MMCCMB3401 Controller and Memory Board (CMB3401) User's Manual

CAUTION: ESD Protection

M•CORE development systems include open-construction printed circuit boards that contain static-sensitive components. These boards are subject to damage from electrostatic discharge (ESD). To prevent such damage, you must use static-safe work surfaces and grounding straps, as defined in ANSI/EOS/ESD S6.1 and ANSI/EOS/ESD S4.1. All handling of these boards must be in accordance with ANSI/EAI 625.

Contents

Section 1 Introduction

Section 2 Configuration

Section 3 Operation

Section 4 Using the FPGA Device

Section 5 Connector Information

Figures

Tables

Section 1 Introduction

This user's manual explains connection, configuration, and operation information for the MMCCMB3401 Controller and Memory Board (CMB3401), a development tool of Motorola's M•CORE™ family. The CMB3401 lets you develop code to be embedded in an MMC3401 microcontroller unit.

The CMB3401 uses an RS232 connection to your computer. This connection lets you use Motorola's M•CORE System Development Software (SysDS) or the GNU source-level debugger. The SysDS consists of a loader, the Picobug monitor, the ESL monitor, and the built-in selftest. The CMB3401 also has a OnCE™ connector, enabling you to use a debugging tool or application that requires one.

Optionally, you may use the CMB3401 with a different emulator product, such as the appropriate Motorola Embedded Background Debug Interface (EBDI).

Motorola's SysDS Loader lets you download your code into the CMB3401's SRAM (for execution) or FLASH memory (for execution or for storage in non-volatile memory). Should your application overwrite system software in the FLASH memory device, you can use the SysDS Loader to restore the system software.

1.1 CMB3401 Features

The CMB3401 features:

- Motorola M340 resident MCU.
- 2 megabytes burst FLASH memory on the external interface module (EIM) bus.
- 2 megabytes FLASH memory on the M•CORE local bus (MLB).
- 2 megabytes fast SRAM on the EIM bus.
- 2 megabytes fast SRAM on the MLB.
- Altera EPF10K100A FPGA device, with a configuration chip.
- Four power regulators that provide four voltages: 5, 3.2, 3.0, and 1.8.
- Power supply that converts line power to 12-volt power.
- Power adapter cable for a bench supply.
- Two RS232 channels for serial communications. These channels use internal universal asynchronous receiver/transmitters (UARTs).
- A NEXUS port (also known as a GEPDIS port).
- A MAPI 400 connector interface ring, on the top and bottom of the CMB3401, for easy connection to other, compatible development boards.

- Motorola's SysDS.
- Altera MAX+plusII development software.
- GNU source-level debugger (from the Free Software Foundation).
- Three 38-pin Mictor logic analyzer connectors on the EIM bus.
- Three 38-pin Mictor logic analyzer connectors on the MLB.
- Altera ByteBlaster cable.

1.2 System and User Requirements

You need an IBM PC or compatible computer, running the Windows 95 or WindowsNT (version 4.0) operating system. The computer requires a Pentium (or equivalent) microprocessor, 16 megabytes of RAM, 50 megabytes of free hard-disk space, an SVGA color monitor, and an RS232 serial-communications port. To use the Picobug monitor, you also need Hyperterminal or a comparable terminal-emulation program.

To get the most from your CMB3401, you should be an experienced C or M•CORE assembly programmer.

The power supply that comes with your CMB3401 converts line power to the input power that the CMB3401 needs: 12 volts at a minimum of 0.5 amperes.

1.3 CMB3401 Layout

Figure 1-1 shows the layout of the CMB3401. Jumper header W1 specifies the EIM FSRAM chip select (0 or 1); jumper header W2 specifies the EIM FLASH chip select (0 or 1). Jumper header W3 enables or disables MLB FLASH. Jumper header W4 enables or disables MLB SRAM. Headers W6 and W7 are for factory use only: you should not remove the jumpers from these headers.

Connectors P1 through P4, on the top of the board, are the MAPI I/O and interrupt connectors (the corresponding MAPI connectors on the bottom of the CMB3401 are J1 through J4).Connector J13 is the OnCE connector. Connector J14 is for reprogramming the FPGA device (at location U2). Connector J23 is the NEXUS connector.

Connectors J25, J28, and J32 are the MLB logic analyzer connectors. Connectors J29, J30, and J33 are the EIM-bus logic analyzer connectors. Connectors J57 and J58 are the RS232 serial connectors. Connector J59 is the connector for 12-volt input power.

J7 is an eyelet connection for the MLB_OLD_TEA signal. J18 is an eyelet connector for the MLB_OLD_TA signal.

Switch S1 specifies memory-access mode, port size, and first address to be accessed. Switch S2 specifies the software module to be run upon reset. Switch S3 controls several aspects of memory configuration. Switch S4 specifies the number of wait states. Switch S5 is the master reset switch (resetting the entire board). Switch S6 resets only the resident M340 MCU (at location U3). Switch S7 reconfigures the FPGA device (at location U2).

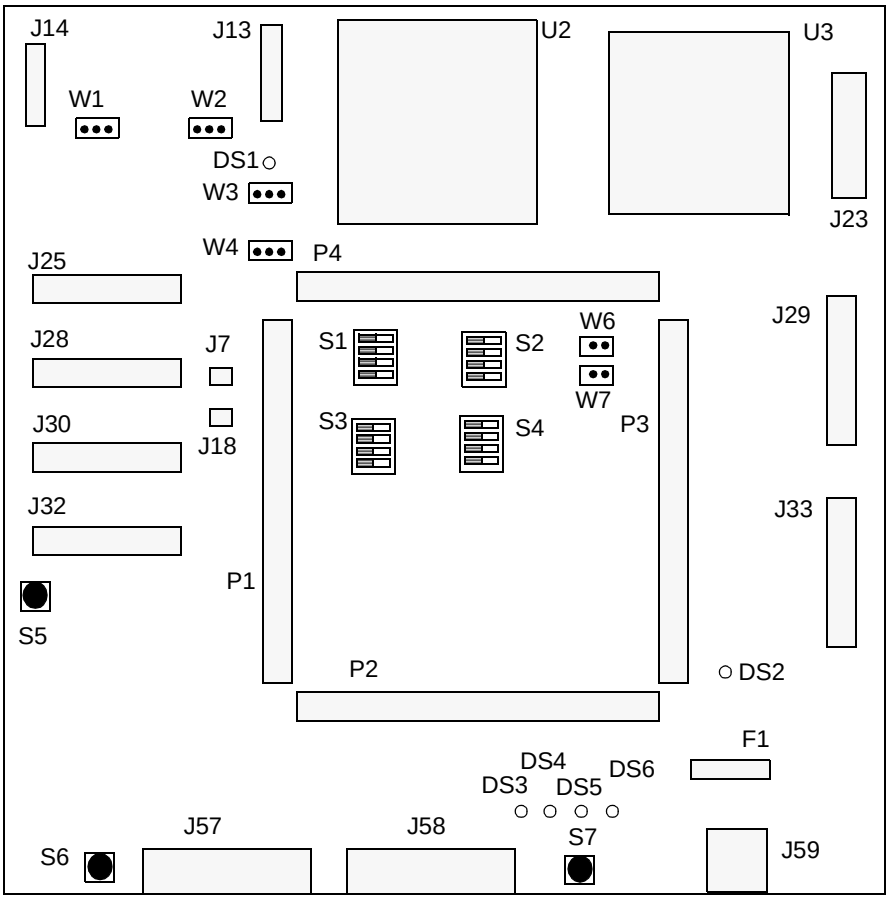


Figure 1-1 MMCCMB3401 Computer and Memory Board

Location F1 is for the CMB3401 fuse.

LED DS1 indicates configuration completion for the FPGA device. LED DS2 confirms board power. LEDs DS3 through DS6 are status indicators.

NOTE: *Some CMB3401 locations are for factory use, so are not populated on your board. Although populated, headers W6 and W7 also are for factory use only: you should not remove the jumpers from these headers.*

Table 1-1 lists CMB3401 specifications.

Table 1-1 MMCCMB3401 Controller and Memory Board Specifications

Characteristic	Specifications
MCU extension I/O ports	HCMOS compatible
Operating temperature	0° to 40° C
Storage temperature	-40° to +85° C
Relative humidity	0 to 90% (non-condensing)
Clock	20 MHz
Power requirements	12 volts dc, at a minimum 0.5 amperes, provided from a separate power source
Dimensions	6.9 x 8.2 inches (175 x 208 mm)

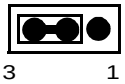
Section 2 Configuration

This chapter explains how to configure your CMB3401, and how to hook it up to your computer system.

2.1 Configuring Board Components

Configuring your CMB3401 involves setting several components. **Table 2-1** is a summary of these settings; paragraphs 2.1.1 through 2.1.8 give additional information.

Table 2-1 Component Configuration Setting

Component	Position	Effect
EIM FSRAM Header, W1		Selects chip-select 1. (So W2 must select chip select 0.) **Factory setting.**
		Selects chip-select 0. (So W2 must select chip select 1.)
EIM FLASH Header, W2		Selects chip-select 1. (So W1 must select chip select 0.)
		Selects chip-select 0. (So W1 must select chip select 1.) **Factory setting.**
MLB FLASH Enable Header, W3		Enables MLB FLASH. **Factory setting.**
		Disables MLB FLASH
MLB SRAM Enable Header, W4		Disables MLB SRAM.
		Enables MLB SRAM. **Factory setting.**

Configuration

Table 2-1 Component Configuration Setting

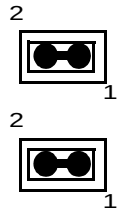
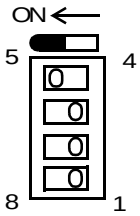
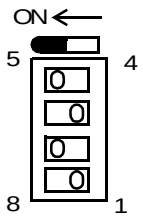
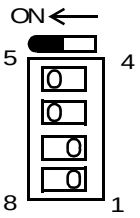
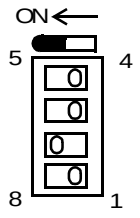
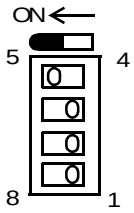
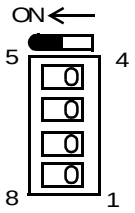
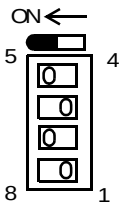
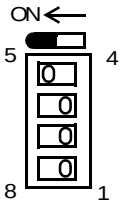
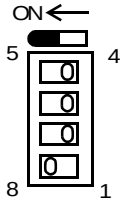
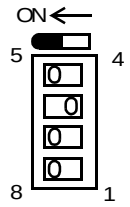
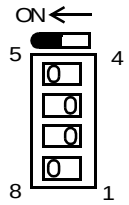
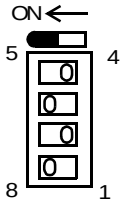
Component	Position	Effect
Factory Headers, W6, W7		Factory use only. **Factory setting. — Do not remove jumpers.**
Memory Access Switch, S1		Configures Big Endian mode, 32-bit port booted internally, and 0x0000_0000 as the first address accessed. (One of many possible S1 configurations.) **Factory setting.**
		Configures Big Endian mode, 8-bit port on pins D[15:8], and 0x1000_0000 as the first address accessed. (Another of many possible S1 configurations.)
		Configures Big Endian mode, 16-bit port on pins D[31:16], and 0x1000_0000 as the first address accessed. (Another of many possible S1 configurations.)
Software Select Switch, S2		Specifies Picobug to be run upon reset. (One of many possible S2 configurations.) **Factory setting.**

Table 2-1 Component Configuration Setting

Component	Position	Effect
Software Select Switch, S2 (continued)		Specifies MLB user code to be run upon reset. (Another of many possible S2 configurations.)
		Specifies EIM user code to be run upon reset. (Another of many possible S2 configurations.)
Memory Configuration Switch, S3		Configures memory mapping starting at address 0x0000_0000, FLASH before SRAM in the memory map, and MLB memory activated in the CMB. (One of many possible S3 configurations.) **Factory setting.**
		Configures memory mapping starting at address 0x0000_0000, FLASH after SRAM in the memory map, and MLB memory activated in the CMB. (Another of many possible S3 configurations.)
		Configures memory mapping starting at address 0x0100_0000 and MLB memory deactivated in the CMB. (Another of many possible S3 configurations.)

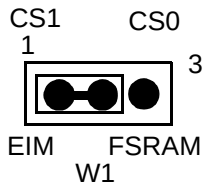
Configuration

Table 2-1 Component Configuration Setting

Component	Position	Effect
Wait State Switch, S4		Configures a wait state of 4 clock cycles. (One of many possible S4 configurations.) **Factory setting.**
		Configures a wait state of 6 clock cycles. (Another of many possible S4 configurations.)
		Configures a wait state of 10 clock cycles. Another of many possible S4 configurations.)
Master Reset Switch, S5		Push to reset all board components.
Reset Switch, S6		Push to reset only the resident MCU (location U3).
FPGA Configuration Switch, S7		Push to reconfigure the FPGA device (location U2)

2.1.1 Setting the EIM FSRAM Chip Select Header (W1)

Jumper header W1 specifies the chip select for the 2 megabytes of FSRAM on the EIM bus. The diagram below shows the factory configuration: the jumper between pins 1 and 2 selects chip select 1.

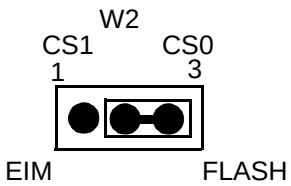


To select instead chip select 0, reposition the W1 jumper between pins 2 and 3.

NOTE: *Jumper headers W1 and W2 must specify opposite chip selects. If W1 specifies chip select 1, as in the diagram above, W2 must specify chip select 0.*

2.1.2 Setting the EIM FLASH Chip Select Header (W2)

Jumper header W2 specifies the chip select for the 2 megabytes of FLASH on the EIM bus. The diagram below shows the factory configuration: the jumper between pins 2 and 3 selects chip select 0.

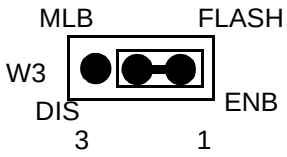


To select instead chip select 1, reposition the W2 jumper between pins 1 and 2.

NOTE: *Jumper headers W1 and W2 must specify opposite chip selects. If W2 specifies chip select 0, as in the diagram above, W2 must specify chip select 1.*

2.1.3 Setting the MLB FLASH Enable Header (W3)

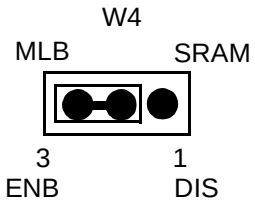
Jumper header W3 enables or disables the 2 megabytes of FLASH memory on the MLB. The diagram below shows the factory configuration: the jumper between pins 1 and 2 enables FLASH.



To disable MLB FLASH, reposition the W3 jumper between pins 2 and 3.

2.1.4 Setting the MLB SRAM Enable Header (W4)

Jumper header W4 enables or disables the 2 megabytes of FSRAM on the MLB. The diagram below shows the factory configuration: the jumper between pins 2 and 3 enables FSRAM.

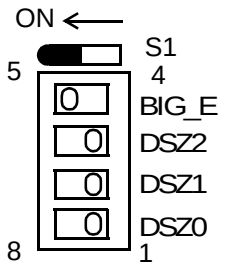


To disable MLB FSRAM, reposition the W4 jumper between pins 1 and 2.

2.1.5 Setting the Memory Access Switch (S1)

Switch S1 specifies memory-access mode, port size, and first address to be accessed. The diagram below shows the factory configuration: the BIG_E subswitch ON, all other subswitches OFF. This configures:

- Big Endian mode,
- 32-bit port, booted internally, and
- 0x0000_0000 to be the first address accessed.



To configure Little Endian mode, set the BIG_E subswitch to OFF.

NOTE: *Do not set switch S1 for Little Endian mode unless you use that mode for all stages of your code development: writing, compiling, and debugging.*

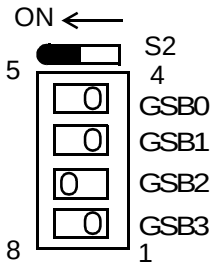
To configure a different port size or first address, set the other S1 subswitches per **Table 2-2**.

Table 2-2 S1 Subswitch Settings

Port Size	Port Location	First Address Accessed	DSZ2 Subswitch	DSZ1 Subswitch	DSZ0 Subswitch
8 bits	EIM D[31:24] pins	0x1000_0000	ON	ON	ON
8 bits	EIM D[23:16] pins	0x1000_ 0000	ON	ON	OFF
8 bits	EIM D[15:8] pins	0x1000_0000	ON	OFF	ON
8 bits	EIM D[7:0] pins	0x1000_0000	ON	OFF	OFF
16 bits	EIM D[31:16] pins	0x1000_0000	OFF	ON	ON
16 bits	EIM D[15:0] pins	0x1000_0000	OFF	ON	OFF
32 bits	EIM D[31:0] pins	0x1000_0000	OFF	OFF	ON
32 bits	internal (MLB)	0x0000_0000	OFF	OFF	OFF

2.1.6 Setting the Software Select Switch (S2)

Switch S2 specifies the software to be run upon a reset. The diagram below shows the factory configuration: the GSB0 and GSB1 subswitches OFF, and the GSB2 subswitch ON. This specifies Picobug.



To specify a different software module, reset the S2 subswitches per **Table 2-3**.

Table 2-3 S2 Subswitch Settings

Software Module	GSB0 Subswitch	GSB1 Subswitch	GSB2 Subswitch
Built-In selftest	ON	ON	ON
SysDS Programmer	OFF	ON	ON
ESL monitor	ON	OFF	ON
Picobug	OFF	OFF	ON
Picobug	ON	ON	OFF
Picobug	OFF	ON	OFF
MLB user code	ON	OFF	OFF
EIM user code	OFF	OFF	OFF

NOTES: *1. The S2 GSB3 subswitch is nonfunctional.*

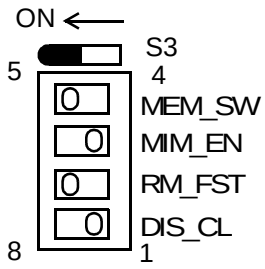
2. ESL monitor requires additional debug software on your computer for compatibility with your CMB3401; for a source of such software, see the product release guide. ESL monitor is compatible only with Big Endian mode, so subswitch S1-1 must be ON.

3. You also may use switch S2 for control of your own application software. Subsection 3.3 explains this additional role.

2.1.7 Setting the MLB Memory Configuration Switch (S3)

Switch S3 specifies several aspects of MLB memory configuration. The diagram below shows the factory configuration: the MEM_SW and RM_FST subswitches ON, and the MIM_EN and DIS_CL subswitches OFF. This configures:

- Memory mapping starting address 0x0000_0000,
- FLASH before SRAM in the memory map, and
- MLB memory enabled in the CMB3401.



For different configurations, set the S3 subswitches per **Table 2-4**.

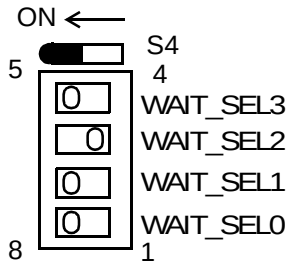
Table 2-4 S3 Subswitch Settings

Subswitch	Effect Set ON	Effect Set OFF
MEM_SW (Memory swap)	Memory mapping begins at address 0x0000_0000.	Memory mapping begins at address 0x0F00_0000.
RM_FST (ROM First)	Puts FLASH (ROM) before SRAM in the memory map (provided that the MEM_SW subswitch starts mapping at address 0x0000_0000.	Puts FLASH (ROM) after SRAM in the memory map (provided that the MEM_SW subswitch starts mapping at address 0x0000_0000.
DIS_CL (Disable CMB)	Disables internal CMB MLB memory.	Enables internal CMB MLB memory.

NOTE: *The S3 MIM_EN subswitch is nonfunctional.*

2.1.8 Setting the Wait State Switch (S4)

Switch S4 specifies the length of wait states. The diagram below shows the factory configuration: all subswitches in the OFF position. This configures a wait state of 4 clock cycles.



To configure a different number of clock cycles, set the S4 subswitches per **Table 2-5**.

Table 2-5 S4 Subswitch Settings

Wait State Clock Cycles	WAIT_SEL3 Subswitch	WAIT_SEL2 Subswitch	WAIT_SEL1 Subswitch	WAIT_SEL0 Subswitch
0	ON	ON	ON	ON
1	ON	ON	ON	OFF
2	ON	ON	OFF	ON
3	ON	ON	OFF	OFF
4	ON	OFF	ON	ON
5	ON	OFF	ON	OFF
6	ON	OFF	OFF	ON
7	ON	OFF	OFF	OFF
8	OFF	ON	ON	ON
9	OFF	ON	ON	OFF
10	OFF	ON	OFF	ON
11	OFF	ON	OFF	OFF
12	OFF	OFF	ON	ON
13	OFF	OFF	ON	OFF
14	OFF	OFF	OFF	ON
15	OFF	OFF	OFF	OFF

NOTE: *The switch S4 setting does not override a software wait-state setting.*

2.2 Making Computer-System Connections

When you have configured your CMB3401, you are ready to connect it to your computer system:

1. Make sure that power is disconnected.
2. If you will use RS232 communication with your host computer, connect an RS232 cable between CMB3401 connector J57 and the appropriate serial port of your computer. (Optional: If your application must have the higher UART addresses, use connector J58 for your RS232 communication.)
3. If you will use a OnCE-compatible emulator with your CMB3401, connect an appropriate 14-lead ribbon cable between CMB3401 connector J13 and your emulator. Then use an appropriate cable to connect your emulator to your host computer.
4. If you will use a NEXUS-compatible emulator with your CMB3401, connect an appropriate NEXUS cable between CMB3401 connector J23 and your emulator. Then use an appropriate cable to connect your emulator to your host computer.
5. Optional: If you use the CMB3401 with another, compatible development board, you must connect the boards via their MAPI rings. To do so, hold the CMB3401 directly above the other board. Turn the CMB3401 so that the right-triangle silk screen markings line up. Then press the CMB3401 down onto the other board. CMB3401 connectors J1 through J4, on the bottom of the board, must connect with the corresponding MAPI connectors P1 through P4, on the top of the other board.
6. Optional: You may use a logic analyzer with the CMB3401. If you do, connect appropriate cables to any of the logic analyzer connectors:
   - J25, 28, or J32 with regard to the MLB.
   - J29, J30, or J33 with regard to the EIM bus.
7. Optional: If you will reprogram the U2 FPGA device, connect the ByteBlaster cable to CMB3401 connector J14. Make sure that the red wire connects to pin 1 of the connector. Connect the other end of the ByteBlaster cable to a parallel port of your computer, per the instructions of your Altera documentation. (Section 4 gives additional information about reprogramming the U2 device.)
8. Connect the +12-volt power supply to CMB3401 connector J59 and to line power. LED DS2 lights to confirm that the CMB3401 is powered and converting input voltage.

   Should DS2 *not* light, you may need to replace the fuse at location F1. (Use a BUS GMA-1.5A fuse, or compatible.)
9. This completes system connections: you are ready to perform a selftest, per the instructions of subsection 2.3, below. You are ready to begin debugging or other development activities, per the instructions of Section 3.

2.3 Performing the CMB3401 Selftest

Once you have configured your CMB3401, you can perform a selftest of its components.

NOTE: *If you open Hyperterminal, per the instructions of subsection 3.1.1, Hyperterminal displays the progress of the selftest. Should the selftest fail, Hyperterminal indicates the address at which the test failed.*

1. Make sure that CMB3401 power is disconnected. The power LED DS2 should be out.
2. Set switch S2 for the built-in selftest: the GSB0, GSB1, and GSB2 subswitches all ON.
3. Apply power. LED DS2 comes on, and the CMB3401 automatically begins its selftest.
4. LEDs DS3 through DS6 (also designated GCB0 through GCB3, respectively) flash in rapid patterns during the test. Then the four LEDs flash each second or two, in unison, confirming that the CMB3401 has passed.
5. Flashing of fewer than four LEDs indicates a failure, per **Table 2-6**. In case of such a failure, you should contact Motorola customer support for assistance.

Table 2-6 CMB3401 Selftest LED Patterns

DS3 (GCB0)	DS4 (GCB1)	DS5 (GCB2)	DS6 (GCB3)	Meaning
OFF	OFF	ON	ON	EIM RAM failure.
OFF	OFF	OFF	ON	MLB RAM failure.
ON	OFF	ON	OFF	MLB FLASH failure.

6. Turn off power.
7. Configure switch S2 for your next development activity before restoring power to the CMB3401.

2.4 Memory Maps

Table 2-7 shows the default memory map for the MBL bus. **Table 2-8** lists the MLB FLASH sector boundaries. **Table 2-9** shows the default memory map for the EIM bus. **Table 2-10** lists the EIM FLASH sector boundaries. The shaded cells of **Table 2-8** and **Table 2-10** indicate the sectors that contain system software.

NOTE: *For either MLB SRAM or EIM SRAM, the first 64 kilobytes contain system software. You should confine your code or data to the remainder of the SRAM.*

Table 2-7 CMB3401 MLB Default Memory Map

Address Range	Contents	Related Signals
0x0000_0000 0x001F_FFFF	FLASH (2M)[1]	FLASH_CS_b
0x0020_0000 0x003F_7FFF	Reserved	
0x0040_0000 0x005F_FFFF	Fast SRAM (2M)[2]	FSRAM_CS_b
0x0060_8000 0x0060_FFFF	Reserved	
0x0100_0000 0x01FF_FFFF	Chip select 0 (16M)	CS_b[0]
0x0200_0000 0x02FF_FFFF	Chip select 1 (16M)	CS_b[1]
0x0300_0000 0x03FF_FFFF	Chip select 2 (16M)	CS_b[2]
0x0400_0000 0x04FF_FFFF	Chip select 3 (16M)	CS_b[3]
0x0500_0000 0x05FF_FFFF	Chip select 4 (16MK)	CS_b[4]
0x0600_0000 0x060F_FFFF	Chip select 5 (1M)	CS_b[5]
0x0610_0000 0x061F_FFFF	Chip select 6 (1M)	CS_b[6]
0x0620_0000 0x062F_FFFF	Chip select 6 (1M)	CS_b[7]
0x0630_0000 0x063F_FFFF	Chip select 6 (1M)	CS_b[8]
0x0640_0000 0x064F_FFFF	Chip select 6 1M)	CS_b[9]
0x0FFF_FFFF		

NOTES:

1. If this address range is FLASH, **Table 2-8** lists the sector (block) boundaries.
2. If this address range is FLASH (that is, switch S3-1 is OFF), **Table 2-8** lists the sector (block) boundaries.

Table 2-8 MLB FLASH Sector Boundaries

Sector (Block)	Range (S3-3 ON)	Range (S3-3 OFF)
0	0x0000_0000 — 0x0000_7FFF	0x0040_0000 — 0x0040_7FFF
1	0x0000_8000 — 0x0000_BFFF	0x0040_8000 — 0x0040_BFFF
2	0x0000_C000 — 0x0000_FFFF	0x0040_C000 — 0x0040_FFFF
3	0x0001_0000 — 0x0001_FFFF	0x0041_0000 — 0x0041_FFFF
4	0x0002_0000 — 0x0003_FFFF	0x0042_0000 — 0x0043_FFFF
5	0x0004_0000 — 0x0005_FFFF	0x0044_0000 — 0x0045_FFFF
6	0x0006_0000 — 0x0007_FFFF	0x0046_0000 — 0x0047_FFFF
7	0x0008_0000 — 0x0009_FFFF	0x0048_0000 — 0x0049_FFFF
8	0x000A_0000 — 0x000B_FFFF	0x004A_0000 — 0x004B_FFFF
9	0x000C_0000 — 0x000D_FFFF	0x004C_0000 — 0x004D_FFFF
10	0x000E_0000 — 0x000F_FFFF	0x004E_0000 — 0x004F_FFFF
11	0x0010_0000 — 0x0011_FFFF	0x0050_0000 — 0x0051_FFFF
12	0x0012_0000 — 0x0013_FFFF	0x0052_0000 — 0x0053_FFFF
13	0x0014_0000 — 0x0015_FFFF	0x0054_0000 — 0x0055_FFFF
14	0x0016_0000 — 0x0017_FFFF	0x0056_0000 — 0x0057_FFFF
15	0x0018_0000 — 0x0019_FFFF	0x0058_0000 — 0x0059_FFFF
16	0x001A_0000 — 0x001B_FFFF	0x005A_0000 — 0x005B_FFFF
17	0x001C_0000 — 0x001D_FFFF	0x005C_0000 — 0x005D_FFFF
18	0x001E_0000 — 0x001F_FFFF	0x005E_0000 — 0x005F_FFFF

Table 2-9 CMB3401 EIM Default Memory Map

Address Range	Contents	Comments
0x0000_0000 0x003F_FFFF	Reserved	
0x0040_0000 0x005F_FFFF	Reserved	
0x0060_0000 0x0060_7FFF	Reserved	
0x0060_8000 0x0060_FFFF	Reserved	
0x1000_0000 0x11FF_FFFF	Chip select 0 (16M)	Active low. Can be used either for FLASH or FSRAM.[1]
0x1200_0000 0x12FF_FFFF	Chip select 1 (16M)	Active low. Can be used either for FLASH or FSRAM.[1.]
0x1300_0000 0x13FF_FFFF	Chip select 2 (16M)	Active low. Can be used for DUART A.[2]
0x1400_0000 0x14FF_FFFF	Chip select 3 (16M)	Active low. Can be used for DUART B.[3]
0x1500_0000 0x15FF_FFFF	Chip select 4 (16M)	Active low.
0x1600_0000 0x16FF_FFFF	Chip select 5 (16M)	Active low.
0x1700_0000 0xFFFF_FFFF	Reserved	

NOTES:

1. **Table 2-10** shows sector (block) boundaries for EIM FLASH memory. These sectors do not take up the entire memory range that either chip select 0 or chip select 1 specifies.
2. The address of DUART A is 0x1300_8000. The DUART A interrupt, associated with connector J57, connects to `p_int_b[23]` of the processor.
3. The address of DUART B is 0x1400_0000. The DUART B interrupt, associated with connector J58, connects to `p_int_b[21]` of the processor.

Table 2-10 EIM FLASH Sector Boundaries

Sector (Block)	Range (W2 Selects CS0)	Range (W2 Selects CS1)
0	0x1000_0000 — 0x1000_3FFF	0x1200_0000 — 0x1200_3FFF
1	0x1000_4000 — 0x1000_7FFF	0x1200_4000 — 0x1200_7FFF
2	0x1000_8000 — 0x1000_BFFF	0x1200_8000 — 0x1200_BFFF
3	0x1000_C000 — 0x1000_FFFF	0x1200_C000 — 0x1200_FFFF
4	0x1001_0000 — 0x1001_3FFF	0x1201_0000 — 0x1201_3FFF
5	0x1001_4000 — 0x1001_7FFF	0x1201_4000 — 0x1201_7FFF
6	0x1001_8000 — 0x1001_BFFF	0x1201_8000 — 0x1201_BFFF
7	0x1001_C000 — 0x1001_FFFF	0x1201_C000 — 0x1201_FFFF
8	0x1002_0000 — 0x1003_FFFF	0x1202_0000 — 0x1203_FFFF
9	0x1004_0000 — 0x1005_FFFF	0x1204_0000 — 0x1205_FFFF
10	0x1006_0000 — 0x1007_FFFF	0x1206_0000 — 0x1207_FFFF
11	0x1008_0000 — 0x1009_FFFF	0x1208_0000 — 0x1209_FFFF
12	0x100A_0000 — 0x100B_FFFF	0x120A_0000 — 0x120B_FFFF
13	0x100C_0000 — 0x100D_FFFF	0x120C_0000 — 0x120D_FFFF
14	0x100E_0000 — 0x101F_FFFF	0x120E_0000 — 0x121F_FFFF
15	0x1010_0000 — 0x1011_FFFF	0x1210_0000 — 0x1211_FFFF
16	0x1012_0000 — 0x1013_FFFF	0x1212_0000 — 0x1213_FFFF
17	0x1014_0000 — 0x1015_FFFF	0x1214_0000 — 0x1215_FFFF
18	0x1016_0000 — 0x1017_FFFF	0x1216_0000 — 0x1217_FFFF
19	0x1018_0000 — 0x1019_FFFF	0x1218_0000 — 0x1219_FFFF
20	0x101A_0000 — 0x101B_FFFF	0x121A_0000 — 0x121B_FFFF
21	0x101C_0000 — 0x101D_FFFF	0x121C_0000 — 0x121D_FFFF
22	0x101E_0000 — 0x101F_FFFF	0x121E_0000 — 0x121F_FFFF
23	0x1020_0000 — 0x1021_FFFF	0x1220_0000 — 0x1221_FFFF
24	0x1022_0000 — 0x1023_FFFF	0x1222_0000 — 0x1223_FFFF
25	0x1024_0000 — 0x1025_FFFF	0x1224_0000 — 0x1225_FFFF
26	0x1026_0000 — 0x1027_FFFF	0x1226_0000 — 0x1227_FFFF
27	0x1028_0000 — 0x1029_FFFF	0x1228_0000 — 0x1229_FFFF
28	0x102A_0000 — 0x102B_FFFF	0x122A_0000 — 0x122B_FFFF
29	0x102C_0000 — 0x102D_FFFF	0x122C_0000 — 0x122D_FFFF
30	0x102E_0000 — 0x102F_FFFF	0x122E_0000 — 0x122F_FFFF
31	0x1030_0000 — 0x1031_FFFF	0x1230_0000 — 0x1231_FFFF
32	0x1032_0000 — 0x1033_FFFF	0x1232_0000 — 0x1233_FFFF
33	0x1034_0000 — 0x1035_FFFF	0x1234_0000 — 0x1235_FFFF
34	0x1036_0000 — 0x1037_FFFF	0x1236_0000 — 0x1237_FFFF
35	0x1038_0000 — 0x1039_FFFF	0x1238_0000 — 0x1239_FFFF
36	0x103A_0000 — 0x103B_FFFF	0x123A_0000 — 0x123B_FFFF
37	0x103C_0000 — 0x103D_FFFF	0x123C_0000 — 0x123D_FFFF
38	0x103E_0000 — 0x103F_FFFF	0x123E_0000 — 0x123F_FFFF

Section 3 Operation

This section explains how to begin using debugging tools available for your MMCCMB3401 Controller and Memory Board, as well as how to use Motorola's SysDS Loader.

3.1 Debugging Embedded Code

With your CMB3401, you may use the Picobug monitor, as standalone software. Optionally, you may use the GNU source-level debugger with the Picobug monitor. Other firms may produce still additional software to run, test, and modify the code you develop for embedding in an MMC3401 MCU.

To use the Motorola System Development Software to download and transfer control to your code, you must be careful to program only the ranges of FLASH memory or SRAM that are allocated for user code or user space. Programming over ranges that contain system software or data storage would impair or destroy the usefulness of the software. (Subsection 3.2.1 identifies the contents of memory ranges; subsection 3.2.2 explains how to use the SysDS Loader to restore factory programming.)

3.1.1 Using the Picobug Monitor

The Picobug monitor comes burned into the FLASH memory device of your CMB3401**.** Before you start the Picobug monitor, make sure that you have an RS232 connection between CMB3401 connector J57 and a serial port of your computer.

To start the Picobug monitor, for use as a standalone debugger:

1. Set switch S2 for the Picobug monitor. The factory setting, for example, specifies Picobug: the GSB0 and GSB1 subswitches OFF, and the GSB2 subswitch ON.
2. Apply power to the CMB3401 (or press the reset switch), then press the enter key. The Picobug monitor starts automatically, displaying the command prompt: `picobug>`.
3. Activate Hyperterminal or a comparable terminal-emulation program. (If you use a different terminal-emulation program, you must make corresponding changes in the commands and menu selections of these instructions, and in the instructions of paragraph 3.1.2.)
4. From the File menu, select Properties. This opens a properties dialog box.
5. Click on the Configure button of the dialog box. This opens a configuration dialog box.
6. Use the configuration dialog box to make these communications settings: 19,200 baud, 8 bits, no parity, 1 stop bit, and no flow control. Also set the correct communications port (for example, COM1). Click the OK button of the dialog box.

Operation

To use the Picobug debug monitor, merely enter commands at the prompt. **Table 3-1** explains these commands. To see a list of these commands on your computer screen, enter a question mark or the extra command he at the command prompt

Table 3-1 Picobug Commands

Command	Explanation
baud [*value*]	Set Baud Rate: • With optional *value* value, sets that rate (9600, 19200, or 38400). • Without any *value* value, sets default rate: 19200 baud.
br [*address*]	Breakpoint: • With optional *address* value, sets a new breakpoint at that address. • Without any *address* value, lists all current breakpoints.
g [*address*]	Go: • With optional *address* value, starts code execution from that address. • Without any *address* value, starts code execution from the current program-counter value. In either case, execution stops when it arrives at a breakpoint.
gr	Go to Return: Executes code from the current program-counter value to the return address of the calling routine. (Should execution arrive at a breakpoint before encountering the return address, execution stops at the breakpoint.)
gt *address*	Go to Address: Executes code from the current program-counter value to the specified *address* value. (Should execution arrive at a breakpoint before encountering the specified address, execution stops at the breakpoint.)
he	Help Displays available commands, identical to the ? command.
lo [*address*]	Download: • With optional *address* value, downloads a binary image to that address in SRAM. • Without any *address* value, downloads to SRAM an S-record text file.
md [*address1* [*address2*]] [;*size*]	Memory Display: • With optional *address1* and *address2* values, displays memory contents between the addresses. • With optional *address1* value, displays contents of 16 memory bytes. • With no address value, defaults to the last address viewed. • The optional *size* value specifies the format: **b** (bytes, the default), **h** (half words), **w** (words), or **i** (instructions).
mds [*address*]	Memory Display 256: • With optional *address* value, displays contents of 256 memory bytes, starting at that address. • With no address value, displays contents of 256 memory bytes, starting from the last address viewed.

Table 3-1 Picobug Commands (Continued)

Command	Explanation
mm [*address* [*value*]] [;*size*]	Modify Memory: • With optional *address* and *value* parameter values, assigns that value to the *address* location. • With optional *address* value but no *value* parameter value, prompts for a value for the *address* location, then prompts for a new value for the next location. To stop modification, enter a period instead of a new value. • With no optional *address* value, prompts for a value for the last address viewed, then prompts for a new value for the next location. To stop modification, enter a period instead of a new value. • The optional *size* value, specifies the format: **b** (bytes, the default), **h** (half words), **w** (words), or **i** (instructions).
nobr [*address*]	No Breakpoint: • With optional *address* value, removes the breakpoint from that address. • Without any *address* value, removes all the breakpoints.
reset	Reset: Resets the CPU and peripherals.
rd [*name*]	Register Display: • With optional *name* value, displays the value of that CPU register. • Without any *name* value, displays the values of all CPU registers.
rm *name value*	Register Modify: Assigns the *value* parameter value to the *name* CPU register.
t	Trace (Step): Single steps one instruction; identical to the `s` command.
s	Step (Trace): Single steps one instruction; identical to the `t` command.
?	Help Displays available commands, identical to the `he` command.

3.1.2 Picobug Sample Session

1. This sample session begins with the Picobug prompt:

```
picobug
```

2. To see the contents of all registers, enter the Register Display (rd) command without any name value:

```
picobug>     rd
```

   The system responds with a display such as this:

```
     pc 00400286   epc       00400286   fpc       0010a000
    psr 80000100   epsr      80000100   fpsr      00020000
ss0-ss4 bad0beef   20000c00  20008000   20010042  00000801   vbr       00405c00
  r0-r7 004027f8   00000050  0000ea60   00405f94  00406708   80070101  00000200  00000040
 r8-r15 0010a000   00020000  20000c00   004067c0  00000000   00405f94  10005000  00400286
```

3. To see the contents of a specific register, such as the epc register, enter the Register Display (rd) command *with* the name value:

```
picobug>  rd epc
```

   The system responds with a display such as this:

```
epc:      00400286
```

4. To see the contents of a specific memory location, enter the Memory Display (md) command with the location address. An optional size value (in this case w, for word) may be part of the command:

```
picobug>  md    0x00401000   ;w
```

   The system responds with a display such as this:

```
00401000:  8EF0B37E
```

5. To see the contents of a memory range, enter the Memory Display (md) command with the beginning and ending addresses. An optional size value (in this case b, for byte) may be part of the command:

```
picobug>  md    0x00400000   0x00400010   ;b
```

   The system responds with a display such as this:

```
00400000: 8E  F0  00  00  55  55  55  55  0E  22  9E  E8  03  20  0D  20  0E $..UUUU.  "…  . .
00400010: F7
```

6. To download into SRAM a program executable, in S-record format, enter the Download (lo) command without any address value:

```
picobug>  lo
```

   The system waits for you to send the program executable file. To do so, open the Transfer menu and select Send Text File. This opens a file-select dialog box. Use this dialog box to specify the appropriate S-record file, then click on the Open button. As soon as the download is complete (this may take several minutes), the Picobug prompt reappears:

```
picobug>
```

7. To see the new contents of registers, enter the Register Display (rd) command again, without any name value:

```
picobug>  rd
```

The system responds with an updated display, which shows that the pc register value reflects the start of the program just downloaded:

```
     pc 0040022a   epc       2d00108a   fpc       0010a000
    psr 80000000   epsr      80070101   fpsr      00020000
ss0-ss4 bad0beef   20000c00  20008000   20010042  00000801   vbr       00405c00
  r0-r7 bad0beef   00000050  00000000   d89f69ab  00405f20   80000000  00000200  00000024
 r8-r15 0010a000   004066b8  004067d7   00406948  00406714   00406708  004067c8  2d0001c4
```

8. To set a breakpoint at address 0x0040025C, enter this address as part of the Breakpoint (br) command:

```
picobug>  br    0x0040025c
```

   The Picobug prompt reappears, confirming that the system set the breakpoint:

```
picobug>
```

9. To see the list of breakpoints, enter the Breakpoint (br) command *without* any address value:

```
picobug>  br
```

   The system responds with the addresses of breakpoints, in this case only the breakpoint set in step 8:

```
0040025C
```

10. To start program execution, enter the Go (g) command:

```
picobug>  g
```

    In this instance, the breakpoint set during step 8 stops code execution. The system responds with this new display of register values:

```
  At breakpoint!!
     pc 0040025c   epc       0040025c   fpc       0010a000
    psr 80000100   epsr      80000100   fpsr      00020000
ss0-ss4 bad0beef   20000c00  20008000   20010042  00000801   vbr       00405c00
  r0-r7 004027f8   00000050  0000ea60   d89f69ab  00405f20   80000000  00000200  00000040
 r8-r15 0010a000   004066b8  004067d7   00406948  00406714   00406708  10005010  004002a2
```

11. To remove all breakpoints, enter the No Breakpoint (nobr) command, without any address value:

```
picobug>  nobr
```

    The Picobug prompt reappears, confirming that the system has removed the breakpoints:

```
picobug>
```

12. To see the list of breakpoints again, once more enter the Breakpoint (br) command without any address value:

```
picobug>  br
```

As there are no longer any breakpoints, the system responds with the Picobug prompt:

```
picobug>
```

13. To continue with this example session, enter another appropriate command. For example, to resume program execution, enter the Go (g) command.
14. To end your Picobug session, remove power from the EVB and close the terminal-emulation program.

3.1.3 Using the GNU Source-Level Debugger

The GNU source-level debugger is on the CD-ROM that comes with your CMB3401. This GNU software works with the Picobug monitor to provide source-level debugging for your code.

The CMB3401 software release guide gives the instructions for loading the GNU software, and for making any connections different from standalone Picobug connections. Make sure that the Picobug communications-speed setting is 19200 baud: this is the only communications speed for the GNU software.

3.2 Downloading to FLASH Memory

The Motorola SysDS Loader lets you program code into FLASH memory, upload FLASH contents to a PC file, verify that FLASH contents match those of a download file, display memory contents, erase FLASH memory, erase a sector of FLASH memory, or blank check a sector of FLASH memory.

3.2.1 Using the SysDS Loader

Follow these steps to use the Loader:

1. If you have not already installed the SysDS Loader onto your computer hard disk, do so. The CMB3401 product release guide includes installation instructions.
2. If the Hyperterminal emulation program is running, stop the program. (The SysDS Loader needs the same computer serial port that Hyperterminal uses.)
3. Set switch S2 for the Picobug monitor. The factory setting, for example, specifies Picobug: the GSB0 and GSB1 subswitches OFF, and the GSB2 subswitch ON.
4. Press switch S5 to reset the CMB3401.

5. Start the SysDS Loader. The main screen (**Figure 3-1**) appears.

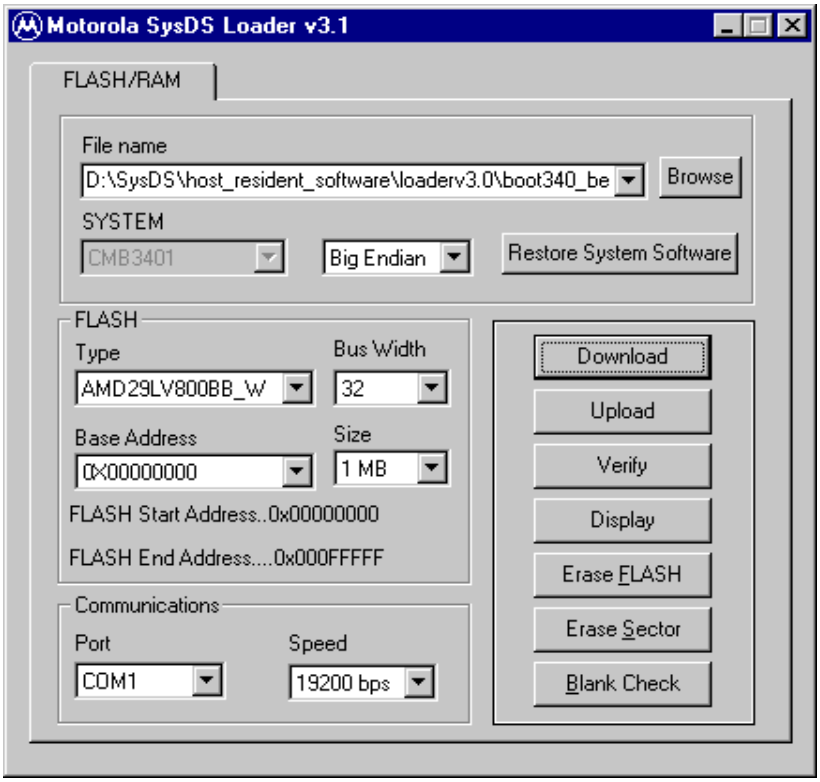


Figure 3-1 SysDS Loader Main Screen

6. Go to the File name field.
   - If you know the full pathname of the file to be programmed, enter the pathname in this field.
   - If you do not know the full pathname of the file to be programmed, click on the Browse button. This brings up a standard file-select dialog box: select the file and click on the OK button. This returns you to the FLASH/RAM page, entering the pathname in the File name field.
   - (If your only action for this Loader session will be uploading FLASH contents, you may leave the File name field blank.)
7. Make sure that the SYSTEM field shows the value CMB3401.
8. Use the Flash area to configure the FLASH type, bus width, and size.

   The value in the Base Address field is automatic, according to the entry in the SYSTEM field. (Optionally, you may select the value <CUSTOM>, which brings up the Custom Address dialog box. Enter an appropriate address, then click on the dialog box OK button to return to the main screen.)

9. In the Communications area, use the Port field to specify the PC serial port, and use the Speed field to specify the communications rate. (The default rate is 19200 baud.)
10. To program FLASH memory, click on the Download button. As the software downloads the file you specified, a progress message appears in a Status dialog box. A Download successful message appears at the end of downloading: you are ready to use the code in FLASH memory.
    - If this is the first action of this Loader session, the software downloads an algorithm file before downloading the file you specified. A progress message appears during the downloading of this algorithm file.
    - If the software cannot find the algorithm file, an appropriate error message identifies the file. Click on the message's OK button to bring up a file-select dialog box, then use this dialog box to specify the location of the algorithm file. If necessary, recopy the file from the transmittal CD-ROM. Click on the OK button to resume programming FLASH memory.
    - The error message Unable to Validate Flash configuration indicates some problem with the programming. A likely such problem is that the chip select base address does not correspond to the configured chip select. Correct the problem, then click again on the Program button.
11. To upload FLASH memory contents to a file in your PC, click on the Upload button. This brings up the Upload To File dialog box, **Figure 3-2**:

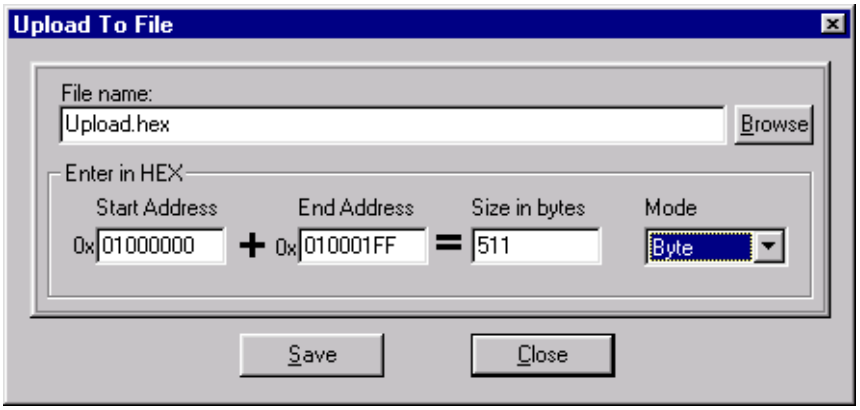


Figure 3-2 Upload To File Dialog Box

- Enter the name of the destination file. Optionally, click on the Browse button, to select a file via a standard file-select dialog box.
- The Start Address field indicates the start of CMB3401 FLASH memory. The default address value corresponds to the value of the SYSTEM field of the main screen FLASH/RAM page, but you may enter a different address, if appropriate.

- The Size in Bytes field value corresponds to the value of the Size field of the FLASH/RAM page. (If appropriate, you may enter a different value.)
- The system determines the value of the End Address field from the Start Address and Size in bytes values.
- The default Mode field value is Byte.
- When the Upload To File dialog box shows appropriate values, click on the Save button. A progress message appears during uploading.

NOTE: *The uploaded values do not include addresses or ASCII representations.*

12. To verify that the contents of Flash memory match the selected download file, click on the Verify button. A progress message appears as verification begins. A Verify successful message appears at the end of verification.
    - If this is the first action of this Loader session, the software downloads an algorithm file before verifying FLASH. A progress message appears during the downloading of this algorithm file. (Should the software be unable to find the algorithm file, an appropriate error message appears, as explained under the program FLASH memory step, above.)
    - If verification fails, an error message specifies the location that did not have the expected contents.
    - To recover from a verification failure, try downloading Flash again, to replace the selected download file.
13. To view the contents of Flash memory, click on the Display button. This brings up the Display Flash/Ram display, **Figure 3-3**:

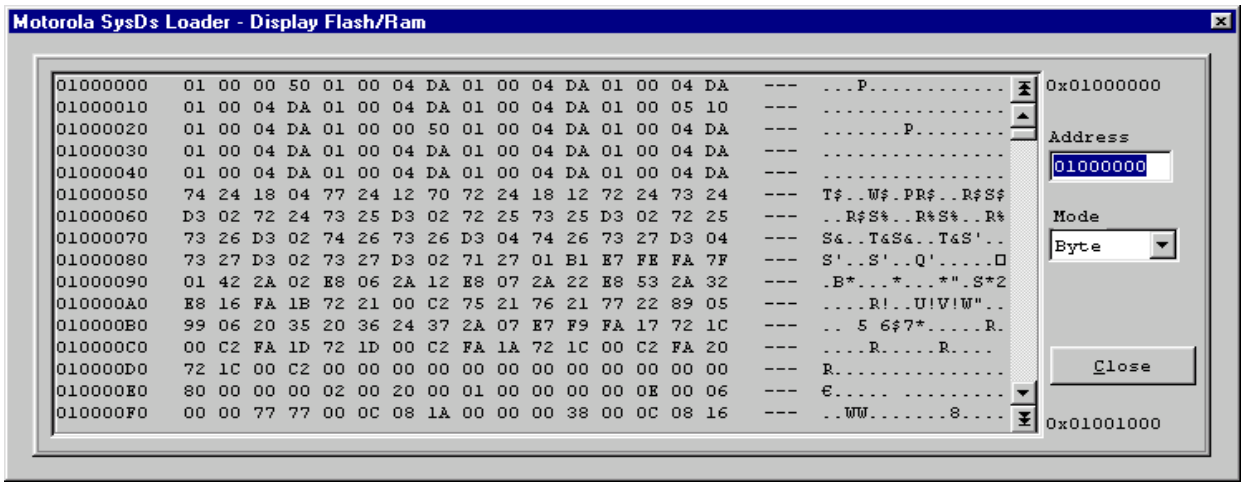


Figure 3-3 Display Flash/Ram Display

- If this is the first action of this Loader session, the software downloads an algorithm file before displaying FLASH contents. A progress message appears during the downloading of this algorithm file. (Should the software be unable to find the algorithm file, an appropriate error message appears, as explained under the program FLASH memory step, above.)
- The Address field shows the first address of the value display. One way to change the display is to enter a different address in this field.
- Another way to change the value display is to use the scroll bars.
- Use the Mode field to specify byte, half-word, or word values in the display.
- When you are done viewing the display, click on the Close button to return to the main screen.

14. To erase FLASH memory, click on the Erase FLASH button. The programmer erases all contents of the FLASH memory except for sectors that contain the system software. Erasing takes 20 to 30 seconds.

    If this is the first action of this Loader session, the software downloads an algorithm file before erasing FLASH. A progress message appears during the downloading of this algorithm file. (Should the software be unable to find the algorithm file, an appropriate error message appears, as explained under the program FLASH memory step, above.)

15. To erase a sector of FLASH memory, click on the Erase Sector button. This brings up the Flash Sector Number dialog box. Enter the number of the sector to be erased, then click on the OK button.

    - If this is the first action of this Loader session, the software downloads an algorithm file before erasing the FLASH sector. A progress message appears during the downloading of this algorithm file. (Should the software be unable to find the algorithm file, an appropriate error message appears, as explained under the program FLASH memory step, above.)
    - **For MLB Flash:** The system does not let you erase any of the sectors that contain system software. (**Table 2-8** shows these sectors.)
    - **For EIM Flash:** If you specify any of the sectors that contain system software, a message so reminds you. (**Table 2-10** shows these sectors.) Buttons of the message box let you cancel the erasure or proceed with the erasure.

NOTE: *Do not erase EIM system-software sectors, unless it is absolutely necessary. If you must erase such a sector, you subsequently can restore factory programming by following the instructions of subsection 3.2.2.*

16. To verify that a FLASH sector is blank, click on the Blank Check button. This brings up a dialog box that asks for a sector number. Enter the number of the sector to be blank checked, then click on the OK button.

    A message tells you the results of the blank check. (If the sector is not blank, you can erase the sector or try a different sector.)

17. To end your Loader session, merely close the main screen.

3.2.2 Restoring EIM System Software

If you must overwrite any of EIM FLASH sectors 0 through 7, you subsequently may use the SysDS Loader to restore SysDS software. To do so, follow these instructions:

1. Set the S1 BIG_E subswitch to ON.
2. Press switch S5, to reset the CMB3401.
3. Look at the SysDS Loader main screen (**Figure 3-1**). Between the SYSTEM field and the Restore System Software button there is an untitled field. Make sure that the value of this untitled field is Big Endian.
4. In the FLASH area of the main screen, set the value of the Type field to INTEL....
5. Click on the Restore System Software button of the main screen.
    - If the system software is in your current hard-disk directory, the Loader automatically restores factory programming to EIM FLASH sectors 0 through 7. The main screen reappears to confirm successful programming.
    - If you receive a message that the system software does not exist, it may be because the software is in a different hard-disk directory. If so, make that directory the active one and click again on the Restore System Software button.

3.3 Controlling CMB3401 LEDs

Section 2 explained how LEDs DS3 through DS6 flicker as part of the CMB3401 self-test. Your own code also can control these LEDs, by assigning values to the four least-significant bits of the global control register (GCR):

- GCR bit 0 controls LED DS3 (GCB0).
- GCR bit 1 controls LED DS4 (GCB1).
- GCR bit 2 controls LED DS5 (GCB2).
- GCR bit 3 controls LED DS6 (GCB3).

The value 0 in any of these bits turns ON the corresponding LED. The value 1 in any of these bits turns OFF the corresponding LED.

The example assembly routine below writes the value 0x0000_1110 to the GCR. This writes 0 to bit 0 and 1 to the other three bits. Accordingly, this routine turns LED DS3 ON, and turns off the other LEDs

```
turn_on_led_0:
      lrw     r2, 0x0E      //LED DS3 or GCB0
      mtcr    r2, GCR
      rts
```

Section 4 Using the FPGA Device

This section explains how to use the Altera Max+plusII software to reprogram the FPGA device at CMB3401 location U2. Additionally, this section explains how to use the periodic interval timers (PITs) of the FPGA device.

4.1 Configuring Your Software

NOTE: *The steps below are guidance for starting to use Altera MAX+plusII software. Should you have difficulty preparing your MAX+plusII software, phone Altera customer service for assistance.*

You must prepare your Altera development software before you can for use it with your CMB3401. Follow the Altera instructions to:

1. Install the Altera development software.
2. Obtain and install your Verilog authorization code file. (If you FAX registration information, this takes only a few hours; Altera customer service can provide the FAX number.)
3. Start the Altera software.
4. Start a project.

This completes software preparation. You are ready to develop an application suitable for downloading to the CMB3401 U2 device.

4.2 Reprogramming the FPGA Device

Follow steps 1 through 29, below, to develop an application suitable for downloading to the FPGA device at location U2. Most of these steps are typical for using the MAX+plusII software to develop any new application project. These steps are *not* rigid instructions. In case of difficulty using the MAX+plusII software, you should call Altera customer service for assistance.

The transmittal CD-ROM that contains this manual also contains example application files: a symbol counter, a Verilog counter, and a Verilog port.

1. Use Windows Explorer to create and name a new folder for the project.
2. Start the MAX+plusII software.
3. Open the File menu and select Project. From the subordinate menu, select Name. This brings up the Project Name dialog box.

4. Use the Project Name dialog box to select the newly created project folder, and to enter a name for the project. (The project name should not contain any spaces; usually it is convenient to give the project the same name as the folder.) Click on the OK button to close the dialog box.
5. Open the File menu and select New. This brings up the New dialog box. Select Text editor file, then click on the OK button. This closes the dialog box and opens the text editor window.
6. Write the Verilog code for your application. (Consult the Altera Verilog manuals for instructions.)
7. When your code is done, leave the text editor window open. Click on the Open Compiler Window toolbar button. The software immediately compiles your code.
8. If the compiler finds errors, correct them in the text editor window, then compile again. When compilation succeeds, your are ready to create a default symbol.
9. Still leaving the text editor window open, open the File menu and select Create Default Symbol. The software automatically creates a graphic representation of the compiled code, a symbol that you later can use in a schematic design.
10. Open the File menu and select Project. From the subordinate menu, select Name. This brings up the Project Name dialog box.
11. Use the Project Name dialog box to select the same folder you selected in Step 4. Enter a new project name: as this project will be for a .hex file, Motorola suggests that you append the letter h to the name you used in Step 4. Click on the OK button to close the dialog box.
12. Open the File menu and select New. This brings up the New dialog box. Select graphic editor file, then click on the OK button. This closes the dialog box and opens the graphic editor window.
13. Open the Symbol menu and select Enter. This brings up the Enter Symbol dialog box. Select the symbol you created in Step 9. The symbol appears in the graphic editor window.
14. Add all the inputs and outputs to the symbol, then compile again. (The only errors likely at this point are mismatched signal names or a forgotten signal. Correct any errors and recompile.) When compilation succeeds, you are ready to assign a device.
15. Open the Assign menu and select Device. This brings up the Device dialog box. In the Device Family area, select FLEX 10KA. In the Devices area, select EPF10K100ABC600-1. Click on the Device Options button, to bring up the Individual Device Options dialog box.

16. In the Individual Device Options dialog box:
    a. Go to the Device Options area. Click to check these items:
        - Release Clears Before Tri-States,
        - Enable Chip-Wide Reset (DEV_CLRn),
        - Enable Chip-Wide Output Enable (DEV_OE),
        - Enable INIT_DONE Output,
        - Use Low-Voltage Configuration Device, and
        - Use Configuration Device Pull-Up Resistor.
    b. Find the Configuration Device field: set the field value to be EPC2LC20.
    c. Elsewhere in the Individual Device Options dialog box, find the Configuration Scheme field: set the field value to be Passive Serial (can use Configuration Device).
    d. In the Not Affected By Configuration Scheme area, make sure that both CLKUSR boxes have grey check marks.
    e. Make sure that the only check marks are those that Steps a through d specify, then click on the OK button to return to the Device dialog box.
17. Click on the Device dialog box OK button to return to the main screen. This completes device assignment. You are ready to assign signals to pins.
18. It is best to give signals (wires) the same names as their corresponding pins. Open the Assign menu and select Pin/Location/Chip. This brings up the Pin/Location/Chip dialog box.
    a. Click on the Search button to bring up a subordinate dialog box that lists the pins. (Click on the LIST button to see the list.) The listed pin names are the inputs and outputs you created as part of Step 14.)
    b. Select (highlight) a pin, then click on the OK button. This returns you to the Pin/Location/Chip dialog box; the selected pin name will be in the Node Name field.
    c. Go to the Chip Resource area of the dialog box. In the Pin field, enter the name of the FPGA pin. (This is the value in the U2 column of cross-reference table 6-5.)

NOTE: *An alternative to using the Search button is to select Pin, activating the Pin Type field. Select the appropriate type from the small pull-down menu, then enter the pin name in the appropriate field, and enter the signal name in the Node Name field.*

This completes assignment for the first pin.

19. Repeat Step 18 for all other signals. When you are done, close the Pin/Location/Chip dialog box.

NOTE: *For each finished design, the Altera software creates a .acf file: a text file that you can edit. For your first design, you must do Step 18 for each signal. But for subsequent designs, you can copy and edit a .acf file.*

20. If you have not already done so, configure ByteBlaster programming hardware per Altera instructions.
21. Compile your application file again. When compilation succeeds, you are ready to create a .pof file.
22. Open the MAX+plusII menu and select Programmer. This brings up the Programmer dialog box. (You will not do anything in this dialog box, but it must be open at this point.)
23. Open the JTAG menu and select Multi-Device JTAG Chain Setup. This brings up the Multi-Device JTAG Chain Setup dialog box.
    a. Use the Device Name pull-down menu to select EPF10K100A, then click on the Add button. The EPF10K100A name appears in the list at the center of the dialog box, but without any associated programming.
    b. Use the Device Name pull-down menu to select EPC2, then click on the ADD button. The EPC2 name appears in the list at the center of the dialog box, but without any associated programming.
    c. Use the Device Name pull-down menu to select EPC2 a second time, then click on the Select Programming File button. This brings up the Select Programming File dialog box.
    d. Use the Select Programming File dialog box to select the .pof file for your project. Click on the OK button to return to the Multi-Device JTAG Chain Setup dialog box.
    e. Click on the Add button. This again adds the EPC2 name to the list at the center of the dialog box, but shows the association with the selected .pof file.
    f. Click on the Save JCF button. This brings up a subordinate dialog box that lets you name and save the listed files as a JTAG chain file. Click on the OK button to return to the Multi-Device JTAG Chain Setup dialog box.
    g. Click on the Multi-Device JTAG Chain Setup dialog box OK button to return to the Programmer dialog box. This completes file creation; you are ready to download the files to the FPGA device.
24. Apply power to your CMB3401.
25. Connect the ByteBlaster between CMB3401 connector J14 and a parallel port of your computer. Make sure that the red wire of the cable connects to J14 pin 1.

26. Click on the Program button of the Programmer dialog box. A percentage indicator shows the progress of downloading the files to a ROM device of the CMB3401.
27. At the end of this downloading, disconnect the ByteBlaster cable from connector J14.
28. Press CMB3401 switch S7 to transfer the downloaded application to the U2 FPGA device.
29. This completes reprogramming of the U2 device. You may close the MAX+plusII software.

4.3 Using the Periodic Interval Timers

The FPGA device at location U2 includes two periodic interval timers (PITs), which can provide precise interrupts with minimal processor intervention. Each timer can either count down from a modulus-latch value or be a free-running down counter. **Figure 4-1** is a diagram of such a PIT.

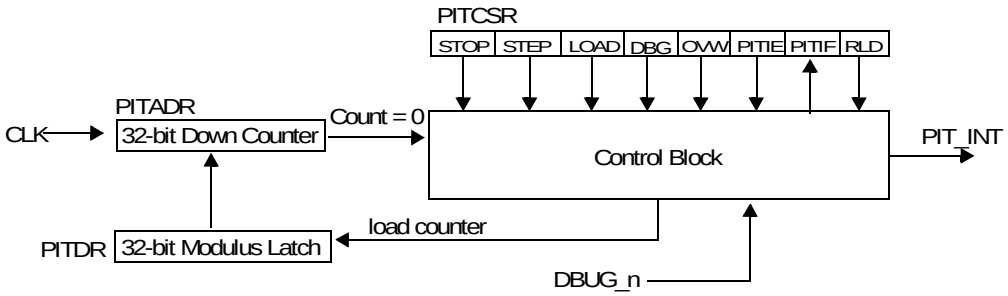


Figure 4-1 PIT Diagram

Each PIT consists of a control block and three registers:

- **PIT data register (PITDR)**, which contains the timer modulus. Your code can set this modulus by writing to this register. Your code can find the modulus by reading from this register.
- **PIT alternate data register (PITADR)**, which contains the current counter value. Your code can find the current timer value by reading from this register.
- **PIT control/status register (PITCSR)**, which controls timer operation. Your code can control the timer by writing to or reading bits 8 through 1 of this register.

Each PIT has a PIT_INT output signal, which connects to a processor interrupt. When the PIT count reaches 0, the control block sets the interrupt flag bit (PITIF, bit 2) of the control/status register. This asserts the PIT_INT signal, which can alert the processor that an interrupt is pending. The signal remains asserted until the control block (or your code) clears the flag bit.

The PIT_INT signal for PIT1 connects to `p_int_b[9]` of the processor. The PIT_INT signal for PIT2 connects to `p_int_b[11]` of the processor.

To use a PIT as a set-and-forget timer, your code must set the reload control bit (RLD, bit 1) of the control/status register. Then, your code must write the appropriate modulus latch value to the data register.

- The alternate data register copies the modulus latch value from the data register.
- Upon each system clock cycle, the alternate data register decrements its value by 1.
- When the counter value reaches 0 (0x0000_0000), the control block sets the interrupt flag bit (PITIF) of the control/status register. If the interrupt enable bit (PITIE, bit 3) also is set, the PIT_INT interrupt-pending signal goes to the processor. The alternate data register again copies the modulus latch value from the data register, and the counting cycle begins again.
- Your code may change the modulus latch value at any time, by writing to the data register.
- To force the count to 0 immediately, your code must set the overwrite enable bit (OVW, bit 3) of the control/status register, then write the value 0 to the data register.

To use a PIT as a free-running timer, your code must clear the reload control bit (RLD, bit 1) of the control/status register. This tells the alternate address register to ignore the modulus latch value in the data register.

- Upon each system clock cycle, the alternate data register decrements its value by 1.
- When the counter value reaches 0, the control block sets the interrupt flag bit (PITIF) of the control/status register. If the interrupt enable bit (PITIE, bit 3) also is set, the PIT_INT interrupt-pending signal goes to the processor.
- At the next clock cycle, the alternate data register decrements its value to 0xFFFF_FFFF, and the counting cycle begins again.
- To force the count to 0 immediately, your code must set the overwrite enable bit (OVW, bit 3) of the control/status register, then write the value 0 to the data register.
- Your code may change the modulus latch value at any time, by writing to the data register. However, the timer will ignore the modulus latch value as long as the RLD bit is set.

Table 4-1 lists the register addresses for both PITs. **Figure 4-2** shows the layout of the control/status register. **Table 4-2** explains the control bits of the control/status register.

Table 4-1 PIT Register Addresses

Registers	PIT1 Addresses	PIT2 Addresses
Control/Status Register (PITCSR)	0x0021_1124	0x0021_2224
Data Register (PITDR)	0x0021_1128	0x0021_2228
Alternate Data Register (PITADR)	0x0021_112C	0x0021_222C

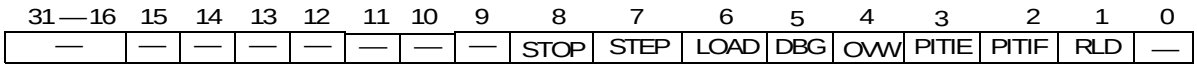


Figure 4-2 PIT Control/Status Register Layout

Table 4-2 Control/Status Register Bit Values

Bit	Bit Name	Bit Value	Effect/Meaning
1	Reload Control (RLD)	0	After reaching 0x0000_0000, the counter decrements to 0xFFFF_FFFF and continues counting down.
		1	After reaching 0x0000_0000, the counter loads the modulus latch value and continues counting down.
2	PIT Interrupt Flag (PITIF)	0	No interrupt is present. (A write of 0 to this bit unasserts an interrupt signal. A write to the data register also clears this bit.)
		1	A PIT interrupt is present. (A write of 1 has no effect.)
3	PIT Interrupt Enable (PITIE)	0	Prevents any interrupt signal from reaching the interrupt controller.
		1	Allows any interrupt signal to reach the interrupt controller.
4	Overwrite Enable (OVW)	0	The data register holds the modulus latch value. When the count in the alternate data register reaches 0, the alternate data register reads the modulus latch value.
		1	The alternate data register immediately reads the modulus latch value from the data register, regardless of the current count value.
5	Debug Mode (DBG)	0	Counter functionality continues while in debug mode.
		1	Debug mode freezes the counter.
6	Load Counter (LOAD)	0	None. (A read of this bit always returns 0; a write of 0 has no effect.)
		1	Copies the modulus latch value from the data register to the alternate data register. (Hardware automatically clears this bit after loading.)
7	Step Counter (STEP)	0	None. (A read of this bit always returns 0; a write of 0 has no effect.)
		1	Steps the counter by one clock cycle. (Hardware automatically clears this bit after stepping. A write of 1 has no effect if the counter is not stopped.)
8	Stop Counter (STOP)	0	Starts counting. Stepping is not possible.
		1	Stops (freezes the counter. Stepping is possible.

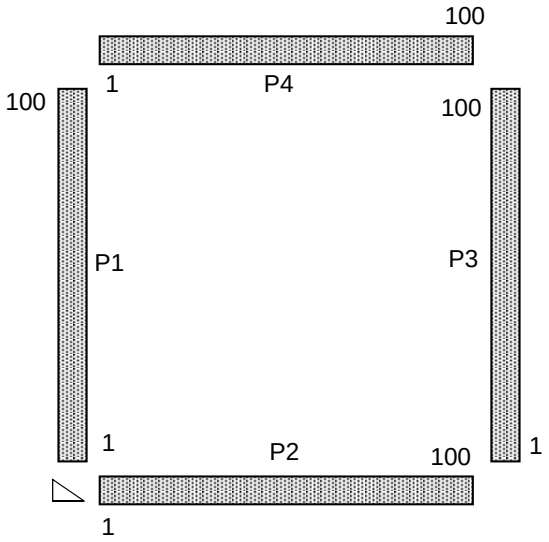
NOTE: *Your code may not step a PIT counter from 1 to 0, nor may it step a PIT counter from 0 to the modulus latch value. Setting the counter value to 0 directly does not cause a PIT interrupt.*

Section 5 Connector Information

This section consists of pin assignments and signal descriptions for CMB3401 connectors.

5.1 MAPI Connectors (P1/J1, P2/J2, P3/J3, P4/J4)

Connectors P1 through P4, all 2-by-50-pin connectors, are the CMB3401 MAPI connectors. (Connectors J1 through J4, on the bottom of the CMB3401, have the same pin assignments.) The diagram below shows the orientation of the CMB3401 MAPI connectors. **Figure 5-1** through **Figure 5-4**, and **Table 5-1** through **Table 5-4**, give the pin assignments and signal descriptions for these connectors.



Connector Information

P1/J1

VPP1	100	99	VDD3V
CS8_b	98	97	CS4_b
CS9_b	96	95	CS5_b
PTJ1[94]	94	93	CS6_b
GND	92	91	CS7_b
VDD5V	90	89	GND
PTJ1[88]	88	87	PTJ1[87]
PTJ1[86]	86	85	PTJ1[85]
PTJ1[84]	84	83	PTJ1[83]
PTJ1[82]	82	81	PTJ1[81]
PTJ1[80]	80	79	PTJ1[79]
PTJ1[78]	78	77	PTJ1[77]
PTJ1[76]	76	75	PTJ1[75]
GND	74	73	DVSP_b[0]
PTJ1[72]	72	71	GND
PTJ1[70]	70	69	RS11
PTJ1[68]	68	67	PTJ1[67]
PTJ1[66]	66	65	PTJ1[65]
PTJ1[64]	64	63	PTJ1[63]
PTJ1[62]	62	61	PTJ1[61]
VDD3V	60	59	VDD3V
MID0	58	57	PTJ1[57]
INT_b[14]	56	55	PTJ1[55]
INT_b[12]	54	53	PTJ1[53]
MID1	52	51	PTJ1[51]
INT_b[10]	50	49	PTJ1[49]
INT_b[8]	48	47	PTJ1[47]
GND	46	45	PTJ1[45]
PTJ1[44]	44	43	PTJ1[43]
PTJ1[42]	42	41	GND
MID2	40	39	PTJ1[39]
PTJ1[38]	38	37	PTJ1[37]
PTJ1[36]	36	35	PTJ1[35]
MID3	34	33	PTJ1[33]
PTJ1[32]	32	31	PTJ1[31]
PTJ1[30]	30	29	PTJ1[29]
GND	28	27	PTJ1[27]
GND1	26	25	PTJ1[25]
PTJ1[24]	24	23	PTJ1[23]
PTJ1[22]	22	21	PTJ1[21]
PTJ1[20]	20	19	PTJ1[19]
PTJ1[18]	18	17	PTJ1[17]
PTJ1[16]	16	15	PTJ1[15]
PTJ1[14]	14	13	PTJ1[13]
GND1	12	11	PTJ1[11]
GND2	10	9	PTJ1[9]
PTJ18]	8	7	PTJ1[7]
PTJ1[6]	6	5	PTJ1[5]
PTJ1[4]	4	3	PTJ1[3]
GND2	2	1	PTJ1[1]

Figure 5-1 MAPI Connector P1/J1 Pin Assignments

Table 5-1 MAPI Connector P1/J1 Signal Descriptions

Pin	Mnemonic	Signal
100	VPP1	Programming Voltage
99, 60, 59,	VDD3V	+3.2-volt power.
98—95, 93, 91	CS9_b — CS4_b (not in exact order)	CHIP SELECTS (lines 9—4) – Active-low output lines that provide chip selects to external devices.
94, 88—75, 72, 70, 68—61, 57, 55, 53, 51, 49, 47, 45—42, 39—35, 33—29, 27, 25—13, 11, 9—3, 1	PTJ1[x]	Pass Through
92, 89, 74, 71, 46, 41, 28	GND	GROUND
90	VDD5V	+5-volt power.
73	DVSP_b[0]	DATA BUS DENY — Active-low signal that requests non-CMB devices to not access the data bus.
69	RS11	Reserved.
58, 52, 40, 34	MID0 — MID3	IDENTIFICATION CODE (lines 0—3) — Signals that identify the host processor board.
56, 54, 50, 48	INT_b[14], INT_b[12], INT_b[10], INT_b[8]	EXTRNAL INTERRUPTS (lines 14, 12, 10, 8) – Bidirectional interrupt lines that form the external interface to the general-purpose I/O module.
26, 12	GND1	GROUND — Connection to the Ground 1 plane.
10, 2	GND2	GROUND — Connection to the Ground 2 plane.

Connector Information

P2/J2

PTJ2[100]	100	99	GND3	
PTJ2[98]	98	97	PTJ2[97]	
PTJ2[96]	96	95	PTJ2[95]	
PTJ2[94]	94	93	PTJ2[93]	
PTJ2[92]	92	91	PTJ2[91]	
PTJ2[90]	90	89	PTJ2[89]	
PTJ2[88]	88	87	PTJ2[87]	
PTJ2[86]	86	85	GND3	
GND	84	83	GND	
VDD3V	82	81	VPP3	
PTJ2[80]	80	79	VDD5V	
PTJ2[78]	78	77	PTJ2[77]	
PTJ2[76]	76	75	PTJ2[75]	
PTJ2[74]	74	73	PTJ2[73]	
PTJ2[72]	72	71	PTJ2[71]	
PTJ2[70]	70	69	PTJ2[69]	
PTJ2[68]	68	67	PTJ2[67]	
PTJ2[66]	66	65	PTJ2[65]	
GND	64	63	GND	
VDD3V	62	61	PTJ2[61]	
RS56	60	59	RS57	
RS54	58	57	RS55	
RS52	56	55	RS53	
RS50	54	53	RS51	
RS48	52	51	RS49	
RS46	50	49	RS47	
RS44	48	47	RS45	
GND	46	45	VDD5V	
VDD3V	44	43	GND	
RS40	42	41	RS41	
RS38	40	39	RS39	
RS36	38	37	RS37	
RS34	36	35	RS35	
RS32	34	33	RS33	
RS30	32	31	SDCPS	
RS28	30	29	VDD5V	
PTJ2[28]	28	27	PTJ2[27]	
PTJ2[26]	26	25	PTJ2[25]	
PTJ2[24]	24	23	PTJ2[23]	
PTJ2[22]	22	21	PTJ2[21]	
VDD3V	20	19	VPP2	
GND	18	17	GND	
PTJ2[16]	16	15	GND2	
PTJ2[14]	14	13	PTJ2[13]	
PTJ2[12]	12	11	PTJ2[11]	
PTJ2[10]	10	9	PTJ2[9]	
PTJ2[8]	8	7	PTJ2[7]	
PTJ2[6]	6	5	PTJ2[5]	
PTJ2[4]	4	3	PTJ2[3]	
PTJ2[2]	2	1	GND2	

Figure 5-2 MAPI Connector P2/J2 Pin Assignments

Table 5-2 MAPI Connector P2/J2 Signal Descriptions

Pin	Mnemonic	Signal
100, 98—86, 80, 78—65, 61, 28—21, 16, 14—2	PTJ2[x]	Pass Through
99, 85	GND3	GROUND — Connection to the Ground 3 plane.
84, 83, 64, 63, 46, 43, 18, 17	GND	GROUND
82, 62, 44, 20	VDD3V	+3.2-volt power.
81	VPP3	Programming Voltage.
79, 45, 29	VDD5V	+5-volt power.
60—47, 40—32, 30	RS57—RS44, RS41—RS32, RS30, RS28 (not in exact order)	Reserved.
31	SDCPS	SHUT DOWN CMB POWER SUPPLY — Signal, from a connected board that supplies power, to disable the on-board CMB power supply.
19	VPP2	Programming Voltage.
15, 1	GND2	GROUND — Connection to the Ground 2 plane.

P3/J3

VDD3V	100	99	VDD3V
VPP4	98	97	GND
CLKCTL	96	95	GND
FREZ_b	94	93	EXTAL
TSC	92	91	GND
GPIO	90	89	GPIOSO
SIZ1_b	88	87	J_TRST_B
SIZ0_b	86	85	J_TCLK
DE_b	84	83	J_TMS
J_TDI	82	81	GND
J_TDO	80	79	RSTOUT_B
VSTBY	78	77	RIN_b
IDVDD	76	75	SHS
VDD5V	74	73	PSTAT3
TBUSY_B	72	71	PSTAT2
RS67	70	69	PSTAT1
GND	68	67	PSTAT0
MAPI_TC2	66	65	GND
MAPI_TC1	64	63	MID9
MAPI_TC0	62	61	MID8
VDD3V	60	59	VDD3V
PTJ3[58]	58	57	PTJ3[57]
PTJ3[56]	56	55	MID4
BKREQ_b	54	53	PTJ3[53]
LPMD0	52	51	PTJ3[51]
LPMD1	50	49	MID5
ABORT_b	48	47	PTJ3[47]
PTJ3[46]	46	45	PTJ3[45]
PTJ3[44]	44	43	GND
PTJ3[42]	42	41	PTJ3[41]
PTJ3[40]	40	39	PTJ3[39]
PTJ3[38]	38	37	MID6
PTJ3[36]	36	35	PTJ3[35]
PTJ3[34]	34	33	PTJ3[33]
PTJ3[32]	32	31	MID7
PTJ3[30]	30	29	PTJ3[29]
PTJ3[28]	28	27	PTJ3[27]
PTJ3[26]	26	25	GND
PTJ3[24]	24	23	GND4
PTJ3[22]	22	21	PTJ3[21]
PTJ3[20]	20	19	PTJ3[19]
PTJ3[18]	18	17	PTJ3[17]
PTJ3[16]	16	15	PTJ3[15]
PTJ3[14]	14	13	PTJ3[13]
PTJ3[12]	12	11	PTJ3[11]
PTJ3[10]	10	9	GND4
PTJ3[8]	8	7	GND3
PTJ3[6]	6	5	PTJ3[5]
PTJ3[4]	4	3	PTJ3[3]
PTJ3[2]	2	1	GND3

Figure 5-3 MAPI Connector P3/J3 Pin Assignments

Table 5-3 MAPI Connector P3/J3 Signal Descriptions

Pin	Mnemonic	Signal
100, 99, 60, 59,	VDD3V	+3.2-volt power
98	VPP4	Programming Voltage
97, 95, 91, 81, 68, 65, 43, 25	GND	GROUND
96	CLKCTL	CLOCK CONTROL — Clock control signal for the MCU clock.
94	FREZ_b	DEBUG MODE INDICATOR — Active-low signal indicating that the processor is in debug mode.
93	EXTAL	EXTERNAL CLOCK — Off-board clock signal.
92	TSC	TRI-STATE CONTROL — Signal that puts the processor in tri-state mode.
90	GPIO	GENERAL PURPOSE INPUT/OUTPUT — General-purpose I/O signal.
89	GPIOSO	GENERAL PURPOSE SERIAL OUTPUT — General-purpose I/O serial output signal.
88, 86	SIZ1_b], SIZ0_b	TRANSFER SIZE (lines 1, 0) — Signals that indicate the size of an external transfer.
87	J_TRST_b	TEST RESET – Active-low input signal to the Schmitt trigger, asynchronously initializing the test controller. The J_TRST_b pin has an internal 47k pullup resistor.
85	J_TCLK	TEST CLOCK – Input signal that synchronizes the JTAG test logic. The TCK pin has an internal 47k pullup resistor.
84	DE_b	DEBUG EVENT – Open-drain, active-low debug signal. If an input signal from an external command controller, causes the EVB to enter debug mode. If an output signal, acknowledges that the MCU is in debug mode.
83	J_TMS	TEST MODE SELECT – Input signal that sequences the test controller's state machine, sampled on the rising edge of the TCK signal. The TMS pin has an internal 47k pullup resistor.
82	J_TDI	TEST DATA INPUT – Serial input signal for test instructions and data, sampled on the rising edge of the TCK signal. The TDI pin has an internal 47k pullup resistor.
80	J_TDO	TEST DATA OUTPUT – Serial output signal for test instructions and data. Three-stateable and actively driven in the Shift-IR and Shift-DR controller states, this signal changes on the falling edge of the TCK signal.
79	RSTOUT_b	RESET OUT – Active-low output signal that resets external components. Activation of any internal reset sources asserts this line.
78	VSTBY	STANDBY VOLTAGE — Standby power for M340 on-board RAM.
77	RIN_b	RESET IN – Active-low input signal that starts a system reset: a reset of the PowerStrike device and most peripherals. This signal does not affect the debug module (which the system provides via the TRST* line).
76	IDVDD	IDENTIFICATION POWER — Special 3-volt power signal for the MAPI identification code (MID) signals.
75	SHS	SHOW CYCLE STROBE — Active-low signal output signal indicating that address and data are valid for show cycles.
74	VDD5V	OPERATING VOLTAGE – Transmission line for +5-volt CMB3401 input power.
73, 71, 69, 67	PSTAT3 — PSTAT0	PROCESSOR STATUS (lines 3—0) — Output signals that provide external status indications for the resident MCU.
72	TBUSY_b	BUSY — Signal indicating that a bus cycle is in progress.

Table 5-3 MAPI Connector P3/J3 Signal Descriptions (Continued)

Pin	Mnemonic	Signal
70	RS67	Reserved.
66, 64, 62	MAPI_TC2 — MAPI_TC0	TRANSFER CODE (lines 2—0) — Signals indicating the general type of transfer.
63, 61, 55, 49, 37, 31	MID9 — MID4 (not in exact order)	IDENTIFICATION CODE (lines 9—4) — Signals that identify the host processor board.
58—56, 53, 51, 47—44, 42—38, 36—32, 30—26, 24, 22—10, 8, 6—2	PTJ3[x]	Pass Through.
54	BKREQ_b	BREAKPOINT REQUEST — Active-low signal that requests a hardware breakpoint.
52, 50	LPMD0, LPMD1	LOW POWER MODE (lines 0, 1) — Signals asserted by the processor upon execution of a doze, stop, or wait instruction. Optionally, an external source can assert these signals to put the CMB in its low-power stopped state.
48	ABORT_b	TRANSFER ABORT — Active-low signal from the processor that a requested access must be aborted.
23, 9	GND4	GROUND — Connection to the Ground 4 plane.
7, 1	GND3	GROUND — Connection to the Ground 3 plane.

P4/J4

VDD5V	100	• •	99	VDD3V
CSE1	98	• •	97	GND
GND	96	• •	95	CLKOUT
CSE0	94	• •	93	GND
RS73	92	• •	91	CS_b[3]
RS72	90	• •	89	CS_b[2]
OE_b[0]	88	• •	87	CS_b[1]
EBD_b	86	• •	85	CS_b[0]
EBC_b	84	• •	83	GND
EBA_b	82	• •	81	R_W_b
EBB_b	80	• •	79	TREQ_b
TEA_b	78	• •	77	MAPI_TA_b
GND	76	• •	75	GND
A30	74	• •	73	A31
A28	72	• •	71	A29
A26	70	• •	69	A27
A24	68	• •	67	A25
A22	66	• •	65	A23
A20	64	• •	63	A21
A18	62	• •	61	A19
A16	60	• •	59	A17
GND	58	• •	57	GND
A14	56	• •	55	A15
A12	54	• •	53	A13
A10	52	• •	51	A11
A8	50	• •	49	A9
A6	48	• •	47	A7
A4	46	• •	45	A5
A2	44	• •	43	A3
A0	42	• •	41	A1
GND	40	• •	39	GND
D30	38	• •	37	D31
D28	36	• •	35	D29
D26	34	• •	33	D27
D24	32	• •	31	D25
D22	30	• •	29	D23
GND	28	• •	27	GND
D20	26	• •	25	D21
D18	24	• •	23	D19
D16	22	• •	21	D17
D14	20	• •	19	D15
D12	18	• •	17	D13
GND	16	• •	15	GND
D10	14	• •	13	D11
D8	12	• •	11	D9
D6	10	• •	9	D7
D4	8	• •	7	D5
D2	6	• •	5	D3
D0	4	• •	3	D1
VDD3V	2	• •	1	VDD3V

Figure 5-4 MAPI Connector P4/J4 Pin Assignments

Table 5-4 MAPI Connector P4/J4 Signal Descriptions

Pin	Mnemonic	Signal
100	VDD5V	+5-volt power.
99, 2, 1	VDD3V	+3.2-volt power.
98, 94	CSE1, CSE0	CHIP SELECT, EMULATION (lines 1, 0) — Emulation chip select signals.
97, 96, 93, 83, 76, 75, 58, 57, 40, 39, 28, 27, 16, 15	GND	GROUND
95	CLKOUT	CLOCK OUTPUT – An external clock source from the processor.
92, 90	RS73, RS72	Reserved.
91, 89, 87, 85	CS_b[3] – CS_B[0]	FPGA CHIP SELECTS (lines 3–0) – Active-low output lines that provide chip selects to external devices. These signals are driven from the FPGA MLB address space.
88	OE_b[0]	FPGA OUTPUT ENABLE – Active-low signal that indicates that a bus access is a read access; enables slave devices to drive the data bus. This signal is driven from the FPGA MLB address space.
86, 84, 82, 80	EBD_b, EBC_b, EBA_b, EBB_b	FPGA ENABLE BYTES D, C, A, B – Active-low outputs active during an operation to corresponding data bits (D31—D24 for enable byte D, D23—D16 for enable byte C, D7–D0 for enable byte B, D15–D8 for enable byte A). (These signals are driven from the FPGA MLB address space.) You can configure these bytes to assert for write cycles or for both read and write cycles.
81	R_W_b	FPGA READ/WRITE ENABLE – Active-low signal that indicates whether the current bus access is a read access or write access. This signal is driven from the FPGA MLB address space.
79	TREQ_b	FPGA TRANSMIT REQUEST — Active-low signal indicating a new access request. The resident MCU drives this signal. This signal is driven from the FPGA MLB address space.
78	TEA_b	TRANSFER ERROR ACKNOWLEDGE — Active-low I/O signal that indicates a bus transfer error.
77	MAPI_TA_b	FPGA TRANSFER ACKNOWLEDGE — Active-low I/O signal indicating completion of a data transfer, for either a read or a write cycle. This signal is driven from the FPGA MLB address space.
74—59, 56—41	A31 — A0 (not in exact order)	ADDRESS BUS (lines 31—0) – Output lines for addressing external devices. These lines change state only during external-memory accesses.
38—29, 26—17, 14—3	D31 — D0 (not in exact order)	DATA BUS (lines 31–0) – Bi-directional data lines for accessing external memory. A hardware reset or no external-bus activity hods these lines in their previous logic state.

5.2 OnCE Connector (J13)

Connector J13, a 2-by-7-pin connector, conveys data and control signals to and from the OnCE control block. **Figure 5-5** and **Table 5-5** give the pin assignments and signal descriptions for this connector.

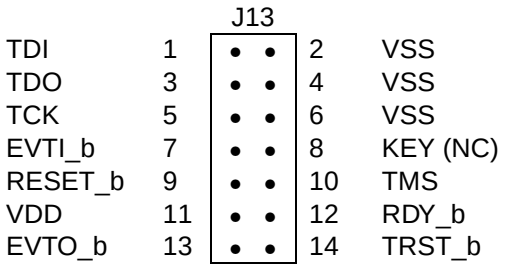


Figure 5-5 OnCE Connector J13 Pin Assignments

Table 5-5 OnCE Connector J13 Signal Descriptions

Pin	Mnemonic	Signal
1	TDI	TEST DATA INPUT – Data and command serial input line to the OnCE controller.
2, 4, 6	VSS	GROUND
3	TDO	TEST DATA OUTPUT – Serial data output line from the OnCE controller.
5	TCK	TEST CLOCK – Serial clock input line to the OnCE control block.
7	EVTI_b	EVENT IN — Active-low signal. At reset, enables or disables the NEXUS block. At other times, provided that NEXUS messages are on, causes a synchronization message.
8	KEY (NC)	No connection
9	RESET_b	RESET IN – Active-low input line to the OnCE controller, signalling a reset.
10	TMS	TEST MODE SELECT – Input signal that tells the OnCE control block to advance one mode state (of the cycle of mode states).
11	VDD	OPERATING VOLTAGE – Transmission line for +3.2-volt MCU operating power.
12	RDY_b	READ/WRITE READY — Active-low signal that a NEXUS read or write access is ready.
13	EVTO_b	WATCHPOINT EVENT OUT — Active-low signal that a NEXUS watchpoint occurred, providing an exact timing reference.
14	TRST_b	TEST RESET – Active-low input line for an external reset signal to the OnCE controller.

5.3 NEXUS Connector (J23)

Connector J23, a 2-by-15-pin connector, conveys data and control signals to and from the NEXUS (GEPDIS) control block. **Figure 5-6** and **Table 5-6** give the pin assignments and signal descriptions for this connector.

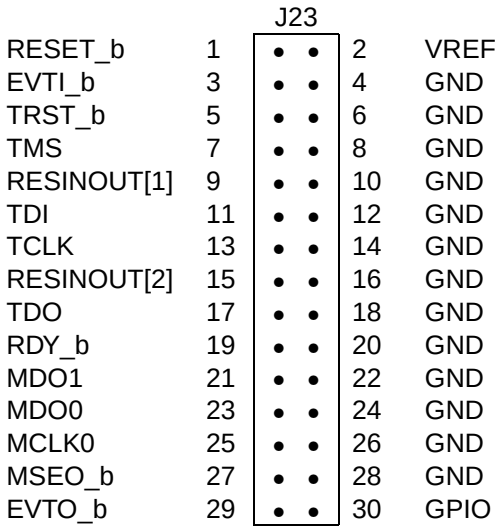


Figure 5-6 NEXUS Connector J23 Pin Assignments

Table 5-6 NEXUS Connector J23 Signal Descriptions

Pin	Mnemonic	Signal
1	RESET_b	RESET IN – Active-low input line to the NEXUS controller, signalling a reset.
2	VREF	VOLTAGE REFERENCE – Transmission line for +3.3-volt MCU operating power.
3	EVTI_b	EVENT IN — Active-low signal. At reset, enables or disables the NEXUS block. At other times, provided that NEXUS messages are on, causes a synchronization message.
even, 4—28	GND	GROUND
5	TRST_b	TEST RESET – Active-low input line for an external reset signal to the NEXUS controller.
7	TMS	TEST MODE SELECT – Input signal that tells the NEXUS control block to advance one mode state (of the cycle of mode states).
9, 15	RESINOUT[1], RESINOUT[2]	RESERVED IN/OUT (lines 1, 2) — No connection.
11	TDI	TEST DATA INPUT – Data and command serial input line to the NEXUS controller.
13	TCLK	TEST CLOCK – Serial clock input line to the NEXUS control block.
17	TDO	TEST DATA OUTPUT – Serial data output line from the NEXUS controller.

Table 5-6 NEXUS Connector J23 Signal Descriptions (Continued)

19	RDY_b	READ/WRITE READY — Active-low signal that a NEXUS read or write access is ready.
21, 23	MDO1, MDO0	MESSAGE DATA OUT (lines 1, 0) — Output signals for NEXUS messages.
25	MCLK0	MCU CLOCK — Clock signal used by the resident MCU.
27	MSEO_b	MESSAGE START/END OUT — Active-low output indicating that a NEXUS message is starting or ending., the end of a variable-length packet, or the end of a message.
29	EVTO_b	WATCHPOINT EVENT OUT — Active-low signal that a NEXUS watchpoint occurred, providing an exact timing reference.
30	GPIO	GENERAL PURPOSE INPUT/OUTPUT — General-purpose I/O signal.

5.4 MLB Logic Analyzer Connectors (J25, J28, J32)

Connectors J25, J28, and J32, all 2-by-19-pin Mictor connectors, are the MLB logic analyzer connectors. **Figure 5-7** through **Figure 5-9** give the pin assignments for these connectors. **Table 5-7** through **Table 5-9** give the signal descriptions for these connectors.

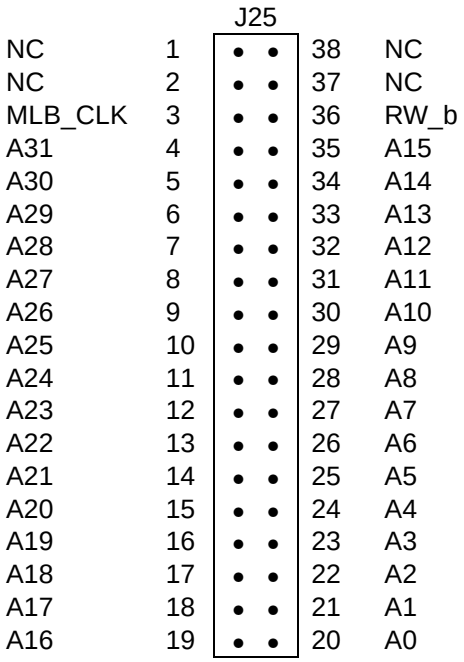


Figure 5-7 MLB Logic Analyzer Connector J25 Pin Assignments

Table 5-7 MLB Logic Analyzer Connector J25 Signal Descriptions

Pin	Mnemonic	Signal
1, 2, 37, 38	NC	No connection
3	MLB_CLK	CLOCK OUTPUT — External clock source.
4 — 35	A31 — A0 (not in exact order)	ADDRESS BUS — Output lines 31—0, for addressing external devices. These lines change state only during external-memory accesses.
36	RW_b	READ/WRITE ENABLE – Active-low signal that indicates whether the current bus access is a read access or write access.

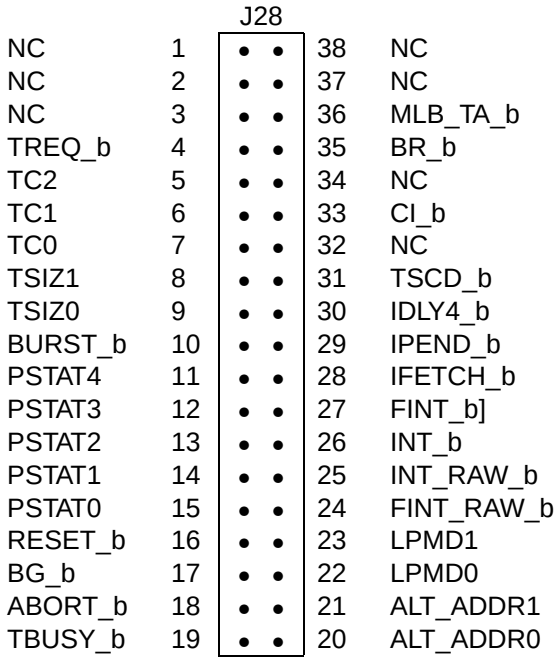


Figure 5-8 MLB Logic Analyzer Connector J28 Pin Assignments

Table 5-8 MLB Logic Analyzer Connector J28 Signal Descriptions

Pin	Mnemonic	Signal
1 — 3, 32, 34, 37, 38	NC	No connection
4	TREQ_b	TRANSMIT REQUEST — Active-low signal indicating a new access request. The resident MCU drives this signal.
5, 6, 7	TC2 — TC0	TRANSFER CODE (lines 2—0) — Signals indicating the general type of transfer.
8, 9	TSIZ1, TSIZ0	TRANSFER SIZE (lines 1, 0) — Signals that indicate the size of an external transfer.
10	BURST_b	BURST — Active-low signal indicating that the current access is a cache line burst.
11 — 15	PSTAT4— PSTAT0	PROCESSOR STATUS (lines 4—0) — Output signals that provide external status indications for the resident MCU.
16	RESET_b	RESET IN — Active-low input signal that resets and initializes most CPU and debug-module logic.
17	BG_b	BUS GRANT — Active-low output signal that grants interface-bus ownership to an alternate master.
18	ABORT_b	TRANSFER ABORT — Active-low signal from the processor that a requested access must be aborted.
19	TBUSY_b	TRANSFER BUSY — Active-low signal indicating that an access is in progress. The resident MCU drives this signal.

Table 5-8 MLB Logic Analyzer Connector J28 Signal Descriptions (Continued)

Pin	Mnemonic	Signal
20, 21	ALT_ADDR0, ALT_ADDR1	ALTERNATE ADDRESS (lines 0,1) — Signals that enable on-chip memory devices to derive addresses of alternate memory blocks. This enables the devices to support both little-endian and big-endian modes.
22, 23	LPMD0, LPM1	LOW POWER MODE (lines 0, 1) — Signals asserted by the processor upon execution of a doze, stop, or wait instruction. Optionally, an external source can assert these signals to put the CMB in its low-power stopped state.
24	FINT_RAW_b	RAW FAST INTERRUPT REQUEST — Active-low input fast-interrupt-request signal to the asynchronous interrupt pending output of the processor core.
25	INT_RAW_b	RAW INTERRUPT REQUEST — Active-low input interrupt-request signal to the asynchronous interrupt pending output of the processor core.
26	INT_b	INTERRUPT REQUEST — Active-low output signal that requests a normal interrupt of the processor core.
27	FINT_b	FAST INTERRUPT REQUEST — Active-low output signal that requests a fast interrupt of the processor core.
28	IFETCH_b	INSTRUCTION FETCH — Active-low output signal that defines the current bus cycle access as an instruction prefetch or cache line fill for an instruction prefetch.
29	IPEND_b	INTERRUPT PENDING — Active-low output signal indicating that an interrupt is pending: the processor has recognized an interrupt request internally, and the appropriate bit PSR has enabled the interrupt.
30	IDLY4_b	IDLY4 STATUS — Active-low output signal indicating that the processor core is in an idly4 instruction sequence.
31	TSCD_b	TRI-STATE CONTROL DATA — Active-low output signal that toggles an alternate master's ability to drive the data bus and the TBUSY_b signal.
33	CI_b	CACHE INHIBIT — Active-low signal (asserted by an accessed lave device) to not cache the current read access.
35	BR_b	BUS REQUEST — Active-low signal from an alternate master, requesting ownership of the interface bus.
36	MLB_TA_b	TRANSMIT ACKNOWLEDGE — Active-low I/O signal that indicates data-transfer completion, for either a read cycle or a write cycle.

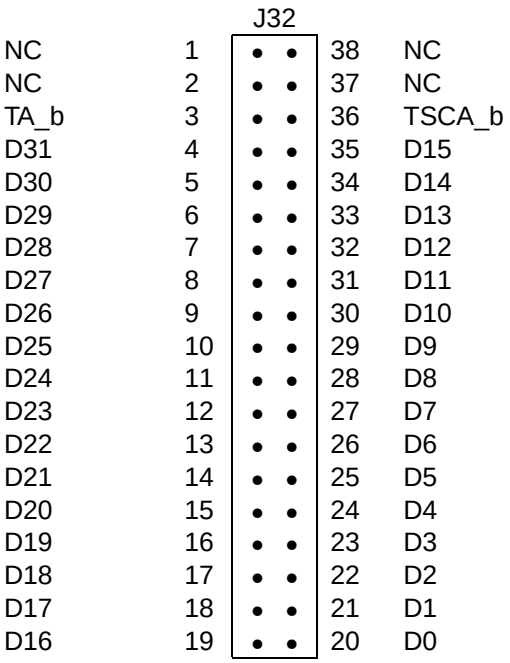


Figure 5-9 MLB Logic Analyzer Connector J32 Pin Assignments

Table 5-9 MLB Logic Analyzer Connector J32 Signal Descriptions

Pin	Mnemonic	Signal
1, 2, 37, 38	NC	No connection
3	TA_b	TRANSMIT ACKNOWLEDGE — Active-low I/O signal that indicates data-transfer completion, for either a read cycle or a write cycle.
4 — 35	D31 — D0 (not in exact order)	DATA BUS — Bi-directional data lines 31—0, for accessing external memory.
36	TSCA_b	TRI-STATE CONTROL ADDRESS — Active-low output signal that toggles an alternate master's ability to drive the address bus and attributes.

5.5 EIM Logic Analyzer Connectors (J29, J30, J33)

Connectors J29, J30, and J33, all 2-by-19-pin Mictor connectors, are the EIM-bus logic analyzer connectors. **Figure 5-10** through **Figure 5-12** give the pin assignments for these connectors. **Table 5-10** through **Table 5-12** give the signal descriptions for these connectors.

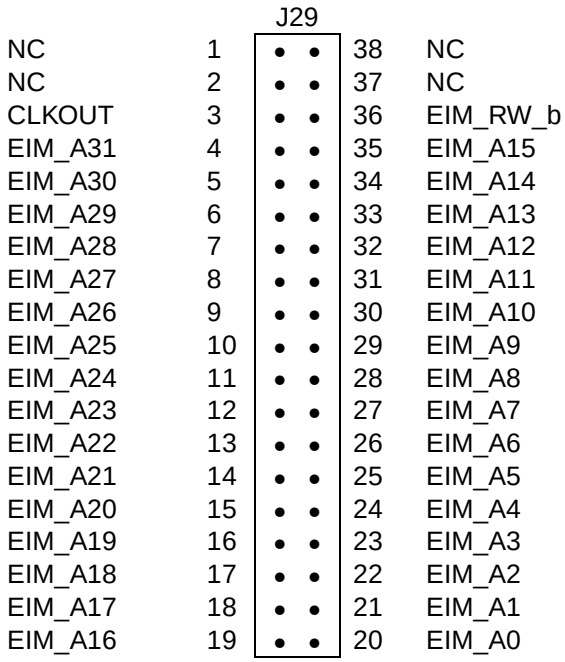


Figure 5-10 EIM Logic Analyzer Connector J29 Pin Assignments

Table 5-10 EIM Logic Analyzer Connector J29 Signal Descriptions

Pin	Mnemonic	Signal
1, 2, 37, 38	NC	No connection
3	CLKOUT	CLOCK OUTPUT — External clock source.
4 — 35	EIM_A31 — EIM_A0 (not in exact order)	EIM ADDRESS BUS — Output lines 31—0, for addressing external devices. These lines change state only during external-memory accesses.
36	EIM_RW_b	EIM READ/WRITE ENABLE – Active-low signal that indicates whether the current EIM bus access is a read access or write access.

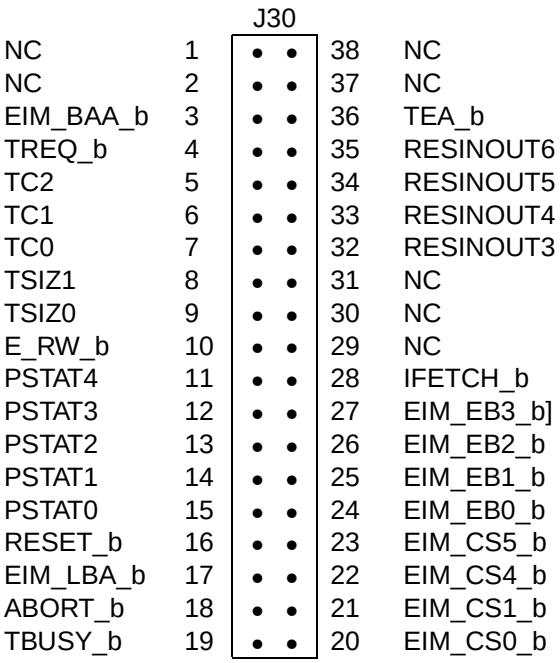


Figure 5-11 EIM Logic Analyzer Connector J30 Pin Assignments

Table 5-11 EIM Logic Analyzer Connector J30 Signal Descriptions

Pin	Mnemonic	Signal
1, 2, 29 — 31, 37, 38	NC	No connection
3	EIM_BAA_b	EIM BURST ADDRESS ADVANCE — Active-low output signal asserted during burst mode accesses, so that burst-capable devices increment internal burst counters to the next sequential memory locations.
4	TREQ_b	TRANSMIT REQUEST — Active-low signal indicating a new access request. The resident MCU drives this signal.
5 — 7	TC2 — TC0	TRANSFER CODE (lines 2—0) — Signals indicating the general type of transfer.
8, 9	TSIZ1, TSIZ0	TRANSFER SIZE (lines 1, 0) — Signals that indicate the size of an external transfer.
10	EIM_RW_b	EIM READ/WRITE ENABLE – Active-low signal that indicates whether the current EIM bus access is a read access or write access.
11 — 15	PSTAT4— PSTAT0	PROCESSOR STATUS (lines 4—0) — Output signals that provide external status indications for the resident MCU.
16	RESET_b	RESET IN — Active-low input signal that resets and initializes most CPU and debug-module logic.
17	EIM_LBA_b	EIM LOAD BURST ADDRESS — Active-low output signal asserted during burst mode accesses so that burst-capable devices load new starting burst addresses.

Table 5-11 EIM Logic Analyzer Connector J30 Signal Descriptions (Continued)

18	ABORT_b	TRANSFER ABORT — Active-low signal from the processor that a requested access must be aborted.
19	TBUSY_b	TRANSFER BUSY — Active-low signal indicating that an access is in progress. The resident MCU drives this signal.
20 — 23	EIM_CS0_b, EIM_CS1_b, EIM_CS4_b, EIM_CS5_b	EIM CHIP SELECTS (lines 0, 1, 4, 5) — Active-low chip-select signals for the EIM bus.
24 — 27	EIM_EB0_b — EIM_EB3_b	EIM ENABLE BYTES 0—3 – Active-low outputs active during an operation to corresponding data bits (D31—D24 for enable byte 3, D23—D16 for enable byte 2, D7–D0 for enable byte 1, D15–D8 for enable byte 0). You can configure these bytes to assert for write cycles or for both read and write cycles.
28	IFETCH_b	INSTRUCTION FETCH — Active-low output signal that defines the current bus cycle access as an instruction prefetch or cache line fill for an instruction prefetch.
32 — 35	RESINOUT3 — RESINOUT6	RESERVED IN/OUT (lines 3—6) — No connection.
36	TEA_b	TRANSFER ERROR ACKNOWLEDGE — Active-low I/O signal that indicates a bus transfer error.

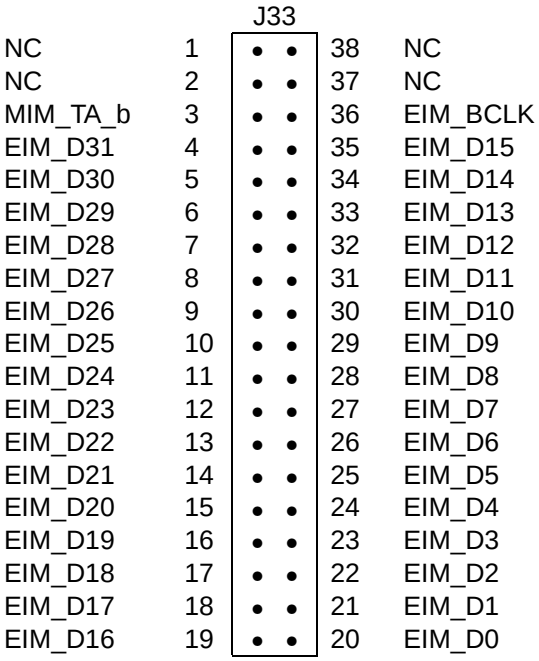


Figure 5-12 EIM Logic Analyzer Connector J33 Pin Assignments

Table 5-12 EIM Logic Analyzer Connector J33 Signal Descriptions

Pin	Mnemonic	Signal
1, 2, 37, 38	NC	No connection
3	MIM_TA_b	MIM TRANSMIT ACKNOWLEDGE — Active-low I/O signal that indicates data-transfer completion for the MIM, for either a read cycle or a write cycle.
4 — 35	EIM_D31 — EIM_D0 (not in exact order)	EIM DATA BUS — Bi-directional data lines 31—0, for accessing external memory.
36	EIM_BCLK	EIM BURST CLOCK — Output clock signal that external burst-capable devices use to synchronize address loading, address incrementing, and burst-read data delivery.

5.6 Old TEA, TA Eyelets (J7, J18)

Figure 5-13 depicts the special eyelet connectors J7 and J18, between Mictor connector J28 and MAPI connector P1.

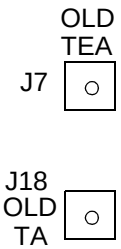


Figure 5-13 Old TEA, TA Eyelets J7, J18

J7 provides the signal MLB_OLD_TEA. This is the TRANSFER ERROR ACKNOWLEDGE signal, double-latched, but before it gets to the M340 core.

J18 provides the signal MLB_OLD_TA. This is the TRANSFER ACKNOWLEDGE signal, double-latched, but before it gets to the M340 core.

5.7 RS232 Connectors (J57, J58)

Connectors J57 and J58, the RS232 connectors, have DCE format. The diagram below shows the pin numbering of these connectors. **Table 5-13** lists the pin assignments and signal directions for these connectors.

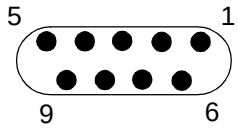


Table 5-13 RS232 Connector J57, J58 Pin Assignments

Pin	Signal	Signal Direction
1	CD Communication Detect	Out
2	TXD Transmitted Data	Out
3	RXD Received Data	In
4	DTR Data Terminal Ready	IN
5	GROUND	—
6	DSR Data Set Ready	Out
7	CTS Clear to Send	In
8	RTS Request to Send	Out
9	RI Ring Indicator	In

NOTE: *Connector J57 is for channel A, and connector J58 is for channel B. Accordingly, the respective pin 1 assignments can be thought of as CDA and CDB. Similarly, the respective pin 2 assignments can be thought of as TXDA and TXDB, and so forth.*

Section 6 Cross Reference Tables

During your application development, you may need to trace a signal from a pin of the resident 3401 device, through the FPGA device, to a MAPI-ring connector. Conversely, you may need to trace a signal in the other direction. The tables of this chapter help such tracing:

- **Table 6-1** lists trace relationships by 3401 device signals.
- **Table 6-2** lists trace relationships by 3401 device pins.
- **Table 6-3** lists trace relationships by MAPI-ring signals.
- **Table 6-4** lists trace relationships by MAPI-ring pins.
- **Table 6-5** lists pins of the FPGA device, showing their relationships to pins of either the 3401 device or of the MAPI-ring connectors.

Table 6-1 Cross Reference: U3 3401, U2 FPGA, MAPI Connectors

U3 3401 Device		U2 FPGA Device		MAPI Connectors	
Signal	Pin	3401 Side	Ring Side	Default Signal	Pin
ITC_IRQ_B1	W28			ITC_IRQ_B1	J1-61
ITC_IRQ_B3	W29			ITC_IRQ_B3	J1-63
ITC_IRQ_B5	V27			ITC_IRQ_B5	J1-65
ITC_IRQ_B7	R29			ITC_IRQ_B7	J1-67
ITC_IRQ_B25	D29			ITC_IRQ_B25	J1-32
ITC_IRQ_B27	C28			ITC_IRQ_B27	J1-36
ITC_IRQ_B29	E26			ITC_IRQ_B29	J1-55
ITC_IRQ_B31	B28			ITC_IRQ_B31	J1-57
ITC_IRQ_B32	D27			ITC_IRQ_B32	J1-31
ITC_IRQ_B34	D26			ITC_IRQ_B34	J2-72
ITC_IRQ_B36	B27			ITC_IRQ_B36	J2-65
ITC_IRQ_B38	C26			ITC_IRQ_B38	J2-74
ITC_IRQ_B40	A28			ITC_IRQ_B40	J2-76
ITC_IRQ_B42	D25			ITC_IRQ_B42	J2-78
ITC_IRQ_B44	E24			ITC_IRQ_B44	J2-80
ITC_IRQ_B46	A26			ITC_IRQ_B46	J1-33
ITC_IRQ_B48	B25			ITC_IRQ_B48	J1-35
ITC_IRQ_B50	A25			ITC_IRQ_B50	J1-37
ITC_IRQ_B52	C23			ITC_IRQ_B52	J1-39
ITC_IRQ_B54	B24			ITC_IRQ_B54	J1-43
ITC_IRQ_B56	B22			ITC_IRQ_B56	J1-45
ITC_IRQ_B58	A22			ITC_IRQ_B58	J1-47
ITC_IRQ_B60	E21			ITC_IRQ_B60	J1-49
ITC_IRQ_B62	D20			ITC_IRQ_B62	J1-51
ITC_IRQ_B63	C20			ITC_IRQ_B63	J1-53
JD_DEBUG_B	C12		AP19	JD_DEBUG_B	J3-84
JD_MCU_DE_B	C10			JD_MCU_DE_B	J3-94
JD_OFF_BUS_B	A10			JD_OFF_BUS_B	J2-77
J_TCLK	A9			J_TCLK	J3-85
J_TDI	A8			J_TDI	J3-82
J_TDO	A11			J_TDO	J3-80
J_TMS	B10			J_TMS	J3-83
J_TRST_B	B11			J_TRST_B	J3-87
MIM_TA_B	C18		C21	MIM_TA_B	J2-73
MIM_TEA_B	B19		D21	MIM_TEA_B	J2-75
mlb_abort_b	F5	V1	U31	MAPI_ABORT_B	J3-48
mlb_addr0	H3	AG2	AL34	FPGA_ADDR0	J4-42
mlb_addr1	H2	AF5	AL35	FPGA_ADDR1	J4-41
mlb_addr2	H1	AG1	AK33	FPGA_ADDR2	J4-44

Table 6-1 Cross Reference: U3 3401, U2 FPGA, MAPI Connectors (Continued)

U3 3401 Device		U2 FPGA Device		MAPI Connectors	
Signal	**Pin**	**3401 Side**	**Ring Side**	**Default Signal**	**Pin**
mlb_addr3	J4	AF4	AH31	FPGA_ADDR3	J4-43
mlb_addr4	J3	AF3	AK34	FPGA_ADDR4	J4-46
mlb_addr5	J2	AF2	AJ33	FPGA_ADDR5	J4-45
mlb_addr6	J1	AE5	AJ34	FPGA_ADDR6	J4-48
mlb_addr7	K5	AE3	AJ35	FPGA_ADDR7	J4-47
mlb_addr8	K4	AE2	AH32	FPGA_ADDR8	J4-50
mlb_addr9	K3	AE1	AG31	FPGA_ADDR9	J4-49
mlb_addr10	K2	AD5	AH33	FPGA_ADDR10	J4-52
mlb_addr11	K1	AD4	AH34	FPGA_ADDR11	J4-51
mlb_addr12	L3	AD3	AH35	FPGA_ADDR12	J4-54
mlb_addr13	L2	AD2	AG32	FPGA_ADDR13	J4-53
mlb_addr14	L1	AD1	AG33	FPGA_ADDR14	J4-56
mlb_addr15	M3	AC5	AG34	FPGA_ADDR15	J4-55
mlb_addr16	M2	AC4	AF31	FPGA_ADDR16	J4-60
mlb_addr17	M1	AC2	AF33	FPGA_ADDR17	J4-59
mlb_addr18	N4	AC1	AF34	FPGA_ADDR18	J4-62
mlb_addr19	N3	AB4	AF35	FPGA_ADDR19	J4-61
mlb_addr20	N2	AB3	AE31	FPGA_ADDR20	J4-64
mlb_addr21	N1	AH3	AM35	FPGA_ADDR21	J4-63
mlb_addr22	P3	AA5	AA31	FPGA_ADDR22	J4-66
mlb_addr23	P2	AA4	AA32	FPGA_ADDR23	J4-65
mlb_addr24	P1	AA1	AA35	FPGA_ADDR24	J4-68
mlb_addr25	R3	Y5	Y32	FPGA_ADDR25	J4-67
mlb_addr26	R2	Y4	Y33	FPGA_ADDR26	J4-70
mlb_addr27	R1	Y3	Y34	FPGA_ADDR27	J4-69
mlb_addr28	T2	W5	Y35	FPGA_ADDR28	J4-72
mlb_addr29	T3	W4	W32	FPGA_ADDR29	J4-71
mlb_addr30	U1	W3	W33	FPGA_ADDR30	J4-74
mlb_addr31	U2	W2	W34	FPGA_ADDR31	J4-73
mlb_alt_addr0	D2	V5	V31	FPGA_ALT_ADDR0	J3-39
mlb_alt-addr1	E3	V4	V33	FPGA_ALT_ADDR1	J3-41
mlb_avec_b	B7	A24	B24	MAPI_AVEC_B	J1-80
mlb_bg_b	D5	AJ1	AL32	MAPI_BG_B	J3-51
MLB_BIGEND_B	C6		AR6	MLB_BIGEND_B	J3-40
mlb_br_b	G2	AH4	AM33	MAPI_BR_B	J3-53
mlb_burst_b	E2	AL25	AR26	MAPI_BURST_B	J3-46
mlb_ci__b	G1	B25	E24	FPGA_CI_B	J3-27
MLB_CLKOUT	J26		E18	MLB_CLKOUT	J4-95
mlb_data0	U3	U4	U32	FPGA_DATA0	J4-4
mlb_data1	U4	U2	U33	FPGA_DATA1	J4-3

Table 6-1 Cross Reference: U3 3401, U2 FPGA, MAPI Connectors (Continued)

U3 3401 Device		U2 FPGA Device		MAPI Connectors	
Signal	Pin	3401 Side	Ring Side	Default Signal	Pin
mlb_data2	V1	U1	U35	FPGA_DATA2	J4-6
mlb_data3	V2	T5	T31	FPGA_DATA3	J4-5
mlb_data4	V3	T3	T33	FPGA_DATA4	J4-8
mlb_data5	W1	T2	T34	FPGA_DATA5	J4-7
mlb_data6	W2	T1	R31	FPGA_DATA6	J4-10
mlb_data7	W3	R5	R33	FPGA_DATA7	J4-9
mlb_data8	Y1	R4	R34	FPGA_DATA8	J4-12
mlb_data9	Y2	R3	R35	FPGA_DATA9	J4-11
mlb_data10	Y3	P3	P31	FPGA_DATA10	J4-14
mlb_data11	Y4	P2	P32	FPGA_DATA11	J4-13
mlb_data12	Y5	P1	P34	FPGA_DATA12	J4-18
mlb_data13	AA1	N5	P35	FPGA_DATA13	J4-17
mlb_data14	AA2	N4	N31	FPGA_DATA14	J4-20
mlb_data15	AA3	N3	N32	FPGA_DATA15	J4-19
mlb_data16	AA4	N2	M35	FPGA_DATA16	J4-22
mlb_data17	AA5	M1	M31	FPGA_DATA17	J4-21
mlb_data18	AB1	M5	M32	FPGA_DATA18	J4-24
mlb_data19	AB2	M4	M33	FPGA_DATA19	J4-23
mlb_data20	AB3	L1	L35	FPGA_DATA20	J4-26
mlb_data21	AC1	L2	L34	FPGA_DATA21	J4-25
mlb_data22	AC2	L3	L33	FPGA_DATA22	J4-30
mlb_data23	AC3	L5	L32	FPGA_DATA23	J4-29
mlb_data24	AD1	K2	K34	FPGA_DATA24	J4-32
mlb_data25	AD2	K3	K33	FPGA_DATA25	J4-31
mlb_data26	AD3	K4	J35	FPGA_DATA26	J4-34
mlb_data27	AE1	J1	K31	FPGA_DATA27	J4-33
mlb_data28	AF1	K5	J34	FPGA_DATA28	J4-36
mlb_data29	AD4	J2	J33	FPGA_DATA29	J4-35
mlb_data30	AE4	H2	H34	FPGA_DATA30	J4-38
mlb_data31	AE3	H3	H33	FPGA_DATA31	J4-37
mlb_fint_b	A6	D23	E23	MAPI_FINT_B	J1-62
mlb_fint_raw_b	B12	A23	B23	MAPI_FINT_RAW_B	J1-70
mlb_idly4_b	C4	AN25	AM25	MAPI_IDLY4_B	J3-26
mlb_ifetch__b	D6	AP25	AL24	MAPI_IFETCH_B	J3-28
mlb_int_b	A7	D22	E22	FPGA_INT_B	J1-79
mlb_int_raw_b	D10	B22	C22	MAPI_INT_RAW_B	J1-72
mlb_ipend_b	B6	AM24	AR25	MAPI_IPEND_B	J1-78
MLB_LPMD0	F4		AP20	MLB_LPMD0	J3-52
MLB_LPMD1	F3		AL19	MLB_LPMD1	J3-50
MLB_PSTAT0	A4			MLB_PSTAT0	J3-67

Table 6-1 Cross Reference: U3 3401, U2 FPGA, MAPI Connectors (Continued)

U3 3401 Device		U2 FPGA Device		MAPI Connectors	
Signal	**Pin**	**3401 Side**	**Ring Side**	**Default Signal**	**Pin**
MLB_PSTAT1	B4			MLB_PSTAT1	J3-69
MLB_PSTAT2	A3			MLB_PSTAT2	J3-71
MLB_PSTAT3	C5			MLB_PSTAT3	J3-73
MLB_PSTAT4	A2			MLB_PSTAT4	J3-70
mlb_reset_b	B5	H4	H32	MAPI_RESET_B	J3-77
mlb_rw_b	C1	J5	J31	FPGA_RW_B	J4-81
mlb_tbusy_b	E1	D29	E29	MAPI_TBUSY_B	J3-72
mlb_tc0	C3	AR24	AN24	MAPI_TC0	J3-62
mlb_tc1	C2	AM23	AL23	MAPI_TC1	J3-64
mlb_tc2	D3	AP23	AN23	MAPI_TC2	J3-66
MLB_TE_B	A1		AM20	MLB_TE_B	J3-57
mlb_treq_b	D1	V2	V35	FPGA_TREQ_B	J4-79
mlb_tsca_b	B3	AM22	AL22	MAPI_TSCA_B	J3-45
mlb_tscd_b	B2	AP22	AN22	MAPI_TSCD_B	J3-47
mlb_tsiz0	E6	E25	A26	FPGA_TSIZ0	J3-86
mlb_tsiz1	D4	B26	C26	FPGA_TSIZ1	J3-88
mlb_vec0	B9	AM17	AN17	MAPI_VEC0	J1-81
mlb_vec1	C9	AR16	AL17	MAPI_VEC1	J1-82
mlb_vec2	A5	AN16	AP16	MAPI_VEC2	J1-83
mlb_vec3	B8	AL16	AM16	MAPI_VEC3	J1-84
mlb_vec4	D9	AP15	AR15	MAPI_VEC4	J1-85
mlb_vec5	E9	AL15	AM15	MAPI_VEC5	J1-86
mlb_vec6	C8	AP14	AR14	MAPI_VEC6	J1-87
MLB_WAKEUP_B	F2		AL21	MLB_WAKEUP_B	J3-38
M_POR	C7			M_POR	J3-32
RESINOUT7	M27			RESINOUT7	J2-22
RESINOUT8	L29			RESINOUT8	J2-24
RESINOUT9	K29			RESINOUT9	J2-26
RESINOUT10	K28			RESINOUT10	J2-28
RESINOUT11	K27			RESINOUT11	J2-66
RESINOUT12	K26			RESINOUT12	J2-68
RESINOUT13	J28			RESINOUT13	J2-70
RESINOUT14	J27			RESINOUT14	J2-21
RESINOUT15	B20			RESINOUT15	J2-23
RESINOUT16	A20			RESINOUT16	J2-25
RESINOUT17	C19			RESINOUT17	J2-27
RESINOUT18	A19			RESINOUT18	J2-71
TC_IN	AG3			TC_IN	J3-30
TC_MUX_MODE	AG2			TC_MUX_MODE	J3-34
TC_OUT	AG4			TC_OUT	J3-29

Table 6-1 Cross Reference: U3 3401, U2 FPGA, MAPI Connectors (Continued)

U3 3401 Device		U2 FPGA Device		MAPI Connectors	
Signal	**Pin**	**3401 Side**	**Ring Side**	**Default Signal**	**Pin**
TC_TEST_EN	C11			TC_TEST_EN	J3-56
TC_TRISTATE	AF4			TC_TRISTATE	J3-92
UARTA_INT10	E29			UARTA_INT10	J1-30
UARTB_INT11	F28			UARTB_INT11	J1-38
			H35	BIG_TEST	J1-27
			A29	FPGA_CS_B0	J4-85
			E28	FPGA_CS_B1	J4-87
			D28	FPGA_CS_B2	J4-89
			C28	FPGA_CS_B3	J4-91
			B28	FPGA_CS_B4	J1-97
			A28	FPGA_CS_B5	J1-95
			AP27	FPGA_CS_B6	J1-93
			AN27	FPGA_CS_B7	J1-91
			AM27	FPGA_CS_B8	J1-98
			AL27	FPGA_CS_B9	J1-96
			E27	FPGA_EB_B0	J4-86
			D27	FPGA_EB_B1	J4-84
			C27	FPGA_EB_B2	J4-80
			B27	FPGA_EB_B3	J4-82
			B30	FPGA_OE_B	J4-88
			C29	MAPI_TA_B	J4-77
			E26	MAPI_TEA_B	J4-78
			AN11	MLB_DEVSP0_B	J1-73

Table 6-2 Cross Reference: U3 3401 U2 FPGA, MAPI Connectors

U3 3401 Device		U2 FPGA Device		MAPI Connectors	
Pin	Signal	3401 Side	Ring Side	Default Signal	Pin
AA1	mlb_data13	N5	P35	FPGA_DATA13	J4-17
AA2	mlb_data14	N4	N31	FPGA_DATA14	J4-20
AA3	mlb_data15	N3	N32	FPGA_DATA15	J4-19
AA4	mlb_data16	N2	M35	FPGA_DATA16	J4-22
AA5	mlb_data17	M1	M31	FPGA_DATA17	J4-21
AB1	mlb_data18	M5	M32	FPGA_DATA18	J4-24
AB2	mlb_data19	M4	M33	FPGA_DATA19	J4-23
AB3	mlb_data20	L1	L35	FPGA_DATA20	J4-26
AC1	mlb_data21	L2	L34	FPGA_DATA21	J4-25
AC2	mlb_data22	L3	L33	FPGA_DATA22	J4-30
AC3	mlb_data23	L5	L32	FPGA_DATA23	J4-29
AD1	mlb_data24	K2	K34	FPGA_DATA24	J4-32
AD2	mlb_data25	K3	K33	FPGA_DATA25	J4-31
AD3	mlb_data26	K4	J35	FPGA_DATA26	J4-34
AE1	mlb_data27	J1	K31	FPGA_DATA27	J4-33
AF1	mlb_data28	K5	J34	FPGA_DATA28	J4-36
AD4	mlb_data29	J2	J33	FPGA_DATA29	J4-35
AE3	mlb_data31	H3	H33	FPGA_DATA31	J4-37
AE4	mlb_data30	H2	H34	FPGA_DATA30	J4-38
AF4	TC_TRISTATE			TC_TRISTATE	J3-92
AG2	TC_MUX_MODE			TC_MUX_MODE	J3-34
AG3	TC_IN			TC_IN	J3-30
AG4	TC_OUT			TC_OUT	J3-29
A1	MLB_TE_B		AM20	MLB_TE_B	J3-57
A2	MLB_PSTAT4			MLB_PSTAT4	J3-70
A3	MLB_PSTAT2			MLB_PSTAT2	J3-71
A4	MLB_PSTAT0			MLB_PSTAT0	J3-67
A5	mlb_vec2	AN16	AP16	MAPI_VEC2	J1-83
A6	mlb_fint_b	D23	E23	MAPI_FINT_B	J1-62
A7	mlb_int_b	D22	E22	FPGA_INT_B	J1-79
A8	J_TDI			J_TDI	J3-82
A9	J_TCLK			J_TCLK	J3-85
A10	JD_OFF_BUS_B			JD_OFF_BUS_B	J2-77
A11	J_TDO			J_TDO	J3-80
A19	RESINOUT18			RESINOUT18	J2-71
A20	RESINOUT16			RESINOUT16	J2-25
A22	ITC_IRQ_B58			ITC_IRQ_B58	J1-47
A25	ITC_IRQ_B50			ITC_IRQ_B50	J1-37
A26	ITC_IRQ_B46			ITC_IRQ_B46	J1-33

Table 6-2 Cross Reference: U3 3401 U2 FPGA, MAPI Connectors (Continued)

U3 3401 Device		U2 FPGA Device		MAPI Connectors	
Pin	Signal	3401 Side	Ring Side	Default Signal	Pin
A28	ITC_IRQ_B40			ITC_IRQ_B40	J2-76
B2	mlb_tscd_b	AP22	AN22	MAPI_TSCD_B	J3-47
B3	mlb_tsca_b	AM22	AL22	MAPI_TSCA_B	J3-45
B4	MLB_PSTAT1			MLB_PSTAT1	J3-69
B5	mlb_reset_b	H4	H32	MAPI_RESET_B	J3-77
B6	mlb_ipend_b	AM24	AR25	MAPI_IPEND_B	J1-78
B7	mlb_avec_b	A24	B24	MAPI_AVEC_B	J1-80
B8	mlb_vec3	AL16	AM16	MAPI_VEC3	J1-84
B9	mlb_vec0	AM17	AN17	MAPI_VEC0	J1-81
B10	J_TMS			J_TMS	J3-83
B11	J_TRST_B			J_TRST_B	J3-87
B12	mlb_fint_raw_b	A23	B23	MAPI_FINT_RAW_B	J1-70
B19	MIM_TEA_B		D21	MIM_TEA_B	J2-75
B20	RESINOUT15			RESINOUT15	J2-23
B22	ITC_IRQ_B56			ITC_IRQ_B56	J1-45
B24	ITC_IRQ_B54			ITC_IRQ_B54	J1-43
B25	ITC_IRQ_B48			ITC_IRQ_B48	J1-35
B27	ITC_IRQ_B36			ITC_IRQ_B36	J2-65
B28	ITC_IRQ_B31			ITC_IRQ_B31	J1-57
C1	mlb_rw_b	J5	J31	FPGA_RW_B	J4-81
C2	mlb_tc1	AM23	AL23	MAPI_TC1	J3-64
C3	mlb_tc0	AR24	AN24	MAPI_TC0	J3-62
C4	mlb_idly4_b	AN25	AM25	MAPI_IDLY4_B	J3-26
C5	MLB_PSTAT3			MLB_PSTAT3	J3-73
C6	MLB_BIGEND_B		AR6	MLB_BIGEND_B	J3-40
C7	M_POR			M_POR	J3-32
C8	mlb_vec6	AP14	AR14	MAPI_VEC6	J1-87
C9	mlb_vec1	AR16	AL17	MAPI_VEC1	J1-82
C10	JD_MCU_DE_B			JD_MCU_DE_B	J3-94
C11	TC_TEST_EN			TC_TEST_EN	J3-56
C12	JD_DEBUG_B		AP19	JD_DEBUG_B	J3-84
C18	MIM_TA_B		C21	MIM_TA_B	J2-73
C19	RESINOUT17			RESINOUT17	J2-27
C20	ITC_IRQ_B63			ITC_IRQ_B63	J1-53
C23	ITC_IRQ_B52			ITC_IRQ_B52	J1-39
C26	ITC_IRQ_B38			ITC_IRQ_B38	J2-74
C28	ITC_IRQ_B27			ITC_IRQ_B27	J1-36
D1	mlb_treq_b	V2	V35	FPGA_TREQ_B	J4-79
D2	mlb_alt_addr0	V5	V31	FPGA_ALT_ADDR0	J3-39
D3	mlb_tc2	AP23	AN23	MAPI_TC2	J3-66

Table 6-2 Cross Reference: U3 3401 U2 FPGA, MAPI Connectors (Continued)

U3 3401 Device		U2 FPGA Device		MAPI Connectors	
Pin	**Signal**	**3401 Side**	**Ring Side**	**Default Signal**	**Pin**
D4	mlb_tsiz1	B26	C26	FPGA_TSIZ1	J3-88
D5	mlb_bg_b	AJ1	AL32	MAPI_BG_B	J3-51
D6	mlb_ifetch__b	AP25	AL24	MAPI_IFETCH_B	J3-28
D9	mlb_vec4	AP15	AR15	MAPI_VEC4	J1-85
D10	mlb_int_raw_b	B22	C22	MAPI_INT_RAW_B	J1-72
D20	ITC_IRQ_B62			ITC_IRQ_B62	J1-51
D25	ITC_IRQ_B42			ITC_IRQ_B42	J2-78
D26	ITC_IRQ_B34			ITC_IRQ_B34	J2-72
D27	ITC_IRQ_B32			ITC_IRQ_B32	J1-31
D29	ITC_IRQ_B25			ITC_IRQ_B25	J1-32
E1	mlb_tbusy_b	D29	E29	MAPI_TBUSY_B	J3-72
E2	mlb_burst_b	AL25	AR26	MAPI_BURST_B	J3-46
E3	mlb_alt-addr1	V4	V33	FPGA_ALT_ADDR1	J3-41
E6	mlb_tsiz0	E25	A26	FPGA_TSIZ0	J3-86
E9	mlb_vec5	AL15	AM15	MAPI_VEC5	J1-86
E21	ITC_IRQ_B60			ITC_IRQ_B60	J1-49
E24	ITC_IRQ_B44			ITC_IRQ_B44	J2-80
E26	ITC_IRQ_B29			ITC_IRQ_B29	J1-55
E29	UARTA_INT10			UARTA_INT10	J1-30
F2	MLB_WAKEUP_B		AL21	MLB_WAKEUP_B	J3-38
F3	MLB_LPMD1		AL19	MLB_LPMD1	J3-50
F4	MLB_LPMD0		AP20	MLB_LPMD0	J3-52
F5	mlb_abort_b	V1	U31	MAPI_ABORT_B	J3-48
F28	UARTB_INT11			UARTB_INT11	J1-38
G1	mlb_ci__b	B25	E24	FPGA_CI_B	J3-27
G2	mlb_br_b	AH4	AM33	MAPI_BR_B	J3-53
H1	mlb_addr2	AG1	AK33	FPGA_ADDR2	J4-44
H2	mlb_addr1	AF5	AL35	FPGA_ADDR1	J4-41
H3	mlb_addr0	AG2	AL34	FPGA_ADDR0	J4-42
J1	mlb_addr6	AE5	AJ34	FPGA_ADDR6	J4-48
J2	mlb_addr5	AF2	AJ33	FPGA_ADDR5	J4-45
J3	mlb_addr4	AF3	AK34	FPGA_ADDR4	J4-46
J4	mlb_addr3	AF4	AH31	FPGA_ADDR3	J4-43
J26	MLB_CLKOUT		E18	MLB_CLKOUT	J4-95
J27	RESINOUT14			RESINOUT14	J2-21
J28	RESINOUT13			RESINOUT13	J2-70
K1	mlb_addr11	AD4	AH34	FPGA_ADDR11	J4-51
K2	mlb_addr10	AD5	AH33	FPGA_ADDR10	J4-52
K3	mlb_addr9	AE1	AG31	FPGA_ADDR9	J4-49
K4	mlb_addr8	AE2	AH32	FPGA_ADDR8	J4-50

Table 6-2 Cross Reference: U3 3401 U2 FPGA, MAPI Connectors (Continued)

U3 3401 Device		U2 FPGA Device		MAPI Connectors	
Pin	Signal	3401 Side	Ring Side	Default Signal	Pin
K5	mlb_addr7	AE3	AJ35	FPGA_ADDR7	J4-47
K26	RESINOUT12			RESINOUT12	J2-68
K27	RESINOUT11			RESINOUT11	J2-66
K28	RESINOUT10			RESINOUT10	J2-28
K29	RESINOUT9			RESINOUT9	J2-26
L1	mlb_addr14	AD1	AG33	FPGA_ADDR14	J4-56
L2	mlb_addr13	AD2	AG32	FPGA_ADDR13	J4-53
L3	mlb_addr12	AD3	AH35	FPGA_ADDR12	J4-54
L29	RESINOUT8			RESINOUT8	J2-24
M1	mlb_addr17	AC2	AF33	FPGA_ADDR17	J4-59
M2	mlb_addr16	AC4	AF31	FPGA_ADDR16	J4-60
M3	mlb_addr15	AC5	AG34	FPGA_ADDR15	J4-55
M27	RESINOUT7			RESINOUT7	J2-22
N1	mlb_addr21	AH3	AM35	FPGA_ADDR21	J4-63
N2	mlb_addr20	AB3	AE31	FPGA_ADDR20	J4-64
N3	mlb_addr19	AB4	AF35	FPGA_ADDR19	J4-61
N4	mlb_addr18	AC1	AF34	FPGA_ADDR18	J4-62
P1	mlb_addr24	AA1	AA35	FPGA_ADDR24	J4-68
P2	mlb_addr23	AA4	AA32	FPGA_ADDR23	J4-65
P3	mlb_addr22	AA5	AA31	FPGA_ADDR22	J4-66
R1	mlb_addr27	Y3	Y34	FPGA_ADDR27	J4-69
R2	mlb_add[26	Y4	Y33	FPGA_ADDR26	J4-70
R3	mlb_addr25	Y5	Y32	FPGA_ADDR25	J4-67
R29	ITC_IRQ_B7			ITC_IRQ_B7	J1-67
T2	mlb_addr28	W5	Y35	FPGA_ADDR28	J4-72
T3	mlb_addr29	W4	W32	FPGA_ADDR29	J4-71
U1	mlb_addr30	W3	W33	FPGA_ADDR30	J4-74
U2	mlb_addr31	W2	W34	FPGA_ADDR31	J4-73
U3	mlb_data0	U4	U32	FPGA_DATA0	J4-4
U4	mlb_data1	U2	U33	FPGA_DATA1	J4-3
V1	mlb_data2	U1	U35	FPGA_DATA2	J4-6
V2	mlb_data3	T5	T31	FPGA_DATA3	J4-5
V3	mlb_data4	T3	T33	FPGA_DATA4	J4-8
V27	ITC_IRQ_B5			ITC_IRQ_B5	J1-65
W1	mlb_data5	T2	T34	FPGA_DATA5	J4-7
W2	mlb_data6	T1	R31	FPGA_DATA6	J4-10
W3	mlb_data7	R5	R33	FPGA_DATA7	J4-9
W28	ITC_IRQ_B1			ITC_IRQ_B1	J1-61
W29	ITC_IRQ_B3			ITC_IRQ_B3	J1-63
Y1	mlb_data8	R4	R34	FPGA_DATA8	J4-12

Table 6-2 Cross Reference: U3 3401 U2 FPGA, MAPI Connectors (Continued)

U3 3401 Device		U2 FPGA Device		MAPI Connectors	
Pin	**Signal**	**3401 Side**	**Ring Side**	**Default Signal**	**Pin**
Y2	mlb_data9	R3	R35	FPGA_DATA9	J4-11
Y3	mlb_data10	P3	P31	FPGA_DATA10	J4-14
Y4	mlb_data11	P2	P32	FPGA_DATA11	J4-13
Y5	mlb_data12	P1	P34	FPGA_DATA12	J4-18
			H35	BIG_TEST	J1-27
			A29	FPGA_CS_B0	J4-85
			E28	FPGA_CS_B1	J4-87
			D28	FPGA_CS_B2	J4-89
			C28	FPGA_CS_B3	J4-91
			B28	FPGA_CS_B4	J1-97
			A28	FPGA_CS_B5	J1-95
			AP27	FPGA_CS_B6	J1-93
			AN27	FPGA_CS_B7	J1-91
			AM27	FPGA_CS_B8	J1-98
			AL27	FPGA_CS_B9	J1-96
			E27	FPGA_EB_B0	J4-86
			D27	FPGA_EB_B1	J4-84
			C27	FPGA_EB_B2	J4-80
			B27	FPGA_EB_B3	J4-82
			B30	FPGA_OE_B	J4-88
			C29	MAPI_TA_B	J4-77
			E26	MAPI_TEA_B	J4-78
			AN11	MLB_DEVSP0_B	J1-73

Table 6-3 Cross Reference: MAPI U2 FPGA, U3 3401

MAPI Connectors		U2 FPGA Device		U3 3401 Device	
Default Signal	**Pin**	**Ring Side**	**3401 Side**	**Signal**	**Pin**
BIG_TEST	J1-27	H35			
FPGA_ADDR0	J4-42	AL34	AG2	mlb_addr0	H3
FPGA_ADDR1	J4-41	AL35	AF5	mlb_addr1	H2
FPGA_ADDR2	J4-44	AK33	AG1	mlb_addr2	H1
FPGA_ADDR3	J4-43	AH31	AF4	mlb_addr3	J4
FPGA_ADDR4	J4-46	AK34	AF3	mlb_addr4	J3
FPGA_ADDR5	J4-45	AJ33	AF2	mlb_addr5	J2
FPGA_ADDR6	J4-48	AJ34	AE5	mlb_addr6	J1
FPGA_ADDR7	J4-47	AJ35	AE3	mlb_addr7	K5
FPGA_ADDR8	J4-50	AH32	AE2	mlb_addr8	K4
FPGA_ADDR9	J4-49	AG31	AE1	mlb_addr9	K3
FPGA_ADDR10	J4-52	AH33	AD5	mlb_addr10	K2
FPGA_ADDR11	J4-51	AH34	AD4	mlb_addr11	K1
FPGA_ADDR12	J4-54	AH35	AD3	mlb_addr12	L3
FPGA_ADDR13	J4-53	AG32	AD2	mlb_addr13	L2
FPGA_ADDR14	J4-56	AG33	AD1	mlb_addr14	L1
FPGA_ADDR15	J4-55	AG34	AC5	mlb_addr15	M3
FPGA_ADDR16	J4-60	AF31	AC4	mlb_addr16	M2
FPGA_ADDR17	J4-59	AF33	AC2	mlb_addr17	M1
FPGA_ADDR18	J4-62	AF34	AC1	mlb_addr18	N4
FPGA_ADDR19	J4-61	AF35	AB4	mlb_addr19	N3
FPGA_ADDR20	J4-64	AE31	AB3	mlb_addr20	N2
FPGA_ADDR21	J4-63	AM35	AH3	mlb_addr21	N1
FPGA_ADDR22	J4-66	AA31	AA5	mlb_addr22	P3
FPGA_ADDR23	J4-65	AA32	AA4	mlb_addr23	P2
FPGA_ADDR24	J4-68	AA35	AA1	mlb_addr24	P1
FPGA_ADDR25	J4-67	Y32	Y5	mlb_addr25	R3
FPGA_ADDR26	J4-70	Y33	Y4	mlb_add[26	R2
FPGA_ADDR28	J4-72	Y35	W5	mlb_addr28	T2
FPGA_ADDR27	J4-69	Y34	Y3	mlb_addr27	R1
FPGA_ADDR29	J4-71	W32	W4	mlb_addr29	T3
FPGA_ADDR30	J4-74	W33	W3	mlb_addr30	U1
FPGA_ADDR31	J4-73	W34	W2	mlb_addr31	U2
FPGA_ALT_ADDR0	J3-39	V31	V5	mlb_alt_addr0	D2
FPGA_ALT_ADDR1	J3-41	V33	V4	mlb_alt-addr1	E3
FPGA_CI_B	J3-27	E24	B25	mlb_ci__b	G1
FPGA_CS_B0	J4-85	A29			
FPGA_CS_B1	J4-87	E28			
FPGA_CS_B2	J4-89	D28			

Table 6-3 Cross Reference: MAPI U2 FPGA, U3 3401 (Continued)

MAPI Connectors		U2 FPGA Device		U3 3401 Device	
Default Signal	**Pin**	**Ring Side**	**3401 Side**	**Signal**	**Pin**
FPGA_CS_B3	J4-91	C28			
FPGA_CS_B4	J1-97	B28			
FPGA_CS_B5	J1-95	A28			
FPGA_CS_B6	J1-93	AP27			
FPGA_CS_B7	J1-91	AN27			
FPGA_CS_B8	J1-98	AM27			
FPGA_CS_B9	J1-96	AL27			
FPGA_DATA0	J4-4	U32	U4	mlb_data0	U3
FPGA_DATA1	J4-3	U33	U2	mlb_data1	U4
FPGA_DATA2	J4-6	U35	U1	mlb_data2	V1
FPGA_DATA3	J4-5	T31	T5	mlb_data3	V2
FPGA_DATA4	J4-8	T33	T3	mlb_data4	V3
FPGA_DATA5	J4-7	T34	T2	mlb_data5	W1
FPGA_DATA6	J4-10	R31	T1	mlb_data6	W2
FPGA_DATA7	J4-9	R33	R5	mlb_data7	W3
FPGA_DATA8	J4-12	R34	R4	mlb_data8	Y1
FPGA_DATA9	J4-11	R35	R3	mlb_data9	Y2
FPGA_DATA10	J4-14	P31	P3	mlb_data10	Y3
FPGA_DATA11	J4-13	P32	P2	mlb_data11	Y4
FPGA_DATA12	J4-18	P34	P1	mlb_data12	Y5
FPGA_DATA13	J4-17	P35	N5	mlb_data13	AA1
FPGA_DATA14	J4-20	N31	N4	mlb_data14	AA2
FPGA_DATA15	J4-19	N32	N3	mlb_data15	AA3
FPGA_DATA16	J4-22	M35	N2	mlb_data16	AA4
FPGA_DATA17	J4-21	M31	M1	mlb_data17	AA5
FPGA_DATA18	J4-24	M32	M5	mlb_data18	AB1
FPGA_DATA19	J4-23	M33	M4	mlb_data19	AB2
FPGA_DATA20	J4-26	L35	L1	mlb_data20	AB3
FPGA_DATA21	J4-25	L34	L2	mlb_data21	AC1
FPGA_DATA22	J4-30	L33	L3	mlb_data22	AC2
FPGA_DATA23	J4-29	L32	L5	mlb_data23	AC3
FPGA_DATA24	J4-32	K34	K2	mlb_data24	AD1
FPGA_DATA25	J4-31	K33	K3	mlb_data25	AD2
FPGA_DATA26	J4-34	J35	K4	mlb_data26	AD3
FPGA_DATA27	J4-33	K31	J1	mlb_data27	AE1
FPGA_DATA28	J4-36	J34	K5	mlb_data28	AF1
FPGA_DATA29	J4-35	J33	J2	mlb_data29	AD4
FPGA_DATA30	J4-38	H34	H2	mlb_data30	AE4
FPGA_DATA31	J4-37	H33	H3	mlb_data31	AE3
FPGA_EB_B0	J4-86	E27			

Table 6-3 Cross Reference: MAPI U2 FPGA, U3 3401 (Continued)

MAPI Connectors		U2 FPGA Device		U3 3401 Device	
Default Signal	Pin	Ring Side	3401 Side	Signal	Pin
FPGA_EB_B1	J4-84	D27			
FPGA_EB_B2	J4-80	C27			
FPGA_EB_B3	J4-82	B27			
FPGA_INT_B	J1-79	E22	D22	mlb_int_b	A7
FPGA_OE_B	J4-88	B30			
FPGA_RW_B	J4-81	J31	J5	mlb_rw_b	C1
FPGA_TREQ_B	J4-79	V35	V2	mlb_treq_b	D1
FPGA_TSIZ0	J3-86	A26	E25	mlb_tsiz0	E6
FPGA_TSIZ1	J3-88	C26	B26	mlb_tsiz1	D4
ITC_IRQ_B1	J1-61			ITC_IRQ_B1	W28
ITC_IRQ_B3	J1-63			ITC_IRQ_B3	W29
ITC_IRQ_B5	J1-65			ITC_IRQ_B5	V27
ITC_IRQ_B7	J1-67			ITC_IRQ_B7	R29
ITC_IRQ_B25	J1-32			ITC_IRQ_B25	D29
ITC_IRQ_B27	J1-36			ITC_IRQ_B27	C28
ITC_IRQ_B29	J1-55			ITC_IRQ_B29	E26
ITC_IRQ_B31	J1-57			ITC_IRQ_B31	B28
ITC_IRQ_B32	J1-31			ITC_IRQ_B32	D27
ITC_IRQ_B34	J2-72			ITC_IRQ_B34	D26
ITC_IRQ_B36	J2-65			ITC_IRQ_B36	B27
ITC_IRQ_B38	J2-74			ITC_IRQ_B38	C26
ITC_IRQ_B40	J2-76			ITC_IRQ_B40	A28
ITC_IRQ_B42	J2-78			ITC_IRQ_B42	D25
ITC_IRQ_B44	J2-80			ITC_IRQ_B44	E24
ITC_IRQ_B46	J1-33			ITC_IRQ_B46	A26
ITC_IRQ_B48	J1-35			ITC_IRQ_B48	B25
ITC_IRQ_B50	J1-37			ITC_IRQ_B50	A25
ITC_IRQ_B52	J1-39			ITC_IRQ_B52	C23
ITC_IRQ_B54	J1-43			ITC_IRQ_B54	B24
ITC_IRQ_B56	J1-45			ITC_IRQ_B56	B22
ITC_IRQ_B58	J1-47			ITC_IRQ_B58	A22
ITC_IRQ_B60	J1-49			ITC_IRQ_B60	E21
ITC_IRQ_B62	J1-51			ITC_IRQ_B62	D20
ITC_IRQ_B63	J1-53			ITC_IRQ_B63	C20
JD_DEBUG_B	J3-84	AP19		JD_DEBUG_B	C12
JD_MCU_DE_B	J3-94			JD_MCU_DE_B	C10
JD_OFF_BUS_B	J2-77			JD_OFF_BUS_B	A10
J_TCLK	J3-85			J_TCLK	A9
J_TDI	J3-82			J_TDI	A8
J_TDO	J3-80			J_TDO	A11

Table 6-3 Cross Reference: MAPI U2 FPGA, U3 3401 (Continued)

MAPI Connectors		U2 FPGA Device		U3 3401 Device	
Default Signal	**Pin**	**Ring Side**	**3401 Side**	**Signal**	**Pin**
J_TMS	J3-83			J_TMS	B10
J_TRST_B	J3-87			J_TRST_B	B11
MAPI_ABORT_B	J3-48	U31	V1	mlb_abort_b	F5
MAPI_AVEC_B	J1-80	B24	A24	mlb_avec_b	B7
MAPI_BG_B	J3-51	AL32	AJ1	mlb_bg_b	D5
MAPI_BR_B	J3-53	AM33	AH4	mlb_br_b	G2
MAPI_BURST_B	J3-46	AR26	AL25	mlb_burst_b	E2
MAPI_FINT_B	J1-62	E23	D23	mlb_fint_b	A6
MAPI_FINT_RAW_B	J1-70	B23	A23	mlb_fint_raw_b	B12
MAPI_IDLY4_B	J3-26	AM25	AN25	mlb_idly4_b	C4
MAPI_IFETCH_B	J3-28	AL24	AP25	mlb_ifetch__b	D6
MAPI_INT_RAW_B	J1-72	C22	B22	mlb_int_raw_b	D10
MAPI_IPEND_B	J1-78	AR25	AM24	mlb_ipend_b	B6
MAPI_RESET_B	J3-77	H32	H4	mlb_reset_b	B5
MAPI_TA_B	J4-77	C29			
MAPI_TBUSY_B	J3-72	E29	D29	mlb_tbusy_b	E1
MAPI_TC0	J3-62	AN24	AR24	mlb_tc0	C3
MAPI_TC1	J3-64	AL23	AM23	mlb_tc1	C2
MAPI_TC2	J3-66	AN23	AP23	mlb_tc2	D3
MAPI_TEA_B	J4-78	E26			
MAPI_TSCA_B	J3-45	AL22	AM22	mlb_tsca_b	B3
MAPI_TSCD_B	J3-47	AN22	AP22	mlb_tscd_b	B2
MAPI_VEC0	J1-81	AN17	AM17	mlb_vec0	B9
MAPI_VEC1	J1-82	AL17	AR16	mlb_vec1	C9
MAPI_VEC2	J1-83	AP16	AN16	mlb_vec2	A5
MAPI_VEC3	J1-84	AM16	AL16	mlb_vec3	B8
MAPI_VEC4	J1-85	AR15	AP15	mlb_vec4	D9
MAPI_VEC5	J1-86	AM15	AL15	mlb_vec5	E9
MAPI_VEC6	J1-87	AR14	AP14	mlb_vec6	C8
MIM_TA_B	J2-73	C21		MIM_TA_B	C18
MIM_TEA_B	J2-75	D21		MIM_TEA_B	B19
MLB_BIGEND_B	J3-40	AR6		MLB_BIGEND_B	C6
MLB_CLKOUT	J4-95	E18		MLB_CLKOUT	J26
MLB_DEVSP0_B	J1-73	AN11			
MLB_LPMD0	J3-52	AP20		MLB_LPMD0	F4
MLB_LPMD1	J3-50	AL19		MLB_LPMD1	F3
MLB_PSTAT0	J3-67			MLB_PSTAT0	A4
MLB_PSTAT1	J3-69			MLB_PSTAT1	B4
MLB_PSTAT2	J3-71			MLB_PSTAT2	A3
MLB_PSTAT3	J3-73			MLB_PSTAT3	C5

Table 6-3 Cross Reference: MAPI U2 FPGA, U3 3401 (Continued)

MAPI Connectors		U2 FPGA Device		U3 3401 Device	
Default Signal	**Pin**	**Ring Side**	**3401 Side**	**Signal**	**Pin**
MLB_PSTAT4	J3-70			MLB_PSTAT4	A2
MLB_TE_B	J3-57	AM20		MLB_TE_B	A1
MLB_WAKEUP_B	J3-38	AL21		MLB_WAKEUP_B	F2
M_POR	J3-32			M_POR	C7
RESINOUT7	J2-22			RESINOUT7	M27
RESINOUT8	J2-24			RESINOUT8	L29
RESINOUT9	J2-26			RESINOUT9	K29
RESINOUT10	J2-28			RESINOUT10	K28
RESINOUT11	J2-66			RESINOUT11	K27
RESINOUT12	J2-68			RESINOUT12	K26
RESINOUT13	J2-70			RESINOUT13	J28
RESINOUT14	J2-21			RESINOUT14	J27
RESINOUT15	J2-23			RESINOUT15	B20
RESINOUT16	J2-25			RESINOUT16	A20
RESINOUT17	J2-27			RESINOUT17	C19
RESINOUT18	J2-71			RESINOUT18	A19
TC_IN	J3-30			TC_IN	AG3
TC_MUX_MODE	J3-34			TC_MUX_MODE	AG2
TC_OUT	J3-29			TC_OUT	AG4
TC_TEST_EN	J3-56			TC_TEST_EN	C11
TC_TRISTATE	J3-92			TC_TRISTATE	AF4
UARTA_INT10	J1-30			UARTA_INT10	E29
UARTB_INT11	J1-38			UARTB_INT11	F28

Table 6-4 Cross Reference: MAPI, U2 FPGA, U3 3401

MAPI Connectors		U2 FPGA Device		U3 3401 Device	
Pin	Default Signal	Ring Side	3401 Side	Signal	Pin
J1-27	BIG_TEST	H35			
J1-30	UARTA_INT10			UARTA_INT10	E29
J1-31	ITC_IRQ_B32			ITC_IRQ_B32	D27
J1-32	ITC_IRQ_B25			ITC_IRQ_B25	D29
J1-33	ITC_IRQ_B46			ITC_IRQ_B46	A26
J1-35	ITC_IRQ_B48			ITC_IRQ_B48	B25
J1-36	ITC_IRQ_B27			ITC_IRQ_B27	C28
J1-37	ITC_IRQ_B50			ITC_IRQ_B50	A25
J1-38	UARTB_INT11			UARTB_INT11	F28
J1-39	ITC_IRQ_B52			ITC_IRQ_B52	C23
J1-43	ITC_IRQ_B54			ITC_IRQ_B54	B24
J1-45	ITC_IRQ_B56			ITC_IRQ_B56	B22
J1-47	ITC_IRQ_B58			ITC_IRQ_B58	A22
J1-49	ITC_IRQ_B60			ITC_IRQ_B60	E21
J1-51	ITC_IRQ_B62			ITC_IRQ_B62	D20
J1-53	ITC_IRQ_B63			ITC_IRQ_B63	C20
J1-55	ITC_IRQ_B29			ITC_IRQ_B29	E26
J1-57	ITC_IRQ_B31			ITC_IRQ_B31	B28
J1-61	ITC_IRQ_B1			ITC_IRQ_B1	W28
J1-62	MAPI_FINT_B	E23	D23	mlb_fint_b	A6
J1-63	ITC_IRQ_B3			ITC_IRQ_B3	W29
J1-65	ITC_IRQ_B5			ITC_IRQ_B5	V27
J1-67	ITC_IRQ_B7			ITC_IRQ_B7	R29
J1-70	MAPI_FINT_RAW_B	B23	A23	mlb_fint_raw_b	B12
J1-72	MAPI_INT_RAW_B	C22	B22	mlb_int_raw_b	D10
J1-73	MLB_DEVSP0_B	AN11			
J1-78	MAPI_IPEND_B	AR25	AM24	mlb_ipend_b	B6
J1-79	FPGA_INT_B	E22	D22	mlb_int_b	A7
J1-80	MAPI_AVEC_B	B24	A24	mlb_avec_b	B7
J1-81	MAPI_VEC0	AN17	AM17	mlb_vec0	B9
J1-82	MAPI_VEC1	AL17	AR16	mlb_vec1	C9
J1-83	MAPI_VEC2	AP16	AN16	mlb_vec2	A5
J1-84	MAPI_VEC3	AM16	AL16	mlb_vec3	B8
J1-85	MAPI_VEC4	AR15	AP15	mlb_vec4	D9
J1-86	MAPI_VEC5	AM15	AL15	mlb_vec5	E9
J1-87	MAPI_VEC6	AR14	AP14	mlb_vec6	C8
J1-91	FPGA_CS_B7	AN27			
J1-93	FPGA_CS_B6	AP27			
J1-95	FPGA_CS_B5	A28			

Table 6-4 Cross Reference: MAPI, U2 FPGA, U3 3401 (Continued)

MAPI Connectors		U2 FPGA Device		U3 3401 Device	
Pin	Default Signal	Ring Side	3401 Side	Signal	Pin
J1-96	FPGA_CS_B9	AL27			
J1-97	FPGA_CS_B4	B28			
J1-98	FPGA_CS_B8	AM27			
J2-21	RESINOUT14			RESINOUT14	J27
J2-22	RESINOUT7			RESINOUT7	M27
J2-23	RESINOUT15			RESINOUT15	B20
J2-24	RESINOUT8			RESINOUT8	L29
J2-25	RESINOUT16			RESINOUT16	A20
J2-26	RESINOUT9			RESINOUT9	K29
J2-27	RESINOUT17			RESINOUT17	C19
J2-28	RESINOUT10			RESINOUT10	K28
J2-65	ITC_IRQ_B36			ITC_IRQ_B36	B27
J2-66	RESINOUT11			RESINOUT11	K27
J2-68	RESINOUT12			RESINOUT12	K26
J2-70	RESINOUT13			RESINOUT13	J28
J2-71	RESINOUT18			RESINOUT18	A19
J2-72	ITC_IRQ_B34			ITC_IRQ_B34	D26
J2-73	MIM_TA_B	C21		MIM_TA_B	C18
J2-74	ITC_IRQ_B38			ITC_IRQ_B38	C26
J2-75	MIM_TEA_B	D21		MIM_TEA_B	B19
J2-76	ITC_IRQ_B40			ITC_IRQ_B40	A28
J2-77	JD_OFF_BUS_B			JD_OFF_BUS_B	A10
J2-78	ITC_IRQ_B42			ITC_IRQ_B42	D25
J2-80	ITC_IRQ_B44			ITC_IRQ_B44	E24
J3-26	MAPI_IDLY4_B	AM25	AN25	mlb_idly4_b	C4
J3-27	FPGA_CI_B	E24	B25	mlb_ci__b	G1
J3-28	MAPI_IFETCH_B	AL24	AP25	mlb_ifetch__b	D6
J3-29	TC_OUT			TC_OUT	AG4
J3-30	TC_IN			TC_IN	AG3
J3-32	M_POR			M_POR	C7
J3-34	TC_MUX_MODE			TC_MUX_MODE	AG2
J3-38	MLB_WAKEUP_B	AL21		MLB_WAKEUP_B	F2
J3-39	FPGA_ALT_ADDR0	V31	V5	mlb_alt_addr0	D2
J3-40	MLB_BIGEND_B	AR6		MLB_BIGEND_B	C6
J3-41	FPGA_ALT_ADDR1	V33	V4	mlb_alt-addr1	E3
J3-45	MAPI_TSCA_B	AL22	AM22	mlb_tsca_b	B3
J3-46	MAPI_BURST_B	AR26	AL25	mlb_burst_b	E2
J3-47	MAPI_TSCD_B	AN22	AP22	mlb_tscd_b	B2
J3-48	MAPI_ABORT_B	U31	V1	mlb_abort_b	F5
J3-50	MLB_LPMD1	AL19		MLB_LPMD1	F3

Table 6-4 Cross Reference: MAPI, U2 FPGA, U3 3401 (Continued)

MAPI Connectors		U2 FPGA Device		U3 3401 Device	
Pin	**Default Signal**	**Ring Side**	**3401 Side**	**Signal**	**Pin**
J3-51	MAPI_BG_B	AL32	AJ1	mlb_bg_b	D5
J3-52	MLB_LPMD0	AP20		MLB_LPMD0	F4
J3-53	MAPI_BR_B	AM33	AH4	mlb_br_b	G2
J3-56	TC_TEST_EN			TC_TEST_EN	C11
J3-57	MLB_TE_B	AM20		MLB_TE_B	A1
J3-62	MAPI_TC0	AN24	AR24	mlb_tc0	C3
J3-64	MAPI_TC1	AL23	AM23	mlb_tc1	C2
J3-66	MAPI_TC2	AN23	AP23	mlb_tc2	D3
J3-67	MLB_PSTAT0			MLB_PSTAT0	A4
J3-69	MLB_PSTAT1			MLB_PSTAT1	B4
J3-70	MLB_PSTAT4			MLB_PSTAT4	A2
J3-71	MLB_PSTAT2			MLB_PSTAT2	A3
J3-72	MAPI_TBUSY_B	E29	D29	mlb_tbusy_b	E1
J3-73	MLB_PSTAT3			MLB_PSTAT3	C5
J3-77	MAPI_RESET_B	H32	H4	mlb_reset_b	B5
J3-80	J_TDO			J_TDO	A11
J3-82	J_TDI			J_TDI	A8
J3-83	J_TMS			J_TMS	B10
J3-84	JD_DEBUG_B	AP19		JD_DEBUG_B	C12
J3-85	J_TCLK			J_TCLK	A9
J3-86	FPGA_TSIZ0	A26	E25	mlb_tsiz0	E6
J3-87	J_TRST_B			J_TRST_B	B11
J3-88	FPGA_TSIZ1	C26	B26	mlb_tsiz1	D4
J3-92	TC_TRISTATE			TC_TRISTATE	AF4
J3-94	JD_MCU_DE_B			JD_MCU_DE_B	C10
J4-3	FPGA_DATA1	U33	U2	mlb_data1	U4
J4-4	FPGA_DATA0	U32	U4	mlb_data0	U3
J4-5	FPGA_DATA3	T31	T5	mlb_data3	V2
J4-6	FPGA_DATA2	U35	U1	mlb_data2	V1
J4-7	FPGA_DATA5	T34	T2	mlb_data5	W1
J4-8	FPGA_DATA4	T33	T3	mlb_data4	V3
J4-9	FPGA_DATA7	R33	R5	mlb_data7	W3
J4-10	FPGA_DATA6	R31	T1	mlb_data6	W2
J4-11	FPGA_DATA9	R35	R3	mlb_data9	Y2
J4-12	FPGA_DATA8	R34	R4	mlb_data8	Y1
J4-13	FPGA_DATA11	P32	P2	mlb_data11	Y4
J4-14	FPGA_DATA10	P31	P3	mlb_data10	Y3
J4-17	FPGA_DATA13	P35	N5	mlb_data13	AA1
J4-18	FPGA_DATA12	P34	P1	mlb_data12	Y5
J4-19	FPGA_DATA15	N32	N3	mlb_data15	AA3

Table 6-4 Cross Reference: MAPI, U2 FPGA, U3 3401 (Continued)

MAPI Connectors		U2 FPGA Device		U3 3401 Device	
Pin	Default Signal	Ring Side	3401 Side	Signal	Pin
J4-20	FPGA_DATA14	N31	N4	mlb_data14	AA2
J4-21	FPGA_DATA17	M31	M1	mlb_data17	AA5
J4-22	FPGA_DATA16	M35	N2	mlb_data16	AA4
J4-23	FPGA_DATA19	M33	M4	mlb_data19	AB2
J4-24	FPGA_DATA18	M32	M5	mlb_data18	AB1
J4-25	FPGA_DATA21	L34	L2	mlb_data21	AC1
J4-26	FPGA_DATA20	L35	L1	mlb_data20	AB3
J4-29	FPGA_DATA23	L32	L5	mlb_data23	AC3
J4-30	FPGA_DATA22	L33	L3	mlb_data22	AC2
J4-31	FPGA_DATA25	K33	K3	mlb_data25	AD2
J4-32	FPGA_DATA24	K34	K2	mlb_data24	AD1
J4-33	FPGA_DATA27	K31	J1	mlb_data27	AE1
J4-34	FPGA_DATA26	J35	K4	mlb_data26	AD3
J4-35	FPGA_DATA29	J33	J2	mlb_data29	AD4
J4-36	FPGA_DATA28	J34	K5	mlb_data28	AF1
J4-37	FPGA_DATA31	H33	H3	mlb_data31	AE3
J4-38	FPGA_DATA30	H34	H2	mlb_data30	AE4
J4-41	FPGA_ADDR1	AL35	AF5	mlb_addr1	H2
J4-42	FPGA_ADDR0	AL34	AG2	mlb_addr0	H3
J4-43	FPGA_ADDR3	AH31	AF4	mlb_addr3	J4
J4-44	FPGA_ADDR2	AK33	AG1	mlb_addr2	H1
J4-45	FPGA_ADDR5	AJ33	AF2	mlb_addr5	J2
J4-46	FPGA_ADDR4	AK34	AF3	mlb_addr4	J3
J4-47	FPGA_ADDR7	AJ35	AE3	mlb_addr7	K5
J4-48	FPGA_ADDR6	AJ34	AE5	mlb_addr6	J1
J4-49	FPGA_ADDR9	AG31	AE1	mlb_addr9	K3
J4-50	FPGA_ADDR8	AH32	AE2	mlb_addr8	K4
J4-51	FPGA_ADDR11	AH34	AD4	mlb_addr11	K1
J4-52	FPGA_ADDR10	AH33	AD5	mlb_addr10	K2
J4-53	FPGA_ADDR13	AG32	AD2	mlb_addr13	L2
J4-54	FPGA_ADDR12	AH35	AD3	mlb_addr12	L3
J4-55	FPGA_ADDR15	AG34	AC5	mlb_addr15	M3
J4-56	FPGA_ADDR14	AG33	AD1	mlb_addr14	L1
J4-59	FPGA_ADDR17	AF33	AC2	mlb_addr17	M1
J4-60	FPGA_ADDR16	AF31	AC4	mlb_addr16	M2
J4-61	FPGA_ADDR19	AF35	AB4	mlb_addr19	N3
J4-62	FPGA_ADDR18	AF34	AC1	mlb_addr18	N4
J4-63	FPGA_ADDR21	AM35	AH3	mlb_addr21	N1
J4-64	FPGA_ADDR20	AE31	AB3	mlb_addr20	N2
J4-65	FPGA_ADDR23	AA32	AA4	mlb_addr23	P2

Table 6-4 Cross Reference: MAPI, U2 FPGA, U3 3401 (Continued)

MAPI Connectors		U2 FPGA Device		U3 3401 Device	
Pin	**Default Signal**	**Ring Side**	**3401 Side**	**Signal**	**Pin**
J4-66	FPGA_ADDR22	AA31	AA5	mlb_addr22	P3
J4-67	FPGA_ADDR25	Y32	Y5	mlb_addr25	R3
J4-68	FPGA_ADDR24	AA35	AA1	mlb_addr24	P1
J4-69	FPGA_ADDR27	Y34	Y3	mlb_addr27	R1
J4-70	FPGA_ADDR26	Y33	Y4	mlb_add[26	R2
J4-71	FPGA_ADDR29	W32	W4	mlb_addr29	T3
J4-72	FPGA_ADDR28	Y35	W5	mlb_addr28	T2
J4-73	FPGA_ADDR31	W34	W2	mlb_addr31	U2
J4-74	FPGA_ADDR30	W33	W3	mlb_addr30	U1
J4-77	MAPI_TA_B	C29			
J4-78	MAPI_TEA_B	E26			
J4-79	FPGA_TREQ_B	V35	V2	mlb_treq_b	D1
J4-80	FPGA_EB_B2	C27			
J4-81	FPGA_RW_B	J31	J5	mlb_rw_b	C1
J4-82	FPGA_EB_B3	B27			
J4-84	FPGA_EB_B1	D27			
J4-85	FPGA_CS_B0	A29			
J4-86	FPGA_EB_B0	E27			
J4-87	FPGA_CS_B1	E28			
J4-88	FPGA_OE_B	B30			
J4-89	FPGA_CS_B2	D28			
J4-91	FPGA_CS_B3	C28			
J4-95	MLB_CLKOUT	E18		MLB_CLKOUT	J26

Table 6-5 Cross Reference:U2 FPGA Device Pins

U2 Pin	Component - Pin	U2 Pin	Component - Pin	U2 Pin	Component - Pin	U2 Pin	Component - Pin
AA1	U3-P1	AH32	J4-50	AN24	J3-62	D23	U3-A6
AA4	U3-P2	AH33	J4-52	AN25	U3-C4	D27	J4-84
AA5	U3-P3	AH34	J4-51	AN27	J1-91	D28	J4-89
AA31	J4-66	AH35	J4-54	AP14	U3-C8	D29	U3-E1
AA32	J4-65	AJ1	U3-D5	AP15	U3-D9	E18	J4-95
AA35	J4-68	AJ33	J4-45	AP16	J1-83	E22	J1-79
AB3	U3-N2	AJ34	J4-48	AP19	J3-84	E23	J1-62
AB4	U3-N3	AJ35	J4-47	AP20	J3-52	E24	J3-27
AC1	U3-N4	AK33	J4-44	AP22	U3-B2	E25	U3-E6
AC2	U3-M1	AK34	J4-46	AP23	U3-D3	E26	J4-78
AC4	U3-M2	AL15	U3-E9	AP25	U3-D6	E27	J4-86
AC5	U3-M3	AL16	U3-B8	AP27	J1-93	E28	J4-87
AD1	U3-L1	AL17	J1-82	AR6	J3-40	E29	J3-72
AD2	U3-L2	AL19	J3-50	AR14	J1-87	H2	U3-AE4
AD3	U3-L3	AL21	J3-38	AR15	J1-85	H3	U3-AE3
AD4	U3-K1	AL22	J3-45	AR16	U3-C9	H4	U3-B5
AD5	U3-K2	AL23	J3-64	AR24	U3-C3	H32	J3-77
AE1	U3-K3	AL24	J3-28	AR25	J1-78	H33	J4-37
AE2	U3-K4	AL25	U3-E2	AR26	J3-46	H34	J4-38
AE3	U3-K5	AL27	J1-96	A23	U3-B12	H35	J1-27
AE5	U3-J1	AL32	J3-51	A24	U3-B7	J1	U3-AE1
AE31	J4-64	AL34	J4-42	A26	J3-86	J2	U3-AD4
AF2	U3-J2	AL35	J4-41	A28	J1-95	J5	U3-C1
AF3	U3-J3	AM15	J1-86	A29	J4-85	J31	J4-81
AF4	U3-J4	AM16	J1-84	B22	U3-D10	J33	J4-35
AF5	U3-H2	AM17	U3-B9	B23	J1-70	J34	J4-36
AF31	J4-60	AM20	J3-57	B24	J1-80	J35	J4-34
AF33	J4-59	AM22	U3-B3	B25	U3-G1	K2	U3-AD1
AF34	J4-62	AM23	U3-C2	B26	U3-D4	K3	U3-AD2
AF35	J4-61	AM24	U3-B6	B27	J4-82	K4	U3-AD3
AG1	U3-H1	AM25	J3-26	B28	J1-97	K5	U3-AF1
AG2	U3-H3	AM27	J1-98	B30	J4-88	K31	J4-33
AG31	J4-49	AM33	J3-53	C21	J2-73	K33	J4-31
AG32	J4-53	AM35	J4-63	C22	J1-72	K34	J4-32
AG33	J4-56	AN11	J1-73	C26	J3-88	L1	U3-AB3
AG34	J4-55	AN16	U3-A5	C27	J4-80	L2	U3-AC1
AH3	U3-N1	AN17	J1-81	C28	J4-91	L3	U3-AC2
AH31	J4-43	AN23	J3-66	C29	J4-77	L5	U3-AC3
AH4	U3-G2	AN22	J3-47	D21	J2-75	L32	J4-29

Table 6-5 Cross Reference:U2 FPGA Device Pins (Continued)

U2 Pin	Component - Pin	U2 Pin	Component - Pin	U2 Pin	Component - Pin	U2 Pin	Component - Pin
L33	J4-30	P2	U3-Y4	T5	U3-V2	V33	J3-41
L35	J4-26	P3	U3-Y3	T31	J4-5	V35	J4-79
M1	U3-AA5	P31	J4-14	T33	J4-8	W2	U3-U2
M4	U3-AB2	P32	J4-13	T34	J4-7	W3	U3-U1
M5	U3-AB1	P34	J4-18	U1	U3-V1	W4	U3-T3
M31	J4-21	P35	J4-17	U2	U3-U4	W5	U3-T2
M32	J4-24	R3	U3-Y2	U4	U3-U3	W32	J4-71
M33	J4-23	R4	U3-Y1	U31	J3-48	W33	J4-74
M35	J4-22	R5	U3-W3	U32	J4-4	W34	J4-73
N2	U3-AA4	R31	J4-10	U33	J4-3	Y3	U3-R1
N3	U3-AA3	R33	J4-9	U35	J4-6	Y4	U3-R2
N4	U3-AA2	R34	J4-12	V1	U3-F5	Y5	U3-R3
N5	U3-AA1	R35	J4-11	V2	U3-D1	Y32	J4-67
N31	J4-20	T1	U3-W2	V4	U3-E3	Y33	J4-70
N32	J4-19	T2	U3-W1	V5	U3-D2	Y34	J4-69
P1	U3-Y5	T3	U3-V3	V31	J3-39	Y35	J4-72

Index

R

S

U

W

Revision History

Revision Number	Date	Author	Summary of Changes
Original	Apr 2000	MTC DDOC	Original document.

Revision History

This manual is a product of the Motorola M•CORE Technology Center Design Documentation team. Technical writing, illustration, and production editing performed with Adobe® Framemaker® running on multiple platforms. Printed by Ken Cook, Inc. in Milwaukee, Wisconsin.