MSC8122/26 ID 6</td>
<td>2M</td>
<td>32/64</td>
</tr>
<tr>
<td>24E00000 - 24FFFFF</td>
<td>Slave7 (off-board) DSI Bus</td>
<td>MSC8122/26 ID 7</td>
<td>2M</td>
<td>32/64</td>
</tr>
<tr>
<td>25000000 - 251FFFFF</td>
<td>Slave8 (off-board) DSI Bus</td>
<td>MSC8122/26 ID 8</td>
<td>2M</td>
<td>32/64</td>
</tr>
<tr>
<td>25200000 - 253FFFFF</td>
<td>Slave9 (off-board) DSI Bus</td>
<td>MSC8122/26 ID 9</td>
<td>2M</td>
<td>32/64</td>
</tr>
<tr>
<td>25400000 - 255FFFFF</td>
<td>Slave10 (off-board) DSI Bus</td>
<td>MSC8122/26 ID 10</td>
<td>2M</td>
<td>32/64</td>
</tr>
<tr>
<td>25600000 - 257FFFFF</td>
<td>Slave11 (off-board) DSI Bus</td>
<td>MSC8122/26 ID 11</td>
<td>2M</td>
<td>32/64</td>
</tr>
<tr>
<td>25800000 - 259FFFFF</td>
<td>Slave12 (off-board) DSI Bus</td>
<td>MSC8122/26 ID 12</td>
<td>2M</td>
<td>32/64</td>
</tr>
<tr>
<td>25A00000 - 25BFFFFF</td>
<td>Slave13 (off-board) DSI Bus</td>
<td>MSC8122/26 ID 13</td>
<td>2M</td>
<td>32/64</td>
</tr>
<tr>
<td>25C00000 - 25DFFFFF</td>
<td>Slave14 (off-board) DSI Bus</td>
<td>MSC8122/26 ID 14</td>
<td>2M</td>
<td>32/64</td>
</tr>
<tr>
<td>25E00000 - 25FFFFF</td>
<td>Slave15 (on-board) DSI Bus</td>
<td>MSC8122/26 ID 15</td>
<td>2M</td>
<td>32/64</td>
</tr>
<tr>
<td>26000000 - 261FFFFF</td>
<td>Broadcast Access over DSI Bus</td>
<td>All MSC8122/26</td>
<td>2M</td>
<td>32/64</td>
</tr>
<tr>
<td>26200000 - FFBFFFFF</td>
<td>Empty Space</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>24000000 - 243FFFFF</td>
<td>Slave0 (off-board) DSI Bus</td>
<td>MSC8122/26 ID 0</td>
<td>4M</td>
<td>32/64</td>
</tr>
<tr>
<td>24400000 - 247FFFFF</td>
<td>Slave1 (off-board) DSI Bus</td>
<td>MSC8122/26 ID 1</td>
<td>4M</td>
<td>32/64</td>
</tr>
<tr>
<td>24800000 - 24BFFFFF</td>
<td>Slave2 (off-board) DSI Bus</td>
<td>MSC8122/26 ID 2</td>
<td>4M</td>
<td>32/64</td>
</tr>
<tr>
<td>24C00000 - 24FFFFF</td>
<td>Slave3 (off-board) DSI Bus</td>
<td>MSC8122/26 ID 3</td>
<td>4M</td>
<td>32/64</td>
</tr>
<tr>
<td>25000000 - 253FFFFF</td>
<td>Slave4 (off-board) DSI Bus</td>
<td>MSC8122/26 ID 4</td>
<td>4M</td>
<td>32/64</td>
</tr>
<tr>
<td>25400000 - 257FFFFF</td>
<td>Slave5 (off-board) DSI Bus</td>
<td>MSC8122/26 ID 5</td>
<td>4M</td>
<td>32/64</td>
</tr>
<tr>
<td>25800000 - 25BFFFFF</td>
<td>Slave6 (off-board) DSI Bus</td>
<td>MSC8122/26 ID 6</td>
<td>4M</td>
<td>32/64</td>
</tr>
<tr>
<td>25C00000 - 25FFFFF</td>
<td>Slave7 (off-board) DSI Bus</td>
<td>MSC8122/26 ID 7</td>
<td>4M</td>
<td>32/64</td>
</tr>
<tr>
<td>26000000 - 263FFFFF</td>
<td>Broadcast Access over DSI Bus</td>
<td>All MSC8122/26</td>
<td>4M</td>
<td>32/64</td>
</tr>
<tr>
<td>26400000 - FFBFFFFF</td>
<td>Empty Space</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>
Table 5-1. Host MSC8103 Memory Map (Continued)

<table>
<thead>
<tr>
<th>ADDRESS RANGE</th>
<th>Memory Type</th>
<th>Device Name</th>
<th>Volume in Byte</th>
<th>Port Size in Bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>24000000 - 247FFFFF</td>
<td>Slave0 (off-board) DSI Bus</td>
<td>MSC8122/26 ID 0</td>
<td>8M³⁹</td>
<td>32/64¹⁰</td>
</tr>
<tr>
<td>24800000 - 247FFFFF</td>
<td>Slave1 (off-board) DSI Bus</td>
<td>MSC8122/26 ID 1</td>
<td>8M</td>
<td>32/64⁶</td>
</tr>
<tr>
<td>25000000 - 257FFFFF</td>
<td>Slave2 (off-board) DSI Bus</td>
<td>MSC8122/26 ID 2</td>
<td>8M</td>
<td>32/64⁶</td>
</tr>
<tr>
<td>25800000 - 25FFFFF</td>
<td>Slave3 (off-board) DSI Bus</td>
<td>MSC8122/26 ID 3</td>
<td>8M</td>
<td>32/64⁶</td>
</tr>
<tr>
<td>26000000 - 26FFFFF</td>
<td>Broadcast Access over DSI Bus</td>
<td>All MSC8122/26</td>
<td>8M</td>
<td>32/64⁶</td>
</tr>
<tr>
<td>27000000 - FFBFFFFF</td>
<td>Empty Space</td>
<td></td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>24000000 - 24FFFFF</td>
<td>Slave0 (off-board) DSI Bus</td>
<td>MSC8122/26 ID 0</td>
<td>16M¹¹</td>
<td>32/64¹²</td>
</tr>
<tr>
<td>25000000 - 25FFFFF</td>
<td>Slave1 (off-board) DSI Bus</td>
<td>MSC8122/26 ID 1</td>
<td>16M</td>
<td>32/64⁶</td>
</tr>
<tr>
<td>26000000 - 26FFFFF</td>
<td>Broadcast Access over DSI Bus</td>
<td>All MSC8122/26</td>
<td>16M</td>
<td>32/64⁶</td>
</tr>
<tr>
<td>27000000 - FFBFFFFF</td>
<td>Empty Space</td>
<td></td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>24000000 - 25FFFFF</td>
<td>Slave0 (on-board) DSI Bus</td>
<td>MSC8122/26 ID 0</td>
<td>32M¹³</td>
<td>32/64¹⁴</td>
</tr>
<tr>
<td>26000000 - FFBFFFFF</td>
<td>Empty Space</td>
<td></td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>FFC00000 - FFFFFFFF</td>
<td>Flash</td>
<td>Am29LV320</td>
<td>4M</td>
<td>16</td>
</tr>
</tbody>
</table>

Notes:  
1. Internal SDRAM mapped to fixed addresses in the SC140 core. See the MSC8103 spec for a complete description of SC140 core internal-memory map.  
2. 32KB device appears repeatedly in port-size byte x depth multiples. For example, BCSR0 appears at memory locations 14700000, 14700010, 14700020... while BCSR1 features at 14700004, 14700014, 14700024...  
3. After a hard reset configuration sequence, MSC8103 internal registers are automatically placed in hF0000000-hF000FFFFF address area. Thereafter the host debugger tool initializes a reconfigured memory space as seen in this table.  
4. MSC8103 spec comprehensively describes MSC8103 internal memory map.  
5. Addresses apply only to MSC8122/26.  
6. DSI port size dependant upon MSC8122/26 power-on-reset configuration pin.  
7. Addresses only applicable to MSC8122/26 extended address 1 bit.  
8. DSI port size dependant upon MSC8122/26 power-on-reset configuration pin.  
9. Addresses only applicable to MSC8122/26 extended address 2 bit.  
10. DSI port size dependant upon MSC8122/26 power-on-reset configuration pin.  
11. Addresses only applicable to MSC8122/26 extended address 3 bit.  
12. DSI port size dependant upon MSC8122/26 power-on-reset configuration pin.  
13. Addresses only applicable to MSC8122/26 extended address 4 bit.  
14. DSI port size dependant upon MSC8122/26 power-on-reset configuration pin.

The memory map defined in Table 5-1, Host MSC8103 Memory Map, on page 5-2 is only a recommendation. Users may choose to work with alternative memory mapping. The described mode is supported by the CodeWarrior debug tool.
Slave System Bus Mapping

The MSC8122/26ADS memory controller governs all access to MSC8122/26 memory slaves, thus the memory map is reprogrammable as per user needs. After hard reset the debug host initializes the memory controller via connector P14 at the JTAG/OnCE port; or via the parallel port in standalone mode as this allows additional access to bus addressable peripherals. SDRAM and FLASH memory respond to all memory access types-program/data and DMA. Slave system bus mapping is shown in Table 5-2.

<table>
<thead>
<tr>
<th>ADDRESS RANGE</th>
<th>Memory Type</th>
<th>Device Name</th>
<th>Config</th>
<th>Volume in Byte</th>
<th>Port Size in Bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>00000000 - 00037FFF</td>
<td>L1 Memory on Internal Bus(^1)</td>
<td></td>
<td>224K</td>
<td>64</td>
<td></td>
</tr>
<tr>
<td>00038000 - 00EFFDFF</td>
<td>Empty Space</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00EFFFE00 - 00EFFFF</td>
<td>EOnCE</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00F00000 - 00F0FFFF</td>
<td>DSP Peripherals (Qbus Bank0)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00F10000 - 00FFFFFF</td>
<td>Empty Space</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>01000000 - 01076FFF</td>
<td>L2 Memory (MQbus)</td>
<td></td>
<td>480 KB</td>
<td></td>
<td></td>
</tr>
<tr>
<td>01077000 - 01077FFF</td>
<td>Boot ROM (MQbus)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>01078000 - 017FFFFF</td>
<td>Empty Space</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>01800000 - 017FFFFF</td>
<td>SQbus Internal Space</td>
<td>Internal Space</td>
<td>Any</td>
<td></td>
<td></td>
</tr>
<tr>
<td>02000000 - 02076FFF</td>
<td>L2 Memory (System bus)</td>
<td></td>
<td>480 KB</td>
<td></td>
<td></td>
</tr>
<tr>
<td>02077000 - 02077FFF</td>
<td>Boot ROM (System bus)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>02080000 - 020B7FFF</td>
<td>L1MEM0 (System bus)</td>
<td></td>
<td>224 KB</td>
<td>x4</td>
<td></td>
</tr>
<tr>
<td>020C0000 - 020F7FFF</td>
<td>L1MEM1 (System bus)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>02100000 - 02137FFF</td>
<td>L1MEM2 (System bus)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>02140000 - 02177FFF</td>
<td>L1MEM3 (System bus)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>02178000 -0217FFFF</td>
<td>Empty Space</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>02180000 - 021BFFFF</td>
<td>CS9 - (IP address space)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>021E0000 - 021FFFFF</td>
<td>CS10 - (FDIR, FDOR)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>02200000 -145FFFFF</td>
<td>Empty Space</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14600000 - 14607FFF</td>
<td>Time-Slot-Interchanger (TSI)</td>
<td>PEF24471</td>
<td>32K(^2)</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>14608000 - 1460FFFFF</td>
<td>FALC56 E1/T1 Framer</td>
<td>PEB2256</td>
<td>32K(^3)</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>14610000 - 146FFFFF</td>
<td>Empty Space</td>
<td></td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>14700000(^1) - 14713FFF</td>
<td>MSC8122/26 System I/F</td>
<td>MSC8122/26 Internal Space</td>
<td>-</td>
<td>128K</td>
<td>32</td>
</tr>
<tr>
<td>14800000 - 148FFFFF</td>
<td>Mezzanine Fast Ethernet</td>
<td>VIA SMII/RMII</td>
<td>-</td>
<td>64</td>
<td>16</td>
</tr>
</tbody>
</table>

\(^1\) MSC8122/26 System I/F

Table 5-2. Slave MSC8122/26 Memory Map
### Memory Maps and Bus Mapping

#### Table 5-2. Slave MSC8122/26 Memory Map (Continued)

<table>
<thead>
<tr>
<th>ADDRESS RANGE</th>
<th>Memory Type</th>
<th>Device Name</th>
<th>Config</th>
<th>Volume in Byte</th>
<th>Port Size in Bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>149000000 - 1FFFFFFF</td>
<td>Empty Space</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>200000000 - 20FFFFFF OR 20000000 - 20FFFFFFF</td>
<td>SDRAM</td>
<td>MT48LC2M32B 2 x 2</td>
<td>64-bit SysBus</td>
<td>16M</td>
<td>64</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MT48LC2M32B 2 x 1</td>
<td>32-bit SysBus</td>
<td>8M</td>
<td>32</td>
</tr>
<tr>
<td>22000000 - FFBFFFFF</td>
<td>Empty Space</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FFC000000 - FFFFFFFF</td>
<td>Flash</td>
<td>Am29LV320</td>
<td></td>
<td>4M</td>
<td>8</td>
</tr>
</tbody>
</table>

**Notes:**

1. L1 memory mapped to fixed addresses in the SC140 core. See the MSC8122/26 spec for a complete description of SC140 core internal-memory map.
2. TSI device internal space is 32 bytes; however, minimum block size controllable by a CS region is 32KB.
3. E1/T1 FALC56 Framer device internal space is 256 bytes; however, minimum block size controllable by a CS region is 32KB.
4. After hard reset configuration sequence MSC8103 internal registers are automatically placed in hF0000000-hF000FFFF address area. Thereafter the host debugger tool initializes a reconfigured memory space as seen in this table.
Host Memory Controller Registers Programming

MSC8103 memory controller in the MSC8122/26ADS initializes for an 100/66 MHz bus operation. For example, register programming is based on a 100/66 MHz timing calculation. Initializations noted in Table 5-3 are based on the ADS board design.

Table 5-3. Host Memory Controller Initialization for 100(66) MHz

<table>
<thead>
<tr>
<th>Reg.</th>
<th>Device Type</th>
<th>Bus</th>
<th>Init Value [hex]</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BR0</td>
<td>Am29LV320 by AMD</td>
<td>Buffered 60-x</td>
<td>FFC01001</td>
<td>Base at FFC0_0000, 16-bit port size, no parity, GPCM.</td>
</tr>
<tr>
<td>OR0</td>
<td></td>
<td></td>
<td>FFC00874 (FFC00854)</td>
<td>4MB block size, CS early negate, 14(10) w.s., timing relax.</td>
</tr>
<tr>
<td>BR1</td>
<td>BCSR0-10 Implemented in Altera EPM3256</td>
<td>Buffered 60-x</td>
<td>14501801</td>
<td>Base at 14500_000, 32-bit port size, no parity, GPCM.</td>
</tr>
<tr>
<td>OR1</td>
<td></td>
<td></td>
<td>FFFFF8820 (FFFF8810)</td>
<td>32 KB block size, all types access, CSNT=1, 2 w.s. (32 KB block size, all types access, CSNT=1, 1 w.s.)</td>
</tr>
<tr>
<td>BR2</td>
<td>SDRAM 64-bit Supported</td>
<td>Non-buff ered 60-x</td>
<td>20000041</td>
<td>Base at 2000_0000, 64-bit port size, no parity, SDRAM machine 1.</td>
</tr>
<tr>
<td>OR2</td>
<td>SDRAM MT48LC2M32B2T6-8x2 by Micron</td>
<td></td>
<td>FF003080</td>
<td>16MB block size, 4 banks per device, row starts at A8, 11 row lines, internal bank interleaving allowed.</td>
</tr>
<tr>
<td>BR3</td>
<td>MSC8122/26 DSI Port broadcast CS</td>
<td>Non-buff ered 60-x</td>
<td>26000001</td>
<td>Base at 2800_0000, 32-bit port size, no parity, GPCM.</td>
</tr>
<tr>
<td>OR3</td>
<td></td>
<td></td>
<td>FFE00844 (FFE00884)</td>
<td>2MB block size, CSNT, four/eight wait states, TRLX.</td>
</tr>
<tr>
<td>BR4</td>
<td>MSC8122/26 DSI Port</td>
<td>Non-buff ered 60-x</td>
<td>24001881</td>
<td>Slaves based at 2400_0000 (on-Board Slave based at 2600_0000), 32-bit port size, no parity, UPMA.</td>
</tr>
<tr>
<td>OR4</td>
<td></td>
<td></td>
<td>FE000100 (FE000100)</td>
<td>32MB block size, non-burst.</td>
</tr>
<tr>
<td>BR5</td>
<td>Reserved</td>
<td></td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>OR5</td>
<td></td>
<td></td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>BR6</td>
<td>User’s peripheral</td>
<td>Buffered PPC</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>OR6</td>
<td></td>
<td></td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>BR10</td>
<td>DSPRAM</td>
<td>Local PPC</td>
<td>020000C1</td>
<td>Base at 0020_0000, 64-bit port size, no parity, UPMC.</td>
</tr>
<tr>
<td>OR10</td>
<td></td>
<td></td>
<td>FFF80000</td>
<td>512KB block size.</td>
</tr>
</tbody>
</table>
### Table 5-3. Host Memory Controller Initialization for 100(66) MHz (Continued)

<table>
<thead>
<tr>
<th>Reg.</th>
<th>Device Type</th>
<th>Bus</th>
<th>Init Value [hex]</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BR11</td>
<td>DSP Peripherals</td>
<td>Local PPC</td>
<td>01F00021</td>
<td>Base at 01F0_0000, 64-bit port size, no parity, GPCM on local PPC bus.</td>
</tr>
<tr>
<td>OR11</td>
<td></td>
<td></td>
<td>FFFF0000</td>
<td>64KB block size</td>
</tr>
<tr>
<td>PSDMR</td>
<td>SDRAM 64-bit</td>
<td>Non-buff</td>
<td>C26B36A7(C2692452)</td>
<td>Page interleaving, refresh enabled, normal operation, address muxing mode SDRAM=2, A(15-17) on BNKSEL(0:2), A8 on PSDA10, 8(4) clocks refresh recovery, 3(2) clocks precharge to activate delay, 3(2) clocks activate to read/write delay, 4 beat burst length, 2(1) clock last data out to precharge, 2(1) clock write recovery time, internal address muxing, normal timing, buffered (non-buffered), 3(2) clocks CAS latency.</td>
</tr>
<tr>
<td>PSRT</td>
<td>SDRAM Supported</td>
<td>All PPC Bus Config.</td>
<td>13</td>
<td>Generates refresh every 14 msec instead of the outside limit of every 15.6 msec. Thus the refresh redundancy is 6.6 msec. The full SDRAM refresh cycle needs 64 msec. For example, application s/w may withhold the 60x bus for up to approx. 6.6 msec in a 57.3 msec period without jeopardizing the contents of the 60x bus SDRAM.</td>
</tr>
<tr>
<td>MPTPR</td>
<td>SDRAM Supported</td>
<td></td>
<td>2800(1300)</td>
<td>Divide Bus clock by 40D (20D)</td>
</tr>
</tbody>
</table>

**Notes:** 1. Table values marked with parentheses are indicative of the lower 66MHz frequency bus.
Host Machine A/B Mode Registers

The MPC8103 (host) memory controller controls various external memory banks that are located on the external system bus and shared by a high-performance SDRAM machine, a General purpose chip select machine (GPCM), and three user-programmable machines (UPMs).

The host machine A mode register (MAMR) configures the UPM and is invoked by chip select 4 (CS4).

### Table 5-4. Host MAMR Programming (CS4:DSI)

<table>
<thead>
<tr>
<th>Step</th>
<th>Register Name</th>
<th>Value to write</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MAMR</td>
<td>0x10048898</td>
<td>Noted value allows an array write operation to control the write routine; UPMWAIT enabled; read/write loop performed twice.</td>
</tr>
<tr>
<td>Single Write</td>
<td>MDR</td>
<td>0x00FFC00</td>
<td>CS active and GPL2 are active</td>
</tr>
<tr>
<td></td>
<td>MDR</td>
<td>0x00CFFC00</td>
<td>BS active, WAEN enable</td>
</tr>
<tr>
<td></td>
<td>MDR</td>
<td>0x00FFDD00</td>
<td>BS active, WAEN disable, begin loop</td>
</tr>
<tr>
<td></td>
<td>MDR</td>
<td>0x00FFC04</td>
<td>BS active, end loop</td>
</tr>
<tr>
<td></td>
<td>MDR</td>
<td>0x0FFFCF00</td>
<td>CS negated, end access</td>
</tr>
<tr>
<td></td>
<td>MDR</td>
<td>0x0FFFC01</td>
<td>End access</td>
</tr>
<tr>
<td></td>
<td>MAMR</td>
<td>0x10C48880</td>
<td>Noted value allows an array write operation to control the read routine; UPMWAIT enabled; read/write loop performed twice.</td>
</tr>
<tr>
<td>Single Read</td>
<td>MDR</td>
<td>0x00FFC00</td>
<td>Only CS active</td>
</tr>
<tr>
<td></td>
<td>MDR</td>
<td>0x00FFEC00</td>
<td>BS active, WAEN enable</td>
</tr>
<tr>
<td></td>
<td>MDR</td>
<td>0x00FFCDD00</td>
<td>BS active, WAEN disable, begin loop</td>
</tr>
<tr>
<td></td>
<td>MDR</td>
<td>0x00FFCC04</td>
<td>BS active, end loop</td>
</tr>
<tr>
<td></td>
<td>MDR</td>
<td>0x0FFFC00</td>
<td>CS negated, end access</td>
</tr>
<tr>
<td></td>
<td>MDR</td>
<td>0x0FFFC01</td>
<td>CS negated, end access</td>
</tr>
<tr>
<td></td>
<td>MAMR</td>
<td>0x10C488BC</td>
<td>Noted value allows an array write operation to control the exception routine: UPMWAIT enabled, read/write loop performed twice.</td>
</tr>
<tr>
<td>Exception</td>
<td>MDR</td>
<td>0xFFFFC05</td>
<td>Access termination</td>
</tr>
<tr>
<td></td>
<td>MAMR</td>
<td>0x00C48880</td>
<td>Normal operation</td>
</tr>
</tbody>
</table>
The host machine B mode register (MBMR) configures the UPM and is invoked by chip select 5 (CS5).

### Table 5-5. Host MBMR Programming (CS5:ATM)

<table>
<thead>
<tr>
<th>Set</th>
<th>Register Name</th>
<th>Value to write</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single Write</td>
<td>MBMR</td>
<td>0x10011018</td>
<td>Noted value allows an array write operation to control the write routine, read/write loop performed four times.</td>
</tr>
<tr>
<td></td>
<td>MDR</td>
<td>0xFFFFFC00</td>
<td>CS active is active</td>
</tr>
<tr>
<td></td>
<td>MDR</td>
<td>0x00FFFC80</td>
<td>BS active, begin loop</td>
</tr>
<tr>
<td></td>
<td>MDR</td>
<td>0x00FFFC80</td>
<td>BS active, end loop</td>
</tr>
<tr>
<td></td>
<td>MDR</td>
<td>0xFFFFFC00</td>
<td>CS negated and BS negated</td>
</tr>
<tr>
<td></td>
<td>MDR</td>
<td>0xFFFFFC05</td>
<td>End access</td>
</tr>
<tr>
<td>Single Read</td>
<td>MBMR</td>
<td>0x10011000</td>
<td>Noted value allows an array write operation to control the read routine; read/write loop performed seven times.</td>
</tr>
<tr>
<td></td>
<td>MDR</td>
<td>0xFFFFFC00</td>
<td>CS active</td>
</tr>
<tr>
<td></td>
<td>MDR</td>
<td>0x0FFCFC80</td>
<td>GPL2 active, begin loop</td>
</tr>
<tr>
<td></td>
<td>MDR</td>
<td>0x00FCFC80</td>
<td>GPL2 active, end loop</td>
</tr>
<tr>
<td></td>
<td>MDR</td>
<td>0x00FFD00</td>
<td></td>
</tr>
<tr>
<td></td>
<td>MDR</td>
<td>0xFFFFFC04</td>
<td>GPL2 active, PSDVAL asserted</td>
</tr>
<tr>
<td></td>
<td>MDR</td>
<td>0xFFFFFC05</td>
<td>End access</td>
</tr>
<tr>
<td>Exception</td>
<td>MBMR</td>
<td>0x1001103C</td>
<td>Noted value allows an array write operation to control the exception routine.</td>
</tr>
<tr>
<td>Run</td>
<td>MBMR</td>
<td>0x00011000</td>
<td>Normal operation</td>
</tr>
</tbody>
</table>
Slave Memory Controller Registers Programming

The MSC8122 or MSC8126 memory controller in the MSC8122/26ADS initializes for an 100/66 MHz bus operation. For example, programming of the registers is based on a 100/66 MHz timing calculation. Initializations noted in Table 5-6 are based on the ADS board design.

**Table 5-6. Slave’s Memory Controller Initialization for 100(66) MHz**

<table>
<thead>
<tr>
<th>Reg.</th>
<th>Device Type</th>
<th>Bus</th>
<th>Init Value [hex]</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BR0</td>
<td>Am29LV320 by AMD</td>
<td>Buffered System Bus</td>
<td>FFC00801</td>
<td>Base at FFC0_0000, 8-bit port size, no parity, GPCM.</td>
</tr>
<tr>
<td>OR0</td>
<td>Am29LV320 by AMD</td>
<td>FFC00874 (FFC00854)</td>
<td>4MB block size, CS early negate, 14(10) w.s., timing relax.</td>
<td></td>
</tr>
<tr>
<td>BR1</td>
<td>Ethernet Switch</td>
<td>Buffered System Bus</td>
<td>1480XXXXX</td>
<td>Base at 1480_0000, 16-bit port size, no parity, GPCM.</td>
</tr>
<tr>
<td>OR1</td>
<td>Ethernet Switch</td>
<td>FFFXXXXX</td>
<td>32 KB block size, non-burst.</td>
<td></td>
</tr>
<tr>
<td>BR2</td>
<td>SDRAM 64-bit Supported</td>
<td>Non-buffered System Bus</td>
<td>20000041</td>
<td>Base at 2000_0000, 64-bit port size, no parity, SDRAM machine 1.</td>
</tr>
<tr>
<td>OR2</td>
<td>SDRAM MT48LC2M32B2T6 -8x2 by Micron</td>
<td>FF0030a0</td>
<td>16MB block size, 4 banks per device, row starts at A8, 11 row lines, internal bank interleaving allowed.</td>
<td></td>
</tr>
<tr>
<td>BR2</td>
<td>SDRAM 32-bit Supported</td>
<td>Non-buffered System Bus</td>
<td>20001841</td>
<td>Base at 2000_0000, 32-bit port size, no parity, SDRAM machine 1.</td>
</tr>
<tr>
<td>OR2</td>
<td>SDRAM MT48LC2M32B2T6 -8 by Micron</td>
<td>FF8032a0</td>
<td>8MB block size, 4 banks per device, row starts at A9, 11 row lines, internal bank interleaving allowed.</td>
<td></td>
</tr>
<tr>
<td>BR3</td>
<td>TSI</td>
<td>Buffered System Bus</td>
<td>14600881</td>
<td>Base at 1460_0000, 8-bit port size, no parity, UPMA.</td>
</tr>
<tr>
<td>OR3</td>
<td>TSI</td>
<td>FFF8100</td>
<td>32 KB block size, non-burst.</td>
<td></td>
</tr>
<tr>
<td>BR4</td>
<td>E1/T1 FALC</td>
<td>Buffered System Bus</td>
<td>14680881</td>
<td>Base at 1468_0000, 8-bit port size, no parity, UPMA.</td>
</tr>
<tr>
<td>OR4</td>
<td>E1/T1 FALC</td>
<td>FFF8100</td>
<td>32 KB block size, non-burst.</td>
<td></td>
</tr>
<tr>
<td>BR5</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>User defined</td>
</tr>
<tr>
<td>OR5</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>User defined</td>
</tr>
<tr>
<td>CS6-CS7</td>
<td>Not used</td>
<td>-</td>
<td>-</td>
<td>User defined</td>
</tr>
<tr>
<td>BR9</td>
<td>IP Bus</td>
<td>Internal System Bus</td>
<td>02181821</td>
<td>Base at 218_0000, 32-bit port size, no parity, GPCM local bus.</td>
</tr>
<tr>
<td>OR9</td>
<td>-</td>
<td>FFFC0008</td>
<td>256KB block size, non-burst.</td>
<td></td>
</tr>
<tr>
<td>BR10</td>
<td>Not used</td>
<td>-</td>
<td>-</td>
<td>User defined</td>
</tr>
<tr>
<td>OR10</td>
<td>Not used</td>
<td>-</td>
<td>-</td>
<td>User defined</td>
</tr>
<tr>
<td>BR11</td>
<td>L1’s and L2</td>
<td>Internal System Bus</td>
<td>020000C1</td>
<td>Base at 2000_0000, 64-bit port size, no parity, UPMC</td>
</tr>
<tr>
<td>OR11</td>
<td>L1’s and L2</td>
<td>FFE00000</td>
<td>2M block size, non-burst.</td>
<td></td>
</tr>
</tbody>
</table>
### Table 5-6. Slave’s Memory Controller Initialization for 100(66) MHz (Continued)

<table>
<thead>
<tr>
<th>Reg.</th>
<th>Device Type</th>
<th>Bus</th>
<th>Init Value [hex]</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSDMR</td>
<td>SDRAM 64-bit</td>
<td>Non-buffered System Bus</td>
<td>C26B36A7 (C2692452)</td>
<td>Page interleaving, refresh enabled, normal operation, address muxing mode SDRAM=2, A(15-17) on BNKSEL(0:2), A8 on PSDA10, 8(4) clocks refresh recovery, 3(2) clocks precharge to activate delay, 3(2) clocks activate to read/write delay, 4 beat burst length, 2(1) clock last data out to precharge, 2(1) clock write recovery time, Internal address muxing, normal timing, 3(2) clocks CAS latency.</td>
</tr>
<tr>
<td>SDRAM 32-bit</td>
<td>Non-buffered System Bus</td>
<td>C28737A7 (C2432552)</td>
<td>Page interleaving, refresh enabled, normal operation, address muxing mode 1, A(13-15) on BNKSEL(0:2), A9 on PSDA10, 8(4) clocks refresh recovery, 3(2) clocks precharge to activate delay, 3(2) clocks activate to read/write delay, 8 beat burst length, 2(1) clock last data out to precharge, 2(1) clock write recovery time, Internal address muxing, normal timing, 3(2) clocks CAS latency.</td>
<td></td>
</tr>
<tr>
<td>PSRT</td>
<td>SDRAM Supported</td>
<td>All System Bus Config.</td>
<td>13</td>
<td>Generates refresh every 14 msec instead of the outside limit of every 15.6 msec. Thus the refresh redundancy is 6.6 msec. The full SDRAM refresh cycle needs 64 msec. For example, application s/w may withhold the System bus for up to approx. 6.6 msec in a 57.3 msec period without jeopardizing the contents of the System bus SDRAM.</td>
</tr>
<tr>
<td>MPTPR</td>
<td>SDRAM Supported</td>
<td></td>
<td>2800(1300)</td>
<td>Divide Bus clock by 40D (20D).</td>
</tr>
</tbody>
</table>

**Notes:** 1. Table values marked with parentheses are indicative of the lower 66MHz frequency bus.
Slave Machine A/C Mode Registers Programming

The MPC8122/26 (slave) memory controller controls various external memory banks that are located on the external system bus and shared by a high-performance SDRAM machine, a general purpose chip select machine (GPCM), and three user-programmable machines (UPMs).

The slave machine A mode register (MAMR) configures the UPM and is invoked by chip select 3 (CS3) for the TSI and chip select 4 (CS4) for E1/T1 FALC.

Table 5-7. Slave MAMR Programming (CS3:TSI and CS4:E1/T1 FALC)

<table>
<thead>
<tr>
<th>Step</th>
<th>Register Name</th>
<th>Value to write [hex]</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single Write</td>
<td>MAMR</td>
<td>10015418</td>
<td>Noted value allows an array write operation to control the write routine, read/write loop performed five times.</td>
</tr>
<tr>
<td></td>
<td>MDR</td>
<td>FFFFFD00</td>
<td>Address setup</td>
</tr>
<tr>
<td></td>
<td>MDR</td>
<td>0FFFFC00</td>
<td>Chip Select active</td>
</tr>
<tr>
<td></td>
<td>MDR</td>
<td>0FF33C80</td>
<td>GPL1(WR TSI) and GPL3(WR FALC) low, start loop</td>
</tr>
<tr>
<td></td>
<td>MDR</td>
<td>0FF33C80</td>
<td>GPL1(WR TSI) and GPL3(WR FALC) low, stop loop</td>
</tr>
<tr>
<td></td>
<td>MDR</td>
<td>0FFFFC00</td>
<td>GPL1(WR TSI) and GPL3(WR FALC) high</td>
</tr>
<tr>
<td></td>
<td>MDR</td>
<td>3FFFFC05</td>
<td>PSDVAL asserted, end access</td>
</tr>
<tr>
<td>Single Read</td>
<td>MAMR</td>
<td>10015400</td>
<td>Noted value allows an array write operation to control the read routine, read/write loop performed five times.</td>
</tr>
<tr>
<td></td>
<td>MDR</td>
<td>FFFFFD00</td>
<td>Address setup</td>
</tr>
<tr>
<td></td>
<td>MDR</td>
<td>0F0CFC80</td>
<td>CS active, GPL0(RD TSI) and GPL2(RD FALC) low, stop loop</td>
</tr>
<tr>
<td></td>
<td>MDR</td>
<td>0F0CFC04</td>
<td>CS active, GPL0(RD TSI) and GPL2(RD FALC) low, PSDVAL asserted</td>
</tr>
<tr>
<td></td>
<td>MDR</td>
<td>FFFFFC00</td>
<td>CS negated, GPL0(RD TSI) and GPL2(RD FALC) high</td>
</tr>
<tr>
<td></td>
<td>MDR</td>
<td>FFFFFD00</td>
<td>Double idle cycle</td>
</tr>
<tr>
<td></td>
<td>MDR</td>
<td>FFFFFC01</td>
<td>End access</td>
</tr>
<tr>
<td>Exception</td>
<td>MAMR</td>
<td>1001543C</td>
<td>Noted value allows an array write operation to control the read routine.</td>
</tr>
<tr>
<td></td>
<td>MDR</td>
<td>FFFFFCC05</td>
<td>Access termination</td>
</tr>
<tr>
<td>Run</td>
<td>MAMR</td>
<td>00015400</td>
<td>Normal operation</td>
</tr>
</tbody>
</table>
Slave machine C mode register (MCMR) configures the UPM and is invoked by chip select 9 (CS9).

### Table 5-8. Slave MCMR Programming (CS9:L1s and L2 Memory)

<table>
<thead>
<tr>
<th>Step</th>
<th>Register Name</th>
<th>Value to write [hex]</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single Write</td>
<td>MCMR</td>
<td>90051258</td>
<td>Noted value allows an array write operation to control the write routine. Read/write loop performed four times.</td>
</tr>
<tr>
<td></td>
<td>MDR</td>
<td>00000040</td>
<td>Exception enable</td>
</tr>
<tr>
<td></td>
<td>MDR</td>
<td>00000045</td>
<td>Access termination</td>
</tr>
<tr>
<td>Burst Write</td>
<td>MCMR</td>
<td>90051260</td>
<td>Noted value allows an array write operation to control the burst write routine; read/write loop performed four times.</td>
</tr>
<tr>
<td></td>
<td>MDR</td>
<td>00000C48</td>
<td>Increment address</td>
</tr>
<tr>
<td></td>
<td>MDR</td>
<td>00000C4C</td>
<td>Increment address with PSDVAL assertion - first beat</td>
</tr>
<tr>
<td></td>
<td>MDR</td>
<td>00000C4C</td>
<td>Increment address with PSDVAL assertion - second beat</td>
</tr>
<tr>
<td></td>
<td>MDR</td>
<td>00000044</td>
<td>PSDVAL assertion - third beat</td>
</tr>
<tr>
<td></td>
<td>MDR</td>
<td>00000045</td>
<td>PSDVAL assertion - fourth beat with access termination</td>
</tr>
<tr>
<td>Single Read</td>
<td>MCMR</td>
<td>90051240</td>
<td>Noted value allows an array write operation to control the read routine; read/write loop performed four times.</td>
</tr>
<tr>
<td></td>
<td>MDR</td>
<td>00030040</td>
<td>GPL2 is high, exception enabled</td>
</tr>
<tr>
<td></td>
<td>MDR</td>
<td>00030045</td>
<td>GPL2 is high, access termination</td>
</tr>
<tr>
<td>Burst Read</td>
<td>MCMR</td>
<td>90051248</td>
<td>Noted value allows an array write operation to control the burst read routine; read/write loop performed four times.</td>
</tr>
<tr>
<td></td>
<td>MDR</td>
<td>00030C48</td>
<td>GPL2 is high, increment address</td>
</tr>
<tr>
<td></td>
<td>MDR</td>
<td>00030C4C</td>
<td>GPL2 is high, increment address with PSDVAL assertion - first beat</td>
</tr>
<tr>
<td></td>
<td>MDR</td>
<td>00030C4C</td>
<td>GPL2 is high, increment address with PSDVAL assertion - second beat</td>
</tr>
<tr>
<td></td>
<td>MDR</td>
<td>00030044</td>
<td>GPL2 is high with PSDVAL assertion - third beat</td>
</tr>
<tr>
<td></td>
<td>MDR</td>
<td>00030045</td>
<td>GPL2 is high, PSDVAL assertion - fourth beat with access termination</td>
</tr>
<tr>
<td>Exception</td>
<td>MCMR</td>
<td>9005127C</td>
<td>Noted value allows an array write operation to control the exception routine.</td>
</tr>
<tr>
<td></td>
<td>MDR</td>
<td>FF000001</td>
<td>Access termination</td>
</tr>
<tr>
<td>Run</td>
<td>MCMR</td>
<td>80011240</td>
<td>Normal operation</td>
</tr>
</tbody>
</table>
Expansion Connectors

Board expansion cPCI connectors (J1-J5) carry MSC8103 60x bus signals and MSC8122/26 peripheral signals for off-board connection. Expansion connector pins, signals, and attributes are described in the following paragraphs.

5.4.1 60x Bus Expansion Connector J1

The pins, signals, and attributes for the 60x bus expansion connector (J1) are listed in Table 5-9.

<table>
<thead>
<tr>
<th>Pin No.</th>
<th>Signal Name</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A2</td>
<td>XTCK</td>
<td>O</td>
<td>JTAG clock for off-board ADS, placed at a backplane.</td>
</tr>
<tr>
<td>A3</td>
<td>-</td>
<td>N.C.</td>
<td>Not connected.</td>
</tr>
<tr>
<td>A4</td>
<td>XCS6</td>
<td>O</td>
<td>Chip Select 6 of the MSC8103.</td>
</tr>
<tr>
<td>A5</td>
<td>XA18</td>
<td>O</td>
<td>60x Buffered Address Line 18.</td>
</tr>
<tr>
<td>A6</td>
<td>IRQcPCIb</td>
<td>I,P.U.</td>
<td>Interrupt 2 to the MSC8103. Pulled up on the ADS with a 10 KW resistor.</td>
</tr>
<tr>
<td>A7</td>
<td>XXD30</td>
<td>I/O</td>
<td>60x Buffered Data Line 30.</td>
</tr>
<tr>
<td>A8</td>
<td>XXD26</td>
<td>I/O</td>
<td>60x Buffered Data Line 26.</td>
</tr>
<tr>
<td>A9</td>
<td>XBS3</td>
<td>O</td>
<td>Buffered Byte Select 3 Strobe.</td>
</tr>
<tr>
<td>A10</td>
<td>XXD21</td>
<td>I/O</td>
<td>60x Buffered Data Line 21.</td>
</tr>
<tr>
<td>A11</td>
<td>XXD18</td>
<td>I/O</td>
<td>60x Buffered Data Line 18.</td>
</tr>
<tr>
<td>A12-A14</td>
<td>Not populated</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>A15, A17, A19, A21, A23</td>
<td>3V3</td>
<td>P</td>
<td>+3.3V power; External power supply can feed the ADS via backplane.</td>
</tr>
<tr>
<td>A16</td>
<td>XHBCSb</td>
<td>O</td>
<td>Broadcast Chip Select for off-board ADS is associated with MSC8103 CS3.</td>
</tr>
<tr>
<td>A18</td>
<td>XA12</td>
<td>O</td>
<td>60x Buffered Address Line 12.</td>
</tr>
<tr>
<td>A20</td>
<td>XXD12</td>
<td>I/O</td>
<td>60x Buffered Data Line 12.</td>
</tr>
<tr>
<td>A22</td>
<td>XXD7</td>
<td>I/O</td>
<td>60x Buffered Data Line 7.</td>
</tr>
<tr>
<td>A24</td>
<td>XXD1</td>
<td>I/O</td>
<td>60x Buffered Data Line 1.</td>
</tr>
<tr>
<td>A25</td>
<td>N.C.</td>
<td>-</td>
<td>Not Connected.</td>
</tr>
<tr>
<td>B1-B4</td>
<td>N.C.</td>
<td>-</td>
<td>Not Connected.</td>
</tr>
<tr>
<td>B5</td>
<td>XA17</td>
<td>O</td>
<td>60x Buffered Address Line 17.</td>
</tr>
<tr>
<td>B6</td>
<td>PCI_PRSNTb</td>
<td>I</td>
<td>PCI Present from Rack.</td>
</tr>
<tr>
<td>B8, B10, B16, B18, B20, B22</td>
<td>GND</td>
<td>P</td>
<td>Digital Ground. Connected to main GND plane of the ADS.</td>
</tr>
</tbody>
</table>
### Table 5-9. 60x Bus System Expansion (Continued)

<table>
<thead>
<tr>
<th>Pin No.</th>
<th>Signal Name</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>B7</td>
<td>XXD29</td>
<td>I/O</td>
<td>60x Buffered Data Line 29.</td>
</tr>
<tr>
<td>B11</td>
<td>XXD17</td>
<td>-</td>
<td>60x Buffered Data Line 17.</td>
</tr>
<tr>
<td>B12-B14</td>
<td>Not populated</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>B15</td>
<td>XHCSb</td>
<td>O</td>
<td>Chip Select for off-board ADS is associated with MSC8103 CS4.</td>
</tr>
<tr>
<td>B17</td>
<td>XA14</td>
<td>O</td>
<td>60x Buffered Address Line 14.</td>
</tr>
<tr>
<td>B19</td>
<td>XXD15</td>
<td>I/O</td>
<td>60x Buffered Data Line 15.</td>
</tr>
<tr>
<td>B21</td>
<td>XXD9</td>
<td>I/O</td>
<td>60x Buffered Data Line 9.</td>
</tr>
<tr>
<td>B23</td>
<td>XXD4</td>
<td>I/O</td>
<td>60x Buffered Data Line 4.</td>
</tr>
<tr>
<td>B24</td>
<td>Not populated</td>
<td></td>
<td>-</td>
</tr>
<tr>
<td>B25</td>
<td>XA9</td>
<td>O</td>
<td>60x Buffered Address Line 9.</td>
</tr>
<tr>
<td>C1</td>
<td>TRSThb</td>
<td>O</td>
<td>Reset to JTAG port for off-board ADS.</td>
</tr>
<tr>
<td>C2</td>
<td>XTMS</td>
<td>O</td>
<td>TMS to JTAG port for off-board ADS.</td>
</tr>
<tr>
<td>C3</td>
<td>N.C.</td>
<td>-</td>
<td>Not Connected.</td>
</tr>
<tr>
<td>C4, C8, C16, C20, C24</td>
<td>GND</td>
<td>P</td>
<td>Digital Ground. Connected to main GND plane of the ADS.</td>
</tr>
<tr>
<td>C5</td>
<td>XPRSTsb</td>
<td>O, O.D.</td>
<td>Power-On-Reset signal for Slaves (MSC8122/26).</td>
</tr>
<tr>
<td>C6, C10, C18, C22</td>
<td>3V3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>C7</td>
<td>XD28</td>
<td>I/O</td>
<td>60x Buffered Data Line 28.</td>
</tr>
<tr>
<td>C9</td>
<td>XD23</td>
<td>I/O</td>
<td>60x Buffered Data Line 23.</td>
</tr>
<tr>
<td>C11</td>
<td>XD16</td>
<td>I/O</td>
<td>60x Buffered Data Line 16.</td>
</tr>
<tr>
<td>C12-C14</td>
<td>Not populated</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>C15</td>
<td>XGPL2</td>
<td>O</td>
<td>Expansion General Purpose Line 2. A buffered strobe that, if necessary, assists MSC8103 UPM control over memory device. May be used as OE for GPCM.</td>
</tr>
<tr>
<td>C17</td>
<td>XA13</td>
<td>O</td>
<td>60x Buffered Address Line 13.</td>
</tr>
<tr>
<td>C19</td>
<td>XXD14</td>
<td>I/O</td>
<td>60x Buffered Data Line 8.</td>
</tr>
<tr>
<td>C21</td>
<td>XXD8</td>
<td>I/O</td>
<td>60x Buffered Data Line 3.</td>
</tr>
<tr>
<td>C23</td>
<td>XXD3</td>
<td>I/O</td>
<td>60x Buffered Data Line 28.</td>
</tr>
<tr>
<td>C25</td>
<td>XA7</td>
<td>O</td>
<td>60x Buffered Address Line 7.</td>
</tr>
<tr>
<td>D1</td>
<td>N.C.</td>
<td>-</td>
<td>Not Connected.</td>
</tr>
<tr>
<td>D2</td>
<td>TDOi</td>
<td>O</td>
<td>TDO signal is driven by off-board ADS.</td>
</tr>
<tr>
<td>D3</td>
<td>N.C.</td>
<td>-</td>
<td>Not Connected.</td>
</tr>
<tr>
<td>D4</td>
<td>XA19</td>
<td>O</td>
<td>60x Buffered Address Line 19.</td>
</tr>
<tr>
<td>Pin No.</td>
<td>Signal Name</td>
<td>Attribute</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>------------------</td>
<td>-----------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>D5, D7, D9, D11, D17, D19</td>
<td>GND</td>
<td>P</td>
<td>Digital Ground. Connected to main GND plane of the ADS.</td>
</tr>
<tr>
<td>D6</td>
<td>N.C.</td>
<td>O</td>
<td>Not Connected.</td>
</tr>
<tr>
<td>D8</td>
<td>XXD25</td>
<td>I/O</td>
<td>60x Buffered Data Line 25.</td>
</tr>
<tr>
<td>D10</td>
<td>XXD20</td>
<td>I/O</td>
<td>60x Buffered Data Line 20.</td>
</tr>
<tr>
<td>D12-D14</td>
<td>Not populated</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>D15</td>
<td>N.C.</td>
<td>-</td>
<td>Not Connected.</td>
</tr>
<tr>
<td>D18</td>
<td>XA15</td>
<td>O</td>
<td>60x Buffered Address Line 15.</td>
</tr>
<tr>
<td>D20</td>
<td>XXD11</td>
<td>I/O</td>
<td>60x Buffered Data Line 11.</td>
</tr>
<tr>
<td>D21</td>
<td>TP90</td>
<td></td>
<td></td>
</tr>
<tr>
<td>D22</td>
<td>XXD6</td>
<td>I/O</td>
<td>60x Buffered Data Line 6.</td>
</tr>
<tr>
<td>D23</td>
<td>Not populated</td>
<td></td>
<td></td>
</tr>
<tr>
<td>D24</td>
<td>XXD0</td>
<td>I/O</td>
<td>60x Buffered Data Line 0.</td>
</tr>
<tr>
<td>D25</td>
<td>3V3Ext</td>
<td>P</td>
<td>+3.3V power; External power supply can feed the ADS via backplane.</td>
</tr>
<tr>
<td>E1</td>
<td>N.C.</td>
<td>-</td>
<td>Not Connected.</td>
</tr>
<tr>
<td>E2</td>
<td>TDIo</td>
<td>I</td>
<td>TDI signal is driven by the ADS to off-board.</td>
</tr>
<tr>
<td>E3</td>
<td>TP93</td>
<td>-</td>
<td>Not Connected.</td>
</tr>
<tr>
<td>E4</td>
<td>TP94</td>
<td></td>
<td></td>
</tr>
<tr>
<td>E5</td>
<td>HRST1b</td>
<td>I/O, O.D.</td>
<td>Hard reset signal for off-board ADS located at first peripheral slot. This signal is pulled-up on the ADS with a 10 KW resistor.</td>
</tr>
<tr>
<td>E6</td>
<td>XXD31</td>
<td>I/O</td>
<td>60x Buffered Data Line 31.</td>
</tr>
<tr>
<td>E7</td>
<td>XXD27</td>
<td>I/O</td>
<td>60x Buffered Data Line 27.</td>
</tr>
<tr>
<td>E8</td>
<td>XXD24</td>
<td>I/O</td>
<td>60x Buffered Data Line 24.</td>
</tr>
<tr>
<td>E9</td>
<td>XXD22</td>
<td>I/O</td>
<td>60x Buffered Data Line 22.</td>
</tr>
<tr>
<td>E10</td>
<td>XXD19</td>
<td>I/O</td>
<td>60x Buffered Data Line 19.</td>
</tr>
<tr>
<td>E11</td>
<td>XBS2</td>
<td>O</td>
<td>Buffered Byte Select 2 Strobe.</td>
</tr>
<tr>
<td>E12-E14</td>
<td>Not populated</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>E15</td>
<td>XBXBCTL0</td>
<td>O</td>
<td>60X BCTL0 (R/W).</td>
</tr>
<tr>
<td>E16</td>
<td>XA11</td>
<td>O</td>
<td>60x Buffered Address Line 11.</td>
</tr>
<tr>
<td>E17</td>
<td>XA10</td>
<td>O</td>
<td>60x Buffered Address Line 10.</td>
</tr>
</tbody>
</table>
### Table 5-9. 60x Bus System Expansion (Continued)

<table>
<thead>
<tr>
<th>Pin No.</th>
<th>Signal Name</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>E18</td>
<td>XBS1</td>
<td>O</td>
<td>Buffered Byte Select 1 Strobe.</td>
</tr>
<tr>
<td>E19</td>
<td>XXD13</td>
<td>I/O</td>
<td>60x Buffered Data Line 13.</td>
</tr>
<tr>
<td>E20</td>
<td>XXD10</td>
<td>I/O</td>
<td>60x Buffered Data Line 10.</td>
</tr>
<tr>
<td>E21</td>
<td>XBS0</td>
<td>O</td>
<td>Buffered Byte Select 0 Strobe.</td>
</tr>
<tr>
<td>E22</td>
<td>XXD5</td>
<td>I/O</td>
<td>60x Buffered Data Line 5.</td>
</tr>
<tr>
<td>E23</td>
<td>XXD2</td>
<td>I/O</td>
<td>60x Buffered Data Line 2.</td>
</tr>
<tr>
<td>E24</td>
<td>XA8</td>
<td>O</td>
<td>60x Buffered Address Line 8.</td>
</tr>
<tr>
<td>E25</td>
<td>N.C.</td>
<td>-</td>
<td>Not Connected.</td>
</tr>
<tr>
<td>F1-F25</td>
<td>Shield</td>
<td>-</td>
<td>Connected to chassis.</td>
</tr>
</tbody>
</table>
### 60x Bus Expansion Connector J2

The pins, signals, and attributes for the 60x bus expansion connector (J2) are listed in Table 5-10.

#### Table 5-10. 60x Bus System Expansion

<table>
<thead>
<tr>
<th>Pin No.</th>
<th>Signal Name</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A4</td>
<td>-</td>
<td>N.C.</td>
<td>Not Connected.</td>
</tr>
<tr>
<td>A5</td>
<td>XBS5</td>
<td>O</td>
<td>Buffered Byte Select 5 Strobe.</td>
</tr>
<tr>
<td>A6</td>
<td>XXD63</td>
<td>I/O</td>
<td>60x Buffered Data Line 63.</td>
</tr>
<tr>
<td>A7</td>
<td>XXD59</td>
<td>I/O</td>
<td>60x Buffered Data Line 59.</td>
</tr>
<tr>
<td>A8</td>
<td>XXD56</td>
<td>I/O</td>
<td>60x Buffered Data Line 56.</td>
</tr>
<tr>
<td>A9</td>
<td>XXD52</td>
<td>I/O</td>
<td>60x Buffered Data Line 52.</td>
</tr>
<tr>
<td>A10</td>
<td>XXD49</td>
<td>I/O</td>
<td>60x Buffered Data Line 49.</td>
</tr>
<tr>
<td>A11</td>
<td>XXD45</td>
<td>I/O</td>
<td>60x Buffered Data Line 45.</td>
</tr>
<tr>
<td>A12</td>
<td>XXD42</td>
<td>I/O</td>
<td>60x Buffered Data Line 42.</td>
</tr>
<tr>
<td>A13</td>
<td>XXD38</td>
<td>I/O</td>
<td>60x Buffered Data Line 38.</td>
</tr>
<tr>
<td>A14</td>
<td>XXD35</td>
<td>I/O</td>
<td>60x Buffered Data Line 35.</td>
</tr>
<tr>
<td>A15</td>
<td>XA21</td>
<td>O</td>
<td>60x Buffered Address Line 21.</td>
</tr>
<tr>
<td>A16</td>
<td>XA24</td>
<td>O</td>
<td>60x Buffered Address Line 24.</td>
</tr>
<tr>
<td>A17</td>
<td>XA25</td>
<td>O</td>
<td>60x Buffered Address Line 25.</td>
</tr>
<tr>
<td>A18</td>
<td>XA29</td>
<td>O</td>
<td>60x Buffered Address Line 29.</td>
</tr>
<tr>
<td>A19</td>
<td>GND</td>
<td>P</td>
<td>Digital Ground. Connected to main GND plane of the ADS.</td>
</tr>
<tr>
<td>A20-A22</td>
<td>-</td>
<td>N.C.</td>
<td>Not connected.</td>
</tr>
<tr>
<td>B1, B3, B5, B7, B9, B11, B13, B15, B17, B19, B20, B21</td>
<td>GND</td>
<td>P</td>
<td>Digital Ground. Connected to main GND plane of the ADS.</td>
</tr>
<tr>
<td>B4</td>
<td>XA20</td>
<td>O</td>
<td>60x Buffered Address Line 20.</td>
</tr>
<tr>
<td>B6</td>
<td>XXD62</td>
<td>I/O</td>
<td>60x Buffered Data Line 62.</td>
</tr>
<tr>
<td>B8</td>
<td>XXD55</td>
<td>I/O</td>
<td>60x Buffered Data Line 55.</td>
</tr>
<tr>
<td>B10</td>
<td>XXD48</td>
<td>I/O</td>
<td>60x Buffered Data Line 48.</td>
</tr>
<tr>
<td>B12</td>
<td>XXD41</td>
<td>I/O</td>
<td>60x Buffered Data Line 41.</td>
</tr>
<tr>
<td>B14</td>
<td>XXD34</td>
<td>I/O</td>
<td>60x Buffered Data Line 34.</td>
</tr>
<tr>
<td>B16</td>
<td>XA23</td>
<td>O</td>
<td>60x Buffered Address Line 23.</td>
</tr>
<tr>
<td>B18</td>
<td>XA28</td>
<td>O</td>
<td>60x Buffered Address Line 28.</td>
</tr>
</tbody>
</table>
### Table 5-10. 60x Bus System Expansion (Continued)

<table>
<thead>
<tr>
<th>Pin No.</th>
<th>Signal Name</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>B22</td>
<td>GA3</td>
<td>I, P.U.</td>
<td>Line 3 of Geographic Addressing sets up on the backplane by jumper. Signal is pulled up on the ADS with a 10 KW resistor.</td>
</tr>
<tr>
<td>C1</td>
<td>IRQ3hb</td>
<td>I, P.U.</td>
<td>Interrupt 3 to the MSC8103. Pulled-up on the ADS with a 10 KW resistor.</td>
</tr>
<tr>
<td>C2</td>
<td>SYSEnb</td>
<td>O</td>
<td>Pin is grounded on the backplane’s System Slot; permits enabling of 60x bus strobes transceiver. Pulled-up on the ADS with a 10 KW resistor.</td>
</tr>
<tr>
<td>C3</td>
<td>XHRESETsb</td>
<td>I/O, O.D.</td>
<td>Buffered hard reset signal for on-board Slave (MSC8122/26). Signal is pulled-up on the ADS with a 1 KW resistor.</td>
</tr>
<tr>
<td>C4</td>
<td>XBS7</td>
<td>O</td>
<td>Buffered Byte Select 7 Strobe.</td>
</tr>
<tr>
<td>C5, C7, C9, C11, C13</td>
<td>Not populated</td>
<td></td>
<td></td>
</tr>
<tr>
<td>C6</td>
<td>XXD61</td>
<td>I/O</td>
<td>60x Buffered Data Line 61.</td>
</tr>
<tr>
<td>C8</td>
<td>XXD54</td>
<td>I/O</td>
<td>60x Buffered Data Line 54.</td>
</tr>
<tr>
<td>C10</td>
<td>XXD47</td>
<td>I/O</td>
<td>60x Buffered Data Line 47.</td>
</tr>
<tr>
<td>C12</td>
<td>XXD40</td>
<td>I/O</td>
<td>60x Buffered Data Line 40.</td>
</tr>
<tr>
<td>C14</td>
<td>XXD33</td>
<td>I/O</td>
<td>60x Buffered Data Line 33.</td>
</tr>
<tr>
<td>C15, C16</td>
<td>N.C.</td>
<td>-</td>
<td>Not Connected.</td>
</tr>
<tr>
<td>C17</td>
<td>BHRESEThb</td>
<td>I/O, O.D.</td>
<td>Buffered hard reset signal for on-board Host (MSC8103) provides Power-On-Reset for off-board ADS. Signal is pulled-up on the ADS with a 1 KW resistor.</td>
</tr>
<tr>
<td>C18</td>
<td>XA27</td>
<td>O</td>
<td>60x Buffered Address Line 27.</td>
</tr>
<tr>
<td>C19</td>
<td>XGPL0</td>
<td>O</td>
<td>Expansion General Purpose Line 0. A buffered strobe assisting, if necessary, MSC8103 UPM control over the memory device. May be used on a rear board.</td>
</tr>
<tr>
<td>C20</td>
<td>XGPL1</td>
<td>O</td>
<td>Expansion General Purpose Line 1. Buffered strobe assisting, if necessary, MSC8103 UPM control over the memory device. May be used on a rear board.</td>
</tr>
<tr>
<td>C21</td>
<td>XGPL3</td>
<td>O</td>
<td>Expansion General Purpose Line 3. Buffered strobe assisting, if necessary, MSC8103 UPM control over the memory device. May be used on a rear board.</td>
</tr>
<tr>
<td>C22</td>
<td>GA2</td>
<td>I, P.U.</td>
<td>Line 2 of Geographic Addressing sets up on the backplane by jumper. Signal is pulled-up on the ADS with a 10 KW resistor.</td>
</tr>
<tr>
<td>D1</td>
<td>HRST2b</td>
<td>I/O, O.D.</td>
<td>Hard reset signal for off-board ADS located at second peripheral slot. Signal is pulled up on the ADS with a 10 KW resistor.</td>
</tr>
<tr>
<td>D2</td>
<td>HRST3b</td>
<td>I/O, O.D.</td>
<td>Hard reset signal for off-board ADS located at third peripheral slot. This signal is pulled up on the ADS with a 10 KW resistor.</td>
</tr>
</tbody>
</table>
## Table 5-10. 60x Bus System Expansion (Continued)

<table>
<thead>
<tr>
<th>Pin No.</th>
<th>Signal Name</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>D3</td>
<td>N.C.</td>
<td>-</td>
<td>Not Connected.</td>
</tr>
<tr>
<td>D4, D6, D8, D10, D12, D14, D16, D18, D20</td>
<td>GND</td>
<td>P</td>
<td>Digital Ground. Connected to main GND plane of the ADS.</td>
</tr>
<tr>
<td>D5</td>
<td>XBS4</td>
<td>O</td>
<td>Buffered Byte Select 4 Strobe.</td>
</tr>
<tr>
<td>D7</td>
<td>XXD58</td>
<td>I/O</td>
<td>60x Buffered Data Line 58.</td>
</tr>
<tr>
<td>D9</td>
<td>XXD51</td>
<td>I/O</td>
<td>60x Buffered Data Line 51.</td>
</tr>
<tr>
<td>D11</td>
<td>XXD44</td>
<td>I/O</td>
<td>60x Buffered Data Line 44.</td>
</tr>
<tr>
<td>D13</td>
<td>XXD37</td>
<td>I/O</td>
<td>60x Buffered Data Line 37.</td>
</tr>
<tr>
<td>D15, D17</td>
<td>N.C.</td>
<td>-</td>
<td>Not Connected.</td>
</tr>
<tr>
<td>D19</td>
<td>N.C.</td>
<td>-</td>
<td>Not Connected.</td>
</tr>
<tr>
<td>D21</td>
<td>N.C.</td>
<td>-</td>
<td>Not Connected.</td>
</tr>
<tr>
<td>D22</td>
<td>GA1</td>
<td>I, P.U.</td>
<td>Line 1 of Geographic Addressing sets up on the backplane by jumper. Signal is pulled up on the ADS with a 10 KW resistor.</td>
</tr>
<tr>
<td>E1</td>
<td>IRQ4hb</td>
<td>I,P.U.</td>
<td>Interrupt 4 to the MSC8103. Pulled-up on the ADS with a 10 KW resistor.</td>
</tr>
<tr>
<td>E2, E3</td>
<td>N.C.</td>
<td>-</td>
<td>Not Connected.</td>
</tr>
<tr>
<td>E4</td>
<td>XBS6</td>
<td>O</td>
<td>Buffered Byte Select 6 Strobe.</td>
</tr>
<tr>
<td>E5</td>
<td>XA16</td>
<td>O</td>
<td>60x Buffered Address Line 16.</td>
</tr>
<tr>
<td>E6</td>
<td>XXD60</td>
<td>I/O</td>
<td>60x Buffered Data Line 60.</td>
</tr>
<tr>
<td>E7</td>
<td>XXD57</td>
<td>I/O</td>
<td>60x Buffered Data Line 57.</td>
</tr>
<tr>
<td>E8</td>
<td>XXD53</td>
<td>I/O</td>
<td>60x Buffered Data Line 53.</td>
</tr>
<tr>
<td>E9</td>
<td>XXD50</td>
<td>I/O</td>
<td>60x Buffered Data Line 50.</td>
</tr>
<tr>
<td>E10</td>
<td>XXD46</td>
<td>I/O</td>
<td>60x Buffered Data Line 46.</td>
</tr>
<tr>
<td>E11</td>
<td>XXD43</td>
<td>I/O</td>
<td>60x Buffered Data Line 43.</td>
</tr>
<tr>
<td>E12</td>
<td>XXD39</td>
<td>I/O</td>
<td>60x Buffered Data Line 39.</td>
</tr>
<tr>
<td>E13</td>
<td>XXD36</td>
<td>I/O</td>
<td>60x Buffered Data Line 36.</td>
</tr>
<tr>
<td>E14</td>
<td>XXD32</td>
<td>I/O</td>
<td>60x Buffered Data Line 32.</td>
</tr>
<tr>
<td>E15</td>
<td>N.C.</td>
<td>-</td>
<td>Not Connected.</td>
</tr>
<tr>
<td>E16</td>
<td>XA22</td>
<td>O</td>
<td>60x Buffered Address Line 22.</td>
</tr>
<tr>
<td>E17</td>
<td>N.C.</td>
<td>-</td>
<td>Not Connected.</td>
</tr>
<tr>
<td>E18</td>
<td>XA26</td>
<td>O</td>
<td>60x Buffered Address Line 26.</td>
</tr>
<tr>
<td>E19</td>
<td>N.C.</td>
<td>-</td>
<td>Not Implemented.</td>
</tr>
</tbody>
</table>
5.4.3 MSC8122/26 Expansion Connector J3

The pins, signals, and attributes for signal expansion connector (J3) are listed in Table 5-11.

Table 5-11. MSC8122/26 Signal Expansion Connector

<table>
<thead>
<tr>
<th>Pin No.</th>
<th>Signal Name</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A1,A5-A14,A17-</td>
<td>GND</td>
<td>P</td>
<td>MCS8122/26ADS Digital Ground.</td>
</tr>
<tr>
<td>A19</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A2</td>
<td>SLED_COL</td>
<td>I</td>
<td>Collision LED from external FETH connector.</td>
</tr>
<tr>
<td>A3</td>
<td>SLED_LINK</td>
<td>I</td>
<td>Link LED from external FETH connector.</td>
</tr>
<tr>
<td>A4</td>
<td>SYSCLK</td>
<td>O</td>
<td>MSC8103 System Clock.</td>
</tr>
<tr>
<td>A15</td>
<td>SRX_P</td>
<td>I</td>
<td>MSC8122/26 SMII 10/100M PHY RX+.</td>
</tr>
<tr>
<td>A16</td>
<td>STX_P</td>
<td>O</td>
<td>MSC8122/26 SMII 10/100M PHY TX+.</td>
</tr>
<tr>
<td>B1,B2,B11-B14,</td>
<td>GND</td>
<td>P</td>
<td>MSC8122/26ADS Digital Ground.</td>
</tr>
<tr>
<td>B17-B19</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B3</td>
<td>SMII_R1</td>
<td>I</td>
<td>L.P.F for 10/100M SMII Connector.</td>
</tr>
<tr>
<td>B4</td>
<td>SMII_R2</td>
<td>I</td>
<td>L.P.F for 10/100M SMII Connector.</td>
</tr>
<tr>
<td>B5</td>
<td>SMII_R3</td>
<td>I</td>
<td>L.P.F for 10/100M SMII Connector.</td>
</tr>
<tr>
<td>B6</td>
<td>SMII_R4</td>
<td>I</td>
<td>L.P.F for 10/100M SMII Connector.</td>
</tr>
<tr>
<td>B7-B10</td>
<td>SHIELD</td>
<td>P</td>
<td>SHIELD for 10/100M SMII Connector.</td>
</tr>
<tr>
<td>B15</td>
<td>SRX_N</td>
<td>I</td>
<td>MSC8122/26 SMII 10/100M PHY RX-.</td>
</tr>
<tr>
<td>B16</td>
<td>STX_N</td>
<td>O</td>
<td>MSC8122/26 SMII 10/100M PHY TX-.</td>
</tr>
<tr>
<td>C1</td>
<td>3V3</td>
<td>P</td>
<td>3V3 from MSC8122/26ADS.</td>
</tr>
<tr>
<td>C2</td>
<td>XTTh1</td>
<td>I/O</td>
<td>TT1 from MSC8103.</td>
</tr>
<tr>
<td>C3</td>
<td>DBG1hb</td>
<td>O</td>
<td>Data Bus Grant from MSC8103.</td>
</tr>
<tr>
<td>C4</td>
<td>ABBVBhb</td>
<td>O</td>
<td>Address Bus Busy from MSC8103.</td>
</tr>
<tr>
<td>C5</td>
<td>DBBVBhb</td>
<td>O</td>
<td>Data Bus Busy from MSC8103.</td>
</tr>
<tr>
<td>Pin No.</td>
<td>Signal Name</td>
<td>Attribute</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
<td>-----------</td>
<td>-------------</td>
</tr>
<tr>
<td>C6</td>
<td>BRhb</td>
<td>I</td>
<td>Bus Request to MSC8103.</td>
</tr>
<tr>
<td>C7</td>
<td>BGh</td>
<td>O</td>
<td>Bus Grant from MSC8103.</td>
</tr>
<tr>
<td>C8</td>
<td>IRQVB5hb</td>
<td>I</td>
<td>Interrupt request 5 to MSC8103.</td>
</tr>
<tr>
<td>C9</td>
<td>IRQ6hb</td>
<td>I</td>
<td>Interrupt request 6 to MSC8103.</td>
</tr>
<tr>
<td>C10</td>
<td>IRQVB7hb</td>
<td>I</td>
<td>Interrupt request 7 to MSC8103.</td>
</tr>
<tr>
<td>C11</td>
<td>DREQVB1b</td>
<td>I</td>
<td>MSC8103 DMA Request 1.</td>
</tr>
<tr>
<td>C12</td>
<td>DREQVB2b</td>
<td>I</td>
<td>MSC8103 DMA Request 2.</td>
</tr>
<tr>
<td>C13</td>
<td>XPSDVALb</td>
<td>O</td>
<td>MSC8103 Data Valid.</td>
</tr>
<tr>
<td>C14-C19</td>
<td>GND</td>
<td>P</td>
<td>Digital Ground. Connected to main GND plane of the ADS.</td>
</tr>
<tr>
<td>D1-D11</td>
<td>3V3</td>
<td>P</td>
<td>3V3 from MSC8122/26ADS.</td>
</tr>
<tr>
<td>D12-D14</td>
<td>Vin+</td>
<td>P</td>
<td>12V from MSC8122/26ADS.</td>
</tr>
<tr>
<td>D15-D18</td>
<td>N.C</td>
<td>-</td>
<td>Not Connected.</td>
</tr>
<tr>
<td>D19</td>
<td>GND</td>
<td>P</td>
<td>Digital Ground. Connected to main GND plane of the ADS.</td>
</tr>
<tr>
<td>E1</td>
<td>TMScc</td>
<td>I</td>
<td>JTAG TMS.</td>
</tr>
<tr>
<td>E2,E4,E6,E8,E10,E19</td>
<td>GND</td>
<td>P</td>
<td>Digital Ground. Connected to main GND plane of the ADS.</td>
</tr>
<tr>
<td>E3</td>
<td>TDIcc</td>
<td>I</td>
<td>JTAG TDI.</td>
</tr>
<tr>
<td>E5</td>
<td>TCKcc</td>
<td>I</td>
<td>JTAG TCK.</td>
</tr>
<tr>
<td>E7</td>
<td>TRSTbcc</td>
<td>I</td>
<td>JTAG TRSTb.</td>
</tr>
<tr>
<td>E9</td>
<td>TDOcc</td>
<td>O</td>
<td>JTAG TDO.</td>
</tr>
<tr>
<td>E11</td>
<td>PRSTb</td>
<td>I/O</td>
<td>MSC8103 Power-On Reset.</td>
</tr>
<tr>
<td>E14</td>
<td>HRESEThb</td>
<td>I</td>
<td>MSC8103 Hard Reset.</td>
</tr>
<tr>
<td>E15-E18</td>
<td>N.C</td>
<td>-</td>
<td>Not Connected.</td>
</tr>
<tr>
<td>F1-F19</td>
<td>Shield</td>
<td>-</td>
<td>Connected to chassis.</td>
</tr>
</tbody>
</table>
## H.110 Bus Expansion Connector J4

The pins, signals, and attributes for the H.110 Signal bus connector (J4) are listed in Table 5-12.

### Table 5-12. H.110 Bus

<table>
<thead>
<tr>
<th>Pin No.</th>
<th>Signal Name</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A1</td>
<td>HTD0</td>
<td>I/O</td>
<td>Bus Data Line 0.</td>
</tr>
<tr>
<td>A3</td>
<td>HTD8</td>
<td>I/O</td>
<td>Bus Data Line 8.</td>
</tr>
<tr>
<td>A4</td>
<td>HTD11</td>
<td>I/O</td>
<td>Bus Data Line 11.</td>
</tr>
<tr>
<td>A5</td>
<td>HTD13</td>
<td>I/O</td>
<td>Bus Data Line 13.</td>
</tr>
<tr>
<td>A6</td>
<td>HTD16</td>
<td>I/O</td>
<td>Bus Data Line 16.</td>
</tr>
<tr>
<td>A7</td>
<td>HTD19</td>
<td>I/O</td>
<td>Bus Data Line 19.</td>
</tr>
<tr>
<td>A8</td>
<td>HTD21</td>
<td>I/O</td>
<td>Bus Data Line 21.</td>
</tr>
<tr>
<td>A9</td>
<td>HTD24</td>
<td>I/O</td>
<td>Bus Data Line 24.</td>
</tr>
<tr>
<td>A10</td>
<td>HTD27</td>
<td>I/O</td>
<td>Bus Data Line 27.</td>
</tr>
<tr>
<td>A11</td>
<td>HTD29</td>
<td>I/O</td>
<td>Bus Data Line 29.</td>
</tr>
<tr>
<td>A23</td>
<td>12V</td>
<td>P</td>
<td>12V from rack to connect Slic-Slac.</td>
</tr>
<tr>
<td>A12-A25</td>
<td>Not in use</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>B1</td>
<td>3V3Ext</td>
<td>P</td>
<td>+3.3V power; External power supply can feed the ADS via backplane.</td>
</tr>
<tr>
<td>B2</td>
<td>HTD5</td>
<td>I/O</td>
<td>Bus Data Line 5.</td>
</tr>
<tr>
<td>B3</td>
<td>HTD9</td>
<td>I/O</td>
<td>Bus Data Line 9.</td>
</tr>
<tr>
<td>B4</td>
<td>5VExt</td>
<td>P</td>
<td>+5.0V power; External power supply can feed the ADS via backplane.</td>
</tr>
<tr>
<td>B5</td>
<td>HTD14</td>
<td>I/O</td>
<td>Bus Data Line 14.</td>
</tr>
<tr>
<td>B6</td>
<td>HTD17</td>
<td>I/O</td>
<td>Bus Data Line 17.</td>
</tr>
<tr>
<td>B7</td>
<td>5VExt</td>
<td>P</td>
<td>+5.0V power; External power supply can feed the ADS via backplane.</td>
</tr>
<tr>
<td>B8</td>
<td>HTD22</td>
<td>I/O</td>
<td>Bus Data Line 22.</td>
</tr>
<tr>
<td>B9</td>
<td>HTD25</td>
<td>I/O</td>
<td>Bus Data Line 25.</td>
</tr>
<tr>
<td>B10</td>
<td>3V3Ext</td>
<td>P</td>
<td>+3.3V power; External power supply can feed the ADS via backplane.</td>
</tr>
<tr>
<td>B11</td>
<td>HTD30</td>
<td>I/O</td>
<td>Bus Data Line 30.</td>
</tr>
<tr>
<td>B12-B21</td>
<td>Not in use</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>B22</td>
<td>TP24</td>
<td></td>
<td></td>
</tr>
<tr>
<td>B23</td>
<td>CT_RESETb</td>
<td>O</td>
<td>Reset to H.110 device on the ADS.</td>
</tr>
<tr>
<td>Pin No.</td>
<td>Signal Name</td>
<td>Attribute</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
<td>-----------</td>
<td>-------------</td>
</tr>
<tr>
<td>B24</td>
<td>N.C.</td>
<td></td>
<td>Not Connected.</td>
</tr>
<tr>
<td>B25</td>
<td>HSGA3</td>
<td>I, P.U.</td>
<td>Line 3 of Shelf Enumeration sets up on the backplane by jumper. Signal is pulled-up on the ADS with a 10 kW resistor.</td>
</tr>
<tr>
<td>C1</td>
<td>HTD1</td>
<td>I/O</td>
<td>Bus Data Line 1.</td>
</tr>
<tr>
<td>C2</td>
<td>HTD6</td>
<td>I/O</td>
<td>Bus Data Line 6.</td>
</tr>
<tr>
<td>C3</td>
<td>HTD10</td>
<td>I/O</td>
<td>Bus Data Line 10.</td>
</tr>
<tr>
<td>C4</td>
<td>HTD12</td>
<td>I/O</td>
<td>Bus Data Line 12.</td>
</tr>
<tr>
<td>C5</td>
<td>HTD15</td>
<td>I/O</td>
<td>Bus Data Line 15.</td>
</tr>
<tr>
<td>C6</td>
<td>HTD18</td>
<td>I/O</td>
<td>Bus Data Line 18.</td>
</tr>
<tr>
<td>C7</td>
<td>HTD20</td>
<td>I/O</td>
<td>Bus Data Line 20.</td>
</tr>
<tr>
<td>C8</td>
<td>HTD23</td>
<td>I/O</td>
<td>Bus Data Line 23.</td>
</tr>
<tr>
<td>C9</td>
<td>HTD26</td>
<td>I/O</td>
<td>Bus Data Line 26.</td>
</tr>
<tr>
<td>C10</td>
<td>HTD28</td>
<td>I/O</td>
<td>Bus Data Line 28.</td>
</tr>
<tr>
<td>C11</td>
<td>HTD31</td>
<td>I/O</td>
<td>Bus Data Line 31.</td>
</tr>
<tr>
<td>C12-C22</td>
<td>Not in use</td>
<td></td>
<td></td>
</tr>
<tr>
<td>C23</td>
<td>CT_ENb</td>
<td>I, P.U.</td>
<td>Signal indicates that the ADS is fully seated into backplane. Pull-up value is 2.4KW resistor.</td>
</tr>
<tr>
<td>C24</td>
<td>N.C.</td>
<td></td>
<td>Not Connected.</td>
</tr>
<tr>
<td>C25</td>
<td>HSGA2</td>
<td>I, P.U.</td>
<td>Line 2 of Shelf Enumeration sets up on the backplane by jumper. Signal is pulled-up on the ADS with a 10 KW resistor.</td>
</tr>
<tr>
<td>D1</td>
<td>HTD3</td>
<td>I/O</td>
<td>Bus Data Line 3.</td>
</tr>
<tr>
<td>D2</td>
<td>HTD7</td>
<td>I/O</td>
<td>Bus Data Line 7.</td>
</tr>
<tr>
<td>D3, D6, D7, D9</td>
<td>GND</td>
<td>P</td>
<td>Digital Ground. Connected to main GND plane of the ADS.</td>
</tr>
<tr>
<td>D4, D5</td>
<td>3V3Ext</td>
<td>P</td>
<td>+3.3V power; External power supply can feed the ADS via backplane.</td>
</tr>
<tr>
<td>D8, D10</td>
<td>5VExt</td>
<td>P</td>
<td>+5.0V power; External power supply can feed the ADS via backplane.</td>
</tr>
<tr>
<td>D11-D21</td>
<td>Not in use</td>
<td></td>
<td></td>
</tr>
<tr>
<td>D22</td>
<td>TP21</td>
<td></td>
<td></td>
</tr>
<tr>
<td>D23-24</td>
<td>Not in use</td>
<td></td>
<td></td>
</tr>
<tr>
<td>D25</td>
<td>HSGA1</td>
<td>I, P.U.</td>
<td>Line 1 of Shelf Enumeration sets up on the backplane by jumper. Signal is pulled-up on the ADS with a 10 KW resistor.</td>
</tr>
<tr>
<td>E1</td>
<td>HTD3</td>
<td>I/O</td>
<td>Bus Data Line 3.</td>
</tr>
</tbody>
</table>
### Table 5-12. H.110 Bus (Continued)

<table>
<thead>
<tr>
<th>Pin No.</th>
<th>Signal Name</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>E2</td>
<td>GND</td>
<td>P</td>
<td>Digital Ground. Connected to main GND plane of the ADS.</td>
</tr>
<tr>
<td>E3</td>
<td>HSCLK_D</td>
<td>I/O</td>
<td>Skewed 8 MHz SCbus compatibility Data Clock.</td>
</tr>
<tr>
<td>E4</td>
<td>HSCLK</td>
<td>I/O</td>
<td>8 MHz SCbus compatibility Data Clock.</td>
</tr>
<tr>
<td>E5</td>
<td>HTNETREF_2</td>
<td>I/O</td>
<td>Secondary Telecom Network Timing Reference.</td>
</tr>
<tr>
<td>E6</td>
<td>HTNETREF_1</td>
<td>I/O</td>
<td>Primary Telecom Network Timing Reference.</td>
</tr>
<tr>
<td>E7</td>
<td>HT_C8_B</td>
<td>I/O</td>
<td>8 MHz redundant Data Clock.</td>
</tr>
<tr>
<td>E8</td>
<td>HT_C8_A</td>
<td>I/O</td>
<td>8 MHz Data Clock.</td>
</tr>
<tr>
<td>E9</td>
<td>HFR_COMP</td>
<td>I/O</td>
<td>8 KHz SCbus compatibility Frame Clock.</td>
</tr>
<tr>
<td>E10</td>
<td>HT_FRAME_B</td>
<td>I/O</td>
<td>Redundant 8 KHz Frame Clock.</td>
</tr>
<tr>
<td>E11</td>
<td>HT_FRAME_A</td>
<td>I/O</td>
<td>8 KHz Frame Clock.</td>
</tr>
<tr>
<td>E12-E21</td>
<td>Not in use</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>E22</td>
<td>TP17</td>
<td></td>
<td></td>
</tr>
<tr>
<td>E23</td>
<td>TP18</td>
<td></td>
<td></td>
</tr>
<tr>
<td>E24</td>
<td>Not in use</td>
<td></td>
<td></td>
</tr>
<tr>
<td>E25</td>
<td>HSGA0</td>
<td>I, P.U.</td>
<td>Line 0 of Shelf Enumeration sets up on the backplane by jumper. Signal is pulled-up on the ADS with a 10 KW resistor.</td>
</tr>
<tr>
<td>F1-F15</td>
<td>GND</td>
<td>P</td>
<td>Digital Ground. Connected to main GND plane of the ADS.</td>
</tr>
<tr>
<td>F12-F20</td>
<td>Not populated</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>F21-F25</td>
<td>Shield</td>
<td>-</td>
<td>Connected to chassis.</td>
</tr>
</tbody>
</table>
Signal expansion connector J5 is the Slic-Slac connector. The pins, signals, and attributes for this connector are listed in Table 5-13.

<table>
<thead>
<tr>
<th>Pin No.</th>
<th>Signal Name</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A1</td>
<td>DSTo</td>
<td>O</td>
<td>Serial Data from CODEC.</td>
</tr>
<tr>
<td>A2</td>
<td>N.C.</td>
<td>-</td>
<td>Not Connected.</td>
</tr>
<tr>
<td>A3</td>
<td>L4TXD</td>
<td>O</td>
<td>MSC8103 TDM Port 4 Transmit Data.</td>
</tr>
<tr>
<td>A4</td>
<td>GND</td>
<td>P</td>
<td>Digital Ground. Connected to main GND plane of the ADS.</td>
</tr>
<tr>
<td>A5</td>
<td>L4RXD</td>
<td>I</td>
<td>MSC8103 TDM Port 4 Receive Data.</td>
</tr>
<tr>
<td>A6</td>
<td>L3TXD</td>
<td>O</td>
<td>MSC8103 TDM Port 3 Transmit Data.</td>
</tr>
<tr>
<td>A7</td>
<td>L3RXD</td>
<td>I</td>
<td>MSC8103 TDM Port 3 Receive Data.</td>
</tr>
<tr>
<td>A8</td>
<td>L2TXD</td>
<td>O</td>
<td>MSC8103 TDM Port 2 Transmit Data.</td>
</tr>
<tr>
<td>A9</td>
<td>L2RXD</td>
<td>I</td>
<td>MSC8103 TDM Port 2 Receive Data.</td>
</tr>
<tr>
<td>A10</td>
<td>L1TXD</td>
<td>O</td>
<td>MSC8103 TDM Port 1 Transmit Data.</td>
</tr>
<tr>
<td>A11</td>
<td>L1RXD</td>
<td>I</td>
<td>MSC8103 TDM Port 1 Receive Data.</td>
</tr>
<tr>
<td>A12</td>
<td>TD_A3</td>
<td>O</td>
<td>MSC8122/26 TDM Port 3 Transmit Data A channel.</td>
</tr>
<tr>
<td>A13</td>
<td>RD_A3</td>
<td>I</td>
<td>MSC8122/26 TDM Port 3 Receive Data A channel.</td>
</tr>
<tr>
<td>A14</td>
<td>TD_INTA2</td>
<td>O</td>
<td>MSC8122/26 TDM Port 2 Transmit Data A channel.</td>
</tr>
<tr>
<td>A15</td>
<td>RD_INTA2</td>
<td>I</td>
<td>MSC8122/26 TDM Port 2 Receive Data A channel.</td>
</tr>
<tr>
<td>A16</td>
<td>TD_A1</td>
<td>O</td>
<td>MSC8122/26 TDM Port 1 Transmit Data A channel.</td>
</tr>
<tr>
<td>A17</td>
<td>RD_A1</td>
<td>I</td>
<td>MSC8122/26 TDM Port 1 Receive Data A channel.</td>
</tr>
<tr>
<td>A18</td>
<td>TD_A0</td>
<td>O</td>
<td>MSC8122/26 TDM Port 0 Transmit Data A channel.</td>
</tr>
<tr>
<td>A19</td>
<td>3V3</td>
<td>P</td>
<td>+3.3V power; External power supply can feed the ADS via backplane.</td>
</tr>
<tr>
<td>A20</td>
<td>RD_A0</td>
<td>I</td>
<td>MSC8122/26 TDM Port 0 Receive Data A channel.</td>
</tr>
<tr>
<td>A21</td>
<td>TESTs</td>
<td>I,O</td>
<td>High active Test Signal for MSC8122/26.</td>
</tr>
<tr>
<td>A22</td>
<td>TIMER0</td>
<td>I/O</td>
<td>MSC8122/26 Timer 0.</td>
</tr>
<tr>
<td>B1</td>
<td>FETH1TXCK</td>
<td>I/O</td>
<td>MSC8103 CLK4 pin.</td>
</tr>
<tr>
<td>B2, B4, B6, B8, B10, B12, B14, B16, B18, B20, B22</td>
<td>GND</td>
<td>P</td>
<td>Digital Ground. Connected to main GND plane of the ADS.</td>
</tr>
<tr>
<td>B3</td>
<td>FETH1RXCK</td>
<td>I/O</td>
<td>MSC8103 CLK3 pin.</td>
</tr>
<tr>
<td>B5</td>
<td>N.C.</td>
<td>-</td>
<td>Not Connected.</td>
</tr>
<tr>
<td>B7</td>
<td>L4CLK</td>
<td>I</td>
<td>MSC8103 TDM Port 4 Data Clock.</td>
</tr>
</tbody>
</table>
### Table 5-13. MSC8122/26 Slic-Slac Connector (Continued)

<table>
<thead>
<tr>
<th>Pin No.</th>
<th>Signal Name</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>B9</td>
<td>L2CLK</td>
<td>I</td>
<td>MSC8103 TDM Port 2 Data Clock.</td>
</tr>
<tr>
<td>B11</td>
<td>TDMCLK3</td>
<td>I</td>
<td>MSC8122/26 TDM Port 3 Data Clock.</td>
</tr>
<tr>
<td>B13</td>
<td>TDMCLK2</td>
<td>I</td>
<td>MSC8122/26 TDM Port 2 Data Clock.</td>
</tr>
<tr>
<td>B15</td>
<td>TDMCLK1</td>
<td>I</td>
<td>MSC8122/26 TDM Port 1 Data Clock.</td>
</tr>
<tr>
<td>B17</td>
<td>TDMCLK0</td>
<td>I</td>
<td>MSC8122/26 TDM Port 0 Data Clock.</td>
</tr>
<tr>
<td>B19</td>
<td>TIMER3</td>
<td>I/O</td>
<td>MSC8122/26 Timer 3.</td>
</tr>
<tr>
<td>B21</td>
<td>TIMER1</td>
<td>I/O</td>
<td>MSC8122/26 Timer 1.</td>
</tr>
<tr>
<td>C1</td>
<td>DSTi</td>
<td>I</td>
<td>Serial Data from CODEC.</td>
</tr>
<tr>
<td>C2</td>
<td>IRQ5sb</td>
<td>I</td>
<td>Interrupt Request 5 to MSC8122/26.</td>
</tr>
<tr>
<td>C3</td>
<td>N.C.</td>
<td>-</td>
<td>Not Connected.</td>
</tr>
<tr>
<td>C4</td>
<td>SL_DX A</td>
<td>I</td>
<td>Slic-Slac input towards the TSI.</td>
</tr>
<tr>
<td>C5</td>
<td>L3CLK</td>
<td>I</td>
<td>MSC8103 TDM Port 3 Data Clock.</td>
</tr>
<tr>
<td>C6, C8, C10, C12 C14, C16, C18</td>
<td>3V3</td>
<td>P</td>
<td>+3.3V power; External power supply can feed the ADS via backplane.</td>
</tr>
<tr>
<td>C7</td>
<td>L1CLK</td>
<td>I</td>
<td>MSC8103 TDM Port 1 Data Clock.</td>
</tr>
<tr>
<td>C9</td>
<td>GPCLK1</td>
<td>O</td>
<td>Programmable General Purpose Clock1 from the TSI part (U16).</td>
</tr>
<tr>
<td>C11</td>
<td>GPCLK0</td>
<td>O</td>
<td>Programmable General Purpose Clock0 from the TSI part (U16).</td>
</tr>
<tr>
<td>C13</td>
<td>TD_B3</td>
<td>O</td>
<td>MSC8122/26 TDM Port 3 Transmit Data B channel.</td>
</tr>
<tr>
<td>C15</td>
<td>TD_B2</td>
<td>O</td>
<td>MSC8122/26 TDM Port 2 Transmit Data B channel.</td>
</tr>
<tr>
<td>C17</td>
<td>TD_B1</td>
<td>O</td>
<td>MSC8122/26 TDM Port 1 Transmit Data B channel.</td>
</tr>
<tr>
<td>C19</td>
<td>TD_B0</td>
<td>O</td>
<td>MSC8122/26 TDM Port 0 Transmit Data B channel.</td>
</tr>
<tr>
<td>C20</td>
<td>CCLK</td>
<td>I</td>
<td>CODEC Data Clock for Serial Microport.</td>
</tr>
<tr>
<td>C21</td>
<td>GND</td>
<td>P</td>
<td>Digital Ground. Connected to main GND plane of the ADS.</td>
</tr>
<tr>
<td>C22</td>
<td>TIMER_CLK2</td>
<td>I/O</td>
<td>MSC8122/26 Timer 2.</td>
</tr>
<tr>
<td>D1</td>
<td>FSYNC2</td>
<td>I</td>
<td>CODEC Frame Alignment.</td>
</tr>
<tr>
<td>D2</td>
<td>N.C.</td>
<td>-</td>
<td>Not Connected.</td>
</tr>
<tr>
<td>D3, D5, D7, D9, D11, D17, D19</td>
<td>GND</td>
<td>P</td>
<td>Digital Ground. Connected to main GND plane of the ADS.</td>
</tr>
<tr>
<td>D4, D6</td>
<td>N.C.</td>
<td>-</td>
<td>Not Connected.</td>
</tr>
<tr>
<td>D8</td>
<td>L3SYNC</td>
<td>I</td>
<td>MSC8103 TDM Port 3 Frame Sync.</td>
</tr>
<tr>
<td>D10</td>
<td>L1SYNC</td>
<td>I</td>
<td>MSC8103 TDM Port 1 Frame Sync.</td>
</tr>
<tr>
<td>D12</td>
<td>TDMSYN3</td>
<td>I</td>
<td>MSC8122/26 TDM Port 3 Frame Sync.</td>
</tr>
</tbody>
</table>
### Table 5-13. MSC8122/26 Slic-Slac Connector (Continued)

<table>
<thead>
<tr>
<th>Pin No.</th>
<th>Signal Name</th>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>D14</td>
<td>TDMSYN2</td>
<td>I</td>
<td>MSC8122/26 TDM Port 2 Frame Sync.</td>
</tr>
<tr>
<td>D16</td>
<td>TDMSYN1</td>
<td>I</td>
<td>MSC8122/26 TDM Port 1 Frame Sync.</td>
</tr>
<tr>
<td>D18</td>
<td>TDMSYN0</td>
<td>I</td>
<td>MSC8122/26 TDM Port 0 Frame Sync.</td>
</tr>
<tr>
<td>D20</td>
<td>CDATA</td>
<td>I/O</td>
<td>CODEC Data I/O.</td>
</tr>
<tr>
<td>D21</td>
<td>EE0s</td>
<td>I/O</td>
<td>MSC8122/26 Debug Request Pin (EE0).</td>
</tr>
<tr>
<td>D22</td>
<td>UARTRX</td>
<td>I</td>
<td>MSC8122/26 UART Receive Pin.</td>
</tr>
<tr>
<td>E1</td>
<td>BCLK</td>
<td>I/O</td>
<td>CODEC Clock to TSI.</td>
</tr>
<tr>
<td>E2</td>
<td>NMIsb</td>
<td>I</td>
<td>MSC8122/26 NMI input.</td>
</tr>
<tr>
<td>E3, E4, E5</td>
<td>N.C.</td>
<td>-</td>
<td>Not Connected.</td>
</tr>
<tr>
<td>E6</td>
<td>12V</td>
<td>P</td>
<td>12V of Slic-Slac.</td>
</tr>
<tr>
<td>E7</td>
<td>L4SYNC</td>
<td>I/O</td>
<td>MSC8103 CPM/GPIO.</td>
</tr>
<tr>
<td>E8, E10, E12, E15, E17, E19</td>
<td>GND</td>
<td>P</td>
<td>Digital Ground. Connected to main GND plane of the ADS.</td>
</tr>
<tr>
<td>E9</td>
<td>L2SYNC</td>
<td>I/O</td>
<td>MSC8103 CPM/GPIO.</td>
</tr>
<tr>
<td>E10</td>
<td>N.C</td>
<td>-</td>
<td>Not Connected.</td>
</tr>
<tr>
<td>E11</td>
<td>CLK2</td>
<td>I/O</td>
<td>MSC8103 CLK2 pin.</td>
</tr>
<tr>
<td>E13</td>
<td>RD_B3</td>
<td>I/O</td>
<td>MSC8122/26 TDM3RSYN.</td>
</tr>
<tr>
<td>E14</td>
<td>3V3</td>
<td>P</td>
<td>+3.3V power; External power supply can feed the ADS via backplane.</td>
</tr>
<tr>
<td>E16</td>
<td>RD_INTB2</td>
<td>I/O</td>
<td>MSC8122/26 TDM2RSYN.</td>
</tr>
<tr>
<td>E18</td>
<td>RD_B1</td>
<td>O</td>
<td>MSC8122/26 TDM1RSYN.</td>
</tr>
<tr>
<td>E20</td>
<td>RD_B0</td>
<td>I/O</td>
<td>MSC8122/26 TDM0RSYN.</td>
</tr>
<tr>
<td>E21</td>
<td>EE1s</td>
<td>I</td>
<td>MSC8122/26 EE1.</td>
</tr>
<tr>
<td>E22</td>
<td>UARTTX</td>
<td>I/O</td>
<td>MSC8122/26 UARTTX pin.</td>
</tr>
<tr>
<td>F1-F22</td>
<td>Shield</td>
<td>-</td>
<td>Connected to chassis.</td>
</tr>
</tbody>
</table>
The MSC8122/26ADS exerts control and samples status through a series of control and status registers.

### 6.1 Board Control and Status Registers

Each board control and status register (BCSR), an 8-bit wide read/write register file, controls or monitors certain ADS hardware options. For a complete ADS board configuration, the BCSR resides on the host 60x bus. The term BCSR includes six registers, BCSR0-BCSR5. Slices of the board control and status registers (miscellaneous) BCSR6-BCSR10 are not implemented.

BCSR0-BCSR6 are duplicated numerously within a CS region. This is due to the region’s 32KB minimum block size and BCSR’s decoding, for register selection, of only address lines A[29:31]. BCSR is implemented on a CPLD Altera device that provides register and logic functions over some ADS signals.

BCSR controls or monitors the following functions:

1. Power-on-reset configuration setting for the slave-side.
2. Hard reset for host and up to four slaves (three of which are ADSs off-board slaves).
3. Soft reset for host and an on-board slave.
4. Hardware reset for the following devices:
   - T1/E1 Framer FALC56
   - Fast Ethernet PHY
   - Fast Ethernet Switch
   - TSI device
5. RS-232 (host and slave) enable/disable port.
6. CODEC enable and reset device found on the serial control port.
7. BCSR provides Flash (host and slave) devices with HW boot protection.
8. BCSR provides TDM port #3 with muxing between the E1/T1 Framer and the TSI device.
9. Two host LEDs (one green, one red) provide SW signaling.
10. Two slave LEDs (one green, one red) provide SW signaling.

12. ADS has special function support for MSC8122/26 (test functions register).

13. Status register includes:
   - Software Option Identification (set by dipswitch SW5)
   - ADS configuration code
   - ADS revision code
   - BCSR revision code

Sections of the BCSR slice-control registers generally have low active notations; a bit function will be realized while the bit is zero. A related function is disabled when a bit is set to “1”. The default setting is assumed to be non-functional.
Board Control/Status Register 0 (BCSR0)

BCSR0 is a control register on the ADS. BCSR0 resides over D(0:7) lines of the 60x data; it’s accessible from the BCSR base address as a byte at offset 0x0. BCSR0 can be read or written at any time. BCSR0 defaults are attributed at the time of main power-on-reset or HRESET from the host. BCSR0 fields are described in Table 6-1.

Table 6-1. BCSR0 Description

<table>
<thead>
<tr>
<th>Bit</th>
<th>Mnemonic</th>
<th>Function</th>
<th>Default</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>FLASHPRT1</td>
<td>Flash 1 HW protection. HCW and boot code, found in the Flash boot sector, are protected in the host-side Flash when the FLASHPRT1 is asserted (low). To unprotect the Flash boot sector, reset unlock bit FLUNLCK1 at BCSR6 [0] and then write high in the FLASHPRT1 bit. This allows for further erase/write operations in the boot sector.</td>
<td>0</td>
<td>R,W</td>
</tr>
<tr>
<td>1</td>
<td>FLASHPRT2</td>
<td>Flash 2 HW protection. HCW and boot code, found in the Flash boot sector, are protected in the slave-side Flash when the FLASHPRT2 is asserted (low). To unprotect the Flash boot sector, reset unlock bit FLUNLCK2 at BCSR6 [1] and then write high in the FLASHPRT2 bit. This allows for further erase/write operations in the boot sector.</td>
<td>0</td>
<td>R,W</td>
</tr>
<tr>
<td>2</td>
<td>FRM_RST</td>
<td>E1/T1 Framer reset. The FALC56 device is reset when the FRM_RST is asserted (low). The HRESET signal of the MSC8103 will assert the FRM_RST signal.</td>
<td>1</td>
<td>R/W</td>
</tr>
<tr>
<td>3</td>
<td>CODEC_EN</td>
<td>CODEC enable. The Mitel MT92303 CODEC chip is selected for the serial control bus and is programmable when the CODEC_EN bit is asserted (low). The CODEC device is disabled for programming purposes when the bit is negated (high).</td>
<td>1</td>
<td>R,W</td>
</tr>
<tr>
<td>4</td>
<td>SIGNALS0</td>
<td>Signal LED slave 0. A dedicated green LED is illuminated when SIGNALS0 is active (low). The LED is unlit when in its inactive (default) state (high). During the reset configuration sequence the LED indicates the state of slave HRESET assertion. Users may utilize the LED for SW slave signalling purposes.</td>
<td>1</td>
<td>R,W</td>
</tr>
<tr>
<td>5</td>
<td>SIGNALS1</td>
<td>Signal LED slave 1. A dedicated red LED is illuminated when SIGNALS1 is active (low). The LED is unlit when in its inactive (default) state (high). During the reset configuration sequence the LED indicates the state of slave SRESETs assertion. Users may utilize the LED for SW slave signalling purposes.</td>
<td>1</td>
<td>R,W</td>
</tr>
<tr>
<td>6</td>
<td>SIGNALH0</td>
<td>Signal LED host 0. A dedicated green LED is illuminated when SIGNALH0 is active (low). The LED is unlit when in its inactive (default) state (high). During the reset configuration sequence the LED indicates the state of host HRESETM assertion. Users may utilize the LED for SW host signalling purposes.</td>
<td>1</td>
<td>R,W</td>
</tr>
<tr>
<td>7</td>
<td>SIGNALH1</td>
<td>Signal LED host 1. A dedicated red LED is illuminated when SIGNALH1 is active (low). The LED is unlit when in its inactive (default) state (high). During the reset configuration sequence the LED indicates the state of host SRESETM assertion. Users may utilize the LED for SW host signalling purposes.</td>
<td>1</td>
<td>R,W</td>
</tr>
<tr>
<td>8-31</td>
<td>-</td>
<td>Not Implemented.</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>
6.1.2 Board Control/Status Register 1 (BCSR1)

BCSR1 is a control register on the ADS and is accessed from the BCSR base address as a byte at offset 0x4. BCSR1 can be read or written at any time. BCSR1 defaults are attributed at main power-on-reset or HRESET. BCSR1 fields are described in Table 6-2.

Table 6-2. BCSR1 Description

<table>
<thead>
<tr>
<th>Bit</th>
<th>Mnemonic</th>
<th>Function</th>
<th>Default</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>RECONF</td>
<td>Start slave re-configuration. Bit RECONF is associated with the slave PRST signal and, when set (low), enables performance of the slave’s secondary power-on-reset configuration. When negated (high) the bit has no influence on the state of the board.</td>
<td>1</td>
<td>W</td>
</tr>
<tr>
<td>1</td>
<td>HRST1</td>
<td>Hard reset to slave1. HRST1 signal is asserted to external off-board slave1 when bit HRST1 is set (low). When negated (high) the bit has no influence on the state of the HRST1 signal. Reading the bit results in the sampling of external off-board slave1 and, as such, provides an indication of the slave processor’s state.</td>
<td>1</td>
<td>R,W</td>
</tr>
<tr>
<td>2</td>
<td>HRST2</td>
<td>Hard reset to slave2. HRST2 signal is asserted to external off-board slave2 when bit HRST2 is set (low). When negated (high) the bit has no influence on the state of the HRST2 signal. Reading the bit results in the sampling of external off-board slave2 and, as such, provides an indication of the slave processor’s state.</td>
<td>1</td>
<td>R,W</td>
</tr>
<tr>
<td>4</td>
<td>HRST3</td>
<td>Hard reset to slave3. HRST3 signal is asserted to external off-board slave3 when bit HRST3 is set (low). When negated (high) the bit has no influence on the state of the HRST3 signal. Reading the bit results in the sampling of external off-board slave3 and, as such, provides an indication of the slave processor’s state.</td>
<td>1</td>
<td>R,W</td>
</tr>
<tr>
<td>5</td>
<td>FETH_RST</td>
<td>Fast Ethernet peripheral reset. Upon activation (low), the host-PHY Davicom DM9161 enters into a reset state and the MII port control bits revert to their default values. The same applies to the FETH_RST signal when the HRESET signal of the MSC8103 is asserted.</td>
<td>1</td>
<td>R,W</td>
</tr>
<tr>
<td>6</td>
<td>RS232EN_1</td>
<td>RS-232 transceiver host-side enable. Upon activation (low), the RS-232 transceiver, using the SCC1 port of the host MSC8103, is enabled. When negated (high), the RS-232 transceiver enters standby mode.</td>
<td>1</td>
<td>R,W</td>
</tr>
<tr>
<td>7</td>
<td>RS232EN_2</td>
<td>UART transceiver slave-side enable. Upon activation (low), the MSC8122/26 UART port transceiver is enabled. When negated (high), the RS-232 transceiver enters standby mode and the UART port pins become available for off-board use (via the J5 expansion connectors).</td>
<td>0</td>
<td>R,W</td>
</tr>
<tr>
<td>8-31</td>
<td>-</td>
<td>Not Implemented.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

6.1.3 Board Control/Status Register 2 (BCSR2)

BCSR2 is both a control register and an MSC8122/26 slave configuration setting on the ADS. BCSR2 is accessed from the BCSR base address as a byte at offset 0x8 and can be read or written at any time. BCSR2 defaults are attributed at main power-on-reset or HRESET from the
depending upon the slave configuration dipswitch SW4/3 (SYS/DSI) setting. BCSR2 fields are described in Table 6-3.

**Table 6-3. BCSR2 Register Description**

<table>
<thead>
<tr>
<th>Bit</th>
<th>Mnemonic</th>
<th>Function</th>
<th>Default</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>RSTCNF</td>
<td>Reset configuration mode. During the slave power-on-reset configuration sequence the RSTCNF signal drives the logical level of the bit’s value. Users may, at any time, change the value of bit RSTCNF.</td>
<td>0</td>
<td>R,W</td>
</tr>
<tr>
<td>1</td>
<td>CNFG</td>
<td>Configuration source. During the slave power-on-reset configuration sequence the CNFG signal drives the logical level of the bit’s value. After the sequence the signal enters the Hi-Z state. Users may, at any time, change the value of bit CNFG.</td>
<td>0, 1</td>
<td>R,W</td>
</tr>
<tr>
<td>2</td>
<td>SWTE</td>
<td>SWTE (Software Watchdog Timer Enable). During the slave power-on-reset configuration sequence the SWTE signal drives the logical level of the bit’s value. After the sequence the signal enters the Hi-Z state. After completing the configuration process negation occurs (low) and the SW watchdog timer is disabled. If asserted (high) then the SW watchdog timer is enabled.</td>
<td>1, 0</td>
<td>R,W</td>
</tr>
<tr>
<td>3</td>
<td>DSI64</td>
<td>DSI /system 64-/32-bit. DSI64 signal indicates DIP switch SW7/3 and is permanently set at the time of main power-on-reset. After the sequence the signal enters the Hi-Z state. When negated (low) the DSI bus configures to 32-bit and the System bus to 64-bit. If asserted (high) the DSI bus configures to 64-bit and the System bus to 32-bit.</td>
<td>-</td>
<td>R</td>
</tr>
<tr>
<td>4</td>
<td>DSISYNC</td>
<td>DSI synchronous mode. During the slave power-on-reset configuration sequence the DSISYNC signal drives the logical level of the bit’s value. After the sequence the signal enters the Hi-Z state. When negated (low) the DSI bus configures to an asynchronous mode or, if asserted (high), to a synchronous mode. This mode is not supported in MSC8103 host.</td>
<td>0</td>
<td>R,W</td>
</tr>
<tr>
<td>5</td>
<td>HRST</td>
<td>Hard reset to slave. The slave HRESET signal is asserted to the on-board MSC8122/26 slave when bit HRESETs is set (low). When negated (high) the bit has no influence on the state of the HRESET signal. Reading the bit results in the sampling of the MSC8122/26 slave and, as such, provides an indication of the slave processor’s state.</td>
<td>1</td>
<td>R,W</td>
</tr>
<tr>
<td>6</td>
<td>SRST</td>
<td>Soft reset to slave. The slave HRESET signal is asserted to the on-board MSC8122/26 slave when bit HRESETs is set (low). When negated (high) the bit has no influence on the state of the HRESET signal. Reading the bit results in the sampling of the MSC8122/26 slave and, as such, provides an indication of the slave processor’s state.</td>
<td>1</td>
<td>R,W</td>
</tr>
<tr>
<td>7</td>
<td>FRMtoTSI</td>
<td>E1/T1 Framer to TSI device. TDM channels of the FALC56 device are connected to a TSI device when the FRM to TSI is asserted (low). When the FRM to TSI is negated (high) the TDM channels of the FALC56 device are tied directly to a TDM3 port of the MSC8122/26.</td>
<td>1</td>
<td>R,W</td>
</tr>
<tr>
<td>8-31</td>
<td>-</td>
<td>Not Implemented.</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>
6.1.4   Board Control/Status Register 3 (BCSR3)

BCSR3 is both a control register and a slave MSC8122/26 configuration setting on the ADS. BCSR3 is accessed from the BCSR base address as a byte at offset 0xc and can be read or written at any time. BCSR3 defaults are attributed upon main power-on-reset or HRESET, depending upon the slave configuration DIP switch SW4 setting. BCSR3 fields are described in Table 6-4.

<table>
<thead>
<tr>
<th>Bit</th>
<th>Mnemonic</th>
<th>Function</th>
<th>Default</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-1</td>
<td>CLKMD[1:2]</td>
<td>Clock mode setting bits 1-2 for slave. During the slave power-on-reset configuration sequence the CLKMD signals drive the logical level of the bit’s value. After the sequence the signals enter the Hi-Z state. Users may change, at any time, the value of bits CLKMD[1:2].</td>
<td>From DIP Switch SW4/1-2</td>
<td>R,W</td>
</tr>
<tr>
<td>2</td>
<td>SEE0</td>
<td>Slave emulation enable 0. Bit SEE0 controls the slave debug request. When bit SEE0 is asserted (low) then PLD logic generates a short positive pulse for the slave EE0 input. Consequently MSC8122/26 enters debug mode despite the DIP switch SW4/4 setting. Toggling the DIP switch SW4/4 (DBG) setting achieves the same results as those described above. Reading bit SEE0 results in the sampling of the MSC8122/26 EE0 signal state.</td>
<td>Slave EE0 signal</td>
<td>R,W</td>
</tr>
<tr>
<td>3-4</td>
<td>Reserved</td>
<td>Not Implemented.</td>
<td>‘11’</td>
<td>R</td>
</tr>
<tr>
<td>5-7</td>
<td>BTMD[0:2]</td>
<td>Boot mode bits 0-2. During the slave power-on-reset configuration sequence the BTMD signals drive the logical level of the bit’s value. After the sequence the signals enter the Hi-Z state. Users may change, at any time, the value of bits BTMD[0:2]</td>
<td>‘000’ ‘001’</td>
<td>R,W</td>
</tr>
<tr>
<td>8-31</td>
<td>-</td>
<td>Not Implemented.</td>
<td>- - -</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 6-5. Summary Slave Configuration Modes

<table>
<thead>
<tr>
<th>DSI/SYS Dipswitch</th>
<th>MSC8122/26 Config. Inputs</th>
<th>Slave Configuration Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>BM[0:2]</td>
<td>SWTE</td>
</tr>
<tr>
<td>On</td>
<td>‘001’</td>
<td>0</td>
</tr>
<tr>
<td>Off</td>
<td>‘000’</td>
<td>1</td>
</tr>
</tbody>
</table>
6.1.5   Board Control/Status Register 4 (BCSR4)

BCSR4 is a status register accessed from the BCSR base address as a byte at offset 0x10. BCSR4 is a read-only register, readable at any time. BCSR4 fields are described in Table 6-6.

**Table 6-6. BCSR4 Description**

<table>
<thead>
<tr>
<th>Bit</th>
<th>Mnemonic</th>
<th>Function</th>
<th>Default Setting</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-1</td>
<td>SEE[0:1]</td>
<td>Slave emulation enable bits 0-1. Bits SEE[0:1] illustrate the state of the slave's EE0 and EE1 lines.</td>
<td>-</td>
<td>R</td>
</tr>
<tr>
<td>2</td>
<td>ETH-ON</td>
<td>ETH-ON. Ethernet-on from DIP-SW when the switch is on (low) and ETH mode is selected. To set ETH-MODE it is necessary to activate BCSR7[5].</td>
<td>1</td>
<td>R</td>
</tr>
<tr>
<td>3-4</td>
<td>SWOPT[1:2]</td>
<td><strong>Software option 1:2.</strong> Bits SWOPT[1:2] illustrate the state of dedicated DIP switch SW7/2-3. This DIP switch provides the option for manually changing a program’s flow.</td>
<td>SW7/2-3 Dipswitch</td>
<td>R</td>
</tr>
<tr>
<td>5-7</td>
<td>Reserved</td>
<td>Not Implemented.</td>
<td>‘111’</td>
<td>R</td>
</tr>
<tr>
<td>8-31</td>
<td>-</td>
<td>Not Implemented.</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>
Board Control/Status Register Identification 5 (BCSR5)

BCSR5 is a read-only register accessed from the BCSR base address as a byte at offset 0x14. The bits of BCSR5 fields are described in Table 6-7.

Table 6-7. BCSR5 Description

<table>
<thead>
<tr>
<th>Bit</th>
<th>Mnemonic</th>
<th>Function</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-1</td>
<td>Reserved</td>
<td>Not Implemented.</td>
<td>R</td>
</tr>
<tr>
<td>2-4</td>
<td>BREVN[0:2]</td>
<td>Board revision number 0-2. Field BREVN[0:2] represents the ADS revision code. See Table 6-8, ADS Revision Encoding, on page 6-8 for revised ADS encoding.</td>
<td>R</td>
</tr>
<tr>
<td>5-7</td>
<td>BCSRREV[0:2]</td>
<td>BCSR revision number 0-2. Field BCSRREV[0:2] represents the revised BCSR code. See Table 6-9, BCSR Revision Encoding, on page 6-8 for revised BCSR encoding.</td>
<td>R</td>
</tr>
<tr>
<td>8-31</td>
<td>-</td>
<td>Not Implemented.</td>
<td>-</td>
</tr>
</tbody>
</table>

ADS and BCSR revision encoding are explained in Table 6-8 and Table 6-9.

Table 6-8. ADS Revision Encoding

<table>
<thead>
<tr>
<th>Revision Number [0:2]</th>
<th>ADS Revision</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>PROTO</td>
</tr>
<tr>
<td>1</td>
<td>PILOT</td>
</tr>
<tr>
<td>2</td>
<td>A</td>
</tr>
<tr>
<td>3 - 7</td>
<td>Reserved</td>
</tr>
</tbody>
</table>

Table 6-9. BCSR Revision Encoding

<table>
<thead>
<tr>
<th>Revision Number [0:2]</th>
<th>BCSR Revision</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-7</td>
<td>1.x - 8.x</td>
</tr>
</tbody>
</table>
### Board Control/Status Register Miscellaneous 6 (BCSR6)

BCSR6 is a service register accessed from the BCSR base address as a byte at offset 0x18. The BCSR6 fields are described in Table 6-10.

#### Table 6-10. BCSR6 Description

<table>
<thead>
<tr>
<th>Bit</th>
<th>Mnemonic</th>
<th>Function</th>
<th>Default</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>FLUNLCK1</td>
<td>Flash 1 protection unlock. Default for hardware write protection bit FLUNLCK1 is negated (high), so that bit BCSR0 [0] FLASHPRT1 is locked. To unlock bit BCSR0 [0] FLASHPRT1, bit FLUNLCK1 must be asserted (low).</td>
<td>1</td>
<td>R,W</td>
</tr>
<tr>
<td>1</td>
<td>FLUNLCK2</td>
<td>Flash 2 protection unlock. Default for hardware write protection bit FLUNLCK2 is negated (high), so that bit BCSR0 [1] FLASHPRT2 is locked. To unlock bit BCSR0 [1] FLASHPRT2, bit FLUNLCK2 must be asserted (low).</td>
<td>1</td>
<td>R,W</td>
</tr>
<tr>
<td>2</td>
<td>LEDEN</td>
<td>LED Enable. When asserted (low) LEDs behave as in normal operation. If negated (high) board LEDs are darkened.</td>
<td>0</td>
<td>R,W</td>
</tr>
<tr>
<td>3</td>
<td>TEST</td>
<td>Test Mode Available. When asserted (high), MSC8122/26 enters test operation mode in accordance with the TSTSIG [1:3] bit settings. If negated (low) normal mode is presented.</td>
<td>0</td>
<td>R,W</td>
</tr>
<tr>
<td>4-6</td>
<td>TSTSIG [1:3]</td>
<td>Test Mode Select. See the MSC8122/26 spec for details.</td>
<td>0</td>
<td>R,W</td>
</tr>
<tr>
<td>7</td>
<td>MARK_BIT</td>
<td>Mark Bit. Serves for identification access to BCSR6 [2:6]. When asserted (high), access to bits BCSR6 [2-6] is possible. If this bit is zero, these bits are hidden.</td>
<td>0</td>
<td>R,W</td>
</tr>
<tr>
<td>8-31</td>
<td>-</td>
<td>Not Implemented.</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>
### Board Control/Status Register Miscellaneous 7 (BCSR7)

BCSR7 is a service register accessed from the BCSR base address as a byte at offset 0x1c. BCSR7 supports the features added by MSC8122/26. Its fields are described in Table 6-11.

#### Table 6-11. BCSR7 Description

<table>
<thead>
<tr>
<th>Bit</th>
<th>Mnemonic</th>
<th>Function</th>
<th>Default</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>I²C_CON</td>
<td>I²C connect. Connects the MSC8122/26 I²C bus with that of the MSC8103. Buses connect when set (high).</td>
<td>0</td>
<td>R/W</td>
</tr>
<tr>
<td>1-2</td>
<td>DSI_EX[0-1]</td>
<td>DSI-extended address mode. Sets DSI address switches to a given mode. Mode: &quot;00&quot; = 1 bit. Mode &quot;01&quot; = 2 bits. Mode &quot;10&quot; = 3 bits. Mode &quot;11&quot; = 4 bits. The MSC8102 sets mode to &quot;00&quot;. HCID3 is set to A10.</td>
<td>0-0</td>
<td>R,W</td>
</tr>
<tr>
<td>3</td>
<td>RPHY_RST</td>
<td>RMII/MII PHY Reset. Reset to RMII/MII Ethernet PHY. When set (low) initiates a reset.</td>
<td>0</td>
<td>R,W</td>
</tr>
<tr>
<td>4</td>
<td>VB_INT</td>
<td>VB-interface. Connects the MSC8103 bus to an external board. When set (high) the buses are connected.</td>
<td>0</td>
<td>R,W</td>
</tr>
<tr>
<td>5</td>
<td>ETH-ON</td>
<td>Ethernet-On. Together with the ETH-ON DIPswitch settings, this bit activates Ethernet MUXes to the ethernet mode. This bit is active high.</td>
<td>0</td>
<td>R,W</td>
</tr>
<tr>
<td>6</td>
<td>SPHY_RST</td>
<td>SMII PHY Reset. When set (low) this bit initiates a reset to the SMII Ethernet PHY.</td>
<td>1</td>
<td>R,W</td>
</tr>
<tr>
<td>7</td>
<td>ETH_RST</td>
<td>SMII/RMII Switch Reset. When set (low), this bit initiates a reset to the Mezzanine Ethernet switch.</td>
<td>1</td>
<td>R,W</td>
</tr>
<tr>
<td>8-31</td>
<td>-</td>
<td>Not Implemented.</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>
### Board Control/Status Register Miscellaneous 8 (BCSR8)

BCSR8 is a service register accessed from the BCSR base address as a byte at offset 0x20. BCSR8 supports the features added by MSC8122/26. Its fields are described in Table 6-12.

#### Table 6-12. BCSR8 Description

<table>
<thead>
<tr>
<th>Bit</th>
<th>Mnemonic</th>
<th>Function</th>
<th>Default</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>cPCI_OFF</td>
<td>cPCI J1/J2 off. Close all buffers to J1/J2. When set (high) the last cPCI buffers to J1/J2 are closed.</td>
<td>0</td>
<td>R/W</td>
</tr>
<tr>
<td>1</td>
<td>RMII</td>
<td>RMII/MII mode. Board Ethernet mode. When set (high) RMII or MII.</td>
<td>0</td>
<td>R/W</td>
</tr>
<tr>
<td>2-3</td>
<td>EREV</td>
<td>Ethernet Revision. Ethernet Mezzanine Revision ID for Switch-Board: 01 = RMII Board 10 = SMII Board</td>
<td>00</td>
<td>R</td>
</tr>
<tr>
<td>4</td>
<td>ETH1_RST</td>
<td>MII Mezzanine PHY Reset. When set (low), this bit initiates a reset to the Mezzanine Ethernet MII PHY.</td>
<td>1</td>
<td>R,W</td>
</tr>
<tr>
<td>5</td>
<td>ETH1_EN</td>
<td>MII Mezzanine PHY Enable. Enable Mezzanine Ethernet MII PHY. When set (high), this bit enables ETH_SW connection to PHY’s RX outputs.</td>
<td>1</td>
<td>R/W</td>
</tr>
<tr>
<td>6</td>
<td>EXTMST</td>
<td>TDM External Clock Master. Enable TDM push TDM clocks to slave. When set (low) this bit enables TDM connection.</td>
<td>0</td>
<td>R/W</td>
</tr>
<tr>
<td>7</td>
<td>ChipID0</td>
<td>8102/22/26 ChipID MSB. Together with BCSR9 [7], this bit displays the chip set for the slave: 00 = MSC8102 01 = MSC8122 10 = MSC8126 11 = N/A</td>
<td>0</td>
<td>R</td>
</tr>
<tr>
<td>8-31</td>
<td>-</td>
<td>Not Implemented.</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>
# Board Control/Status Register Ethernet 9 (BCSR9)

BCSR9 is a service register accessed from the BCSR base address as a byte at offset 0x24. BCSR9 supports the features added by the MSC8122/26. Its fields are described in Table 6-13.

## Table 6-13. BCSR9 Description

<table>
<thead>
<tr>
<th>Bit</th>
<th>Mnemonic</th>
<th>Function</th>
<th>Default</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-3</td>
<td>ETH_MODE</td>
<td>Ethernet mode set. Sets all board switches and strobes to current mode:</td>
<td>0000</td>
<td>R/W</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0000 - MSC8122/26ADS compatible mode (Ethernet OFF)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0001 - MAC2MAC RMII DSI-port to mezzanine switch</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0010 - MAC2MAC RMII TDM-port to mezzanine switch</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0011 - MAC2MAC SMII TDM-port to mezzanine switch</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0100 - As 0010 + mezzanine loopback</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0101 - As 0001 + mezzanine loopback</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0110 - As 0011 + mezzanine loopback</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0111 - MAC2PHY SMII TDM-port to SMII PHY</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1000 - MAC2PHY RMII TDM-port to RMII PHY (Davicom)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1001 - MAC2PHY RMII DSI-port to RMII PHY (Davicom)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1010 - MAC2PHY MII TDM-port to MII PHY (Davicom)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1011 - MAC2PHY MII DSI-port to MII PHY (Davicom)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1100 - MAC2MAC MII TDM-port to 8103</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1101 - MAC2MAC MII DSI-port to 8103</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>NOTE: For modes 1100 and 1101 of MAC-to-MAC, users must reset the SMII-PHY - Write &quot;0&quot; to BCSR7-6.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1110 - Spare</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1111 - Spare</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>FETH1_EN</td>
<td>MII host PHY enable. Enables host Ethernet MII-PHY. When set (high) enables PHY. Should be OFF for</td>
<td>0</td>
<td>R/W</td>
</tr>
<tr>
<td></td>
<td></td>
<td>I2C from host operation.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>FETH2_EN</td>
<td>RMII/MII Slave PHY Enable. When set (high), this bit enables the RMII/MII Ethernet slaves's PHY.</td>
<td>0</td>
<td>R/W</td>
</tr>
<tr>
<td>6</td>
<td>CODEC_16K</td>
<td>CODEC 16 KHz Enable. Enables 8KHz to TSI when CODEC is in 16KHZ. When set (high), this bit enables</td>
<td>0</td>
<td>R/W</td>
</tr>
<tr>
<td></td>
<td></td>
<td>16KHz mode.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>ChipID1</td>
<td>8102/22/26 ChipID MSB. Together with BCSR7 [7], this bit displays the chip set for the slave:</td>
<td>0</td>
<td>R</td>
</tr>
<tr>
<td></td>
<td></td>
<td>00 = MSC8102</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>01 = MSC8122</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>10 = MSC8126</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>11 = N/A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8-31</td>
<td>-</td>
<td>Not Implemented.</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>
## 1 Board Control/Status Register Miscellaneous 10 (BCSR10)

BCSR10 is a service register accessed from the BCSR base address as a byte at offset 0x28. BCSR10 supports the features added by MSC8122/26. BCSR10 fields are described in Table 6-14.

<table>
<thead>
<tr>
<th>Bit</th>
<th>Mnemonic</th>
<th>Function</th>
<th>Default</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>DSI_EX[2]</td>
<td>DSI-extended address mode: Sets DSI address switches as per mode. Mode: &quot;000&quot;=1 bit. Mode &quot;001&quot;=2 bits. Mode &quot;010&quot;=3 bits. Mode &quot;011&quot;=4 bits. MSC8122/26 Mode &quot;100&quot;.</td>
<td>1</td>
<td>R,W</td>
</tr>
<tr>
<td>1</td>
<td>MICTOREN</td>
<td>Enable Ds[63:0] Mictor Buffers. Active High.</td>
<td>0</td>
<td>R,W</td>
</tr>
<tr>
<td>2</td>
<td>CODECMOD</td>
<td>Connects the Codec Direct to TDM. Active Low. When High the Codec is routed Via TSI.</td>
<td>0</td>
<td>R,W</td>
</tr>
<tr>
<td>3-7</td>
<td>Reserved</td>
<td>-</td>
<td>0</td>
<td>R,W</td>
</tr>
</tbody>
</table>
Both the MSC8122/26ADS and the MSC8102/22/26ADS contain one programmable logic device, the Altera CPLD, whose code contains the ADS control and status functions. The CPLD device implemented in U37 is an EPM3512AFC256-7. Code design is undertaken in Verilog HDL. For each ADS board, the program format is listed in an attached disk.

### A.1 MSC8102/22/26ADS CPLD Code

The traditional BCSR (Altera) CPLD code version 1.0 for the MSC8102/22/26ADS is as follows:

```verilog
// hds header_start
// BCSR2_main Rev 012 for 8122ADS board
// hds header_end

`resetall
`timescale 1ns/10ps
module BCSR_8122_12(
    clk, extclk, Data, reset, XAt7, XAt8, XAt9, XAt10, A27, A28, A29,
    nWE, nR_W, nCS0, nBCSR_CS, nFCSh, nHCS1, nHBCS,
    DSItoSYS, nSYS64, nFCFG, MODCK1s, MODCK2s, SWOPT,
    nRS232EN_1, nRS232EN_2, nFETH_RST, nFRM_RST, FRMtoTSI, nCODEC_EN,
    nBOOTPh, nBOOTPs, nDBUFXEN, nDBUFBEN, GA, nPRSTs,
    HEE0_In, HEE0_Out, HEE1_In, HEE1_Out,
    SEE0_In, SEE0_Out, SEE1_In, EE1_misc_In, EE1_misc_Out, n32to64En,
    SIGH_LED, SIGS_LED, SEE1_LED,
    Aborth_In, HReseth_In, SReseth_In, Aborts_In, HResets_In, SResets_In,
    MODCK1h, MODCK2h, MODCK3h, MODCK4h, MODCK5h, MODCK6h, MODCKh_Out,
    nHRESETh, nSRESETh, nRESETs, nRESETs_In, nHRST, nHRST_In, nNMIh, nNMIs,
    BM_Out, CNFGS_Out, DSI64_Out, DSISYNC_Out, SWTE_Out, MODCK1s_Out, MODCK2s_Out,
    RSTCFG_Out, TEST_EN, TEST_SIG, BCTL1_In, nCS0s, BCTL1_Outb
    ,B_8102, B_8122_1, B_8122_2, B_8126, BCSR_CORE1, BCSR_CORE2, SUP1_IND,
    SUP2_IND, SUP3_IND, CS6hb, SYSENb, PCI_PRSNTb, DACK4b, DACK3b, DACK2b, DACK1b,

```

MSC8122/26ADS Reference Manual, Rev. B0.1
XDBUFXENb, EXT_PCI1, EXT_PCI, ESW_CSb, BCTRL1_XOutb, A26, XHRESETsb, BHRESEThb, IRQ1hb, IRQ2hb, IRQ1cPCIb, RSTCNFhb, SYSBUFONb, RMMI2, ETH_IND, Ah5, Ah6, TTS0, HCID, BCONF, BREVN, ABUFONb, ESW_RESETb, ChainSel0, ChainSel1, ChainSel3, ChainSel4, ETH_SW_ON, EREV, LEDDRVh, LEDDRVs, DBBhb, DBBsb, CORELEDh, CORELEDs, RUNhLED, RUNsLED, LED3V3, ESW1RSTb, ESW1EN, nFETH1RST, SMIIPHYRSTb, BCSR_PPC_Sb, BCSR_TDM_Sb, BCSR_SMIIb, BCSR_MSEL, BCSR_MB, BCSR_RSMIIb, EXT_MSTb, FETH1EN, FETH2EN, ESW_RDb, ESW_WRh, BCTL0sb, SPR1, SMII_LEdb, I2C_CONT, I2C_CONT1, FSYNC, FSYNC2, GPL1s

/**/ General Definition /**/
`define ASSERTED 0
`define NEGATED 1
`define ACTIVE_LOW 0
`define ACTIVE_HIGH 1
`define PRESSED 0
`define RELEASED 1
`define WRITE 1
`define READ 0
`define From_BCSR 1
`define From_FLASH 0
`define S0 7'b00000111
`define S1 7'b0000111
`define S2 7'b0000001
`define S3 7'b0000010
`define S4 7'b0001010
`define S5 7'b0001001
`define S6 7'b0001111
`define S7 7'b0010010
`define S8 7'b0001001
`define S9 7'b00101111
`define S10 7'b0010001
`define S11 7'b0100001
`define S12 7'b0110001
`define S13 7'b0110010
/** State Registers Definition **/

reg [6:0] state;

parameter BCSR_ver = 3'b010; // Current version of code is 2 (1.2)
parameter divpr = 12;
parameter divsee0 = 5;
parameter divhee0 = 5;
parameter divhresets = 10;
parameter divhreseth = 10;
parameter div2st = 5; // 4 // to apply delay equals 'delay = extclk * 2**10' or 50usec@20Mhz
parameter div1st = 13; // 2; //16 // to apply delay equals 'delay = extclk * 2**20' or 50msec@20Mhz
parameter zero = 1'b0;
parameter divd = div2st;
parameter divc = div1st;

/** Inputs **/
input clk, // From Host clock buffer
    extclk, // Clock from external clock oscillator
    reset; // Main Power-on-Reset
input Ah5,Ah6,A26,A27,A28,A29, // 60-x bus addresses lines
    nWE,nR_W,BCTL0sb, // and controls
    nCS0,nBCSR_CS; // Chip selects
    DSItoSYS, // DIP-switch selectes Slave boot source
    nSYS64, // DIP-switch selectes Slave data bus distribution
    nFCFG, // DIP-switch selectes Host boot source (BCSR or Flash)
    MODCK1s,MODCK2s, // DIP-switch select MODCK1 for the Slave
    MODCK1h,MODCK2h,MODCK3h,
    MODCK4h,MODCK5h,MODCK6h; // DIP-switch select MODCK1 for the Host
input nHCS1,nHBCS; // Chip Selects for Slaves
XAt7,XAt8,XAt9,XAt10; // Addresses together with nHCS1,2 & nHBCS for select Slaves on/off- board

input [0:3] GA;       // Geographic Address of System Board setting by on-CPCI backplane jumpers,
                      // default value equals 0xf

input Aborth_In, Aborts_In, // Pushbuttons of NMI for Host & Slave
       HReseth_In, HResets_In, // Pushbuttons of HReset for Host & Slave
       SReseth_In, SResets_In; // Pushbuttons of Sreset for Host & Slave

input HEE0_In,HEE1_In; // Inputs from DIP-switch for Host EE0,1 control
input SEE0_In,SEE1_In; // Inputs from DIP-switch for Slave EE0,1 control
input EE1_misc_In;  // Debug acknowledge comming from another ADS
input [1:2] SWOPT;    // Software options applied from DIP-switch
input [1:3] nHRST_In; // Three HRESETs from external boards
input BCTL1_In, nCS0s; // Added for 8102 BCTL1 bug workaround

input B_8102,
    B_8122_1,
    B_8122_2,
    B_8126;    // Set Chip

input CS6hb,        // Rack Host CS6
       SYSENb,     // System slot indication
       PCI_PRSNTb, // PCI Rack present indication
       DACK4b,DACK3b,
       DACK2b,DACK1b; // DMA Acknowledge from Host

input ESW_CSb;      // Mezzanine Ethernet Switch Chip Select
input IRQ1hb;       // INT_OUT from Slave

input [0:1] BCONF;   // Board Configuration from Pull-Ups
input [0:2] BREVN;   // Board Configuration from Pull-Ups
input ChainSel1;     // JTAG Chain Select

input SPR1;          // Spare DIP-SW for internal test only
input FSYNC;         // 8KHz/16KHz Sync from Codec

/*** Ethernet Pins ***/
input ETH_SW_ON;     // Ethernet DIP-SW On
input [0:1] EREV;     // Ethernet Mezzanine Board ID

/*** LED Indication strobes ***/
```vhdl
LEDDRVh, LEDDRVVs, DBBhb,  // LED Indication strobes
DBBsbg;

input GPL1s;

//*** Bidirectional ***//
inout [0:7] Data;  // Eight bit of buffered 60-x bus
inout nHRESETh, nHRESETs,  // HRESET for Host & Slave
     nSRESETh, nSRESETs,  // SRESET for Host & Slave
     BHRESETb;  // HRESET from Host to Slave in cPCI Rack
inout nPRSTs;  // Wire connected to PORRESET of Slave
inout HEE1_Out;  // EE1 for Host
inout IRQ1cPCIb;  // Interrupt from Slave on Peripheral-Slot to cPCI

//*** Outputs ***//
output nNMIh, nNMIs;  // IRQ0 for Host & Slave
output nFCSH,  // Host Side Flash chip enable
     nRS232EN_1, nRS232EN_2,  // Enable for RS232 transmitters
     nFETH_RST,  // Reset for Ethernet Phy
     nFRM_RST,  // E1/T1 Farmer Reset
     FRMtoTSI,  // Switch FALC56 channels to TSI or to TDM3 of 8102
     nCODEC_EN,  // Chip select to CODEC device
     nBOOTPh, nBOOTPs;  // Boot Sector protection for Flashes
output HEE0_Out;  // EE0 for Host
output SEE0_Out;  // EE0 for Slave
output nDBUFXEN,  // Enable for on-board peripherals buffer
     nDBUFBEN,  // Enable for off-board peripherals buffer to access to another boards
     ABUFXENb;  // Disable Addr Buffers of 8103 when Slave Board
output [3:1] MODCKh_Out;  // MODCK1-3 comming from DIP-switch for setting Host(8101) PLL mode
output [1:3] nHRST;  // Three HRESETs to external boards
output BCSR_CORE1,  // Slave Core Voltage selector
     BCSR_CORE2;
output SUP1_IND,
     SUP2_IND,
     SUP3_IND;  // Slave Core Supply Indication
```
output XDBUXENb, // Buffer Enable to cPCI J1/J2
EXT_PCI1,EXT_PCI; // Direction of cPCI-J1/J2 Buffers
output BCTL1_XOutb;// Mezzanine Ethernet Switch Buffer Enable
output IRQ2hb; // Interrupt from Peripheral slot to Host on SYS-Slot
output RSTCNFhb; // RSTCNF for Host (Set on Peripheral Slot)
output XHRESETsb; // XHRESETsb is spare reset to cPCI
output SYSBUFFXENb; // Enable CS6h towards Rack Slave
output RMII2; // RMII/MII Mode for RPHY
output ETH_IND; // Ethernet Indication
output [0:3] HCID; // Host ID
output TTs0; // HA7 / TT0 Slave
output ChainSel0, // JTAG Chain Select
    ChainSel3,
    ChainSel4;
output I2C_CONT, // Connects Host&Slave I2C Bus
    I2C_CONT1; // Connects Host&Slave I2C Bus (or Host MII)
output FSYNC2; // 8KHz Sync to TSI From Codec

    //*** Ethernet Pins ***/
output ESW_RESETb, // Reset Mezzanine Switch
    nFETH1RST, // Reset RMII/MII PHY
    SMIIPHYRSTb, // Reset SMII PHY
    ESW1RSTb, // Reset Mezzanine MII-PHY
    FETH1EN, // Enable Host MII PHY
    FETH2EN, // Enable Slave R/MII PHY
    ESW1EN; // Enable Mezzanine MII-PHY
output BCSR_PPC_Sb,// ETH-SW1 Mux Enable
    BCSR_TDM_Sb, // ETH-SW2 Mux Enable
    BCSR_SMIIb, // ETH-SW5 Mux Enable
    BCSR_RSMIIb; // ETH-SW4 Mux Enable
output [0:3] BCSR_MSEL,// ETH-MUX1 Clock Enables
    BCSR_MB; // ETH-MUX1 Clock selectors
output EXT_MSTb; // TDM Clock Master Enable
output ESW_RDb,
    ESW_WRb; // Ethernet SW Controls
/*** LED Indication strobes ***/
output CORELEDh, CORELEDS,
      RUNhLED, RUNsLED, LED3V3, SMII_LEDb;

**** LEDs drive pins ****/
//output nLED_EN;   // Allow to switch off all LEDs on the ADS accordingly to TTM req.
output [0:1] SIGH_LED,    // Misc two LEDs for SW indication for Host
       SIGS_LED;   // Misc two LEDs for SW indication for Slave
output SEE1_LED;  // Slave in-Debug Mode indication

**** Slave configuration pins ****/
output [0:2] BM_Out;       // Boot Mode selection three bits for the
output n32to64En;   // Output for MUX control
output CNFGS_Out,    // Configuration Source
       DSI64_Out,   // DSI 64bit setting
       DSISYNC_Out, // DSI Synchronous Mode
       SWTE_Out,    // Software Watchdog Timer Enable
       RSTCFG_Out,  // Reset Configuration Mode
       MODCK1s_Out, // Two lines of MODCK
       MODCK2s_Out;

`define HD_PINS
{CNFGS_Out, DSI64_Out, DSISYNC_Out, SWTE_Out, RSTCFG_Out, MODCK1s_Out, MODCK2s_Out}
// All the pins defined Slave PORESET configuration

**** Test mode ****/
output TEST_EN;    // Enter 8102 to test mode
output [1:3] TEST_SIG;  // Select test mode
output EE1_misc_Out;
output BCTLL_Outb;  // Enable BDs[0:15] Buffer

**** Registers Definition ****/
reg [0:7] Data0;

***** BCSR0 Description *****
reg [0:7] BCSR0;
`define       FLASHPRT1 BCSR0[0] // Flash 1 H/W Protection


`define     FLASHPRT2  BCSR0[1]  // Flash 2 H/W Protection
`define     FRM_RST    BCSR0[2]  // Framer E1/T1 Reset control
`define     CODEC_EN   BCSR0[3]  // CODEC Chip Select
`define     SIGNALS0   BCSR0[4]  // Signal-0 LED Slave
`define     SIGNALS1   BCSR0[5]  // Signal-1 LED Slave
`define     SIGNALH0   BCSR0[6]  // Signal-0 LED Host
`define     SIGNALH1   BCSR0[7]  // Signal-1 LED Host

//**** BCSR1 Description ****/
reg [0:7] BCSR1;
`define     RECONF   BCSR1[0]  // Start Slave Re-configuration
`define     HRST1    BCSR1[1]  // Hard Reset to External Slave1
`define     HRST2    BCSR1[2]  // Hard Reset to External Slave2
`define     ATM_RST  BCSR1[3]  // SPARE N/A
`define     HRST3    BCSR1[4]  // Hard Reset to External Slave3
`define     FETH_RST BCSR1[5]  // Fast Ethernet phy Reset
`define     RS232EN_1 BCSR1[6]  // RS232 Transceiver Host Side Enable
`define     RS232EN_2 BCSR1[7]  // UART Transceiver Slave Side Enable

//**** BCSR2 Description ****/
reg [0:7] BCSR2;
`define     RSTCNF   BCSR2[0]  // Reset Configuration Mode
`define     CNFG     BCSR2[1]  // Configuration Source
`define     SWTE     BCSR2[2]  // Software Watchdog Timer Enable
`define     DSI64    BCSR2[3]  // System/DSI 64/32 bit
`define     DSI SYNC BCSR2[4]  // DSI Synchronous Mode
`define     HRST     BCSR2[5]  // Hard Reset to Slave
`define     SRST     BCSR2[6]  // Soft Reset to Slave
`define     FRMtoTSI BCSR2[7]  // FALC56 connects to TSI or to TDM3 of 8102

//**** BCSR3 Description ****/
reg [0:7] BCSR3;
`define     CLKMD    BCSR3[0:1]  // Clock Mode Setting Bits 1-2 for Slave
`define     SEE0     BCSR3[2]    // Slave Emulation Enable 0
`define     RSV34    BCSR3[3:4]  // Reserved two bits

//**** BCSR6 Description ****/
reg [0:7] BCSR6;
`define FLUNLCK1 BCSR6[0]   // Flash 1 Protection Unlock
`define FLUNLCK2 BCSR6[1]   // Flash 2 Protection Unlock
`define LEDEN   BCSR6[2]   // LED Enable | Hidden
`define TEST     BCSR6[3]   // Test Mode Available
`define TESTSIG BCSR6[4:6]  // Test Mode Select | bits
`define MARK    BCSR6[7]   // Mark Bit |

***** BCSR7 Description *****

reg [0:7] BCSR7;
`define I2C_CON   BCSR7[0]   // I2C Connect (Slave+Host)
`define DSI_EX    BCSR7[1:2] // DSI Extended Mode
`define RPHY_RST  BCSR7[3]   // R/MII PHY Reset
`define VB_INT    BCSR7[4]   // VB-Interface Connect
`define ETH_ON    BCSR7[5]   // Ethernet ON
`define SPHY_RST  BCSR7[6]   // SMII PHY Reset
`define ETH_RST   BCSR7[7]   // Reset Eth Switch

***** BCSR8 Description *****

reg [0:7] BCSR8;
`define cPCI_OFF  BCSR8[0]   // cPCI J1/J2 Buffers Close
`define RMII     BCSR8[1]   // RMII/MII Mode
`define EREV_BCSR BCSR8[2:3] // Ethernet Board Revision/ID
`define ETH1_RST BCSR8[4]   // Reset Mezzanine MII-PHY
`define ETH1_EN  BCSR8[5]   // Enable Mezzanine MII-PHY
`define EXTMST   BCSR8[6]   // TDM Clock Master
`define CHIPID0  BCSR8[7]   // Chip ID 8102/22/26

***** BCSR9 Description *****

reg [0:7] BCSR9;
`define ETH_MODE BCSR9[0:3] // Ethernet Mode
`define FETHL_EN BCSR9[4]   // Enable Host MII PHY
`define FETH2_EN BCSR9[5]   // Enable Slave R/MII PHY
`define CODEC_16K BCSR9[6]  // Codec is in 16KHz Sync Mode
`define CHIPID1  BCSR9[7]   // Chip ID 8102/22/26

***** BCSR10 Description *****

reg [0:7] BCSR10;
`define DSI_EX2  BCSR10[0]  // DSI Extended Mode
`define BCSR10_SPR BCSR10[1:7] // Spare
reg        HEE0_Out;
reg        SEE0_Out;
reg        EE1misc_Out;
reg        BCSRCsyn;
reg        HEE0_In_syn;
reg        SEE0_In_syn;
reg        nHRESETh_d;
reg        nHRESETs_d;
reg [0:3]  GA_r;           // GA latch

reg [divsee0:0] SEE0_count;
reg [divhee0:0] HEE0_count;
reg [divhreseth:0] nHRESETh_count;
reg [divhresets:0] nHRESETs_count;
reg        NM1h,NM1s,HRESETh,SRESETh,HRESETs,SRESETs;
reg        clock_divider;
reg [divc:0]  counter;
reg [divd:0]  hhr,shr,nhr,hsr,ssr,nsr;
reg [0:3]    HCID;
reg        TTS0;
reg        FSYNC_8;        // 8MHz for TSI

reg        BCSR_PPC_Sb,    // ETH-SW1 Mux Enable
            BCSR_TDM_Sb,    // ETH-SW2 Mux Enable
            BCSR_SMIIb,    // ETH-SW5 Mux Enable
            BCSR_RSMIIb;   // ETH-SW4 Mux Enable
reg [0:3]  BCSR_MSEL,     // ETH-MUX1 Clock Enables
            BCSR_MB;      // ETH-MUX1 Clock selectors

wire [1:0] CFG_ADDR = {A27,A28};
wire [3:0] BCSR_ADDR = {A26,A27,A28,A29};
wire READ_CFG_FROM_BCSR;
wire [0:7] CFG_BYTE3_DEF;
wire [0:7] BCSR4;
wire [5:7] BCSR5; // Bits 0-4 are comming via ext. buffer
wire [0:3] SSEL; // Address vector for Slaves Select
wire [0:3] ON_BRD_SLAVE_ADDR;
wire ON_BRD_HOST_PRPH; // Active when select any of Host peripheral
wire OFF_BRD_SLAVE; // Active when access to off-board Slave
wire Write_to_BCSR;
wire Read_from_BCSR;
wire [0:7] BCSR1r, BCSR2r;
wire [0:7] BCSR2_PON_DSI, BCSR2_PON_SYS; // BCSR3_PON;
wire [0:1] BCSR3_PONh;
wire [0:4] BCSR3_PONl;
wire [0:2] BTMD;
wire CNFG, DSI64, DSISYNC, SWTE, RSTCNF;
wire [0:1] CLKMD;
wire ADDR3, ADDR5;
wire WRITE3;
wire [1:10] CFGREG;
wire CONFIG_EN;
wire CoreConfig1;
wire CoreConfig2;
wire SYS_SLOT;
wire PER_SLOT;
wire VB_ACCESS;
wire Activate_ETH;
wire RST_FROM_cPCI;
wire Internal_Test;
wire CID1, CID0;

/** Hard Reset Configuration Word (HRCW) for 810l **/

parameter EARB_DEF = 1'b0, // bit 0
                 EXMC_DEF = 1'b0,
                 nIRQ7INT_DEF = 1'b1,
                 EBM_DEF = 1'b0,
BPS_DEF = 2'b10,
SCDIS_DEF = 1'b0,
ISPS_DEF = 1'b0, // bit 7
CFG_BYTE0_DEF = // 0x28
{EARB_DEF,EXMC_DEF,nIRQ7INT_DEF,EBM_DEF,BPS_DEF,SCDIS_DEF,ISPS_DEF},
IRPC_DEF = 2'b00, // bit 8
DPPC_DEF = 2'b00,
NMIOUT_DEF = 1'b0,
ISB_DEF = 3'b000, // bit 15
CFG_BYTE1_DEF = // 0x0
{IRPC_DEF,DPPC_DEF,NMIOUT_DEF,ISB_DEF},
RSV16_DEF = 1'b0, // bit 16
BBD_DEF = 1'b0,
MMR_DEF = 2'b11,
RSV20_21_DEF = 2'b00,
TCPC_DEF = 2'b10, // bit 23
CFG_BYTE2_DEF = // 0x32
{RSV16_DEF,BBD_DEF,MMR_DEF,RSV20_21_DEF,TCPC_DEF},
BC1PC_DEF = 2'b00, // bit 24
RSV26_DEF = 1'b0,
DLLDIS_DEF = 1'b1,
// MODCK_H_DEF = 3'b100,
RSV31_DEF = 1'b0, // bit 31
CFG_BYTE3_0_3DEF = // 0x01
{BC1PC_DEF,RSV26_DEF,DLLDIS_DEF},
CFG_BYTE3_7DEF = RSV31_DEF;
/*** Power-on-Reset value of BCSR0-3, BCSR6 ***/
//*** BCSR0 PON value ***/
FLASHPRT1_PON = 1'b0,
FLASHPRT2_PON = 1'b0,
FRM_RST_PON = 1'b1,
CODEC_EN_PON = 1'b1, // CODEC disable
SIGNALS0_PON = 1'b1,
SIGNALS1_PON = 1'b1,
SIGNALH0_PON = 1'b1,
SIGNALH1_PON = 1'b1,

BCSR0_PON = {FLASHPR1_PON, FLASHPR2_PON, FRM_RST_PON,
CODEC_EN_PON, SIGNALS0_PON, SIGNALS1_PON,
SIGNALH0_PON, SIGNALH1_PON
},  // 0x2f

//*** BCSR1 PON value ***/
RECONF_PON = 1'b1,
HRST1_PON = 1'b1,
HRST2_PON = 1'b1,
ATM_RST_PON = 1'b1,
HRST3_PON = 1'b1,
FETH_RST_PON = 1'b1,
RS232EN_1_PON = 1'b1, // RS232-1 disable
RS232EN_2_PON = 1'b0, // RS232-2 Enable

BCSR1_PON = {RECONF_PON, HRST1_PON, HRST2_PON, ATM_RST_PON,
HRST3_PON, FETH_RST_PON, RS232EN_1_PON,
RS232EN_2_PON
}, // 0xff

//*** BCSR2 PON value ***/
RSTCNF_PON = 1'b0, // BCSR2.0
DSISYN_PON = 1'b0, // BCSR2.4
HRST_PON = 1'b1, // BCSR2.5
SRST_PON = 1'b1, // BCSR2.6
FRMtoTSI_PON = 1'b1, // BCSR2.7

//*** BCSR3 PON value ***/
RSV34_PON = 2'b11,

//*** BCSR6 PON value ***/
FLUNLCK1_PON = 1'b1,
FLUNLCK2_PON = 1'b1,
LEDEN = 1'b0,
TEST = 1'b0, // Disable Test Mode
TESTSIG = 3'b0,
MARK = 1'b0,

BCSR6_PON = {FLUNLCK1_PON, FLUNLCK2_PON, LEDEN, TEST,
TESTSIG, MARK}, // 0xd0

/** BCSR7 PON value ***/
I2C_CON_PON = 1'b0,
DSI_EX_PON = 2'b00,
RPHY_RST_PON = 1'b0, // Reset PHY and CODEC

//
RPHY_RST_PON = 1'b1,
VB_INT_PON = 1'b0,
ETH_ON_PON = 1'b0,
SPHY_RST_PON = 1'b1,
ETH_RST_PON = 1'b1,

BCSR7_PON =
{I2C_CON_PON,DSI_EX_PON,RPHY_RST_PON,VB_INT_PON,ETH_ON_PON,SPHY_RST_PON,
  ETH_RST_PON}, // 0x0B

/** BCSR8 PON value ***/
cPCI_OFF_PON = 1'b0,
RMII_PON = 1'b0,
EREV_PON = 2'b00,
ETH1_RST_PON = 1'b1,
ETH1_EN_PON = 1'b1,
EXTMST_PON = 1'b0,

x = 1'b0,

BCSR8_PON =
cPCI_OFF_PON,RMII_PON,EREV_PON,ETH1_RST_PON,ETH1_EN_PON,EXTMST_PON,
x}, // 0x0C

/** BCSR9 PON value ***/
ETH_MODE_PON = 4'b0000,
FETH1_EN_PON = 1'b0,
FETH2_EN_PON = 1'b0,
CODEC_16K_PON = 1'b0,

BCSR9_PON = {ETH_MODE_PON,FETH1_EN_PON,FETH2_EN_PON,CODEC_16K_PON,x},

/** BCSR10 PON value ***/
DSI_EX2_PON = 1'b1,

BCSR10_PON = {DSI_EX2_PON,7'b0000000};

/** Nodes ***/
assign READ_CFG_FROM_BCSR = (nCS0 == 'ASSERTED) && (nHRESETh == 'ASSERTED)
&& (nFCFG == `From_BCSR) && !nR_W;
assign SSEL[0:3] = {XAt7,XAt8,XAt9,XAt10};
assign SYS_SLOT = !(PCI_PRSNTb || SYSENb);
assign PER_SLOT = !PCI_PRSNTb && SYSENb;
assign ON_BRD_SLAVE_ADDR = GAr[0:3]; //Apply 0xf for any out-of-CPCI backplane configuration
assign ON_BRD_HOST_PRPH = !(nCS0 && nBCSR_CS);
assign XDBUFXENb = !(OFF_BRD_SLAVE || PER_SLOT);
assign EXT_PCI1 = (OFF_BRD_SLAVE && nR_W) || ((PER_SLOT || VB_ACCESS) && (!nR_W));
assign EXT_PCI = !PER_SLOT; // "Read" Addr & Cntl from cPCI only in peripheral slot

// External Board Present and has access on
assign VB_ACCESS = (!DACK4b || !DACK3b || !DACK2b || !DACK1b) && `VB_INT ;

// OFF Board Slave means Local 8103 is hosting a 8122 slave from rack
assign OFF_BRD_SLAVE = !CS6hb ||
(SYS_SLOT && ((SSEL != ON_BRD_SLAVE_ADDR) && !nHCS1)) && nHBCS ; //If CS to 8102 on board then
assign "0"
assign nDBUFBEN = nHCS1 && nHBCS && (!ON_BRD_HOST_PRPH || OFF_BRD_SLAVE ||
VB_ACCESS));

// nDBUFXEN prevents contention on the Data bus to external boards
assign nDBUFXEN = nHCS1 && nHBCS && (!OFF_BRD_SLAVE || VB_ACCESS));
assign Write_to_BCSR = BCSRCSsyn;
assign Read_from_BCSR = !nR_W && !nBCSR_CS && nHRESETh;
assign BCSR2_PON_DSI = {RSTCNF_PON,2'b10,nSYS64,DSISYN_PON,HRST_PON,SRST_PON,FRMtoTSI_PON};
assign BCSR2_PON_SYS = {RSTCNF_PON,2'b01,nSYS64,DSISYN_PON,HRST_PON,SRST_PON,FRMtoTSI_PON};
assign BTMD = DSItoSYS ? 3'b0 : 3'b001;
assign BCSR3_PONh = {MODCK1s,MODCK2s};
assign BCSR3_PONl = {RSV34_PON,BTMD};
assign BCSR4 = {SEE0_In,SEE1_In,ETH_SW_ON,SWOPT[1:2],3'b111};
assign BCSR5 = BCSR_ver;
assign ADDR3 = !A27 && A28 && A29;
assign ADDR5 = A27 && !A28 && A29;
WRITE3 = Write_to_BCSR && ADDR3 && (nWE == `ASSERTED);

assign BCSR1r[0:7] = `{RECONF, nHRST_In[1:2], `ATM_RST, nHRST_In[3], `FETH_RST, `RS232EN_1, `RS232EN_2};
assign BCSR2r[0:7] = `{RSTCNF, `CNFG, `SWTE, nSYS64, `DSISYNC, nHRESETs, nRESets, `FRMtoTSI};
assign CFG_BYTE3_DEF = {`RECONF, `CNFG, `SWTE, nSYS64, `DSISYNC, `RSTCNF, `CLKMD};
assign CONFIG_EN = nHRESETs && nPRSTs;
assign BM_Out[0:2] = (CONFIG_EN == `ASSERTED)? `BTMD : 4'bz;
assign `HD_PINS = (CONFIG_EN == `ASSERTED)&&(!PER SLOT)? CFGREG[4:10] : (Internal_Test ?
{1'b0, MODCK1h, MODCK3h, MODCK6h, 1'b0, MODCK2h, 1'b0} : 7'bz);

assign n32to64En = !(DSI64) || Activate_ETH && ((ETH_MODE == 4'b0001) || (ETH_MODE == 4'b0101) ||
(ETH_MODE == 4'b1011) || (ETH_MODE == 4'b1101));

//** Outputs ***/
assign TEST_EN = `TEST || Internal_Test;
assign TEST_SIG[1:3] = (`TEST == `ACTIVE_HIGH)? TESTSIG : 3'bz;
assign EE1_misc_In = EE1_misc_In; // temporary Null-Logic
assign MODCKh_Out = (nHRESETs == `ASSERTED)? {MODCK3h, MODCK2h, MODCK1h} : 3'bz;
assign nFCSh = (nFCFG == `From_FLASH)? nCS0 : ((nHRESETs == `ASSERTED)? 1'b1 :
modCS);
assign FRMtoTSI = `FRMtoTSI;
assign nBOOTPh = (`FLASHPRT1 == 0)? `ASSERTED : `NEGATED;
assign nBOOTPs = (`FLASHPRT2 == 0)? `ASSERTED : `NEGATED;
assign SIGH_LED[0] = (`LEDEN == `ACTIVE_LOW)? ((nSRESETs == `ASSERTED)? `ASSERTED :
`SIGNALH0): `NEGATED;
   // Also indicates SRESET assertion
assign SIGH_LED[1] = (`LEDEN == `ACTIVE_LOW)? ((nSRESETs == `ASSERTED)? `ASSERTED :
`SIGNALH1): `NEGATED;
   // Also indicates HRESET assertion
assign SIGS_LED[0] = (`LEDEN == `ACTIVE_LOW)? ((nSRESETs == `ASSERTED)? `ASSERTED :
`SIGNALS0): `NEGATED;
   // Also indicates SRESET assertion
assign SIGS_LED[1] = (`LEDEN == `ACTIVE_LOW)? ((nHRESETs == `ASSERTED)? `ASSERTED : `NEGATED);
  // Also indicates HRESET assertion
assign SEE1_LED = (`LEDEN == `ACTIVE_LOW)? !SEE1_In : `NEGATED;
  // Debug Acknowledge indication for the Slave
assign nCODEC_EN = (`CODEC_EN == `ACTIVE_LOW)? `ASSERTED : `NEGATED;
assign nRS232EN_1 = (`RS232EN_1 == `ACTIVE_LOW)? `ASSERTED : `NEGATED;
assign nRS232EN_2 = (`RS232EN_2 == `ACTIVE_LOW)? `ASSERTED : `NEGATED;
assign nFETH1RST = (`FETH_RST == `ACTIVE_LOW)||(nHRESETh == `ASSERTED)? `ASSERTED : `NEGATED;
assign nFRM_RST = (`FRM_RST == `ACTIVE_LOW)||(nHRESETs == `ASSERTED)? `ASSERTED : `NEGATED;
assign HEE1_Out = (HEE1_In == `ASSERTED)? HEE1_In : 1'bz;
assign nNMIh = ( NMIh == `ASSERTED)? `ASSERTED : `NEGATED;
assign nNMIs = ( NMIs == `ASSERTED)? `ASSERTED : 1'bz;

  //***  Resets Output ***  //
assign nHRESETh = ( HRSETh == `ASSERTED || reset ==`ASSERTED)? `ASSERTED : 1'bz ;
assign nSRESETh = ( SRESETh == `ASSERTED)? `ASSERTED : 1'bz ;
assign nSRESETs = (`SRST == `ACTIVE_LOW || SRESETs == `ASSERTED)? `ASSERTED : 1'bz;
assign nHRESETs = ( RST_FROM_cPCI || (`HRST == `ACTIVE_LOW) || (HRESETs == `ASSERTED))? `ASSERTED : 1'bz;

// PowerOnReset could be given from Rack to SyS-Slot
assign nPRSTs = (`RECONF == `ACTIVE_LOW) || (!PER_SLOT)&&(nHRESETh == `ASSERTED)) ||
                SYS_SLOT && (BHRESEThb == `ASSERTED))? `ASSERTED : 1'bz;

//Identify Reset from System Slot
assign RST_FROM_cPCI = PER_SLOT && !nHRST_In[1] ;

assign nHRST[1] = (`cPCI_OFF||(!SYS_SLOT)) ? 1'b1 : (`HRST1 == `ASSERTED)|| (nHRESETs ==
                  `ASSERTED)? 1'b0 : `NEGATED);
assign nHRST[2] = (`cPCI_OFF||(!SYS_SLOT)) ? 1'b1 : (`HRST2 == `ASSERTED)? `HRST2 :
                  `NEGATED);
assign nHRST[3] = (`cPCI_OFF||(!SYS_SLOT)) ? 1'b1 : (`HRST3 == `ASSERTED)? `HRST3 :
                  `NEGATED);
assign BCTL1_Outb    = BCTL1_In && nCS0s && ESW_CSb;
assign BCTL1_XOutb  = ESW_CSb;

//*** Execution Section ***//

always @ (posedge clk) BCSRCSsyn = !nBCSR_CS;

always @ (posedge nHRESETs) GA[0:3] = GA[0:3];  //*** Latch GA lines ***//

//*** Generation Debug Request for Host & Slave***//

//*** Delayed Host Hard Reset ***//
always @ (posedge extclk)
begin
  if (nHRESETh == `ASSERTED)
    begin nHRESETh_count = 0; nHRESETh_d = 0; end
  else begin
    if (nHRESETh_count[divhreseth] != 1) begin
      nHRESETh_count = nHRESETh_count + 1;
      nHRESETh_d = 0;
    end
    else nHRESETh_d = 1;
  end
end

always @ (posedge extclk)
begin
  if (HEE0_In_syn ^ HEE0_In) HEE0_count = 0; // Produce short negage pulse
  else begin
    if (HEE0_count[divhee0] != 1)
      begin
        HEE0_count = HEE0_count + 1; HEE0_Out = 1'b1; // Activate Debug Request at rising edge
      end
end
end
else if (nHRESETh_d == `ASSERTED) HEE0_Out = !HEE0_In;
else
    HEE0_Out = 0;
end
HEE0_In_syn <= HEE0_In;
end

//*** Delayed Slave Hard Reset ***/
always @(posedge extclk)
begin
    if (nHRESETs == `ASSERTED)
        begin
            nHRESETs_count = 0; nHRESETs_d = 0;
        end
    else begin
        if (nHRESETs_count[divhresets] != 1) begin
            nHRESETs_count = nHRESETs_count + 1;
            nHRESETs_d = 0;
        end
        else nHRESETs_d = 1;
    end
end

//*** Slave Debug Request logic ***/
always @(posedge clk)
begin
    if (WRITE3 && (Data[2] == 1) || (SEE0_In_syn ^ SEE0_In))
        SEE0_count = 0;
    else begin
        if (SEE0_count[divsee0] != 1)
            begin
                SEE0_count = SEE0_count + 1; SEE0_Out = 1'b1; // Activate Debug Request at rising edge
            end
        else if (nHRESETs_d == `ASSERTED) SEE0_Out = !SEE0_In;
        else
            SEE0_Out = 0;
    end
SEE0_In_syn <= SEE0_In;
if (Internal_Test) SEE0_Out = 1'b1;
end
//**** BCSR Register Main ****//
assign Data[0:7] = DataO[0:7];

always @ (CFG_ADDR or READ_CFG_FROM_BCSR or BCSR_ADDR or Read_from_BCSR)
if(READ_CFG_FROM_BCSR)
  case (CFG_ADDR)
    2'b00 : DataO[0:7] = CFG_BYTE0_DEF;
    2'b01 : DataO[0:7] = CFG_BYTE1_DEF;
    2'b10 : DataO[0:7] = CFG_BYTE2_DEF;
    2'b11 : DataO[0:7] = CFG_BYTE3_DEF;
  endcase
else if(Read_from_BCSR)
  case (BCSR_ADDR)
    0 : DataO[0:7] = BCSR0[0:7];
    1 : DataO[0:7] = BCSR1r[0:7];
    2 : DataO[0:7] = BCSR2r[0:7];
    3 : DataO[0:7] = {BCSR3[0:1],!SEE0_In,BCSR3[3:7]};
    4 : DataO[0:7] = BCSR4[0:7];
    5 : DataO[0:7] = {BCONF,BREVN,BCSR5[5:7]};
    6 : DataO[0:7] = BCSR6[0:7];
    7 : DataO[0:7] = BCSR7[0:7];
    8 : DataO[0:7] = {BCSR8[0:6],CID0};
    9 : DataO[0:7] = {BCSR9[0:6],CID1};
   10: DataO[0:7] = BCSR10[0:7];
    default : DataO[0:7] = 8'bz;
  endcase
else DataO[0:7] = 8'bz;

//**** BCSR Register Power-on-Reset setting & Write ****//
always @ (posedge nWE or negedge nHRESETh) // or negedge reset)
begin
  if(nHRESETh == `ASSERTED)
    begin

  end
BCSR0[0:7] = BCSR0_PON; // general
BCSR1[0:7] = BCSR1_PON; // board
BCSR6[0:7] = BCSR6_PON; // initialize
BCSR3[0:1] = BCSR3_PONh;
BCSR3[3:7] = BCSR3_PONl;

if (!DSItoSYS) BCSR2[0:7] = BCSR2_PON_DSI;
else BCSR2[0:7] = BCSR2_PON_SYS;

BCSR7[0:7] = BCSR7_PON; // 8122/26
BCSR8[0:7] = BCSR8_PON; // 8122/26
BCSR9[0:7] = BCSR9_PON; // 8122/26 Ethernet
BCSR10[0:7] = BCSR10_PON;

else begin

if (Write_to_BCSR)
    case (BCSR_ADDR)
    0: begin
         if (`FLUNLCK1 == `ASSERTED) `FLASHPRT1 = Data[0];
         else `FLASHPRT1 = `ASSERTED;
         if (`FLUNLCK2 == `ASSERTED) `FLASHPRT2 = Data[1];
         else `FLASHPRT2 = `ASSERTED;
         BCSR0[2:7] = Data[2:7];
    end
    1: BCSR1[0:7] = Data[0:7];
    2: begin
        BCSR2[0:2] = Data[0:2];
        BCSR2[4:7] = Data[4:7];
    end
    3: BCSR3[0:7] = Data[0:7];
    6: begin
        BCSR6[0:1] = Data[0:1];
    end
    7: BCSR7[0:7] = Data[0:7];
    8: begin
        BCSR8[0:1] = Data[0:1];
    end

end
BCSR8[2:3] = EREV[0:1];
BCSR8[4:6] = Data[4:6];
end
9: BCSR9[0:6] = Data[0:6];
10: BCSR10[0:7] = Data[0:7];

// default :
endcase
end

//**** Debounce functions ***//
function hhtime_elaps;
input i;
input [divd:0] d;
begin
  if (i == 0) hhr = 0;
  else begin
    if (hhr[d] != 1) begin
      hhr = hhr + 1; hhtime_elaps = 1'b0;
    end else hhtime_elaps = 1'b1;
  end
endfunction

function shtime_elaps;
input i;
input [divd:0] d;
begin
  if (i == 0) shr = 0;
  else begin
    if (shr[d] != 1) begin
      shr = shr + 1; shtime_elaps = 1'b0;
    end else shtime_elaps = 1'b1;
  end
end
endfunction
function nhtime_elaps;
input i;
input [divd:0] d;
begin
  if (i == 0) nhr = 0;
  else begin
    if (nhr[d] != 1) begin
      nhr = nhr + 1; nhtime_elaps = 1'b0;
    end else nhtime_elaps = 1'b1;
  end
endfunction

function hstime_elaps;
input i;
input [divd:0] d;
begin
  if (i == 0) hsr = 0;
  else begin
    if (hsr[d] != 1) begin
      hsr = hsr + 1; hstime_elaps = 1'b0;
    end else hstime_elaps = 1'b1;
  end
endfunction

function sstime_elaps;
input i;
input [divd:0] d;
begin
  if (i == 0) ssr = 0;
  else begin
    if (ssr[d] != 1) begin
      ssr = ssr + 1; sstime_elaps = 1'b0;
    end
  end
endfunction
end else sstime_elaps = 1'b1;
end
endfunction

function nstime_elaps;
input i;
input [divd:0] d;
begin
if (i == 0) nsr = 0;
else begin
if (nsr[d] != 1) begin
nsr = nsr + 1; nstime_elaps = 1'b0;
end else nstime_elaps = 1'b1;
end
end
endfunction

always @ (posedge extclk)
begin
if (reset == `ASSERTED)
begin
counter = (1 << div1st) - 2;
end
counter = counter + 1;
if (counter[div1st] != 1) clock_divider = 0;
else clock_divider = 1;
end

always @ (posedge clock_divider)
begin
if (HReseth_In == `PRESSED) HRESETh = hhtime_elaps(0,div2st);
else HRESETh = hhtime_elaps(1,div2st);
if (SReseth_In == `PRESSED) SRESETh = shtime_elaps(0,div2st);
SRESETh = shite_elaps(1,div2st);

if (Aborth_In == 'PRESSED) NMIh = nhtime_elaps(0,div2st);
else
  NMIh = nhtime_elaps(1,div2st);

if (HResets_In == 'PRESSED) HRESETs = hstime_elaps(0,div2st);
else
  HRESETs = hstime_elaps(1,div2st);

if (SResets_In == 'PRESSED) SRESETs = sstime_elaps(0,div2st);
else
  SRESETs = sstime_elaps(1,div2st);

if (Aborts_In == 'PRESSED) NMIs = nstime_elaps(0,div2st);
else
  NMIs = nstime_elaps(1,div2st);

if (Internal_Test) NMIs = 1'b1;
end

/* NEW FUNCTIONS FOR 8122ADS */
assign CID1 = (CoreConfig1 || !CoreConfig2) & B_8126;

// Assign Core Voltage Selectors
assign CoreConfig1 = B_8122_1 && !B_8126 && !B_8122_2 && !B_8102;
assign CoreConfig2 = CoreConfig1 || (!B_8122_1 && !B_8126 && !B_8122_2 && B_8102);
assign BCSR_CORE1 = CoreConfig1;
assign BCSR_CORE2 = CoreConfig2;
assign SUP1_IND = !CoreConfig1;
assign SUP2_IND = CoreConfig1 || CoreConfig2;
assign SUP3_IND = CoreConfig1 || !CoreConfig2;
assign CID0 = (CoreConfig1 || !CoreConfig2) & (B_8122_1 || B_8122_2);

assign IRQ2hb = ((!SYS_SLOT) || IRQ1cPCIb) ? 1'b0 : 1'bz;
assign IRQ1cPCIb = PER_SLOT ? IRQ1hb : 1'bz;
assign RSTCNPhb = PER_SLOT; // Place Host in Default Slave Config for PER_SLOT
assign XHRESETsb = 1'bz; // Spare Slave Reset to Connector
assign SYSBUFXXENb = `cPCI_OFF; // Close buffer in cPCI OFF mode
assign ABUFXXENb = PER_SLOT; // Close host buffer in Peripheral Slot

// JTAG Chain
assign ChainSel0 = (SYSENb || `SWOPT[2]) && ChainSell;
// Ethernet Functions

assign Activate_ETH = (`ETH_ON && !ETH_SW_ON && !B_8102); // 3 operations to Validate ETH_ON

assign RMII2 = `RMII ? 1'b1 : 1'b0; // Set RMII/MII for RPHY

//assign RMII2 = (SWOPT[1] == `ASSERTED) ? 1'b1 : 1'b0; // Set RMII/MII for RPHY

// Reset Mezzanine Ethernet switch
assign ESW_RESETb = (`ETH_RST == `ACTIVE_LOW) || (nHRESETh == `ASSERTED)? `ASSERTED : `NEGATED;

// Reset Mezzanine MII-PHY
assign ESW1RSTb = (`ETH1_RST == `ACTIVE_LOW) || (nHRESETh == `ASSERTED)? `ASSERTED : `NEGATED;

// Reset SMII-PHY
assign SMIIHYRSTb = (`SPHY_RST == `ACTIVE_LOW) || (nHRESETh == `ASSERTED)? `ASSERTED : `NEGATED;

always //@(``ETH_MODE or Activate_ETH)
if (Activate_ETH)
begin

  case (`ETH_MODE)
    4'b0000: begin // 8102 Compatible
      BCSR_PPC_Sb = 1'b1; // DSI ETH closed
      BCSR_TDM_Sb = 1'b1; // 8103 side closed
      BCSR_SMIIb = 1'b1; // Not SMII Mode
      BCSR_MSEL = 4'b0011; // 25MHz to Host PHY
      BCSR_MB = 4'b0010; // 25MHz to Host PHY
      BCSR_RSMIIb = 1'b1; // Not R/SMII PHY
    end
    4'b0001: begin // Mac2Mac RMII DSI to ETH-SW
      BCSR_PPC_Sb = 1'b0; // DSI open
  endcase
end
BCSR_TDM_Sb = 1'b1;       //8103 side closed
BCSR_SMIIb = 1'b0;       //Like SMII Mode
BCSR_MSEL = 4'b0111;     //25MHz to Host PHY, 50MHz 8122 & ETH-SW

BCSR_MB = 4'b1010;       //50MHz 8122 & ETH-SW
BCSR_MB = 4'b0010;       //50MHz from ETH-SW to 8122
BCSR_RSMIIb= 1'b1;       //Not R/SMII PHY
end

4'b0010: begin           // Mac2Mac RMII TDM to ETH-SW
  BCSR_PPC_Sb = 1'b1;       //DSI closed
  BCSR_TDM_Sb = 1'b1;       //8103 side closed
  BCSR_SMIIb = 1'b0;       //SMII Mode
  BCSR_MSEL = 4'b0111;     //25MHz to Host PHY, 50MHz 8122 & ETH-SW

  BCSR_MB = 4'b1010;       //50MHz 8122 & ETH-SW
  BCSR_MB = 4'b0010;       //50MHz from ETH-SW to 8122
  BCSR_RSMIIb= 1'b1;       //Not R/SMII PHY
end

4'b0011: begin           // Mac2Mac SMII TDM to ETH-SW
  BCSR_PPC_Sb= 1'b1;       //DSI closed
  BCSR_TDM_Sb = 1'b1;       //8103 side closed
  BCSR_SMIIb = 1'b0;       //SMII Mode
  BCSR_MSEL = 4'b0111;     //25MHz to Host PHY

  BCSR_MB = 4'b0010;       //125MHz to 8122 from ETH-SW
  BCSR_RSMIIb= 1'b1;       //Not R/SMII PHY
end

4'b0100: begin           // Mac2Mac RMII TDM to ETH-SW LOOPBACK
  BCSR_PPC_Sb = 1'b1;       //DSI closed
  BCSR_TDM_Sb = 1'b1;       //8103 side closed
  BCSR_SMIIb = 1'b0;       //Not SMII Mode
  BCSR_MSEL = 4'b0111;     //25MHz to Host PHY, 50MHz 8122 & ETH-SW

  BCSR_MB = 4'b1010;       //50MHz 8122 & ETH-SW
  BCSR_MB = 4'b0010;       //50MHz from ETH-SW to 8122
  BCSR_RSMIIb= 1'b1;       //Not R/SMII PHY
end

4'b0101: begin           // Mac2Mac RMII DSI to ETH-SW LOOPBACK
  BCSR_PPC_Sb = 1'b0;       //DSI open
  

MSC8102/22/26ADS CPLD Code

MSC8122/26ADS Reference Manual, Rev. B0.1
BCSR_TDM_Sb = 1'b1;  // 8103 side closed
BCSR_SMIIb = 1'b0;  // Like SMII Mode
BCSR_MSEL = 4'b0111; // 25MHz to Host PHY, 50MHz 8122 & ETH-SW

BCSR_MB = 4'b1010; // 50MHz 8122 & ETH-SW
BCSR_MB = 4'b0010; // 50MHz from ETH-SW to 8122
BCSR_RSMIIb= 1'b1;  // Not R/SMII PHY

end

4'b0110: begin  // Mac2Mac SMII TDM to ETH-SW
BCSR_PPC_Sb= 1'b1;  // DSI closed
BCSR_TDM_Sb = 1'b1;  // 8103 side closed
BCSR_SMIIb = 1'b0;  // SMII Mode
BCSR_MSEL = 4'b0011; // 25MHz to Host PHY
BCSR_MB = 4'b0010; // 125MHz to 8122 from ETH-SW
BCSR_RSMIIb= 1'b1;  // Not R/SMII PHY

end

4'b0111: begin  // Mac2Phy SMII TDM to PHY
BCSR_PPC_Sb= 1'b1;  // DSI closed
BCSR_TDM_Sb = 1'b1;  // 8103 side closed
BCSR_SMIIb = 1'b0;  // SMII Mode
BCSR_MSEL = 4'b1011; // 25MHz to Host PHY, 125MHz From Osc.
BCSR_MB = 4'b1110; // 125MHz to 8122 and PHY
BCSR_RSMIIb= 1'b0;  // SMII PHY

end

4'b1000: begin  // Mac2Phy RMII TDM to PHY
BCSR_PPC_Sb= 1'b1;  // DSI closed
BCSR_TDM_Sb = 1'b1;  // 8103 side closed
BCSR_SMIIb = 1'b1;  // Not SMII Mode
BCSR_MSEL = 4'b0111; // 25MHz to Host PHY, 50MHz From Osc.
BCSR_MB = 4'b1010; // 50MHz to 8122 and PHY
BCSR_RSMIIb= 1'b0;  // R/SMII PHY

end

4'b1001: begin  // Mac2Phy RMII DSI to PHY
BCSR_PPC_Sb= 1'b0;  // DSI open
BCSR_TDM_Sb = 1'b1;  // 8103 side closed
BCSR_SMIIb = 1'b0;  // Like SMII Mode

end
BCSR_MSEL  = 4'b0111;  // 25MHz to Host PHY, 50MHz From Osc.
BCSR_MB    = 4'b1010;   // 50MHz to 8122 and RPHY
BCSR_RSMIIb= 1'b0;     // R/SMII PHY
end

4'b1010: begin   // Mac2Phy MII TDM to RPHY(MII)
BCSR_PPC_Sb= 1'b1;  // DSI closed
BCSR_TDM_Sb = 1'b1;  // 8103 side closed
BCSR_SMIIb = 1'b1;   // Not SMII Mode
BCSR_MSEL  = 4'b0001;  // 25MHz to Host PHY, 25MHz From Osc.
BCSR_MB    = 4'b0011;   // 25MHz from PHY
BCSR_RSMIIb= 1'b0;     // R/SMII PHY
end

4'b1011: begin   // Mac2Phy MII DSI to RPHY(MII)
BCSR_PPC_Sb= 1'b0;  // DSI open
BCSR_TDM_Sb = 1'b1;  // 8103 side closed
BCSR_SMIIb = 1'b1;   // Not SMII Mode
BCSR_MSEL  = 4'b0001;  // 25MHz to Host PHY, 25MHz From Osc.
BCSR_MB    = 4'b0011;   // 25MHz from PHY
BCSR_RSMIIb= 1'b0;     // R/SMII PHY
end

4'b1100: begin   // Mac2Mac MII TDM 8122 to 8103
BCSR_PPC_Sb= 1'b1;  // DSI closed
BCSR_TDM_Sb = 1'b0;  // 8103 side open
BCSR_SMIIb = 1'b1;   // Not SMII Mode
BCSR_MSEL  = 4'b0011;  // 25MHz to Host PHY, 25MHz From Osc.
BCSR_MB    = 4'b1111;   // 25MHz from Osc
BCSR_RSMIIb= 1'b1;     // Not R/SMII PHY
end

4'b1101: begin   // Mac2Mac MII DSI to RPHY(MII)
BCSR_PPC_Sb= 1'b0;  // DSI open
BCSR_TDM_Sb = 1'b0;  // 8103 side open
BCSR_SMIIb = 1'b1;   // Not SMII Mode
BCSR_MSEL  = 4'b0011;  // 25MHz to Host PHY, 25MHz From Osc.
BCSR_MB    = 4'b1111;   // 25MHz from Osc
BCSR_RSMIIb= 1'b1;     // Not R/SMII PHY
end

default: begin

// 8102 Compatible

BCSR_PPC_Sb = 1'b1; //DSI ETH closed
BCSR_TDM_Sb = 1'b1; //8103 side closed
BCSR_SMIIb = 1'b1; //Not SMII Mode
BCSR_MSEL = 4'b0001; //25MHz to Host PHY
BCSR_MB = 4'b0010; //25MHz to Host PHY
BCSR_RSMIIb = 1'b1; //Not R/SMII

endcase
end

// else if (SWOPT[1] == `ASSERTED) // Set 50MHz clock to RPHY
begin

// Mac2Phy RMII TDM/DSI to RPHY

BCSR_MSEL = 4'b0111; //25MHz to Host PHY, 50MHz From Osc.
BCSR_MB = 4'b1010; //50MHz to 8122 and RPHY

end

else

begin // 8102 Compatible

BCSR_PPC_Sb = 1'b1; //DSI ETH closed
BCSR_TDM_Sb = 1'b1; //8103 side closed
BCSR_SMIIb = 1'b1; //Not SMII Mode
BCSR_MSEL = 4'b0001; //25MHz to Host PHY
BCSR_MB = 4'b0010; //25MHz to Host PHY
BCSR_RSMIIb = 1'b1; //Not R/SMII

end

// TDM External Clock master when not Ethernet mode
assign EXT_MSTb = `EXTMST || !ETH_SW_ON; //Activate_ETH;
assign ESW1EN = Activate_ETH && `ETH1_EN; // Enable Mezzanine MII-PHY
assign FETH1EN = `FETH1_EN; // Enable Host MII PHY
assign FETH2EN = Activate_ETH && `FETH2_EN; // Enable Slave R/MII PHY
assign ESW_RDb = state[1] || BCTL0sb;
assign ESW_WRb = state[0] || SPR1 || !BCTL0sb;

// Ethernet Mezzanine RD/WR state machine
always @(posedge clk)
    if (nHRESETh == `ASSERTED)
        state = `S0;
    else
        case (state)
            `S0: if (!ESW_CSb)
                state = `S1;
            else
                state = `S0;
            `S1: if (BCTL0sb)
                state = `S3;
            else if (!BCTL0sb)
                state = `S2;
            `S2: state = `S7;
            // wait total 6 cycles (133MHZ Bus will give 40ns RD cycle)
            `S7: state = `S8;
            `S8: state = `S9;
            `S9: state = `S10;
            `S10: state = `S5;
            // wait total 6 cycles (133MHZ Bus will give 40ns WR cycle)
            `S3: state = `S11;
            `S11: state = `S12;
            `S12: state = `S13;
            `S13: state = `S14;
            `S14: state = `S4;
            `S4: if (!ESW_CSb)
                state = `S6;
            else
                state = `S0;
            `S5: if (!ESW_CSb)
state = `S6;
else
state = `S0;

`S6: if (!ESW_CSb)
state = `S6;
else
state = `S0;
default:
state = `S0;
endcase

// Internal Test
assign Internal_Test = 0; // !SPR1;

// I2C
assign I2C_CONT = `I2C_CON ;
assign I2C_CONT1 = `I2C_CON && (!`FETH1_EN);

// DSI Functions
always // @(B_8102 or `DSI_EX)
if (B_8102 || `DSI_EX2)
begin
   HCID[0] = XAt7 ;
   HCID[1] = XAt8 ;
   HCID[2] = XAt9 ;
   HCID[3] = XAt10 ;
   TTs0  = 1'bz;
end
else
  case ( `DSI_EX )
  2'b00: begin
    HCID[0] = XAt7 ;
    HCID[1] = XAt8 ;
    HCID[2] = XAt9 ;
    HCID[3] = 1'b0 ;
TTs0  = 1'bz;
   end

2'b01: begin
   HCID[0] = XAt7 ;
   HCID[1] = XAt8 ;
   HCID[2] = 1'b0 ;
   HCID[3] = 1'b0 ;
   TTs0  = 1'bz;
   end

2'b10: begin
   HCID[0] = XAt7 ;
   HCID[1] = 1'b0 ;
   HCID[2] = 1'b0 ;
   HCID[3] = XAt8;
   TTs0  = 1'bz;
   end

2'b11: begin
   HCID[0] = Ah5 ;
   HCID[1] = Ah6 ;
   HCID[2] = 1'b0 ;
   HCID[3] = XAt8;
   TTs0  = XAt7;
   end
endcase

// LED Indications
assign CORELEDh = (`LEDEN == `ACTIVE_LOW)? LEDDRVh : `NEGATED;
assign CORELEDs = (`LEDEN == `ACTIVE_LOW)? LEDDRVs : `NEGATED;
assign RUNhLED = (`LEDEN == `ACTIVE_LOW)? DBBhb : `NEGATED;
assign RUNsLED = (`LEDEN == `ACTIVE_LOW)? DBBsb : `NEGATED;
assign LED3V3 = (`LEDEN == `ACTIVE_LOW)? `ASSERTED : `NEGATED;
assign SMII_LEDb = (`LEDEN == `ACTIVE_LOW)? BCSR_SMIIb : `NEGATED;
assign ETH_IND = !Activate_ETH ; // Ethernet-ON LED
dffFSYNC_8_FF(.d(!FSYNC_8), .q(FSYNC_8), .clk(FSYNC), .clrn(reset));
assign FSYNC2 = `CODEC_16K ? FSYNC_8 : FSYNC ;
A.2 MSC8122/26ADS CPLD Code

BCSR (Altera) code version 1.2 for the MSC8122/26ADS is in the disk attached. It replaces the traditional version for the MSC8102/22/26ADS.

```vhdl
// hds header_start
//BCSR2_main Rev 012 for 8122ADS board
// hds header_end

`resetall
`timestcale 1ns/10ps

module BCSR_8122_12 (  
clk, extclk, Data, reset, XAt7, XAt8, XAt9, XAt10, A27, A28, A29,  
nWE, nR_W, nCS0, nBCSR_CS, nFCSh, nHCS1, nHBCS,  
DS1toSYS, nSYS64, nFCFG, MODCK1s, MODCK2s, SWOPT,  
nRS232EN_1, nRS232EN_2, nFETH_RST, nFRM_RST, FRMtoTSI, nCODEC_EN,  
nBOOTPh, nBOOTPs, nDBUFxEN, nDBUFxEN, GA, nPRSTs,  
HEE0_In, HEE0_Out, HEE1_In, HEE1_Out,  
SEE0_In, SEE0_Out, SEE1_In, SEE1_Out, nMODCK1s, nMODCK2s, nMODCK3s, nMODCK4s, nMODCK5s, nMODCK6s, nMODCK7s, nMODCK8s,  
nHRSTPh, nHRSTPs, nSRESETh, nSRESETs, nHRST, nHRST_In, nNMPh, nNMIh,  
BM_Out, CNFGS_Out, DSI64_Out, DSI6SYNC_Out, SWTE_Out, MODCK1s_Out, MODCK2s_Out,  
RSTCFG_Out, TEST_EN, TEST_SIG, BCTL1_In, nCS0s, BCTL1_Outb  
);  
B_8102, B_8122_1, B_8122_2, B_8126, BCSR_CORE1, BCSR_CORE2, SUP1_IND,  
SUP2_IND, SUP3_IND, CS6hb, SYSEXb, PCI_PRSNTb, DACK4b, DACK3b, DACK2b, DACK1b,  
XDBUFxENb, EXT_PCI1, EXT_PCI, ESW_CSb, BCTL1_Xoutb, A26, AHRESLETb, AHRESLETb,  
IRQ1hb, IRQ2hb, IRQ1cPCIb, RSTCNFhb, SYSEXb, MI2, ETH_IND, Ah5, Ah6, TTs0,  
HCID, BCONF, BREV, BREV, ESW_RESETh, ChainSel0, ChainSel1, ChainSel3,  
ChainSel4, ETH_SW_ON, EREV, LEDDRVh, LEDDRVl, DBBhb, DBBsb, CORELEDh, CORELEDs,  
RUNhLED, RUNsLED, LED3V3, ESW1RSTb, ESW1EN, nFETH1RST, SMIIHYRSTb, BCSR_PPC_Sb,  
BCSR_TDM_Sb, BCSR_SMIib, BCSR_MSEL, BCSR_MB, BCSR_RSMIIb,  
EXT_MSTb, FETH1EN, FETH2EN, ESW_RDb, ESW_Wrb, BCTL0sb, SPR1, SMI2_LEDb, I2C_CONT,  
);  
```

MSC8122/26ADS Reference Manual, Rev. B0.1
I2C_CONT1, FSYNC, FSYNC2, GPL1s

); //*** General Definition ***//
`define ASSERTED 0
`define NEGATED 1
`define ACTIVE_LOW 0
`define ACTIVE_HIGH 1
`define PRESSED 0
`define RELEASED 1
`define WRITE 1
`define READ 0
`define From_BCSR 1
`define From_FLASH 0
`define S0 7'b0000011
`define S1 7'b0000111
`define S2 7'b0000001
`define S3 7'b0000010
`define S4 7'b0001010
`define S5 7'b0001001
`define S6 7'b0001111
`define S7 7'b0010001
`define S8 7'b0100001
`define S9 7'b0110001
`define S10 7'b1000001
`define S11 7'b0010010
`define S12 7'b0100010
`define S13 7'b0110010
`define S14 7'b1000010

//*** State Registers Definition ***//
reg [6:0] state;
parameter BCSR_ver = 3'b010; // Current version of code is 2 (1.2)

parameter divpr = 12;
parameter divsee0 = 5;
parameter divhee0 = 5;
parameter divhresets = 10;
parameter divhreseth = 10;
parameter div2st = 5; // 50usec@20Mhz
parameter div1st = 13; // 50msec@20Mhz
parameter zero = 1'b0;
parameter divd = div2st;
parameter divc = div1st;

// *** Inputs ***
input clk, // From Host clock buffer
    extclk, // Clock from external clock oscillator
     reset; // Main Power-on-Reset
input Ah5,Ah6,A26,A27,A28,A29, // 60-x bus addresses lines
    nWE,nR_W,BCTL0sb, // and controls
 nCS0,nBCSR_CS, // Chip selects
    DSItoSYS, // DIP-switch selectes Slave boot source
 nSYS64, // DIP-switch selectes Slave data bus distribution
 nFCFG, // DIP-switch selectes Host boot source (BCSR or Flash)
 MODCK1s,MODCK2s, // DIP-switch select MODCK1 for the Slave
    MODCK1h,MODCK2h,MODCK3h,
 MODCK4h,MODCK5h,MODCK6h; // DIP-switch select MODCK1 for the Host
input nHCS1,nHBCS; // Chip Selects for Slaves
input XAt7,XAt8,XAt9,XAt10; // Addresses together with nHCS1,2 & nHBCS for select Slaves on/ off- board
input [0:3] GA; // Geographic Address of System Board setting by on-CPCI backplane jumpers,
                  // default value equals 0xf
input Aborth_In, Aborts_In, // Pushbuttons of NMI for Host & Slave
   HReseth_In, HResets_In, // Pushbuttons of HReset for Host & Slave
 SReseth_In, SResets_In; // Pushbuttons of Sreset for Host & Slave
HEE0_In,HEE1_In; // Inputs from DIP-switch for Host EE0,1 control
input SEE0_In,SEE1_In; // Inputs from DIP-switch for Slave EE0,1 control
input EEl_misc_In; // Debug acknowledge comming from another ADS
input [1:2] SWOPT;    // Software options applied from DIP-switch
input [1:3] nHRST_In;  // Three HRESETs from external boards
input BCTL1_In, nCS0s; // Added for 8102 BCTL1 bug workaround
inputB_8102,
B_8122_1,
B_8122_2,
B_8126; // Set Chip
inputCS6hb,    // Rack Host CS6
SYSENb,  // System slot indication
PCI_FRSNTb,   // PCI Rack present indication
DACK4b,DACK3b,
DACK2b,DACK1b; // DMA Acknowledge from Host
inputESW_CSb; // Mezzanine Ethernet Switch Chip Select
inputIRQ1hb; // INT_OUT from Slave
input [0:1] BCONF; // Board Configuration from Pull-Ups
input [0:2] BREVN; // Board Configuration from Pull-Ups
inputChainSel1;  // JTAG Chain Select
inputSPR1; // Spare DIP-SW for internal test only
inputFSYNC; // 8KHz/16KHz Sync from Codec
/*** Ethernet Pins ***/
inputETH_SW_ON; // Ethernet DIP-SW On
input [0:1] EREV; // Ethernet Mezzanine Board ID

/*** LED Indication strobes ***/
inputLEDDRVh,LEDDRVs,DBBhb, // LED Indication strobes
DBBsb;
inputGPL1s;

/*** Bidirectional ***/
inout [0:7] Data; // Eight bit of buffered 60-x bus
inout nHRESETh,nHRESETs, // HRESET for Host & Slave
nSRESETh,nSRESETs; // SRESET for Host & Slave
BHRESEThb;         // HRESET from Host to Slave in cPCI Rack
inout nPRSTs;      // Wire connected to FORESET of Slave
inout HEE1_Out;    // EE1 for Host
inoutIRQ1cPCIb;   // Interrupt from Slave on Peripheral-Slot to cPCI

//*** Outputs ***/
output nNM1h,nNMI0s; // IRQ0 for Host & Slave
output nFCSh,        // Host Side Flash chip enable
nRS232EN_1,nRS232EN_2, // Enable for RS232 transmitters
nFETH_RST, // Reset for Ethernet Phy
nFRM_RST,      // E1/T1 Farmer Reset
FRMtoTSI,       // Switch FALC56 channels to TSI or to TDM3 of 8102
nCODEC_EN,     // Chip select to CODEC device
nBOOTPh,nBOOTPs; // Boot Sector protection for Flashes
output HEE0_Out;   // EE0 for Host
output SEE0_Out;   // EE0 for Slave
output nDBUFXEN,   // Enable for on-board peripherals buffer
nDBUFBEN,        // Enable for off-board peripherals buffer to access to another boards
ABUFXEnb;        // Disable Addr Buffers of 8103 when Slave Board
output [3:1] MODCKh_Out; // MODCK1-3 comming from DIP-switch for setting Host(8101) PLL mode
output [1:3] nHRST;    // Three HRESETs to external boards
output BCSR_CORE1, // Slave Core Voltage selector
        BCSR_CORE2;
output SUP1_IND,
        SUP2_IND,
        SUP3_IND; // Slave Core Supply Indication
output XDBUFXEnb, // Buffer Enable to cPCI J1/J2
        EXT_PCI1,EXT_PCI;  // Direction of cPCI-J1/J2 Buffers
output BCTL1_XOutb;  // Mezzanine Ethernet Switch Buffer Enable
output IRQ2hb; // Interrupt from Peripheral slot to Host on SYS-Slot
output RSTCNFhb; // RSTCNF for Host (Set on Peripheral Slot)
output XHRESETsb; // XHRESETsb is spare reset to cPCI
output SYSBUFxFFEnb; // Enable CS6h towards Rack Slave
output RMII2;       // RMII/MII Mode for RPHY
: ETH_IND;  // Ethernet Indication

output [0:3] HCID;  // Host ID
output TTs0;  // HA7 / TT0 Slave
output ChainSel0,  // JTAG Chain Select
ChainSel1,
ChainSel3,
ChainSel4;

output I2C_CONT,  // Connects Host&Slave I2C Bus
    I2C_CONT1;  // Connects Host&Slave I2C Bus (or Host MII)
output FSYNC2;  // 8KHz Sync to TSI From Codec

/*** Ethernet Pins /**/

    output ESW_RESETb,  // Reset Mezzanine Switch
    nFETH1RST,  // Reset RMII/MII PHY
    SMIIPHYRSTb,  // Reset SMII PHY
    ESW1RSTb,  // Reset Mezzanine MII-PHY
    FETH1EN,  // Enable Host MII PHY
    FETH2EN,  // Enable Slave R/MII PHY
    ESW1EN;  // Enable Mezzanine MII-PHY

output BCSR_PPC_Sb,  // ETH-SW1 Mux Enable
    BCSR_TDM_Sb,  // ETH-SW2 Mux Enable
    BCSR_SMIIb,  // ETH-SW5 Mux Enable
    BCSR_RSMIIb;  // ETH-SW4 Mux Enable
output [0:3] BCSR_MSEL,  // ETH-MUX1 Clock Enables
    BCSR_MB;  // ETH-MUX1 Clock selectors
output EXT_MSTb;  // TDM Clock Master Enable
output ESW_RDb,
    ESW_WRb;  // Ethernet SW Controls

/*** LED Indication strobes /**/
output CORELEDh, CORELEDs,
RUNhLED, RUNsLED, LED3V3, SMII_LEDb;

/*** LEDs drive pins /**/
(output nLED_EN;  // Allow to switch off all LEDs on the ADS accordingly to TTM req.
output [0:1] SIGH_LED;  // Misc two LEDs for SW indication for Host)
SIGS_LED; // Misc two LEDs for SW indication for Slave
output SEE1_LED; // Slave in-Debug Mode indication

///*** Slave configuration pins ***///
output [0:2] BM_Out; // Boot Mode selection three bits for the
output n32to64En; // Output for MUX control
output CNFGS_Out; // Configuration Source
output DSI64_Out; // DSI 64bit setting
DSISYNC_Out; // DSI Synchronous Mode
SWTE_Out, // Software Watchdog Timer Enable
RSTCFG_Out, // Reset Configuration Mode
MODCK1s_Out, // Two lines of MODCK
MODCK2s_Out;
#define HD_PINS
{CNFGS_Out,DSI64_Out,DSISYNC_Out,SWTE_Out,RSTCFG_Out,MODCK1s_Out,MODCK2s_Out}
  // All the pins defined Slave FORESET configuration

///*** Test mode ***///
output TEST_EN; // Enter 8102 to test mode
output [1:3] TEST_SIG; // Select test mode
output EE1_misc_Out;
output BCTL1_Outb; // Enable BDs[0:15] Buffer

///*** Registers Definition ***///
reg [0:7] Data0;

//**** BCSR0 Description ****///
reg [0:7] BCSR0;
#define FLASHPRT1 BCSR0[0] // Flash 1 H/W Protection
#define FLASHPRT2 BCSR0[1] // Flash 2 H/W Protection
#define FRM_RST BCSR0[2] // Framer E1/T1 Reset control
#define CODEC_EN BCSR0[3] // CODEC Chip Select
#define SIGNALS0 BCSR0[4] // Signal-0 LED Slave
#define SIGNALS1 BCSR0[5] // Signal-1 LED Slave
#define SIGNALH0 BCSR0[6] // Signal-0 LED Host
#define SIGNALH1 BCSR0[7] // Signal-1 LED Host

//**** BCSR1 Description ****///
```c
// **** BCSR1 Description ****

reg [0:7] BCSR1;

`define RECONF     BCSR1[0] // Start Slave Re-configuration
`define HRST1      BCSR1[1] // Hard Reset to External Slave1
`define HRST2      BCSR1[2] // Hard Reset to External Slave2
`define ATM_RST    BCSR1[3] // SPARE N/A
`define HRST3      BCSR1[4] // Hard Reset to External Slave3
`define FETH_RST   BCSR1[5] // Fast Ethernet phy Reset
`define RS232EN_1  BCSR1[6] // RS232 Transceiver Host Side Enable
`define RS232EN_2  BCSR1[7] // UART Transceiver Slave Side Enable

// **** BCSR2 Description ****

reg [0:7] BCSR2;

`define RSTCNF     BCSR2[0] // Reset Configuration Mode
`define CNFG       BCSR2[1] // Configuration Source
`define SWTE      BCSR2[2] // Software Watchdog Timer Enable
`define DSI64     BCSR2[3] // System/DSI 64/32 bit
`define DSISYNC   BCSR2[4] // DSI Synchronous Mode
`define HRST      BCSR2[5] // Hard Reset to Slave
`define SRST      BCSR2[6] // Soft Reset to Slave
`define FRMtoTSI  BCSR2[7] // FALC56 connects to TSI or to TDM3 of 8102

// **** BCSR3 Description ****

reg [0:7] BCSR3;

`define CLKMD     BCSR3[0:1] // Clock Mode Setting Bits 1-2 for Slave
`define SEE0      BCSR3[2]  // Slave Emulation Enable 0
`define RSV34     BCSR3[3:4] // Reserved two bits

// **** BCSR6 Description ****

reg [0:7] BCSR6;

`define FLUNLCK1  BCSR6[0] // Flash 1 Protection Unlock
`define FLUNLCK2  BCSR6[1] // Flash 2 Protection Unlock
`define LEDEN     BCSR6[2] // LED Enable| Hidden
`define TEST      BCSR6[3] // Test Mode Available|
`define TESTSIG   BCSR6[4:6] // Test Mode Select  | bits
`define MARK     BCSR6[7] // Mark Bit |

// **** BCSR7 Description ****

reg [0:7] BCSR7;
```
`define I2C_CON BCSR7[0]  // I2C Connect (Slave+Host)
`define DSI_EX BCSR7[1:2]  // DSI Extended Mode
`define RPHY_RST BCSR7[3]  // R/MII PHY Reset
`define VB_INT BCSR7[4]    // VB-Interface Connect
`define ETH_ON BCSR7[5]    // Ethernet ON
`define SPHY_RST BCSR7[6]  // SMII PHY Reset
`define ETH_RST BCSR7[7]   // Reset Eth Switch

//**** BCSR8 Description ****//
reg [0:7] BCSR8;
`define cPCI_OFF BCSR8[0]  // cPCI J1/J2 Buffers Close
`define RMII BCSR8[1]     // RMII/MII Mode
`define EREV_BCSR BCSR8[2:3]  // Ethernet Board Revision/ID
`define ETH1_RST BCSR8[4] // Reset Mezzanine MII-PHY
`define ETH1_EN BCSR8[5]  // Enable Mezzanine MII-PHY
`define EXTMST BCSR8[6]   // TDM Clock Master
`define CHIPID0 BCSR8[7]  // Chip ID 8102/22/26

//**** BCSR9 Description ****//
reg [0:7] BCSR9;
`define ETH_MODE BCSR9[0:3] // Ethernet Mode
`define FETH1_EN BCSR9[4]  // Enable Host MII PHY
`define FETH2_EN BCSR9[5]  // Enable Slave R/MII PHY
`define CODEC_16K BCSR9[6] // Codec is in 16KHz Sync Mode
`define CHIPID1 BCSR9[7]   // Chip ID 8102/22/26

//**** BCSR10 Description ****//
reg [0:7] BCSR10;
`define DSI_EX2 BCSR10[0] // DSI Extended Mode
`define BCSR10_SPR BCSR10[1:7] // Spare

reg HEE0_Out;
reg SEE0_Out;
reg EE1misc_Out;
reg BCSRCSsyn;
reg HEE0_In_syn;
reg SEE0_In_syn;
nHRESETh_d;
reg nHRESETs_d;
reg [0:3] GAr;       // GA latch

reg [divsee0:0] SEE0_count;
reg [divhee0:0] HEE0_count;
reg [divhreseth:0] nHRESETh_count;
reg [divhresets:0] nHRESETs_count;
reg NMIh,NMIs,HRESETh,SRESETh,HRESETs,SRESETs;
reg clock_divider;
reg [divc:0] counter;
reg [divd:0] hhr,shr,nhr,hsr,ssr,nsr;
reg [0:3] HCID;
reg TTs0;
reg FSYNC_8;      // 8MHz for TSI

reg BCSR_PPC_Sb;// ETH-SW1 Mux Enable
   BCSR_TDM_Sb;// ETH-SW2 Mux Enable
   BCSR_SMIIb;// ETH-SW5 Mux Enable
   BCSR_RSMIIb; // ETH-SW4 Mux Enable
reg [0:3] BCSR_MSEL; // ETH-MUX1 Clock Enables
   BCSR_MB;// ETH-MUX1 Clock selectors

wire [1:0] CFG_ADDR = {A27,A28};
wire [3:0] BCSR_ADDR = {A26,A27,A28,A29};
wire READ_CFG_FROM_BCSR;
wire [0:7] CFG_BYTE3_DEF;
wire [0:7] BCSR4;
wire [5:7] BCSR5;       // Bits 0-4 are comming via ext. buffer
wire [0:3] SSEL;       // Address vector for Slaves Select
wire [0:3] ON_BRD_SLAVE_ADDR;
wire ON_BRD_HOST_PRPH; // Active when select any of Host peripheral
wire OFF_BRD_SLAVE;    // Active when access to off-board Slave
wire Write_to_BCSR;
wire Read_from_BCSR;
wire [0:7] BCSR1r, BCSR2r;
wire [0:7] BCSR2_PON_DSI, BCSR2_PON_SYS; // BCSR3_PON;
wire [0:1] BCSR3_PONh;
wire [0:4] BCSR3_PONl;
wire [0:2] BTMD;
wire CNFG, DSI64, DSISYNC, SWTE, RSTCNF;
wire [0:1] CLKMD;
wire ADDR3, ADDR5;
wire WRITE3;
wire [1:10] CFGREG;
wire CONFIG_EN;
wire CoreConfig1;
wire CoreConfig2;
wire SYS_SLOT;
wire PER_SLOT;
wire VB_ACCESS;
wire Activate_ETH;
wire RST_FROM_cPCI;
wire Internal_Test;
wire CID1, CID0;

// *** Hard Reset Configuration Word (HRCW) for 8101 *** //

parameter EARB_DEF = 1'b0, // bit 0
    EXMC_DEF = 1'b0,
    nIRQ7INT_DEF = 1'b1,
    EBM_DEF = 1'b0,
    BPS_DEF = 2'b10,
    SCDIS_DEF = 1'b0,
    ISPS_DEF = 1'b0, // bit 7
    CFG_BYTE0_DEF = // 0x28

{EARB_DEF, EXMC_DEF, nIRQ7INT_DEF, EBM_DEF, BPS_DEF, SCDIS_DEF, ISPS_DEF},
    IRPC_DEF = 2'b00, // bit 8
    DPPC_DEF = 2'b00,
    NMIOUT_DEF = 1'b0,
ISB_DEF = 3'b000, // bit 15
CFG_BYTE1_DEF = // 0x0

{IRPC_DEF, DPPC_DEF, NMIOUT_DEF, ISB_DEF},
RSV16_DEF = 1'b0, // bit 16
BBD_DEF = 1'b0,
MMR_DEF = 2'b11,
RSV20_21_DEF = 2'b00,
TCPC_DEF = 2'b10, // bit 23
CFG_BYTE2_DEF = // 0x32

{RSV16_DEF, BBD_DEF, MMR_DEF, RSV20_21_DEF, TCPC_DEF},
BC1PC_DEF = 2'b00, // bit 24
RSV26_DEF = 1'b0,
DLLDIS_DEF = 1'b1,
// MODCK_H_DEF = 3'b100,
RSV31_DEF = 1'b0, // bit 31
CFG_BYTE3_0_3DEF = // 0x01

{BC1PC_DEF, RSV26_DEF, DLLDIS_DEF},
CFG_BYTE3_7DEF = RSV31_DEF;

/*** Power-on-Reset value of BCSR0-3, BCSR6 ***/
parameter

/*** BCSR0 PON value ***/
FLASHPRT1_PON = 1'b0,
FLASHPRT2_PON = 1'b0,
FRM_RST_PON = 1'b1,
CODEC_EN_PON = 1'b1, // CODEC disable
SIGNALS0_PON = 1'b1,
SIGNALS1_PON = 1'b1,
SIGNALH0_PON = 1'b1,
SIGNALH1_PON = 1'b1,
BCSR0_PON = {FLASHPRT1_PON, FLASHPRT2_PON, FRM_RST_PON,
CODEC_EN_PON, SIGNALS0_PON, SIGNALS1_PON,
SIGNALH0_PON, SIGNALH1_PON
}, // 0x2f

/*** BCSR1 PON value ***/
RECONF_PON = 1'b1,
HRST1_PON = 1'b1,
HRST2_PON = 1'b1,
ATM_RST_PON = 1'b1,
HRST3_PON = 1'b1,
FETH_RST_PON = 1'b1,
RS232EN_1_PON = 1'b1, // RS232-1 disable
RS232EN_2_PON = 1'b0, // RS232-2 Enable
BCSR1_PON = {RECONF_PON, HRST1_PON, HRST2_PON, ATM_RST_PON,
HRST3_PON, FETH_RST_PON, RS232EN_1_PON,
RS232EN_2_PON}
}, // 0xff
//*** BCSR2 PON value ***//
RSTCNF_PON = 1'b0, // BCSR2.0
DSISYN_PON = 1'b0, // BCSR2.4
HRST_PON = 1'b1, // BCSR2.5
SRST_PON = 1'b1, // BCSR2.6
FRMtoTSI_PON = 1'b1, // BCSR2.7
//*** BCSR3 PON value ***//
RSV34_PON = 2'b11,
//*** BCSR6 PON value ***//
FLUNLCK1_PON = 1'b1,
FLUNLCK2_PON = 1'b1,
LEDEN = 1'b0,
TEST = 1'b0, // Disable Test Mode
TESTSIG = 3'b0,
MARK = 1'b0,
BCSR6_PON = {FLUNLCK1_PON, FLUNLCK2_PON, LEDEN, TEST,
TESTSIG, MARK}, // 0xd0
//*** BCSR7 PON value ***//
I2C_CON_PON= 1'b0,
DSI_EX_PON= 2'b00,
RPHY_RST_PON = 1'b0, // Reset PHY and CODEC
// RPHY_RST_PON = 1'b1,
VB_INT_PON= 1'b0,
ETH_ON_PON= 1'b0,
//_RST_PON = 1'b1,
ETH_RST_PON = 1'b1,

BCSR7_PON= {I2C_CON_PON,DSI_EX_PON,RPHY_RST_PON,VB_INT_PON,ETH_ON_PON,SPHY_RST_PON
,ETH_RST_PON}, // 0x0B

    //*** BCSR8 PON value ***/
    cPCI_OFF_PON= 1'b0,
    RMII_PON= 1'b0,
    EREV_PON= 2'b00,
    ETH1_RST_PON= 1'b1,
    ETH1_EN_PON= 1'b1,
    EXTMST_PON= 1'b0,
    x= 1'b0,
    BCSR8_PON= {cPCI_OFF_PON,RMII_PON,EREV_PON,ETH1_RST_PON,ETH1_EN_PON,EXTMST_PON,
    x}, // 0x0C

    //*** BCSR9 PON value ***/
    ETH_MODE_PON = 4'b0000,
    FETH1_EN_PON = 1'b0,
    FETH2_EN_PON = 1'b0,
    CODEC_16K_PON = 1'b0,
    BCSR9_PON= {ETH_MODE_PON,FETH1_EN_PON,FETH2_EN_PON,CODEC_16K_PON,x},

    //*** BCSR10 PON value ***/
    DSI_EX2_PON= 1'b1,
    BCSR10_PON = {DSI_EX2_PON,7'b0000000};

    //*** Nodes ***
assign READ_CFG_FROM_BCSR = (nCS0 == `ASSERTED) && (nHRESETh == `ASSERTED)
    && (nPCFG == `From_BCSR) && !nR_W;
assign SSEL[0:3] = {XAt7,XAt8,XAt9,XAt10};
assign SYS_SLOT = !(PCI_PRSNTb || SYSENb);
assign PER_SLOT = !(PCI_PRSNTb || SYSENb);
assign ON_BRD_SLAVE_ADDR = GA[0:3]; //Apply 0xf for any out-of-CPCI backplane configuration
assign ON_BRD_HOST_PRPH  = !(nCS0 && nBCSR_CS);
assign XDBUFXENb = !(OFF_BRD_SLAVE || PER_SLOT);

assign EXT_PCI1 = (OFF_BRD_SLAVE && nR_W) || ((PER_SLOT || VB_ACCESS) && (!nR_W));
\ EXT_PCI = !PER_SLOT ; // "Read" Addr&Cntl from cPCI only in peripheral slot

//External Board Present and has access on
assign VB_ACCESS = (!DACK4b || !DACK3b || !DACK2b || !DACK1b) && `VB_INT;

// OFF Board Slave means Local 8103 is Hosting a 8122 slave from rack
assign OFF_BRD_SLAVE = !CS6hb ||
(SYS_SLOT && ((SSEL != ON_BRD_SLAVE_ADDR) && !nHCS1)) && nHBCS ; //If CS to 8102 on board then assign "0"
assign nDBUFBN = nHCS1 && nHBCS && (!ON_BRD_HOST_PRPH || OFF_BRD_SLAVE || VB_ACCESS));

// nDBGFXEN prevents contention on the Data bus to external boards
assign nDBGFXEN = nHCS1 && nHBCS && (!OFF_BRD_SLAVE || VB_ACCESS));
assign Write_to_BCSR = BCSRCSyn;
assign Read_from_BCSR = !R_W && !BCSR_CS && nHRESETh;
assign BCSR2_PON_DSI = {RSTCNF_PON,2'b10,nSYS64,DSISYN_PON,HRST_PON,SRST_PON,FRMtoTSI_PON};
assign BCSR2_PON_SYS = {RSTCNF_PON,2'b01,nSYS64,DSISYN_PON,HRST_PON,SRST_PON,FRMtoTSI_PON};

assign BTMD = DSItoSYS ? 3'b0 : 3'b001;
assign BCSR3_PONh = {MODCK1s,MODCK2s};
assign BCSR3_PONl = {RSV34_PON,BTMD};
assign BCSR4 = {SEE0_In,SEE1_In,ETH_SW_ON,SWOPT[1:2],3'b111};
assign BCSR5 = BCSR_ver;
assign ADDR3 = !A27 && A28 && A29;
assign ADDR5 = A27 && !A28 && A29;
assign WRITE3 = Write_to_BCSR && ADDR3 && (nWE == `ASSERTED);

assign BCSR1r[0:7] = `{RECONF,nHRST_In[1:2],`ATM_RST,nHRST_In[3],`FETH_RST,`RS232EN_1,`RS232EN_2};
assign BCSR2r[0:7] = `{RSTCNF, `CNFG, `SWTE , nSYS64, `DSISYN, nHRESETs, nRESETs, `FRMtoTSI};
assign CFG_BYTE3_DEF = {CFG_BYTE3_0_3DEF,MODCK4h,MODCK5h,MODCK6h,CFG_BYTE3_7DEF};
assign CFGREG[1:10] = `{`BTMD,`CNFG,`DSI64,`DSISYN,`SWTE,`RSTCNF,`CLKMD};
assign CONFIG_EN = nHRESETs && nPRSTs;
```verilog
// BM_Out[0:2] = (CONFIG_EN == 'ASSERTED)? 'BTMD : 4'bz;
assign `HD_PINS = (CONFIG_EN == 'ASSERTED)&(!PER_SLOT)? CFGREG[4:10] : (Internal_Test ? 
{1'bz,MODCK1h,MODCK3h,MODCK6h,1'b0,MODCK2h,1'bz} : 7'b1);
//define HD_PINS
(CNFREG_Out,DSI64_Out,DSISYNC_Out,SWTE_Out,RSTCFG_Out,MODCK1s_Out,MODCK2s_Out)
assign n32to64En = !(DSI64) || Activate_ETH && (ETH_MODE == 4'b0001)|| (ETH_MODE == 4'b1011) ||
(`ETH_MODE == 4'b1011) || (`ETH Mode == 4'b1110));

//**** Outputs ****/
assign TEST_EN = `TEST || Internal_Test;
assign TEST_SIG[1:3] = (`TEST == `ACTIVE_HIGH)? `TESTSIG : 3'bz;
assign EE1_misc_Out = EE1_misc_In; // temporary Null-Logic
assign MODCKh_Out = (nHRESETh == `ASSERTED)? {MODCK3h,MODCK2h,MODCK1h} : 3'bz;
assign nFCSh = (nFCFG == `From_FLASH)? nCS0 : ((nHRESETh == `ASSERTED)? 1'b1 : nCS0);
assign FRMtoTSI = `FRMtoTSI;
assign nBOOTPh = (`FLASHPRT1 == 0)? `ASSERTED : `NEGATED;
assign nBOOTPs = (`FLASHPRT2 == 0)? `ASSERTED : `NEGATED;
assign SIGH_LED[0] = (`LEDEN == `ACTIVE_LOW)? ((nSRESETh == `ASSERTED)? `ASSERTED :
`SIGNALH0): `NEGATED;
    // Also indicates SRESET assertion
assign SIGH_LED[1] = (`LEDEN == `ACTIVE_LOW)? ((nHRESETh == `ASSERTED)? `ASSERTED :
`SIGNALH1): `NEGATED;
    // Also indicates HRESET assertion
assign SIGS_LED[0] = (`LEDEN == `ACTIVE_LOW)? ((nSRESETs == `ASSERTED)? `ASSERTED :
`SIGNALS0): `NEGATED;
    // Also indicates SRESET assertion
assign SIGS_LED[1] = (`LEDEN == `ACTIVE_LOW)? ((nHRESETs == `ASSERTED)? `ASSERTED :
`SIGNALS1): `NEGATED;
    // Also indicates HRESET assertion
assign SEE1_LED = (`LEDEN == `ACTIVE_LOW)? !SEE1_In : `NEGATED;
    // Debug Acknowledge indication for the Slave
assign nCODEC_EN = (`CODEC_EN == `ACTIVE_LOW)? `ASSERTED : `NEGATED;
assign nRS232EN_1 = (`RS232EN_1 == `ACTIVE_LOW)? `ASSERTED : `NEGATED;
assign nRS232EN_2 = (`RS232EN_2 == `ACTIVE_LOW)? `ASSERTED : `NEGATED;
assign nFETH1RST = (`FETH_RST == `ACTIVE_LOW)||(nHRESETh == `ASSERTED)? `ASSERTED :
`NEGATED;
```
assign nFRM_RST = (`FRM_RST == `ACTIVE_LOW) || (nHRESETh == `ASSERTED)? `ASSERTED :
`NEGATED;

assign HEE1_Out = (HEE1_In == `ASSERTED)? HEE1_In : 1'bz;
assign nNMTh  = ( NMTh == `ASSERTED)? `ASSERTED : `NEGATED;
assign nNMIs  = ( NMIs == `ASSERTED)? `ASSERTED : 1'bz;

//*** Resets Output *** //
assign nHRESETh = ( HRSETh == `ASSERTED || reset ==`ASSERTED)? `ASSERTED : 1'bz ;
assign nSRESETh = ( SRSETh == `ASSERTED)? `ASSERTED : 1'bz ;
assign nRESETs = (`SRST == `ACTIVE_LOW || SRESETs == `ASSERTED)? `ASSERTED : 1'bz;
assign nRESETs = (RST_FROM_cPCI || (`HRST == `ACTIVE_LOW) || (HRESETs == `ASSERTED))
? `ASSERTED : 1'bz;

// PowerOnReset could be given from Rack to SyS-Slot
assign nPRSTs = ((`RECONF == `ACTIVE_LOW) || ((!PER_SLOT)&&(nHRESETh == `ASSERTED)) ||
SYS_SLOT && (BHRESEThb == `ASSERTED)) ? `ASSERTED : 1'bz;

// Identify Reset from System Slot
assign RST_FROM_cPCI = PER_SLOT & nHRST_In[1];

assign nHRST[1]  = (`cPCI_OFF || (!SYS_SLOT)) ? 1'b1 : ((`HRST1 == `ASSERTED) || (nHRSTs ==
`ASSERTED)
? 1'b0 : `NEGATED);
assign nHRST[2]  = (`cPCI_OFF || (!SYS_SLOT)) ? 1'b1 : ((`HRST2 == `ASSERTED)? `HRST2 :
`NEGATED);
assign nHRST[3]  = (`cPCI_OFF || (!SYS SLOT)) ? 1'b1 : ((`HRST3 == `ASSERTED)? `HRST3 :
`NEGATED);

assign BCTL1_Outb = BCTL1_In && nCS0s && ESW_CSb;
assign BCTL1_XOutb = ESW_CSb;

//*** Execution Section ***//

always @ (posedge clk) BCSRCSyn = !nBCSR_CS;
always @ (posedge nHRESETs) GAr[0:3] = GA[0:3];  //*** Latch GA lines ***//

//*** Generation Debug Request for Host & Slave***//

//*** Delayed Host Hard Reset ***//
always @ (posedge extclk)
begin
    if (nHRESETh == `ASSERTED)
        begin nHRESETh_count = 0; nHRESETh_d = 0; end
    else begin
        if (nHRESETh_count[divhreseth] != 1) begin
            nHRESETh_count = nHRESETh_count + 1;
            nHRESETh_d = 0;
        end
        else nHRESETh_d = 1;
    end
end

always @ (posedge extclk)
begin
    if (HEE0_In_syn ^ HEE0_In) HEE0_count = 0;  // Produce short negage pulse
    else begin
        if (HEE0_count[divhee0] != 1) begin
            HEE0_count = HEE0_count + 1; HEE0_Out = 1'b1; // Activate Debug Request at rising edge
        end
        else if (nHRESETh_d == `ASSERTED) HEE0_Out = !HEE0_In;
                    else                    HEE0_Out = 0;
    end
    HEE0_In_syn <= HEE0_In;
end

//*** Delayed Slave Hard Reset ***//
always @ (posedge extclk)
if (nHRESETs == `ASSERTED)
  begin nHRESETs_count = 0; nHRESETs_d = 0; end
else begin
  if (nHRESETs_count[divhresets] != 1) begin
    nHRESETs_count = nHRESETs_count + 1;
    nHRESETs_d = 0;
  end
  else nHRESETs_d = 1;
end
end

//*** Slave Debug Request logic ***//
always @ (posedge clk)
begin
  if (WRITE3 && (Data[2] == 1) || (SEE0_In_syn ^ SEE0_In)) SEE0_count = 0;
  else begin
    if (SEE0_count[divsee0] != 1)
      begin
        SEE0_count = SEE0_count + 1; SEE0_Out = 1'b1; // Activate Debug Request at rising edge
      end
    else if (nHRESETs_d == `ASSERTED) SEE0_Out = !SEE0_In;
      elseSEE0_Out = 0;
end
SEE0_In_syn <= SEE0_In;
if (Internal_Test) SEE0_Out = 1'b1;
end

//**** BCSR Register Main ****//
assign Data[0:7] = DataO[0:7];
always @ (CFG_ADDR or READ_CFG_FROM_BCSR or BCSR_ADDR or Read_from_BCSR)
begin
  if (READ_CFG_FROM_BCSR)
    case (CFG_ADDR)
      2'b00 : DataO[0:7] = CFG_BYTE0_DEF;
      2'b01 : DataO[0:7] = CFG_BYTE1_DEF;
      2'b10 : DataO[0:7] = CFG_BYTE2_DEF;
    endcase
end
l :  DataO[0:7] = CFG_BYTE3_DEF;
endcase
else if(Read_from_BCSR)
    case (BCSR_ADDR)
        0 :  DataO[0:7] = BCSR0[0:7];
        1 :  DataO[0:7] = BCSR1r[0:7];
        2 :  DataO[0:7] = BCSR2r[0:7];
        3 :  DataO[0:7] = {BCSR3[0:1],!SEE0_In,BCSR3[3:7]};
        4 :  DataO[0:7] = BCSR4[0:7];
        5 :  DataO[0:7] = {BCONF,BREVN,BCSR5[5:7]};
        6 :  DataO[0:7] = BCSR6[0:7];
        7 :  DataO[0:7] = BCSR7[0:7];
        8 :  DataO[0:7] = {BCSR8[0:6],CID0};
        9 :  DataO[0:7] = {BCSR9[0:6],CID1};
       10:  DataO[0:7] = BCSR10[0:7];
         default : DataO[0:7] = 8'bz;
    endcase
else DataO[0:7] = 8'bz;

**** BCSR Register Power-on-Reset setting & Write ****/

always @ (posedge nWE or negedge nHRESETh) // or negedge reset
begin
    if(nHRESETh == `ASSERTED)
        begin
            BCSR0[0:7] = BCSR0_PON; // general
            BCSR1[0:7] = BCSR1_PON; // board
            BCSR6[0:7] = BCSR6_PON; // initialize
            BCSR3[0:1] = BCSR3_PONh;
            BCSR3[3:7]= BCSR3_PONl;
            if (!DSItoSYS) BCSR2[0:7] = BCSR2_PON_DSI;
            else            BCSR2[0:7] = BCSR2_PON_SYS;
            BCSR7[0:7] = BCSR7_PON; // 8122/26
            BCSR8[0:7] = BCSR8_PON; // 8122/26
\[ R9[0:7] = \text{BCSR9	extunderscore PON}; \quad /\!/ \text{8122/26 Ethernet} \]

\[
\text{BCSR10}[0:7] = \text{BCSR10	extunderscore PON};
\]

\begin{verbatim}
end
else begin
if (Write_to_BCSR)
    case (BCSR_ADDR)
        0: begin
            if (FLUNLCK1 == ASSERTED) `FLASHPR1 = Data[0];
            else `FLASHPR1 = ASSERTED;
            if (FLUNLCK2 == ASSERTED) `FLASHPR2 = Data[1];
            else `FLASHPR2 = ASSERTED;
            BCSR0[2:7] = Data[2:7];
        end
        1: BCSR1[0:7] = Data[0:7];
        2: begin
            BCSR2[0:2] = Data[0:2];
            BCSR2[4:7] = Data[4:7];
        end
        3: BCSR3[0:7] = Data[0:7];
        6: begin
            BCSR6[0:1] = Data[0:1];
        end
        7: BCSR7[0:7] = Data[0:7];
        8: begin
            BCSR8[0:1] = Data[0:1];
            BCSR8[2:3] = EREV[0:1];
            BCSR8[4:6] = Data[4:6];
        end
        9: BCSR9[0:6] = Data[0:6];
        10: BCSR10[0:7] = Data[0:7];
        // default : ;
    endcase
end
\end{verbatim}
' Debounce functions ***//

function hhtime_elaps;
  input i;
  input [divd:0] d;
  begin
    if (i == 0) hhr = 0;
    else begin
      if (hhr[d] != 1) begin
        hhr = hhr + 1; hhtime_elaps = 1'b0;
      end else hhtime_elaps = 1'b1;
    end
  end
endfunction

function shtime_elaps;
  input i;
  input [divd:0] d;
  begin
    if (i == 0) shr = 0;
    else begin
      if (shr[d] != 1) begin
        shr = shr + 1; shtime_elaps = 1'b0;
      end else shtime_elaps = 1'b1;
    end
  end
endfunction

function nhtime_elaps;
  input i;
  input [divd:0] d;
  begin
    if (i == 0) nhr = 0;
    else begin
      if (nhr[d] != 1) begin
        nhr = nhr + 1; nhtime_elaps = 1'b0;
      end
    end
  end
endfunction
end else nhtime_elaps = 1'b1;
end
end
endfunction

function hstime_elaps;
input i;
input [divd:0] d;
begin
  if (i == 0) hsr = 0;
  else begin
    if (hsr[d] != 1) begin
      hsr = hsr + 1; hstime_elaps = 1'b0;
    end else hstime_elaps = 1'b1;
  end
end
endfunction

function sstime_elaps;
input i;
input [divd:0] d;
begin
  if (i == 0) ssr = 0;
  else begin
    if (ssr[d] != 1) begin
      ssr = ssr + 1; sstime_elaps = 1'b0;
    end else sstime_elaps = 1'b1;
  end
end
endfunction

function nstime_elaps;
input i;
input [divd:0] d;
begin
if (i == 0) nsr = 0;
else begin
  if (nsr[d] != 1) begin
    nsr = nsr + 1; nstime_elaps = 1'0;
  end else nstime_elaps = 1'1;
end
end

definition

always @ (posedge extclk)
begin
  if (reset == `ASSERTED)
    begin
      counter = (1 << div1st) - 2;
    end
  counter = counter + 1;
  if (counter[div1st] != 1) clock_divider = 0;
  else clock_divider = 1;
end

always @ (posedge clock_divider)
begin
  if (HReseth_In == `PRESSED) HRESETh = hhtime_elaps(0,div2st);
else HRESETh = hhtime_elaps(1,div2st);
if (SReseth_In == `PRESSED) SRESETh = shtime_elaps(0,div2st);
else SRESETh = shtime_elaps(1,div2st);
if (Aborth_In == `PRESSED) NMIh = nhtime_elaps(0,div2st);
else NMIh = nhtime_elaps(1,div2st);
if (HResets_In == `PRESSED) HRESETs = hstime_elaps(0,div2st);
else HRESETs = hstime_elaps(1,div2st);
if (SResets_In == `PRESSED) SRESETs = sstime_elaps(0,div2st);
else SRESETs = sstime_elaps(1,div2st);
if (Aborts_In == `PRESSED) NMIs = nstime_elaps(0,div2st);
else NMIs = nstime_elaps(1,div2st);
if (Internal_Test)   NMIs = 1'b1;
end

/* NEW FUNCTIONS FOR 8122ADS */
assign CID1  = (CoreConfig1 || !CoreConfig2)&& B_8126 ;

// Assign Core Voltage Selectors
assign CoreConfig1 = B_8122_1 && !B_8126 && !B_8122_2 && !B_8102;
assign CoreConfig2 = CoreConfig1 || (!B_8122_1 && !B_8126 && !B_8122_2 && B_8102);
assign BCSR_CORE1 = CoreConfig1;
assign BCSR_CORE2 = CoreConfig2;
assign SUP1_IND = !CoreConfig1;
assign SUP2_IND = CoreConfig1 || CoreConfig2;
assign SUP3_IND = CoreConfig1 || !CoreConfig2;
assign CID0  = (CoreConfig1 || !CoreConfig2)&& (B_8122_1 || B_8122_2);

assign IRQ2hb = ((!SYS_SLOT) || IRQ1cPCIb) ? 1'b0 : 1'bz;
assign IRQ1cPCIb = PER_SLOT ? IRQ1hb : 1'bz;
assign RSTCNFhb = PER_SLOT;// Place Host in Default Slave config for PER_SLOT
assign XHRESETsb = 1'bz; // Spare Slave Reset to Connector
assign SYSBUFXENb = `cPCI_OFF ; // Close buffer in cPCI OFF mode
assign ABUFXENb= PER_SLOT;// Close host buffer in Peripheral Slot

//JTAG Chain
assign ChainSel0 = (SYSENb || !SWOPT[2] )&& ChainSel1;
assign ChainSel3 = (!ChainSel1) && PER_SLOT;
assign ChainSel4 = SWOPT[2] && ChainSel1 && PCI_PRSNTb ;

//Ethernet Functions
assign Activate_ETH = (`ETH_ON && !ETH_SW_ON && !B_8102);// 3 operations to Validate ETH_ON
assign RMII2 = `RMII ? 1'b1 : 1'b0;  // Set RMII/MII for RPHY
//assign RMII2 = (SWOPT[1] == `ASSERTED) ? 1'b1 : 1'b0;  // Set RMII/MII for RPHY
Mezzanine Ethernet switch

assign ESW_RESETb = (`ETH_RST == `ACTIVE_LOW)|| (nHRESETh == `ASSERTED) ? `ASSERTED : `NEGATED;

//Reset Mezzanine MII-PHY
assign ESW1RSTb  = (`ETH1_RST == `ACTIVE_LOW)|| (nHRESETh == `ASSERTED) ? `ASSERTED : `NEGATED;

//Reset SMII-PHY
assign SMIIPHYRSTb  = (`SPHY_RST == `ACTIVE_LOW)|| (nHRESETh == `ASSERTED) ? `ASSERTED : `NEGATED;

//Reset SMII-PHY
assign nFETH_RST = (`RPHY_RST == `ACTIVE_LOW)|| (nHRESETh == `ASSERTED) ? `ASSERTED : `NEGATED;

always //@(`ETH_MODE or Activate_ETH)
if (Activate_ETH)
begin
  case (`ETH_MODE)
    4'b0000: begin // 8102 Compatible
      BCSR_PPC_Sb = 1'b1;//DSI ETH closed
      BCSR_TDM_Sb = 1'b1;//8103 side closed
      BCSR_SMIIb = 1'b1;//Not SMII Mode
      BCSR_MSEL = 4'b0011;//25MHz to Host PHY
      // BCSR_MB= 4'b1010;//50MHz 8122 & ETH-SW
      BCSR_MB= 4'b0010;//50MHz from ETH-SW to 8122
      BCSR_RSMIIb= 1'b1;//Not R/SMII PHY
    end
    4'b0001: begin // Mac2Mac RMII DSI to ETH-SW
      BCSR_PPC_Sb = 1'b0;//DSI open
      BCSR_TDM_Sb = 1'b1;//8103 side closed
      BCSR_SMIIb = 1'b0;//Like SMII Mode
      BCSR_MSEL = 4'b0011;//25MHz to Host PHY,50MHz 8122 & ETH-SW
      // BCSR_MB= 4'b1010;//50MHz 8122 & ETH-SW
      BCSR_MB= 4'b0010;//50MHz from ETH-SW to 8122
      BCSR_RSMIIb= 1'b1;//Not R/SMII PHY
    end
    4'b0010: begin // Mac2Mac RMII TDM to ETH-SW
      BCSR_PPC_Sb = 1'b1;//DSI closed
BCSR_TDM_Sb = 1'b1;//8103 side closed
BCSR_SMIIb = 1'b1;//Not SMII Mode
BCSR_MSEL  = 4'b0111;//25MHz to Host PHY,50MHz 8122 & ETH-SW
// BCSR_MB= 4'b1010;//50MHz 8122 & ETH-SW
// BCSR_MB= 4'b0010;//50MHz from ETH-SW to 8122
BCSR_RSMIIb= 1'b1;//Not R/SMII PHY
end

4'b0011: begin // Mac2Mac SMII TDM to ETH-SW
BCSR_PPC_Sb= 1'b1;//DSI closed
BCSR_TDM_Sb = 1'b1;//8103 side closed
BCSR_SMIIb = 1'b0;//SMII Mode
BCSR_MSEL  = 4'b0011;//25MHz to Host PHY
BCSR_MB= 4'b0010;//125MHz to 8122 from ETH-SW
BCSR_RSMIIb= 1'b1;//Not R/SMII PHY
end

4'b0100: begin // Mac2Mac RMII TDM to ETH-SW LOOPBACK
BCSR_PPC_Sb = 1'b1;//DSI closed
BCSR_TDM_Sb = 1'b1;//8103 side closed
BCSR_SMIIb = 1'b1;//Not SMII Mode
BCSR_MSEL  = 4'b0011;//25MHz to Host PHY,50MHz 8122 & ETH-SW
// BCSR_MB= 4'b1010;//50MHz 8122 & ETH-SW
// BCSR_MB= 4'b0010;//50MHz from ETH-SW to 8122
BCSR_RSMIIb= 1'b1;//Not R/SMII PHY
end

4'b0101: begin // Mac2Mac RMII DSI to ETH-SW LOOPBACK
BCSR_PPC_Sb = 1'b0;//DSI open
BCSR_TDM_Sb = 1'b1;//8103 side closed
BCSR_SMIIb = 1'b1;//Like SMII Mode
BCSR_MSEL  = 4'b0011;//25MHz to Host PHY,50MHz 8122 & ETH-SW
// BCSR_MB= 4'b1010;//50MHz 8122 & ETH-SW
// BCSR_MB= 4'b0010;//50MHz from ETH-SW to 8122
BCSR_RSMIIb= 1'b1;//Not R/SMII PHY
end

4'b0110: begin // Mac2Mac SMII TDM to ETH-SW
BCSR_PPC_Sb = 1'b1;//DSI closed
```
.TDM_Sb = 1'b1;//8103 side closed
BCSR_SMIib = 1'b0;//SMII Mode
BCSR_MSEL = 4'b0011;//25MHz to Host PHY
BCSR_MB= 4'b0010;//125MHz to 8122 from ETH-SW
BCSR_RSMIIb= 1'b1;//Not R/SMII PHY
end
4'b0111: begin  // Mac2Phy SMII TDM to SPHY
    BCSR_PPC_Sb= 1'b1;//DSI closed
    BCSR_TDM_Sb = 1'b1;//8103 side closed
    BCSR_SMIib = 1'b0;//SMII Mode
    BCSR_MSEL = 4'b1011;//25MHz to Host PHY,125MHz From Osc.
    BCSR_MB= 4'b1110;//125MHz to 8122 and SPHY
    BCSR_RSMIIb= 1'b0;//SMII PHY
end
4'b1000: begin  // Mac2Phy RMII TDM to RPHY
    BCSR_PPC_Sb= 1'b1;//DSI closed
    BCSR_TDM_Sb = 1'b1;//8103 side closed
    BCSR_SMIib = 1'b1;//Not SMII Mode
    BCSR_MSEL = 4'b0111;//25MHz to Host PHY,50MHz From Osc.
    BCSR_MB= 4'b1010;//50MHz to 8122 and RPHY
    BCSR_RSMIIb= 1'b0;//R/SMII PHY
end
4'b1001: begin  // Mac2Phy RMII DSI to RPHY
    BCSR_PPC_Sb= 1'b0;//DSI open
    BCSR_TDM_Sb = 1'b1;//8103 side closed
    BCSR_SMIib = 1'b0;//Like SMII Mode
    BCSR_MSEL = 4'b0111;//25MHz to Host PHY,50MHz From Osc.
    BCSR_MB= 4'b1010;//50MHz to 8122 and RPHY
    BCSR_RSMIIb= 1'b0;//R/SMII PHY
end
4'b1010: begin  // Mac2Phy MII TDM to RPHY(MII)
    BCSR_PPC_Sb= 1'b1;//DSI closed
    BCSR_TDM_Sb = 1'b1;//8103 side closed
    BCSR_SMIib = 1'b1;//Not SMII Mode
    BCSR_MSEL = 4'b0001;//25MHz to Host PHY,25MHz From Osc.
```
BCSR_MB = 4'b0011; //25MHz from PHY
BCSR_RSMIIb = 1'b0; //R/SMII PHY
end

4'b1011: begin // Mac2Phy MII DSI to RPHY(MII)
    BCSR_PPC_Sb = 1'b0; //DSI open
    BCSR_TDM_Sb = 1'b1; //8103 side closed
    BCSR_SMIIb = 1'b1; //Not SMII Mode
    BCSR_MSEL = 4'b0001; //25MHz to Host PHY, 25MHz From Osc.
    BCSR_MB = 4'b0011; //25MHz from PHY
    BCSR_RSMIIb = 1'b0; //R/SMII PHY
end

4'b1100: begin // Mac2Mac MII TDM 8122 to 8103
    BCSR_PPC_Sb = 1'b1; //DSI closed
    BCSR_TDM_Sb = 1'b0; //8103 side open
    BCSR_SMIIb = 1'b1; //Not SMII Mode
    BCSR_MSEL = 4'b0011; //25MHz to Host PHY, 25MHz From Osc.
    BCSR_MB = 4'b1111; //25MHz from Osc
    BCSR_RSMIIb = 1'b1; //Not R/SMII PHY
end

4'b1101: begin // Mac2Mac MII DSI to RPHY (MII)
    BCSR_PPC_Sb = 1'b0; //DSI open
    BCSR_TDM_Sb = 1'b0; //8103 side open
    BCSR_SMIIb = 1'b1; //Not SMII Mode
    BCSR_MSEL = 4'b0011; //25MHz to Host PHY, 25MHz From Osc.
    BCSR_MB = 4'b1111; //25MHz from Osc
    BCSR_RSMIIb = 1'b1; //Not R/SMII PHY
end

default: begin
    // 8102 Compatible
    BCSR_PPC_Sb = 1'b1; //DSI ETH closed
    BCSR_TDM_Sb = 1'b1; //8103 side closed
    BCSR_SMIIb = 1'b1; //Not SMII Mode
    BCSR_MSEL = 4'b0001; //25MHz to Host PHY
    BCSR_MB = 4'b0010; //25MHz from host PHY
    BCSR_RSMIIb = 1'b1; //Not R/SMII
endcase
end

// else if (SWOPT[1] == `ASSERTED)      // Set 50MHz clock to RPHY
// begin // Mac2Phy RMII TDM/DSI to RPHY
// BCSR_MSEL  = 4'b0111;//25MHz to Host PHY,50MHz From Osc.
// BCSR_MB= 4'b1010;//50MHz to 8122 and RPHY
// end
//
else
begin    // 8102 Compatible
  BCSR_PPC_Sb = 1'b1;//DSI ETH closed
  BCSR_TDM_Sb = 1'b1;//8103 side closed
  BCSR_SMIib = 1'b1;//Not SMII Mode
  BCSR_MSEL  = 4'b0001;//25MHz to Host PHY
  BCSR_MB= 4'b0010;//25MHz to Host PHY
  BCSR_RSMIIb= 1'b1;//Not R/SMII
end

// TDM External Clock master when not Ethernet mode
assign EXT_MSTb = `EXTMST || !ETH_SW_ON ;//Activate_ETH ;
assign ESW1EN = Activate_ETH && `ETH1_EN; // Enable Mezzanine MII-PHY
assign FETH1EN = `FETH1_EN; // Enable Host MII PHY
assign FETH2EN = Activate_ETH && `FETH2_EN; // Enable Slave R/MII PHY
assign ESW_RDb = state[1];//SPR1 || BCTL0sb;
assign ESW_WRb = state[0];//SPR1 || !BCTL0sb;

// Ethernet Mezzanine RD/WR state machine
always @(posedge clk)
if (nHRESETh == `ASSERTED)
  state = `S0;
else
  case (state)
    `S0: if (!ESW_CSb)

: `S1;
else
  state = `S0;
`S1: if (BCTL0sb)
  state = `S3;
else if (!BCTL0sb)
  state = `S2;
`S2: state = `S7;
// wait total 6 cycles (133MHZ Bus will give 40ns RD cycle)
`S7: state = `S8;
`S8: state = `S9;
`S9: state = `S10;
`S10: state = `S5;

  // wait total 6 cycles (133MHZ Bus will give 40ns WR cycle)
`S3: state = `S11;
`S11: state = `S12;
`S12: state = `S13;
`S13: state = `S14;
`S14: state = `S4;

`S4: if (!ESW_CSb)
  state = `S6;
else
  state = `S0;
`S5: if (!ESW_CSb)
  state = `S6;
else
  state = `S0;
`S6: if (!ESW_CSb)
  state = `S6;
else
  state = `S0;
default:
  state = `S0;


// Internal Test
assign Internal_Test = 0; // !SPR1;

// I2C
assign I2C_CONT = `I2C_CON;
assign I2C_CONT1 = `I2C_CON && (!`FETH1_EN);

// DSI Functions
always // @(B_8102 or `DSI_EX)
if (B_8102 || `DSI_EX2)
begin
    HCID[0] = XAt7;
    HCID[1] = XAt8;
    HCID[2] = XAt9;
    HCID[3] = XAt10;
    TTs0 = 1'bx;
end
else
    case (`DSI_EX)
    2'b00: begin
        HCID[0] = XAt7;
        HCID[1] = XAt8;
        HCID[2] = XAt9;
        HCID[3] = 1'b0;
        TTs0 = 1'bx;
    end
    2'b01: begin
        HCID[0] = XAt7;
        HCID[1] = XAt8;
        HCID[2] = 1'b0;
        HCID[3] = 1'b0;
        TTs0 = 1'bx;
    end
endcase
2'b10: begin
    HCID[0] = XAt7;
    HCID[1] = 1'b0;
    HCID[2] = 1'b0;
    HCID[3] = XAt8;
    TTs0 = 1'bz;
end

2'b11: begin
    HCID[0] = Ah5;
    HCID[1] = Ah6;
    HCID[2] = 1'b0;
    HCID[3] = XAt8;
    TTs0 = XAt7;
end
endcase

// LED Indications
assign CORELEDh = (`LEDEN == `ACTIVE_LOW)? LEDDRVh : `NEGATED;
assign CORELEDs = (`LEDEN == `ACTIVE_LOW)? LEDRVs : `NEGATED;
assign RUNhLED = (`LEDEN == `ACTIVE_LOW)? DBBhb : `NEGATED;
assign RUNsLED = (`LEDEN == `ACTIVE_LOW)? DBBsb : `NEGATED;
assign LED3V3 = (`LEDEN == `ACTIVE_LOW)? `ASSERTED : `NEGATED;
assign SMII_LEDb = (`LEDEN == `ACTIVE_LOW)? BCSR_SMIIb : `NEGATED;
assign ETH_IND = !Activate_ETH ; // Ethernet-ON LED

dffFSYNC_8_FF(.d(!FSYNC_8), .q(FSYNC_8), .clk(FSYNC), .clrn(reset));
assign FSYNC2 = `CODEC_16K ? FSYNC_8 : FSYNC ;
endmodule