Intel® IQ80331 I/O Processor Evaluation Platform

Board Manual - DDR-II 400 MHz SDRAM

September 2004

Document Number: 273948-004US

INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL® PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER, AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

Intel products are not intended for use in medical, life saving, life sustaining applications.

Intel may make changes to specifications and product descriptions at any time, without notice.

Designers must not rely on the absence or characteristics of any features or instructions marked “reserved” or “undefined.” Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them.

The Intel® 80331 I/O Processor may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Contact your local Intel sales office or your distributor to obtain the latest specifications and before placing your product order.

Copies of documents which have an ordering number and are referenced in this document, or other Intel literature may be obtained by calling 1-800-548-4725 or by visiting Intel's website at http://www.intel.com.

Contents

Figures

Tables

Revision History

Date	Revision	Description
September 2004	004	Updated Switch and Jumper settings, Various Typo corrections.
April 2004	003	Revised for DDR-II 400 MHz SDRAM feature.
April 2004	002	Revised Appendix B.
December 2003	001	Initial Release.

Introduction 1

1.1 Document Purpose and Scope

This document describes the Intel® IQ80331 I/O processor evaluation platform board (IQ80331) using DDR-II 400 MHz SDRAM. The IQ80331 is intended for rapid intelligent I/O development. The design is based on the Intel® 80331 I/O processor (80331), which is a multi-function device, that integrates the Intel XScale® core (ARM* architecture compliant) with intelligent peripherals, including a PCI bus application bridge. For more information on the 80331, please see http://developer.intel.com/design/iio/iop331.htm.

1.2 Related Documents

Table 1. Intel® 80331 I/O Processor Related Documentation List

Document	Number
Intel® 80331 I/O Processor Developer's Manual	273942
Intel® 80331 I/O Processor Datasheet	273943
Intel® 80331 I/O Processor Design Guide	273823
Intel® 80331 I/O Processor Specification Update	273930
Intel® 80331 I/O Processor JTAG Support White Paper	273961
Intel® 80331 I/O Processor Product Brief	253413
Intel® 80321 Software Conversion to the Intel® 80331 I/O Processor Application Note	273914
Intel® Flash Recovery Utility (FRU) Reference Manual	273551
PCI Local Bus Specification, Revision 2.3	http://www.pcisig.com/specifications
PCI-X Addendum to the PCI Local Bus Specification, Revision 1.0a	

Intel documentation is available from the local Intel Sales Representative or Intel Literature Sales. To obtain Intel literature write to or call:

Intel Corporation
Literature Sales
P.O. Box 5937
Denver, CO 80217-9808

(1-800-548-4725) or visit the Intel website at http://www.intel.com

1.3 Electronic Information

Table 2. Electronic Information

Support Type	Location/Contact
The Intel World-Wide Web (WWW) Location:	http://www.intel.com
Customer Support (US and Canada):	1-916-377-7000

1.4 Component References

Table 3 provides additional information on the major components of the IQ80331.

Table 3. Component Reference

Component	Part Number	Additional Information
Intel StrataFlash® Memory	28F640J3C	• Manufacturer: Intel Corporation • URL: http://developer.intel.com/design/flcomp/prodbref/298044.htm
Intel® Gigabit Ethernet Controller	82545EM	• Manufacturer: Intel Corporation • URL: http://developer.intel.com/design/network/products/lan/controllers/82545.htm
Rotary Switch	DR FC 16	• Manufacturer: Grayhill* • URL: http://embrace.grayhill.com/embrace/Item/ASP/Item-Detail.asp?PartNo=94HAB16W&CatalogGroupID=Series94HBinaryCoded&GroupDisplayLabel=&RestSes=No
Hex Display	HDSP-A103	• Manufacturer: Agilent Technologies* • URL: http://www.semiconductor.agilent.com/cgi-bin/morpheus/home/home.jsp?pSection=LED
AudioBuzzer	DMT 1206 SMT	• Manufacturer: RDI* • URL: http://www.rdi-electronics.com/products/Audio/DMT-1206-SMT.html
NVSRAM	STK14C88-3 N 35	• Manufacturer: SIMTEK* • URL: http://www.simtek.com/product-information/datasheets/256K-PDF/STK14C88-3.pdf
CPLD	XC9572XL - 10TQ100C	• Manufacturer: XILINK* • URL: http://www.xilinx.com/bvdocs/publications/ds057.pdf
Temperature Sensor	LM75CIMX-3	• Manufacturer: National* • URL: http://www.national.com/pf/LM/LM75.html
Programmable Reset IC	MAX6306UK29D3	• Manufacturer: Maxim* • URL: http://www.maxim-ic.com/quick_view2.cfm/qv_pk/1524
Registered Buffer	IDT74SSTU32864BF	• Manufacturer: IDT* (Integrated Device Technology) • URL: http://www1.idt.com/pcms/products.taf?catID=97&genID=74SSTU32864
Programmable PLL	IDTCSPU877BV	• Manufacturer: IDT* (Integrated Device Technology • URL: http://www1.idt.com/pcms/products.taf?catID=112&genID=CSPU877
256 bit 1-wire EEPROM	DS2430A_TSOC	• Manufacturer: Maxim* • URL: http://www.maxim-ic.com/quick_view2.cfm?qv_pk=2913
3.3V Transceiver	MAX561	• Manufacturer: Maxim* • URL: http://www.maxim-ic.com/quick_view2.cfm?qv_pk=1544
Battery Charger	ADP3801	• Manufacturer: Analog Devices* • URL: http://www.analog.com/UploadedFiles/Data_Sheets/308746738ADP3801_2_0.pdf

1.5 Terms and Definitions

Table 4. Terms and Definitions

Acronym/Term	Definition
ARM	Refers to both the microprocessor architecture and the company that licenses it.
IOP	I/O processor
ICE	In-Circuit Emulator – A piece of hardware used to mimic all the functions of a microprocessor.
JTAG	Joint Test Action Group – A hardware port supplied on Intel XScale® microarchitecture evaluation boards used for in-depth testing and debugging.
PPCI-X	Primary PCI-X.
PSU	Power Supply Unit
SPCI-X	Secondary PCI-X.

1.6 Intel® 80331 I/O Processor

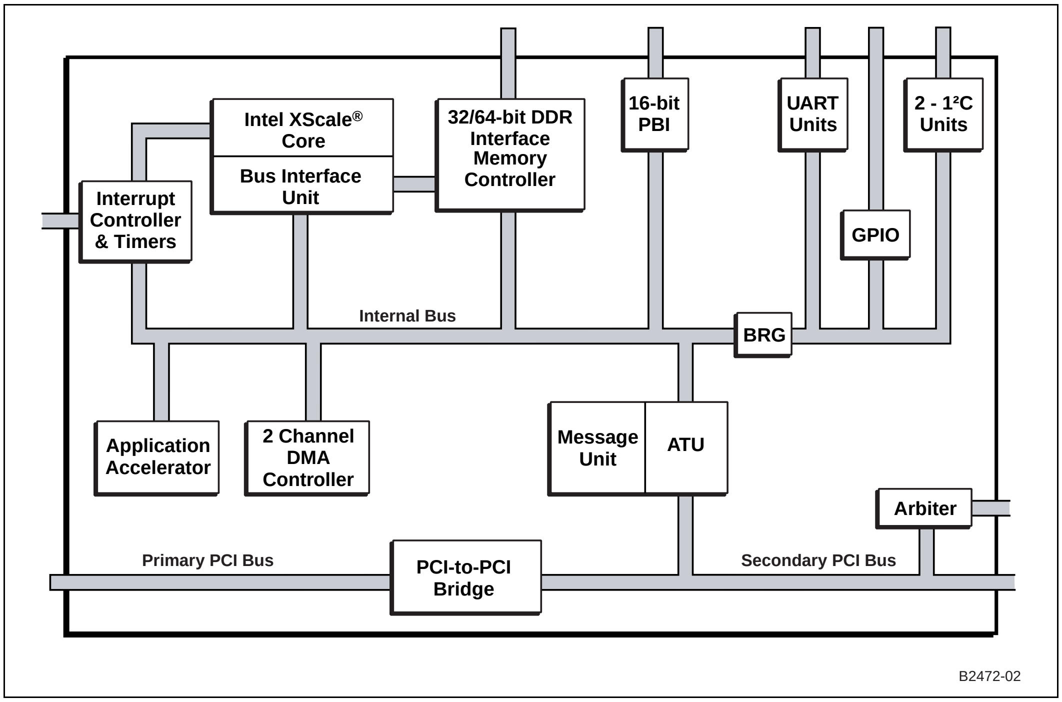
The 80331 combines the Intel XScale® core with powerful new features to create an intelligent I/O processor. This multi-function PCI device integrates a PCI-to-PCI Bridge and is fully compliant with the *PCI Local Bus Specification*, Revision 2.3, the *PCI-X Addendum to the PCI Local Bus Specification*, Revision 1.0a, and the *PCI Bridge Specification*, Revision 1.1.

The 80331 consolidates into a single system:

- Intel XScale® core
- PCI-to-PCI Bridge supporting PCI-X interfaces on the Primary and Secondary bus
- Address Translation Unit (ATU)
- High-Performance Memory Controller
- Interrupt Controller with 13 external interrupt inputs
- Two Direct Memory Access (DMA) Controller
- Peripheral Bus Interface (PBI) Unit
- Performance Monitor Unit (PMU)
- Application Accelerator Unit (AAU)
- Two I^{2}C Bus Interface Units (BIU)
- Two 16550 Compatible UARTs with flow control (4 pins)
- Eight General Purpose Input Output (GPIO) Ports

For more in depth information in regards to the 80331, please see the *Intel® 80331 I/O Processor Developer's Manual*.

Figure 1. Intel® 80331 I/O Processor Block Diagram



The 80331 is an integrated processor that addresses the needs of intelligent I/O applications and helps reduce intelligent I/O system costs.

The instruction cache is 32 Kbytes (KB) in size and is 32-way set associative. Also, the core processor includes a data cache that is 32 KB and is 32-way set associative and a mini data cache that is 2 KB and is 2-way set associative.

The 80331 dual-ported memory controller can be configured for DDR SDRAM at 333 MHz (with a core processor speed of 500 MHz or 667 MHz) and Double Data Rate (DDR-II) SDRAM at 400 MHz (with a core processor speed of 500 Mhz, or 800 MHz). However, this manual only covers the configuration for DDR-II SDRAM at 400 MHz and with a core processor speeds of 500 MHz or 800 Mhz.

1.7 Intel® IQ80331 I/O Processor Evaluation Platform Board Features

Table 5. Summary of Features

Feature	Definition
Battery Backup Unit:	Battery back up circuit for SDRAM – 64 MB for 18 hours.
Ethernet	Intel® 82545EM Gigabit Ethernet Controller
Flash ROM:	8 MB Flash ROM 3.3 V – 16-bit Flash I/F.
Form Factor:	PCI-X card (312 X 107 mm)
General Purpose I/O:	GPIO Pins are used as described in the appropriate section in this document
Hex Display:	Two 7-segment Hex LED displays.
JTAG Port:	ARM compliant JTAG Header.
Logic Analyzer:	Logic analyzer connectors on the DDR-II SDRAM 400 MHzinterface. Interposer Card may be used for the memory bus – Information supplied separately.
Memory:	256 MB (512 Mb x 16) DDR-II SDRAM 400 MHz soldered onto the board. ECC Registered
Onboard Power:	Board sources +1.25 V, +2.5 V, +3.3 V, +5 V, +12 V, and -12 V from primary PCI connector. All core voltages are derived from 3.3 V supply. Auxiliary power for the Secondary PCI slot.
Power LED:	Power on (green).
Primary PCI:	64 bits 3.3 V[a] 133/100/66 MHz PCI-X or PCI 66 MHz
Secondary PCI:	1 x 64-bit PCI-X connector - 133 MHz. Intel® 82545EM Gigabit Ethernet Controller also on the Secondary PCI.
Serial Port:	Dual RJ11 serial port connectors. The 80331 has two integrated UART serial ports which are 16550 compatible.

[a] Please note that the Intel® IQ80331 I/O processor evaluation platform board is **NOT** 5 V tolerant.

Getting Started 2

The Intel® IQ80331 I/O processor evaluation platform board (IQ80331) is a software development environment for Intel® 80331 I/O processor[1] (80331).

2.1 Kit Contents

The IQ80331 Kit contains the following items:

- IQ80331 with 400 MHz DDR-II SDRAM.
- Code|Lab* Development Environment from Accelerated Technology Incorporated* (ATI).
- JTAG MPDemon Emulation unit.
- Serial Cable and RJ11 Adapter.

2.2 Hardware Installation

Warning: Static charges can severely damage the boards. Be sure to properly ground before removing the board from the anti-static bag.

2.2.1 First-Time Installation and Test

For first-time installation, visually inspect the IQ80331 for any damage made during shipment. Follow the host system manufacturer instructions for installing a PCI adapter. The board is a full-length PCI/PCI-X adapter and requires a PCI/PCI-X slot free from obstructions.

2.2.2 Power and Backplane Requirements

The IQ80331 requires a 3.3 V supply coming through the PCI/PCI-X primary connector. The board can be plugged into either a backplane or a desktop 64-bit PCI/PCI-X slot. When using a backplane, an ATX rated power supply is required. The IQ80331 only draws from the 3.3 V line of the power supply. Most ATX power supply units (PSUs) regulate off the 5 V signal. When there is nothing drawing from the 5 V signal most ATX PSU do not supply the 3.3 V correctly. To overcome this, it is recommended to put a load on the 5 V line of the PSU. An old IDE Hard drive can be used for this.

Warning: The PCI buses on the IQ80331 are **NOT** 5 V tolerant. The primary PCI bus **MUST** be plugged into a 3.3 V PCI/PCI-X slot.

Caution: When plugging the power supply into the backplane, make sure that the power supply is disconnected from the mains. Most ATX PSUs supply 5 V standby current even when turned Off, backplane damage is possible.

The IQ80331 has an auxiliary power receptacle (J1A1, see Section 3.9.4, “Connector Summary”) that is used to power the secondary PCI-X slot. This connector is compatible with a standard ATX hard drive power connector.

Caution: Before connecting power to the entire system, verify that the auxiliary system power to the secondary PCI-X slot and the main power to the IQ80331 are both connected. Both power rails should come up at the same time. When there is not a card plugged into the secondary PCI-X slot, then the auxiliary power can be left unconnected.

[1] ARM* architecture compliant.

2.3 Factory Settings

Make sure that the switch/jumper settings are set to proper positions as explained in Section 3.9, “Switches and Jumpers” on page 33.

2.4 Development Strategy

2.4.1 Supported Tool Buckets

For developing and debugging software application, the production version of the IQ80331 kit includes the Code|Lab Development Environment. Support for the Code|Lab development environment is available from ATI. Please refer to the enclosed package.

The kit also contains evaluation copies for several Software Development Tools. These tools are for evaluation purposes and do not include any support. Please contact the vendor directly for additional information and support. They include, but not limited to:

- RedHat* GNUPro* tools
- ARM RealView Developer Suite
- WindRiver* VxWorks* RTOS and Tornado* Development Tools
- Wasabi Systems NetBSD* ODS
- TimeSys* Linux* RTOS
- ATI, Nucleus Plus* RTOS and Development Tools

It is recommended to frequently check with the Intel Development Tools web site located at http://www.intel-ioprocessortools.com/kshowcase/view/ for current software and vendor updates.

2.4.2 Contents of the Flash

The production version of the board contains an image for RedHat* RedBoot target monitor.

2.5 Target Monitors

2.5.1 Red Hat RedBoot

RedBoot is an acronym for “Red Hat Embedded Debug and Bootstrap”, and is the standard embedded system debug/bootstrap environment from Red Hat, replacing the previous generation of debug firmware: CygMon and GDB stubs. It provides a bootstrap environment for a range of embedded operating systems, such as embedded Linux and eCos*, and includes facilities such as network downloading and debugging. It also provides a simple Flash file system for boot images.

RedBoot provides a set of tools for downloading and executing programs on embedded target systems, as well as tools for manipulating the target system's environment. It can be used for both product development (debug support) and for end product deployment (Flash and network booting).

Here are some highlights of RedBoot capabilities:

- Boot scripting support.
- Simple command line interface for RedBoot configuration and management, accessible via serial (terminal) or Ethernet (telnet) (see Section 2.6.4, “GNUPro GDB/Insight” on page 19).
- Integrated GDB stubs for connection to a host-based debugger (GBD/Insight) via serial or Ethernet. (Ethernet connectivity is limited to local network only).
- Attribute Configuration - user control of aspects such as system time and date (when applicable), default Flash image to boot from, default fail-safe image, static IP address, etc.
- Configurable and extensible, specifically adapted to the target environment.
- Network bootstrap support including setup and download, via BOOTP, DHCP and TFTP.
- X/Y-Modem support for image download via serial port.
- Power On Self Test.

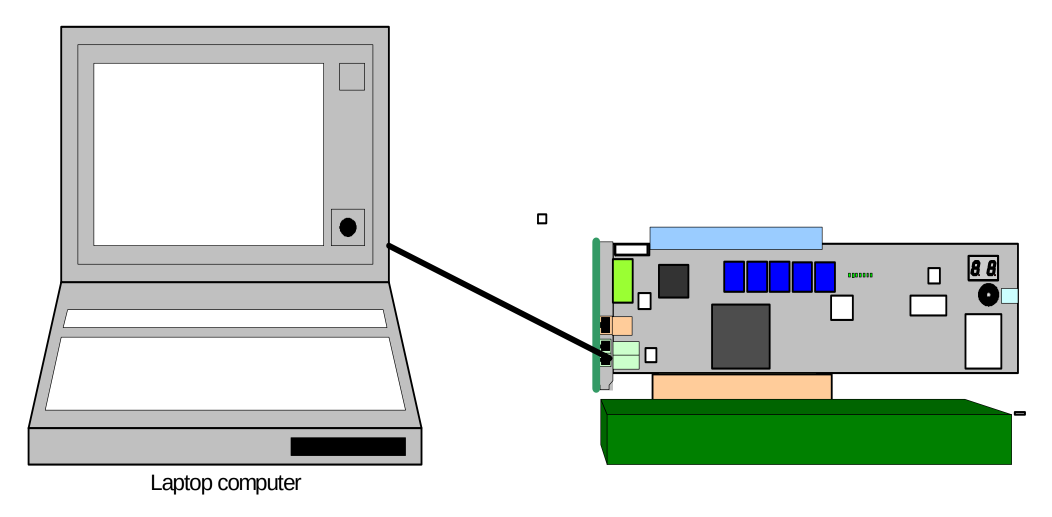
2.6 Host Communications Examples

How to communicate to the host.

2.6.1 Serial-UART Communication

Using a serial connection, while the IQ80331 is in a backplane or a host computer.

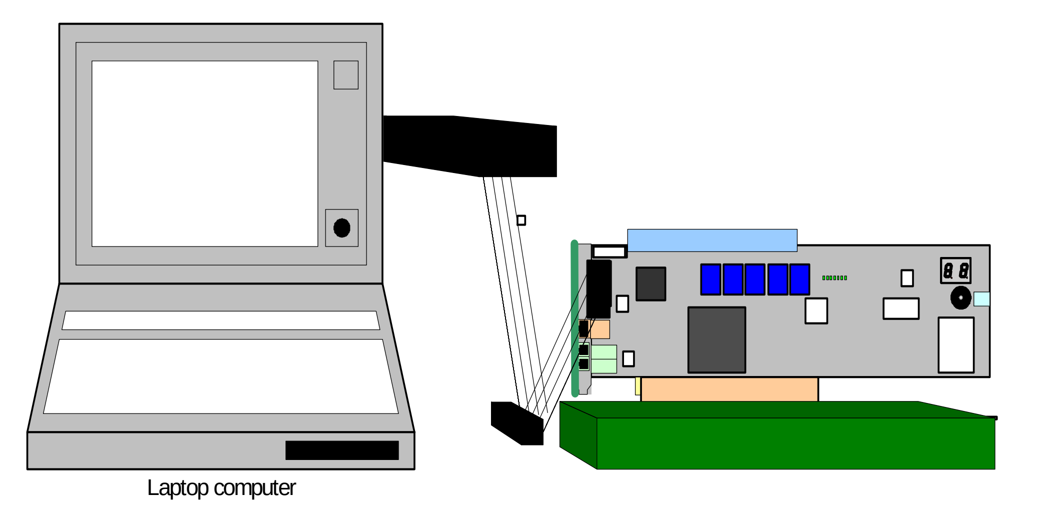
Figure 2. Serial-UART Communication



2.6.2 JTAG Debug Communication

Using a JTAG Emulator, while the IQ80331 is plugged into a backplane or a host computer:

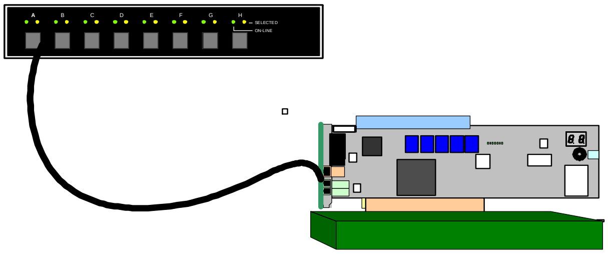
Figure 3. JTAG Debug Communication



2.6.3 Ethernet Communication

A standard network cable can be used to access the IQ80331 Ethernet connection. Redboot allows remote booting off a BOOTP server, Figure 4.

Figure 4. Communication via Ethernet



2.6.4 GNUPro GDB/Insight

2.6.4.1 Communicating with RedBoot

Hardware Setup:

- Host with UNIX*/Linux* or Win32* installed.
- IQ80331 with serial cable.
- Redhat RedBoot monitor Flashed to the platform board.

Recommended Mapping of UART Ports to Host Com Ports:

- Host port connected to the platform board UART.

The following communication tools can be used:

- Win32 using HyperTerminal*.
- UNIX using Kermit*.
- Linux using Minicom*.
- Solaris* using Tip*.

RedBoot Monitor startup:

Description: Terminal emulator runs on host and communicates with the board via the serial cable.

Start: Power up the IQ80331. While the 'reset' is asserted, the two 7-segment LEDs sequentially display “SL” (Scrub loop) and then “A1”. When RedBoot is successfully booted, then it displays the characters “A1” on the LEDs. When the final state of “A1” does not occur, then cycle the power again.

The time for reset is approximately 1 or 2 seconds.

Win32 on Host Connecting with HyperTerminal*.

To bring up a HyperTerminal session on a Win32 platform: Go to Start, Programs, Accessories, Communications, HyperTerminal.

- HyperTerminal setup screens:
    - — “Connection Description” Panel:
        - Enter name.
    - — “Connect To” Panel:
        - Select host com2 port (or whichever port is being used).
    - — Port Settings:
        - Bits per second: 115200
        - Data Bits: 8
        - Parity: none
        - Stop Bits: 1
        - Flow Control: none
    - — Start HyperTerminal:
        - Select Call from HyperTerminal panel.
    - — Reset or power up IQ80331 board.
    - — The Host screen reads:

```
RedBoot(tm) debug environment - built dd:mm:yy, Mon dd 2004
Platform: IQ80331
Copyright (C) 2004, Red Hat, Inc.
RAM: 0xa0000000-0xa2000000
FLASH: 0x00000000 - 0x00800000, 64 blocks of 0x00020000 bytes each.
IP: 192.168.0.1, Default server: 0.0.0.0
RedBoot>
```

For further information on the GDB/Insight Debugger, refer to the content of the GNUPro CD and/or the GNUPro Debugging Tools manual. This setup assumes that RedBoot* is Flashed on the board.

2.6.4.2 Connecting with GDB

Below are the GDB commands entered from the command prompt. Be sure system path is set to access “xscale-elf-gdb.exe”. File name in example “hello”. Bold type represents input by user:

>**xscale-elf-gdb -nw hello**[1]

- Start GDB executable, loads debug information and symbols.

(GDB) **set remotebaud 115200**

- Set baud rate for the IQ80331.

Connect COM port:

- When using Windows command prompt:
  (GDB) **target remote com1**
  Example: screen output from board to host (GDB) target remote com1:
  ```
  Remote debugging using com1.
  (GDB)
  ```
- When using Linux
  (GDB) **target remote /dev/ttyS0**

(GDB) **load**

- Load the program to the board, may have to wait a few seconds.

(GDB) **break main**

- Set breakpoint at main.

(GDB) **continue**

- Start the program using 'continue' verse the usual 'run'.
- Program hits break at main() and wait.

[1] To be supplied separately.

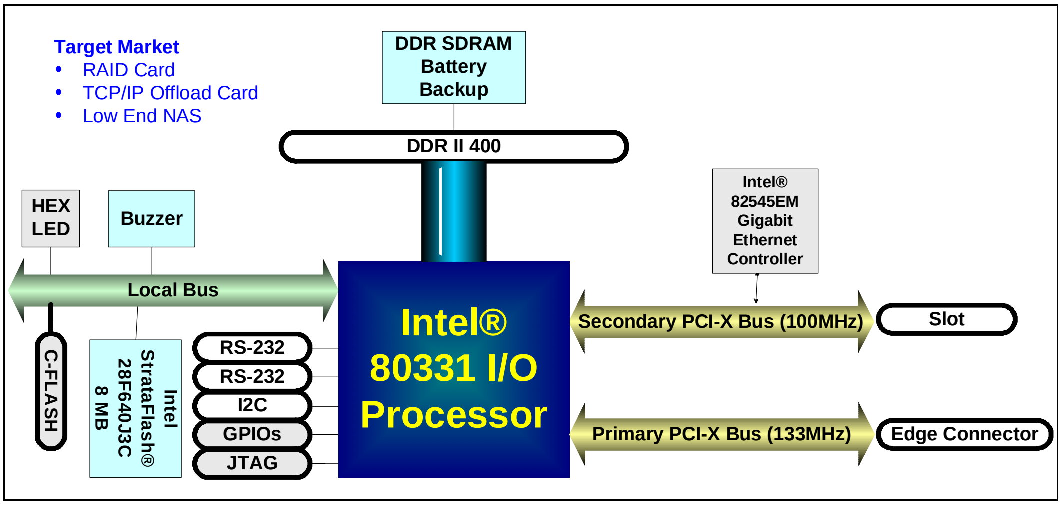
This Page Left Intentionally Blank

Hardware Reference Section 3

3.1 Functional Diagram

Figure 5 shows the functional block for the IQ80331.

Figure 5. Functional Block Diagram



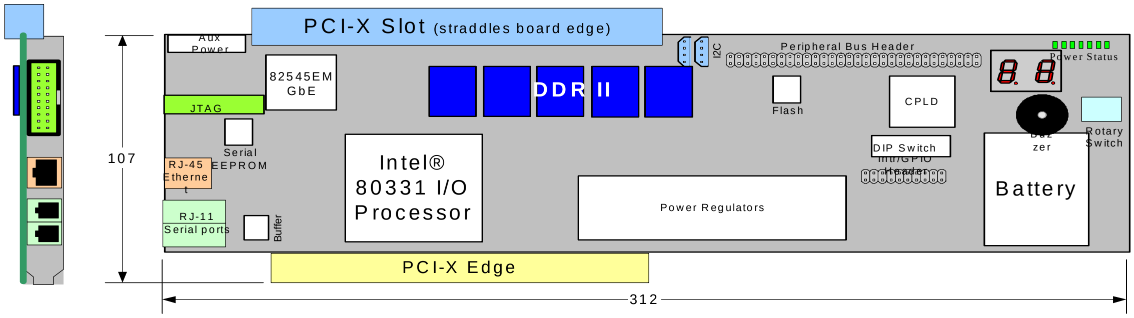
3.2 Board Form-Factor/Connectivity

Table 6 summarizes the form-factor and connectivity features for the IQ80331.

Table 6. Form-Factor/Connectivity Features

Description
The Intel® IQ80331 I/O processor evaluation platform board is a 64-bit PCI card with form factor depicted by Figure 6.
The IQ80331 connects to the Primary PCI-X (PPCI-X) bus a PCI-X.
The IQ80331 has one PCI-X expansion slot.
The IQ80331 has two serial ports and one RJ-45 Ethernet port.
The IQ80331 has one JTAG port compliant with ARM Multi-ICE 20-pin connector standard. The JTAG is targeted for the Intel XScale® core and the CPLD, and is used for software debug purposes.

Figure 6. Board Form Factor



3.3 Power

The IQ80331 draws power from the PCI-X bus. The power requirements for the IQ80331 are shown in Table 7 below. The numbers do not include the power required by a PCI-X card mounted on the expansion slot.

Table 7. Power Features

Voltage Rails	Typical Current	Maximum Current
+3.3 V	TBD mA	6237 mA
+5 V	TBD mA	6 mA
+12 V	TBD mA	103 mA

Note: Maximum values were calculated, not measured. Does not include additional power required by a PCI-X card mounted on the expansion slot.

3.4 Memory Subsystem

The memory subsystem consists of the DDR-II SDRAM as well as the Flash memory.

3.4.1 DDRII SDRAM

The DDR-II SDRAM interface consists of a 64-bit wide data path to support up to 3.2 Gbytes/sec throughput. An 8-bit Error Correction Code (ECC) is stored into the DDR-II SDRAM array along with the data and is checked when the data is read.

The Intel® IQ80331 I/O processor evaluation platform board features on board registered DDR-II 400 MHz SDRAM, arranged 512 Mbit x16 in density (256 MB), and with ECC.

3.4.1.1 Battery Backup

Battery backup is provided to save any information in DDR during a power failure. The evaluation board contains a 4 V Li-ion battery, a charging circuit and a regulator circuit.

DDR-II technology provides enabling data preservation through the self-refresh command. When the processor receives an active Primary PCI-X reset, the self-refresh command issues, driving SCKE signals low. Upon seeing this condition, the board logic circuit holds SCKE low before the processor loses power. Batteries maintain power to DDR-II and logic, to ensure self-refresh mode. When the circuit detects PRST# returning to inactive state, the circuit releases the hold on SCKE. Removing the battery can disable the battery circuit. When the battery remains in the platform and it is de-powered and/or removed from the chassis, the battery maintains DDR-II for about four hours. Once power is reapplied, the battery is fully charged.

The CPLD contains information in regards to the battery status. Please see Section 3.6.7, “Battery Status” on page 30 for more details.

3.4.2 Flash Memory Requirements

Total Flash memory size is 8 MB.

Table 8. Flash Memory Requirements

Description
Intel® IQ80331 I/O processor evaluation platform board Total Flash size is 8 MB
IQ80331 Flash technology is based on Intel StrataFlash® family
IQ80331 Flash uses a 16-bit interface
IQ80331 Flash utilizes the 80331 Peripheral Bus
IQ80331 May be programmed using the PCI-X interface – Flash Recovery Utility (FRU) Utility
IQ80331 May be programmed using a RAM based software target monitor – Redhat RedBoot and ARM Firmware Suite
IQ80331 May be programmed using a JTAG emulation/debug device

3.5 Interrupt Routing

The IQ80331 Interrupt routing.

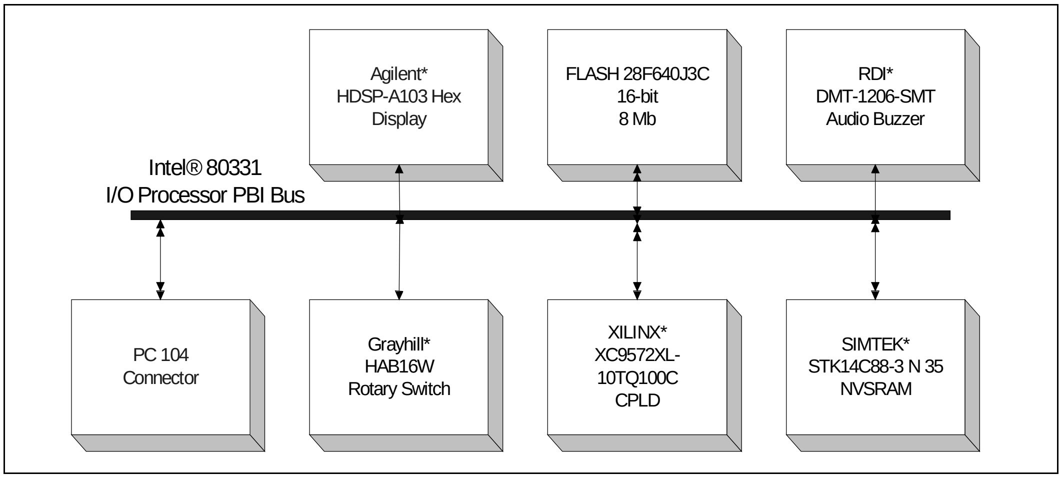
Table 9. External Interrupt Routing to Intel® 80331 I/O Processor

Interrupt	System Resource
HPI#	Temperature Sensor, Header
S_INTA#	PCI-X Slot INTB#, Header
S_INTB#	PCI-X Slot INTC#, Header
S_INTC#	PCI-X Slot INTD#, Header
S_INTD#	PCI-X Slot INTA#, Header
P_INTA#	PCI-X Card Edge INTA#, Header
P_INTB#	PCI-X Card Edge INTB#, Header
P_INTC#	PCI-X Card Edge INTC#, Header
P_INTD#	PCI-X Card Edge INTD#, Header

3.6 Intel® IQ80331 I/O Processor Evaluation Platform Board Peripheral Bus

The IQ80331 populates the peripheral bus as depicted by Figure 7.

Figure 7. Intel® IQ80331 I/O Processor Evaluation Platform Board Peripheral Bus Topology



The devices on the bus include Flash ROM, audio buzzer, CPLD, HEX display, NVSRAM, and rotary switch.

Table 10. Peripheral Bus Features

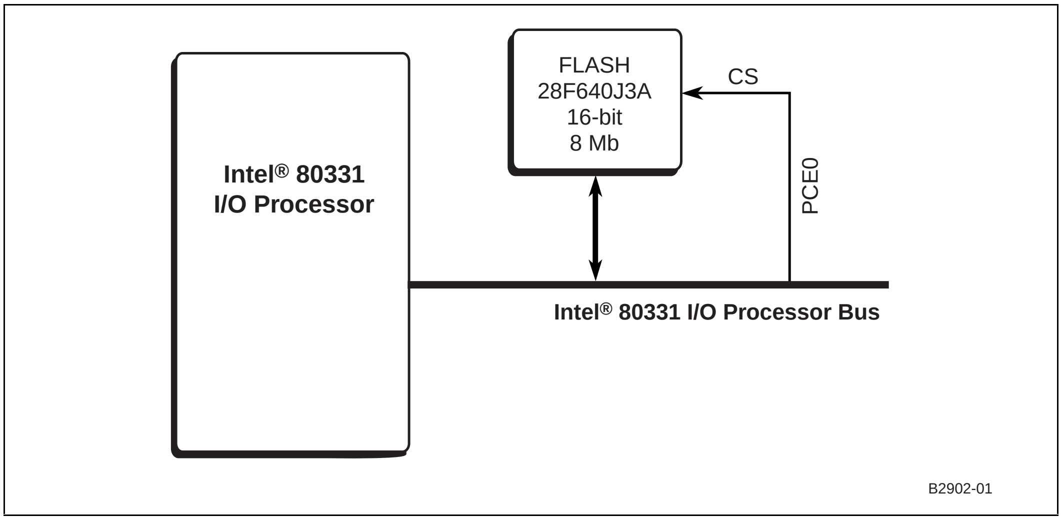
Description
The bus width can be 8-bit or 16-bit and runs at 66 MHz.
The bus is utilized for attaching debug and Flash devices.
The interfaces/devices that are utilized include an audio buzzer, CPLD, a rotary switch, a HEX Display, and NVSRAM.

3.6.1 Flash ROM

Table 11. Flash ROM Features

Description
Flash is an Intel StrataFlash® technology – Part number: 28F640J3C
Flash size is 8 MB
The connection to the peripheral bus is depicted by Figure 8

Figure 8. Flash Connection on Peripheral Bus



3.6.2 UART

The 80331 has two integrated UARTs. Each asynchronous serial port supports all the functions of a 16550 UART. The UART signals are connected to a dual RS-232 buffer and then to a RJ-11 serial port connector mounted on the bracket of the evaluation board. The serial ports and GPIO signals are muxed on the same pins. Jumper J1D2, located next to the serial port buffer can disable the buffer to allow the signals to be used as GPIO signals. Please see Section 3.9.3, "Jumper Summary" on page 34 for more details.

3.6.3 Non-Volatile RAM

In addition to the 8 MB Flash device, the Intel® IQ80331 I/O processor evaluation platform board has a separate 32 K x 8 non-volatile RAM device on the peripheral bus. The NVRAMs address range is from CE87 0000 to CE87 FFFF (in hex). Please see Section 4.2.2, "Peripheral Bus Memory Map" on page 42 for more details.

3.6.4 Audio Buzzer

The IQ80331 has an audio buzzer that is turned on and off by writing to the Buzzer Control Register located in the CPLD. Jumper J9C2 enables the buzzer and jumper J9C1 adjusts the volume. Please see Section 3.9.3, "Jumper Summary" on page 34 for more details. The audio buzzer's address range is from CE86 0000 to CE86 FFFF (in hex). Please see Section 4.2.2, "Peripheral Bus Memory Map" on page 42 for more details.

3.6.5 HEX Display

The two pairs of Agilent* HDSP-A103 seven segment LEDs are used for displaying POST codes or other software generated debug codes. Both HEX displays are individually addressed. The left HEX display address range is CE84 0000 to CE84 FFFF (in hex). The right HEX display address range is CE85 0000 to CE85 FFFF (in hex). Please see Section 4.2.2, "Peripheral Bus Memory Map" on page 42 for more details.

3.6.6 Rotary Switch

The IQ80331 provides a Rotary Switch for the user to select from different boot-up flavors. Please see Section 4.2.2, "Peripheral Bus Memory Map" on page 42 for more details on addressing the rotary switch.

Table 12. Rotary Switch Requirements

Description
Rotary switch has a 4-bit resolution (16 positions).
The connection to the peripheral bus is depicted by Figure 7.

3.6.7 Battery Status

A CPLD on the Intel® IQ80331 I/O processor evaluation platform board provides the following status for the battery. Please see Section 4.2.2, “Peripheral Bus Memory Map” on page 42 for more details on addressing the CPLD.

Table 13. Battery Status Buffer Requirements

BIT	Read/ Write	Name	Description
0	R	Battery Present	• 0 = No backup battery • 1 = Battery backup is present
1	R	Battery Charged	• 0 = Battery is not fully charged • 1 = Battery is fully charged
2	R	Battery Discharged	• 0 = Battery backup is not fully discharged • 1 = Battery backup is fully discharged
3	R/W	Battery Enable	• 0 = Disable battery backup • 1 = Enable battery backup
4-7	N/A	Reserved	Undefined

3.7 Debug Interface

3.7.1 Console Serial Port

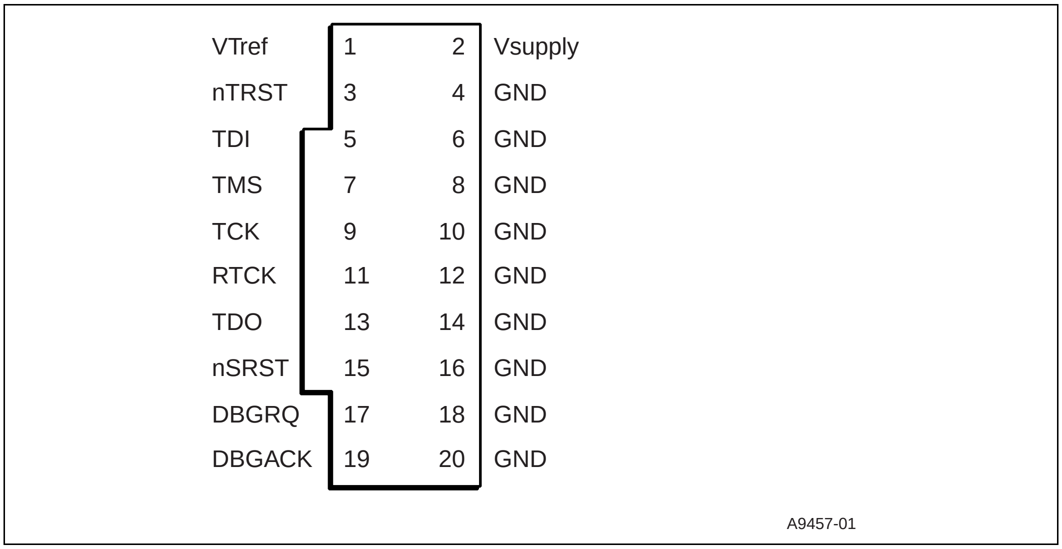
The platform has two serial ports for debug purposes as described in Section 3.6, “Intel® IQ80331 I/O Processor Evaluation Platform Board Peripheral Bus” on page 27.

3.7.2 JTAG Debug

The IQ80331 has a 20-pin JTAG connector that is in compliant with ARM* Multi-ICE guidelines.

3.7.2.1 JTAG Port

Figure 9. JTAG Port Pin-out



VTref	1	2	Vsupply
nTRST	3	4	GND
TDI	5	6	GND
TMS	7	8	GND
TCK	9	10	GND
RTCK	11	12	GND
TDO	13	14	GND
nSRST	15	16	GND
DBGRQ	17	18	GND
DBGACK	19	20	GND

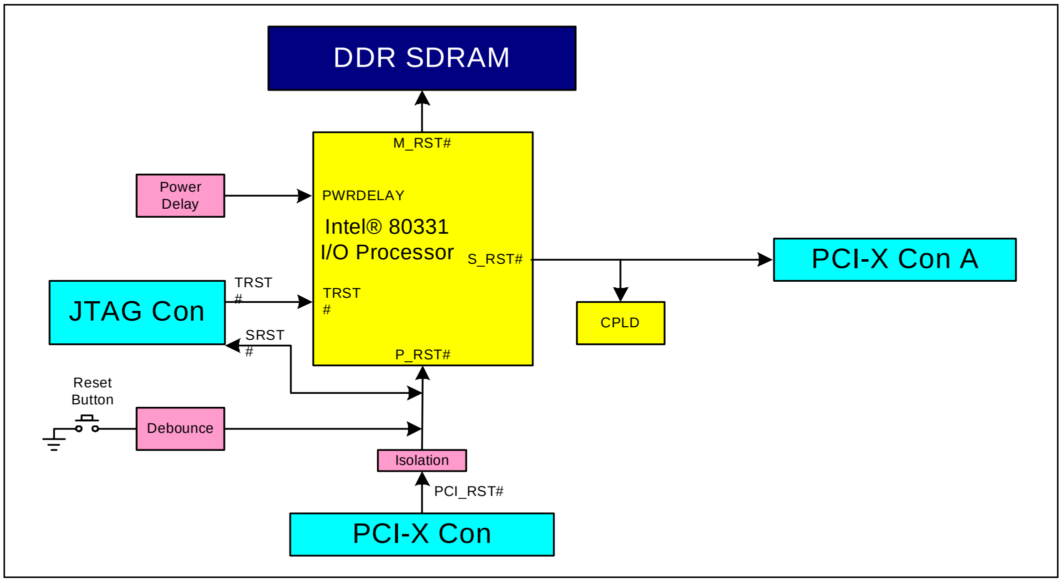
3.8 Board Reset Scheme

Figure 10 depicts the reset scheme for the IQ80331. Table 14 list the reset schemes for the IQ80331.

Table 14. Reset Requirements/Schemes

Description
Primary PCI reset, resets all devices on the board. It occurs during the power-up.
The SRST signal from the JTAG connector is a bi-directional signal that can force a reset similar to the power-up reset on the board.

Figure 10. RESET Sources



3.9 Switches and Jumpers

3.9.1 Switch Summary

Please note that the term ‘open’ refers to the individual pins of Switch S7C1being pushed in at the bottom (small dot on pin away from the ‘open’ label on the switch). The term ‘closed’ refers to the pin being pushed in at the top. Please see Figure 11, “Default Switch Settings” on page 33, for more details.

Table 15. Switch Summary

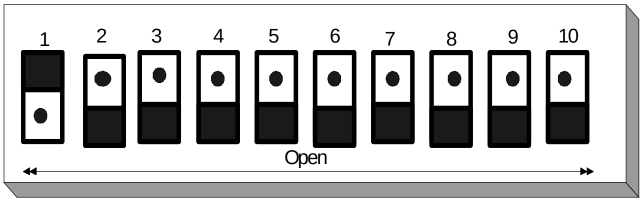
Switch	Association	Description	Factory Default
S7C1-1	SPCI-X Bus	Sets secondary PCI bus to 100 MHz	Closed
S7C1-2	IOP	Reset: Sets IOP Reset-Mode operation	Open
S7C1-3	IOP	Retry: ATU/PCI configuration cycle-Mode operation	Open
S7C1-4	PPCI-X Bus	BRG_EN: Enables or disables bridge	Open
S7C1-5	SPCI-X Bus	PRIVDEV: Enables private or normal devices	Open
S7C1-6	SPCI-X Bus	PRIVMEM: Enables normal or private memory addressing	Open
S7C1-7	PPCI-X Bus	Enables 64-bit or 32-bit on primary PCI-X bus	Open
S7C1-8		Spare	Open
S7C1-9		Spare	Open
S7C1-10	Flash	Sets flash width (x8 or x16) Enable	Open

3.9.2 Default Switch Settings - Visual

Table 16. Switch S7C1

Closed	Open	Open	Open	Open	Open	Open	Open	Open	Open
S7C1 1	S7C1 2	S7C1 3	S7C1 4	S7C1 5	S7C1 6	S7C1 7	S7C1 8	S7C1 9	S7C1 10

Figure 11. Default Switch Settings



3.9.3 Jumper Summary

Table 17. Jumper Summary

Jumper	Description	Factory Default
J1B1	U1_EN_N bypass	Open
J1B2	JTAG 2x4 Header	1-2, 7-8
J1D2	UART Enable	Open
J9C1	Buzzer Volume	Open

3.9.4 Connector Summary

Table 18. Connector Summary

Connector	Description
J1A1	Secondary PCI-X Expansion Slot Power. Please see Section 2.2.2, “Power and Backplane Requirements” for more details.
J1B1, J1B2, J1D2, J9C1	Jumper Settings. Please see Section 3.9.3, “Jumper Summary” on page 34 for more details.
J1D1	RJ45 Network Connector for GbE NIC.
J1E1	UART Headers.
J2C1	Interrupt Header
J1E2	RJ11 Dual Serial Port Connector.
J1E3	Edge connector for primary PCI-X Bus.
J2A1	Secondary PCI-X Expansion bus Slot.
J2D1	Power header for fan.
J4M1, J4N1, J5M1, J5N1, J5N2	Agilent mictor connectors - no pop
J6A1, J6A2	I^{2}C 4 pin connectors.
J7C1	CPLD Header.
J8B1	Not populated.
J8D1	Power header for battery.
J9A1	PC104 Mod connector, can NOT be used with standard PC104 applications. This is not a PC104 valid connector.

3.9.5 General Purpose Input/Output Header

The following table in Section 19, “GPIO Header Definition” on page 34 shows the GPIO signal assignments. The GPIO signals are muxed with the serial port signals. The serial port must be disabled to use the GPIO signals.

Table 19. GPIO Header Definition

Pin	Signal	Pin	Signal
1	GPIO0	5	GPIO4
2	GPIO1	6	GPIO5
3	GPIO2	7	GPIO6
4	GPIO3	8	GPIO7

3.9.6 Detail Descriptions of Switches/Jumpers

3.9.6.1 Switch S7C1

This switch allows the user to enable or disable various features. All associated nets can be cross-referenced from the Intel® IQ80331 Evaluation Platform Board Schematics.

3.9.6.1.1 S7C1-1: SECONDARY BUS SPEED - associated net PD_PBI_AD3

This switch allows the user to force the secondary PCI-X bus to run at 133 MHz or 100 MHz. However, since the 82545EM resides on the secondary PCI-X bus, it is unlikely that an additional device plugged into the secondary PCI-X slot could run at 133 MHz.

Table 20. S7C1-1: SECONDARY BUS SPEED: Settings and Operation Mode

S7C1-1	Operation Mode
Open	Enables 133 MHz on secondary bus
Closed	**Enables 100 MHz on secondary bus (Default mode).**

3.9.6.1.2 S7C1-2: RESET - associated net PD_AD5

RESET MODE is latched at the de-asserting edge of P_RST# and it determines when the 80331 is held in reset until the Intel XScale® core Reset bit is cleared in the PCI Configuration and Status Register. When the core is held in reset, the LEDs (DS9A1 and DS9A2)are not illuminated.

Table 21. Switch S7C1-2: RESET: Settings and Operation Mode

S7C1-2	Operation Mode
Open	**Don't hold in reset - enables core to execute (Default mode).**
Closed	Hold in reset.

3.9.6.1.3 S7C1-3: RETRY - associated net PD_AD6

RETRY is latched at the de-asserting edge of P_RST# and it determines when the Primary PCI interface disable PCI configuration cycles by signaling a Retry until the Configuration Cycle Retry bit is cleared in the PCI Configuration and Status Register.

Table 22. Switch S7C1-3: RETRY: Settings and Operation Mode

S7C1-3	Operation Mode
Open	**Retry enabled - use when booting in a host (Default mode).**
Closed	Configuration retry disabled - use when booting in a backplane.

3.9.6.1.4 S7C1-4: BRIDGE ENABLE - associated net PD_AD0

This switch allows the user to enable or disable the PCI-X to PCI-X bridge inside the I/O processor.

Table 23. S7C1-4: BRIDGE ENABLE: Settings and Operation Mode

S7C1-4	Operation Mode
Open	**Bridge enabled(Default mode).**
Closed	Bridge disabled.

3.9.6.1.5 Switch S7C1- 5: PRIVATE DEVICES - associated net PD_A0

This switch allows the user to enable a private device on the secondary PCI-X bus.

Note: Please note, Redboot expects private devices to be enabled. If this switch is not properly set correctly with Redboot running, then the Redboot kernel crashes when the software addresses the GigE device. When the kernel crashes, garbage is seen on the serial port. Try properly setting this switch, and resetting the system.

Table 24. Switch S7C1- 5: PRIVATE DEVICES: Settings and Operation Mode

S7C1-5	Operation Mode
Open	**Enables private devices (Default Mode).**
Closed	Disables private devices. All devices are public.

3.9.6.1.6 Switch S7C1 - 6: PRIVATE MEMORY - associated net PD_A1

This allows 80331 to hide devices in PCI-X by enabling private memory spaces.

Note: Again, Redboot expects this switch to be open.

Table 25. Switch S7C1 - 6: PRIVATE MEMORY: Settings and Operation Mode

S7C1-6	Operation Mode
Open	**Private memory enable on secondary PCI-X bus** (Default mode)**.**
Closed	Normal addressing mode.

3.9.6.1.7 Switch S7C1 - 7: PRIMARY PCI BUS WIDTH - associated net PD_PBI_A2

This allows 80331 to force the secondary PCI-X bus to operate in 32-bit or 64-bit mode.

Table 26. Switch S7C1 - 7: PRIMARY PCI BUS WIDTH: Settings and Operation Mode

S7C1-7	Operation Mode
Open	**Enables 64-bit Secondary PCI-X bus width (Default mode).**
Closed	Enables 32-bit Secondary PCI-X bus width.

3.9.6.1.8 Switch S7C1- 8/9

These switches (S7C1-8 and S7C1-9) are spares and should be left open.

3.9.6.1.9 Switch S7C1- 10: FLASH WIDTH - associated net P3_3V

This switch allows the user to choose 8-bit or 16-bit Flash.

Table 27. Switch S7C1 - 10: FLASH WIDTH: Settings and Operation Mode

S7C1-10	Operation Mode
Open	**Enables 8-bit Flash (Default mode).**
Closed	Enables 16-bit Flash.

3.9.6.2 Jumper J1B1- UART U1 RXD ENABLE

Install this jumper to disable UART U1 RXD. The associated net is U1_EN_N.

Table 28. Jumper J1B1: Descriptions

Jumper	Description	Factory Default
J1B1	UART U1 RXD disable	Open

Table 29. Jumper J1B1: Settings and Operation Mode

J1B2	Operation Mode
NC	UART transceiver enabled (Factory Default).
Pins 1,2	UART transceiver disabled.

3.9.6.3 Jumper J1B2 - JTAG COMMUNICATION OPTIONS

Table 30. Jumper J1B2: Descriptions

Jumper	Description	Factory Default
J1B2	JTAG 2 x 4 Header	1-2, 7-8

Table 31. Jumper J1B2: Settings and Operation Mode

Pins	Operation Mode
1-2	JTAG communication to IOP only (Factory Default).
3-4	JTAG communication to IOP and GigE
5-6	Spare
7-8	Connects TRST_N through a 1 K ohm resistor, pulled down to ground (Factory Default).

3.9.6.4 Jumper J1D2 - UART TRANSCEIVER ENABLE

Install J1D2 to enable UART transceiver.

Table 32. Jumper J1D2: Descriptions

Jumper	Description	Factory Default
J1D2	UART transceiver disable	Open

Table 33. Jumper J1D2: Settings and Operation Mode

J1D2	Operation Mode
NC	UART transceiver disabled (Factory Default).
Pins 1,2	UART transceiver enabled.

3.9.6.5 Jumper J9C1 - BUZZER VOLUME

Table 34. Jumper J9C1: Descriptions

Jumper	Description	Factory Default
J9C1	Buzzer Volume	Open

Table 35. Jumper J9C1: Settings and Operation Mode

J9C1	Operation Mode
Pins 1,2	Buzzer Volume LOUD.
Pins 2,3	Buzzer Volume QUIET.
NC	Buzzer Volume OFF(Factory Default).

Software Reference 4

4.1 DRAM

For DDR SDRAM and DDR-II sizes and configurations that the Intel® 80331 I/O processor supports, please see section 8.3.3 of the *Intel® 80331 I/O Processor Developer's Manual*. This section also contains multiple examples of Address Register Programming.

See the *Intel® 80331 I/O Processor Design Guide*, section 7.1, Table 16 for supported DDR and SDRAM configurations.

For all registers relating to DRAM and other MCU related registers, see Section 8.7, Table 234 of the *Intel® 80331 I/O Processor Developer's Manual*.

4.2 Components on the Peripheral Bus

The 80331 has a peripheral bus which contains the following peripheral devices:

- Flash ROM
- CPLD
- Audio Buzzer
- Rotary Switch
- Hex Display

Peripheral memory-Mapped Register Locations for the Peripheral Bus Interface Unit can be found in the *Intel® 80331 I/O Processor Developer's Manual*, Section 7.5, Table 298.

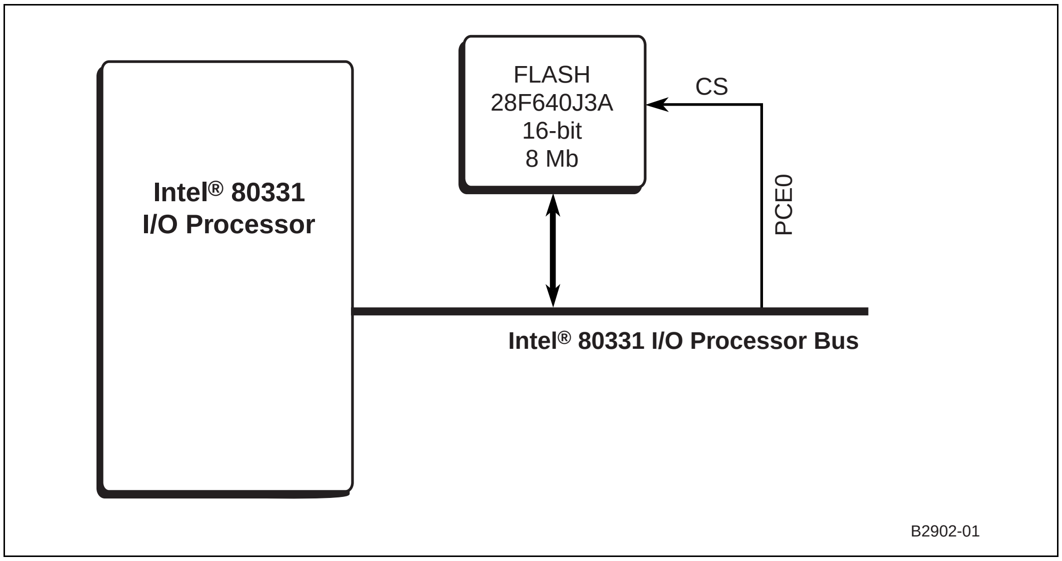
All registers associated with the PBI can be found in the *Intel® 80331 I/O Processor Developer's Manual*, Section 9.4, Table 272.

4.2.1 Flash ROM

The Flash ROM is an 8 MB Intel StrataFlash® (part# 38F640) that sits on the Peripheral Bus and is accessed using PCE0.

Note: The current version of the Intel® Flash Recovery Utility (FRU) can be set for 8-bit or 16-bit access.

Figure 12. Flash Connection to Peripheral Bus



Under normal operation, the very first instruction access by the Intel XScale® core begins at location 0x0 on the 80331 Internal Bus. By default, address 0x0 is pointing to PCE0 where flash is located.

See the *Intel® Flash Recovery Utility (FRU) Reference Manual* for details on how to upload / download Flash images:

http://developer.intel.com/design/iio/devtools/iq80310/273551.htm

4.2.2 Peripheral Bus Memory Map

The following table Table 36 is the physical memory map of the devices on the IQ80331 Peripheral Bus:

Table 36. Peripheral Bus Memory Map

Address Range (in Hex)	Size	Data Bus Width	Description
C000 0000 - C07F FFFF	8 MB	8-bit or 16-bit	Flash memory (re-mapped)
CE80 0000 -CE80 FFFF	64 KB	8-bit	Product Code
CE81 0000 -CE81 FFFF	64 KB	8-bit	Board Stepping
CE24 0000 -CE82 FFFF	64 KB	8-bit	CPLD Firmware Revision
CE83 0000 -CE83 FFFF	64 KB	8-bit	Discrete LEDs
CE84 0000 -CE84 FFFF	64 KB	8-bit	Hex Display Left
CE85 0000 -CE85 FFFF	64 KB	8-bit	Hex Display Right
CE86 0000 -CE86 FFFF	64 KB	8-bit	Buzzer Control
CE87 0000 -CE87 FFFF	64 KB	8-bit	32KB NV RAM
CE8D 0000 -CE8D FFFF	64 KB	8-bit	Rotary Switch
CE8E 0000 -CE8E FFFF	64 KB	8-bit	ESN I/O
CE8F 0000 -CE8F FFFF	64 KB	8-bit	Battery Status

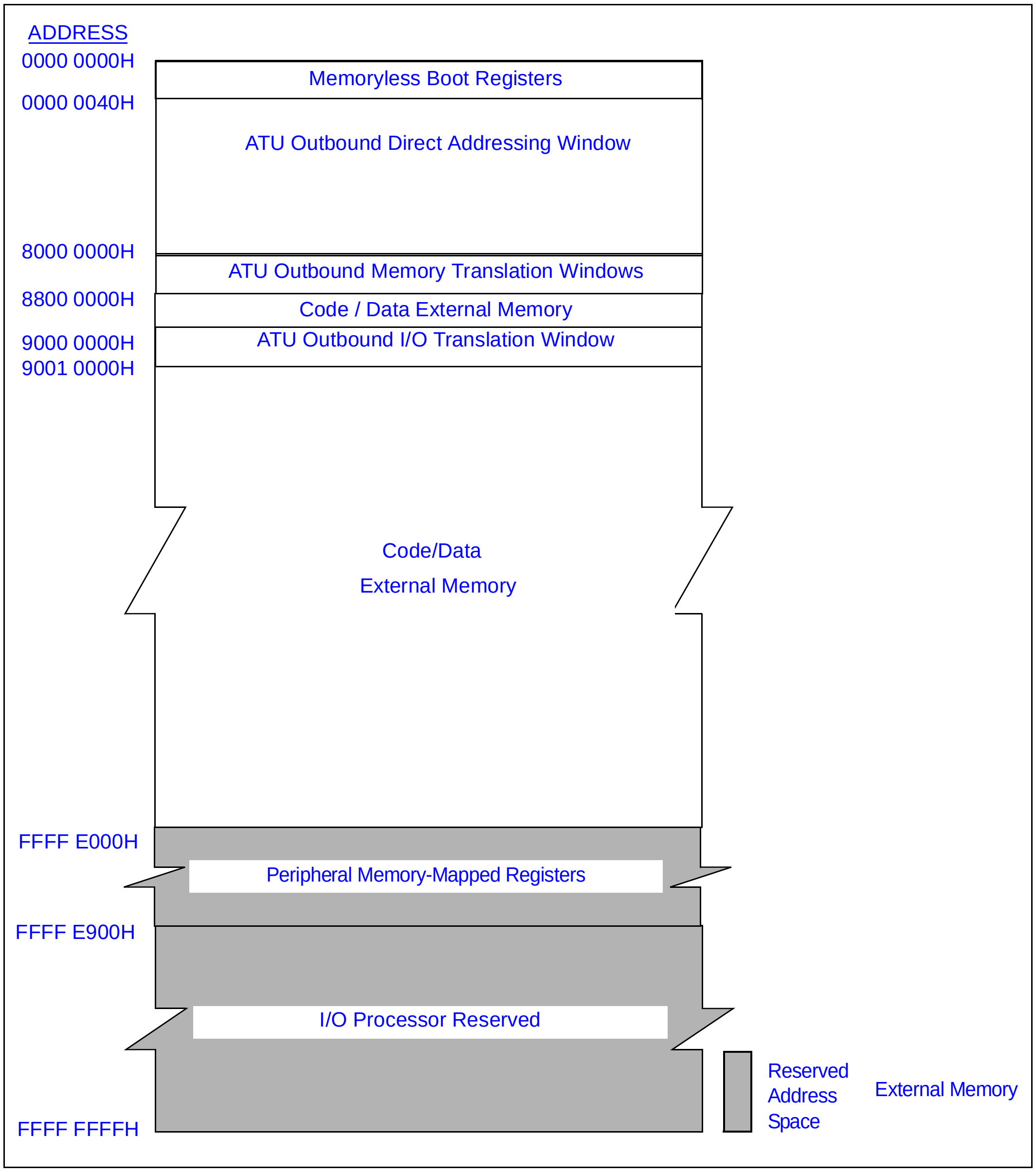
4.3 Board Support Package (BSP) Examples

Examples provided in this section are based on RedHat RedBoot software running on the IQ80331.

4.3.1 Intel® 80331 I/O Processor Memory Map

Figure 13 depicts the memory space for the IQ80331 (before RedBoot* boots):

Figure 13. Intel® 80331 I/O Processor Memory Map


ADDRESS
0000 0000H
Memoryless Boot Registers
0000 0040H
ATU Outbound Direct Addressing Window
8000 0000H
ATU Outbound Memory Translation Windows
8800 0000H
Code / Data External Memory
9000 0000H
ATU Outbound I/O Translation Window
9001 0000H
Code/Data
External Memory
FFFF E000H
Peripheral Memory-Mapped Registers
FFFF E900H
I/O Processor Reserved
FFFF FFFFH
Reserved Address Space
External Memory

4.3.2 Intel® IQ80331 Memory Map for RedBoot

Virtual Address	Physical Address	Size (MB)	Description
0x0000 0000	0x0000 0000		SDRAM
0x8000 0000	0x8000 0000	128	ATU Outbound Memory Translation Windows
0x8800 0000	*	128	Unused
0x9000 0000	0x9000 0000	1	ATU OUtbound I/O Translation Window.
0xA000 0000	*	255	Unused
0xC000 0000	0xC000 0000*	8	Flash (PCE0#)
0xC080 0000	*	224	Unused
0xCE80 0000	0xCE80 0000	1	PCE1# - Uncached
0xCE90 0000	*	23	Unused
0xD000 0000	*	256	Unused
0xE00 0000	0xE00 0000	1	Cache flush
0xE010 0000	*	255	Unused
0xF000 0000	*	255	Unused
0xFFF0 0000	0xFFF0 0000	1	Intel® 80331 I/O Processor Memory Mapped Registers. Please see Chapter 17 of the *Intel® 80331 I/O Processor Developer's Manual* for more details.

4.3.3 Intel® IQ80331 Files for RedBoot

Attached in the kit, find a copy of the Red Hat* eCos for Intel® 80331 I/O processor CD. Once the CD is installed, look for the:

- **The RedBoot initialization code source files** from the following location:
  From the installed directory:
  ..\Red Hat\eCos\packages\hal\arm\xscale\iq80331\current\include
- **The RedBoot binary image files** (downloadable onto Flash) from the following location:
  From the installed directory:
  ..\Red Hat\eCos\loaders\iq80331

To access Red Hat* GNUPro tools including RedBoot binaries and source code, also go to the following location on the Intel site:

- http://developer.intel.com/design/intelxscale/dev_tools/021022/index.htm

4.3.4 Intel® IQ80331 DDR Memory Initialization Sequence for RedBoot

In order to set the correct ECC bits, a DDR memory system (DIMM or discrete components) must be written to with a known value. This process requires 64-bit writes to the entire DDR memory intended for use. The following explains the sequence for memory initialization by RedBoot on an IQ80331 board with an ECC DIMM. It also includes an example for the scrub (ECC initialization) code.

Initialization Sequence:

1. Disable interrupts. (Technically they are disabled at reset, but for soft reset this is included.
2. Init PBIU (Peripheral Bus Interface Unit) chip selects.
3. Enable I cache.
4. Move Flash to 0xF0000000.
5. Set TTB and Enable MMU.
6. Read DIM for memory parameters.
7. Set Memory Parameters.
8. Delay.
9. Turn DDRAM on.
10. Delay.
11. Enable Data Cache.
12. Enable BTB.
13. Flush all.
14. Clear ECC error logs.
15. Battery Test.
16. Enable ECC.
17. Scrub loop: Write zeros to all memory locations

```
    mov     r8, r4       // save DRAM size
    mov     r0, #-1
    mov     r1, #-1
    mov     r2, #-1
    mov     r3, #-1
    mov     r4, #-1
    mov     r5, #-1
    mov     r6, #-1
    mov     r7, #-1

    ldr     r11, = SDRAM_BASE

    // scrub Loop
0:
    stmia   r11!, {r0-r7}
    subs    r12, r12, #32
    bne     0
```

This Page Left Intentionally Blank

IQ80321 and IQ80331 Comparisons A

This appendix provides a brief description for differences between IQ80331 and IQ80321. Please also refer to application note: *Intel® 80321 Software Conversion to the Intel® 80331 I/O Processor Application Note* 273562.

Table 37. Intel® IQ80310 and Intel® IQ80331 I/O Processor Evaluation Platform Board Comparisons

Features	Intel® IQ80331 I/O Processor Evaluation Platform Board	Intel® IQ80321 Evaluation Platform Board
I/O Processor	Intel® 80331 I/O processor	Intel® 80321 I/O processor
Core/Microprocessor Technology	Intel XScale® microarchitecture	Intel XScale® microarchitecture
Memory Technology	DDR-II 400 MHz SDRAM	PC1600 DDR SDRAM (100 MHz Clock)
Form Factor	PC board that attaches to a PC/Server/Backplane – One PCI-X Expansion Slot	Extended PC board that attaches to a PC/Server/Backplane – One PCI-X Expansion Slot
PC/Server/Backplane Connection	3.3 V PCI-X 133-MHz/64-Bits or 3.3V PCI 66 MHz/64 Bits	PCI-X 133-MHz/64-Bits or PCI 66 MHz/64 Bits
Expansion Card Slot	One PCI-X 133-MHz/64-bit	One PCI-X 133-MHz/64-bit
PCI/PCI-X Bridge	PCI-X to PCI-X Bridge integrated with the Intel® 80331 I/O processorl	IBM PCI-X Bridge Reference: IBM 133 PCI-X Bridge http://www.chips.ibm.com/
Interrupt Routing	External interrupts are routed through the XINT pins on the 80331. PLease see Table 9 for more details.	External interrupts are routed through the XINT pins on the 80321. They include INTA, INTB form PCI-X expansion slot, INTA from 82544 GBE, and UART interrupt – Steering and Status registers are in 80321 – see *Intel® 80321 I/O Processor Developer's Manual*
Timers	Internal to 80331 – Refer to *Intel® 80331 I/O Processor Developer's Manual*	Internal to the 80321 - please refer to the *Intel® 80321 I/O Processor Developer's Manual*
Local/Peripheral Bus	66 MHz multiplexed bus with two chip-enables, Synch/Asynchronous (IQ80331 operates in 66 MHz Asynchronous mode) – Refer to PBI section in *Intel® 80331 I/O Processor Developer's Manual*	2-bit/33-100MHz multiplexed bus with six chip-enables, Synch/Asynchronous (IQ80321 operates in 33 MHz Asynchronous mode) – Refer to PBI section in the I*Intel® 80321 I/O Processor Developer's Manual*.
Flash Memory	8-bit or 16-bit, 8 MB accessed through Peripheral Bus with chip-enable 0 (PCE0)	16-bit, 8 MB accessed through Peripheral Bus with chip-enable 0 (PCE0))
Serial Debug Port	Two UARTs integrated within the 80331.	One UART on the Peripheral bus – 16C550 device
Network Debug Port	Intel® 82545EM GbE on the PCI-X bus	Intel® 82544 GbE on the PCI-X bus
Rotary Switch	Same	Same
LED HEX Display	Same	Same
JTAG	20-PIN ARM Compliant	20-PIN ARM Compliant
Logic Analyzer Connection	Various Mictors.	Various Mictors

This Page Left Intentionally Blank

Getting Started and Debugger B

B.1 Introduction

This appendix pertains to Code|Lab version 2.3 (and later versions) which uses Microsoft* Visual Studio* .NET.

For more detailed information on JTAG and the IQ80331, please see the *Intel® 80331 I/O Processor JTAG Support White Paper*.

B.1.1 Purpose

The purpose of this appendix is to help the user setup and become familiar with the Intel® IQ80331 I/O processor evaluation platform board (IQ80331) and, other related hardware and software. This appendix provides steps through an example program using:

- Code|Lab EDE
- Code|Lab EDE debugger
- Macraigor* MPDemon* JTAG

Programming information also includes:

- software setup
- compiling
- linking
- debugging example code

The user tours the major features of the debugger and explores some of the basics of debugging. By the end of this exercise, the user has been given a general understanding of the ATI* development tools and can begin working on new applications.

B.1.2 Necessary Hardware and Software

The setup example described in this appendix uses the ATI Code|Lab plug-in for Microsoft Visual Studio*, the GNU* Pro compiler, the Macraigor MPDemon JTAG connector, and the IQ80331.

B.1.2.1 Additional Information

Table 38 and Section B.1.2.1 provide additional information to consult when performing a setup.

Table 38. Related Documents

Document Title	Document #
Intel® 80331 I/O Processor Developer's Manual	273517
Intel® 80200 Processor based on Intel XScale® Microarchitecture Developer's Manual	273411
Hot-Debug for Intel XScale® Core Debug White Paper	273539
ARM Assemblers Guide (http://www.arm.com/support/574FKU/$File/ADS_AssemblerGuide_B.pdf)	
ADS Debug Target Guide (http://www.arm.com/support/574FWT/$File/ADS_DebugTargetGuide_D.pdf)	
Code|Lab Debug for ARM[a]	

[a] This document installs to C:\Ati\docs\codelab debug.pdf.

Many of these documents load as part of ATI Code|Lab install (Start/Programs/ Accelerated Technology/Documentation). This menu contains both the ARM* ADS and Code|Lab documents.

B.1.3 Related Web Sites

- Macraigor: http://www.ocdemon.net/
- http://developer.intel.com/design/intelxscale/dev_tools/021022/index.htm
- http://developer.intel.com/design/iio/docs/iop331.htm
- http://developer.intel.com/design/iio/papers/273961.htm

B.2 Setup

B.2.1 Hardware Setup

Use Figure 14 and the rest of this manual to set up the hardware.

1. Connect the MPDemon to the host via the parallel port and to the evaluation board via the 20-pin JTAG connector.

Warning: Carefully examine the OC Demon ribbon cable. The end marked "OCD Connection End" must be connection on the MPDemon base station. The cable quickly smokes, when not connected correctly.

Note: The parallel port must be configured to ECP mode for the Macraigor MPDemon to work properly. Also, the Ethernet and serial connections are optional. Refer to the MPDemon manual to correctly configure the selected connection type. The parallel port setting can be changed in the BIOS setup program or in the Control Panel.

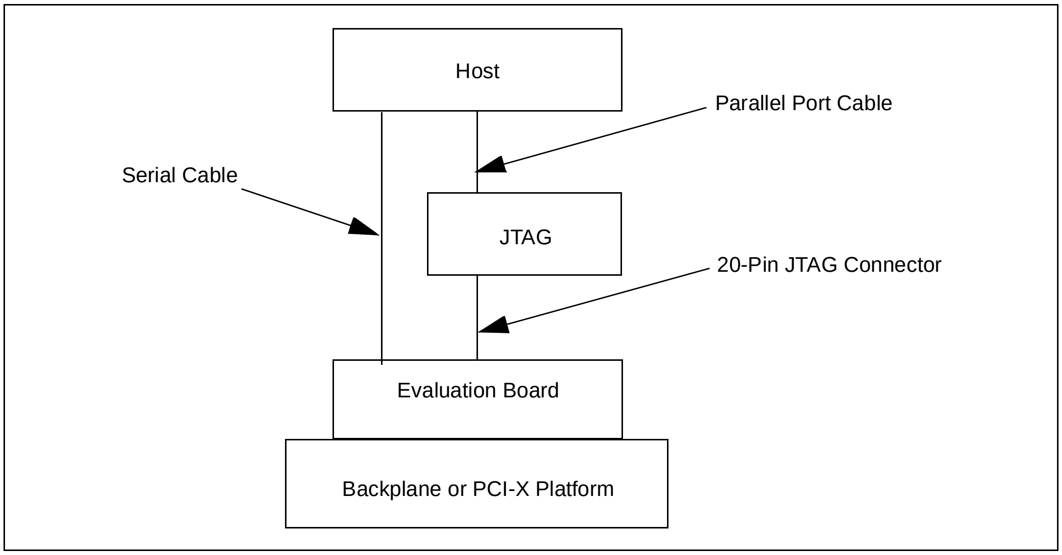
Note: The parallel port setting can be changed in the BIOS setup program or in Control Panel. More information on the MPDemon can be found at the Macraigor web site. Test software for the MPDemon is free and available for download at: http://www.ocdemon.net/Merchant2/merchant.mv?Screen=CTGY&Store_Code=MTS&Category_Code=pinouts.

2. Connect a serial cable from the evaluation board to the host.

Note: The serial cable connects to the evaluation board with an RJ11 connector and connects to the host computer serial port via an RJ11 to DB9F adaptor. The serial port configuration is covered in Section B.3.2, "Configuration" on page 55.

3. Plug the IQ80331into a bus master PCI or PCI-X slot on the backplane or platform.

Figure 14. Intel® 80331 I/O Processor Hardware Setup Flow Chart



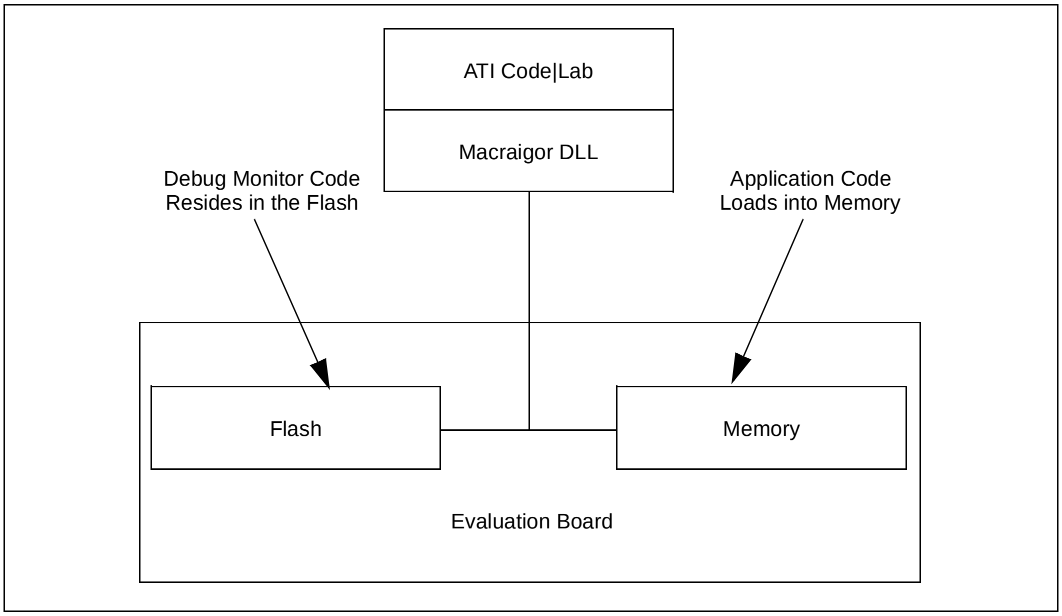
B.2.2 Software Setup

ATI Code|Lab is a plug-in to Microsoft Visual Studio .NET, therefore Microsoft Visual Studio .NET must already be loaded on the system. To load ATI Code|Lab, run *setup.exe* under the Program directory.

Note: Do not install this software over an old version of ATI Code|Lab. When necessary, uninstall Code|Lab using the Add/Remove programs under the Control Panel before reinstalling.

Note: To view the soft copies of this document, Adobe Acrobat Reader is required. The latest version can be downloaded at (http://www.adobe.com).

Figure 15. Software Flow Diagram


ATI Code|Lab
Macraigor DLL
Debug Monitor Code Resides in the Flash
Application Code Loads into Memory
Flash
Memory
Evaluation Board

B.3 New Project Setup

The following sections describe how to set up and configure a new project.

B.3.1 Creating a New Project

1. Launch Code|Lab EDE for .NET.
2. On the Start Page, select “New Project”.
   a. The “New Projects” window appears.
   b. Select “Code|Lab Projects” under Project Types and name the project “Test80331” in the name field.

Note: The directory “Test80331” is created under the path specified in the Location box.

   c. Click OK.

3. In the Code|Lab EDE Project Wizard Window:
   a. Select “Red Hat* GNU Tools for XScale” under “Build Toolset”.
   b. Select IQ80331 under “Project Template”.
   c. Select “Application” under “Project Type”.
   d. Click “Finish”.
4. Close the “Start Page” by clicking on the X in the top right corner of the Start Page window.
5. The new project is now in the “Solution Explorer” window. When this window is not open, open it by “View,”, then “Solution Explorer”.
6. Right click on “Test80331” and select “Save Test80331”.
7. From http://developer.intel.com/design/iio/swsup/Timer80331.htm, download the following zip file (…/Timer80331) from the Software Support section, containing the example code files to the newly created project folder:

   Timer80331.zip

   timer.c

   blink.h

   led.c

   led.h

   These files can be placed in any directory on the hard drive.
8. Add the newly downloaded files to the project:
   a. In the “Solution Explorer” window, right click on “Test80331” and select “Add, Add Existing Item”.
   b. In the “Add Existing Item” window, use the drop-down menu under “Look In” to find the four files listed in Step 7 on the hard drive. Select all four files and click “open”. The “Solution Explorer” window now shows these files under “Test80331”.

B.3.2 Configuration

Examine the main menu of Code|Lab EDE for .NET.

- File
- Edit
- Project
- View
- code|lab EDE
- Build, Debug
- Tools
- Window
- Help

Since Code|Lab is a plug-in to Visual Studio, some of these menu items are Visual Studio and some are specific to Code|Lab. Click on any of these menu items and the drop-down menu displays the subordinate menu items. Many of these items have defined tool bar symbols, function keys, and keyboard patterns as alternatives.

Note: Projects can be built under the "code|lab EDE" menu or under the "build" menu. Always use the "code|lab EDE" menu to perform Code|Lab project builds. Builds under the "build" menu invoke the Visual Studio C compiler.

To configure code|lab EDE, perform the following steps:

1. On the main menu, select "code|lab EDE, Configuration".
2. When "code|lab EDE Configuration" window appears, click on each word in the left box.

Note: Notice that the rest of the window changes when clicking on different parts of the menu tree. This is a typical feature of Code|Lab EDE for .NET.

3. Click on Toolsets.
4. Click on the drop-down arrow and select "Red Hat GNU Tools for XScale". The build tool paths now appear in the box and must be modified as stated below in bold. Note that the assembler and the linker are invoked by GCC.
   a. "Compiler path: $(ToolDir)\BIN\XSCALE-ELF-GCC.EXE".
   b. "Assembler path: $(ToolDir)\BIN\XSCALE-ELF-GCC.EXE".
   c. "Linker path: $(ToolDir)\BIN\XSCALE-ELF-GCC.EXE".
   d. "Librarian path: $(ToolDir)\BIN\XSCALE-ELF-AR.EXE".
5. Click "Apply" then click "OK".
6. On the main menu, click "code|lab EDE, Project Settings".
7. When the "code|lab Project Settings" window appears, click on "C/C++/Assembler" in the left box. Use the drop-down arrow to select "C compiler" for "Build Tool".
8. Edit the command line box at the bottom so that it contains the following:

   ```
   -v -Wall -specs=redboot.specs -gdwarf-2 -O0 -c -mcpu=xscale $(InputRelPath) -o
   $(OutDir)\$(InputName)$(OutputExt)
   ```
9. Use the drop-down arrow to select "Assembler" for "Build Tool". Edit the command line box at the bottom so that it contains the following:

   ```
   -v -specs=redboot.specs -o $(OutDir)\$(InputName)$(OutputExt) $(InputRelPath)
   ```
10. In left box, click on "Linker". Edit command line box at bottom so it contains the following:

    ```
    -v -specs=redboot.specs -o $(OutDir)\$(ProjectName).elf $(ObjectFiles) $(Libraries)
    ```
11. Click "Apply" and then click "OK".
12. Under "Project Settings", select Code|Lab Debugger ARM". Set all four debug options to "false".
13. In "Solution Explorer" window, right click "Test80331" and select "Save Test80331".

B.4 Flashing with JTAG

B.4.1 Overview

Code|Lab and MPDemon are capable of reading from, writing to, and erasing the contents of the Flash on the evaluation board. The board comes with RedBoot loaded in the Flash. RedBoot is the Red Hat debug monitor which initializes the board and has some debug and diagnostic functions. It is capable of serial communication with the console of a debug program or with Microsoft* HyperTerminal*, and it prepares the board for accepting an application program.

Code|Lab invokes a Flash programmer written by Macraigor. More information on the Flash programmer is located at:

http://www.ocdemon.net/Merchant2/merchant.mv?Screen=CTGY&Store_Code=MTS&Category_C ode=Software

This Flash programmer only supports certain file formats: Intel Hex, Motorola srec and standard elf (executable and linking format). RedBoot.s19 and RedBoot.srec are both srec files. TBD.i32 is an ARM BootMonitor Intel Hex file. BootMonitor is an ARM version of a debug monitor, which is similar but not identical to RedBoot.

Macraigor offers conversion tools to convert existing file types to a supported file type. These conversion tools are located at:

C:\ATI\codelab\codelab Debug\Macraigor\Flash Programmer

The ReadMe.txt file describes the conversions tools. BinToS19.exe converts binary files to srec files and MakeIntelHex.exe converts a.out files to Intel Hex files. When using the BinToS19.exe conversion tool, use 0x0 for the starting address. For example, at the CMD prompt in the directory where BinToS19.exe is located, the command line looks like this:

```
C:\ATI\codelab\codelab Debug\Macraigor\Flash Programmer>bintos19
C:\temp\redboot_ROM.bin 0x0 c:\temp\redboot_ROM.s19
```

B.4.2 Using Flash Programmer

Note: The parallel port must be set to EPP mode or the Macraigor MPDemon does not work properly. Refer to the MPDemon manual to check the settings for the connection type selected. This example assumes a Parallel connection with the MPDemon. ECP mode should be selected when using the parallel port.

The 80331 processor has two TAP controllers in the TAP chain. Refer to section 19.3.3. in the *Intel® 80331 I/O Processor Developer's Manual*. The following connection steps place the Test Logic Unit in bypass mode and then communicates with the core.

Download the RedBoot executable files from the following location: http://developer.intel.com/design/intelxscale/dev_tools/021022/index.htm RedBoot Debug Monitor for the IQ80331.

To use Flash Programmer, perform the following steps:

1. Double click on the "Code|Lab Debug" icon on the desktop.
   The Connection Window appears.
2. Select Macraigor JTAG Connect
3. Click Setup.
4. Select "MPDemon Parallel"
5. Do not check "Script Options" or "Enable Hot Debug"
6. Click "Configure"
   The Console Options windows now appears.
7. Select "Parallel Connection" and "LPT1".
8. OCD Speeds should be '1' in both cases. (A larger number equals a slower speed which can help when there are connection problems.)
9. Click "OK"
10. Select "XScale (7-bit JTAG)"
11. Click "Additional Options"
12. Check "Enable OPtion" under "Scan Chain Options"
13. Click "Configure"
14. For "JTAG Instr Register Length", enter '7'
15. For "Bypass Value", enter 0x7f (or 0xff - either works)
16. For "Bypass Length", enter '1'.
17. Click "Add before"
18. Click "OK"
19. Check "Enable Option" under "Console Options"
    The Console Options window appears

20. Select COM1 or COM2, as appropriate

    Baud Rate: 115200

    Data Bits: 8

    Parity: None

    Stop Bits: 1

    Then Press OK, OK, OK (this returns to the Connect window).

21. Now press Connect.

    Assembly code now visible.

22. Select “Memory/Flash…”

    The OCDemon Flash Memory Programmer window appears.

23. The Flash programmer needs a file which is architecture specific, in this case. In the Flash programmer window, select “File/Open”, then choose the file “Xscale TBD.ocd” at:

    “C:\MGC\Embedded\codelab\codelab Debug\Macraigor\”.

24. Click the Program button.

25. Click Browse and “Files of type:” All Files, then choose the “redboot_ROM.srec” file

    (downloaded and uncompressed from developer.com).

26. Check box “Erase Target Flash Sector(s) Before Programming”.

27. Click Program.

    The Flash now programs and verifies; click Close when 100% complete.

28. Cycle power to the board to see that the LEDs on the board sequence “SL”, then “A1”. This is the normal LED sequence of RedBoot. The board may need to be reset more than once. Explore the other features of the Flash programming window. The contents of the Flash can be erased, copied to a file on the host, and verified against a file on the host.

B.5 Debugging Out of Flash

JTAG debuggers can be used on two levels; with or without the source code. When the Flash is programmed, the debugger can monitor the executable code, halt it, step through it, and monitor the memory and registers. The executable code is disassembled so that the assembly code can be examined.

Debugging with source code allows the user to examine the C code that is being executed. This requires that the source code is available and linked by the debugger to the executable code that is running on the evaluation board.

B.6 Building an Executable File From Example Code

To build an executable file, perform the following steps:

1. Launch Code|Lab EDE and open “Test80331”.
2. Select “code|lab EDE, Rebuild Project”.

Note: A project can have more than one solution, but in this example, there is only one solution for the project which means there is no difference between “Build Project” and “Build Solution”.

Note: Rebuild cleans and builds files:

— Clean deletes the old .o files in the project

— Build compiles, links, and produces the executable files.

3. When errors occur, carefully go back through Section B.3.2, “Configuration”.

B.7 Running the Code|Lab Debugger

This section is provided to get the system up and running in the Code|Lab Debug environment, but it is not intended as a full-functional tutorial. Please refer to the *ATI Code|Lab Debug Reference Manual* for more detailed information.

B.7.1 Launching and Configuring Debugger

1. In EDE, click on the icon that looks like a red bug. The “Connect” window appears.
2. When not configured from Section B.4.2, “Using Flash Programmer”, go to Section B.4.2 and perform steps 2-5.
3. Press Connect to enter debug mode.

   The Code|Lab Debug environment appears with the Assembly window open.

Note: Mouseovers are available for most of the toolbar icons. (Leave the mouse over the debug icons across the top on the toolbar to see a brief explanation of each.)

4. Click on the go icon and let RedBoot boot (takes a minute) until the RedBoot prompt “RedBoot>” appears in the Console window (click the Console tab at the bottom of the Debug window to view the Console window).
5. From the console window, type “diag” then press “Enter”.

   The RedBoot Diagnostic function is invoked.

Note: Try out a few of the tests as desired.

6. Close the Debugger and EDE environment.
7. Reset the board (cycle power).

B.7.2 Manually Loading and Executing an Application Program

1. Launch the Code|Lab Debug Environment from the desktop icon.
2. Ensure “File…/Program Load Options/Load Executable and Symbols” is checked.
3. File, program load options, load executable and symbols.
   a. Select “file, open program, browse”.
   b. Find c:\<RedBoot downloaded Files>…\Test80331\O\Test80331.elf.
4. Press Go (80, 3, 32, and 21 cycle on the LEDs).
5. Reset the board (cycle power).

B.7.3 Displaying Source Code

1. Launch the Code|Lab EDE Debugger and open the “Test80331” ELF program.

Note: Use the File/Recent Programs menu for quick access.

2. Select the “Files” view in the lower tab of the WorkSpace window.
3. Bring up “timer.c” and “led.c” source code by double-clicking each filename.
4. Use the “Windows” Menu to arrange the windows, or maximize, minimize, and resize manually as desired.
5. Press the “Mixed” tab at the bottom of the “timer.c” window.

   Notice the assembly along with each C statement.
6. Press the “Source” tab to revert back to C code only.

B.7.4 Using Breakpoints

Note: Notice the small gray circles on the sidebar beside each line of source code. Single-click any of these gray circles and a red dot appears. The red dot represents a break point. Single-click the red dot to remove it, or click the “Remove all breakpoints” icon.

Place a breakpoint on the following lines of code in “timer.c”:

```
displayLED(leds[8],leds[0]); /* LED display '80' */
displayLED(leds[0],leds[3]); /* LED display '03' */
displayLED(leds[3],leds[3]); /* LED Display '33' */
displayLED(leds[3],leds[1]); /* LED display '31' */
displayLED(leds[16],leds[16]); /* LED display ' ' */
```

To insert breakpoints, perform these steps:

1. Click the “Go” icon.

   The yellow arrow stops at the first break point and the HEX display does not change.
2. Click the “Go” icon again.

   The last instruction has now been executed and an “80” is displayed.
3. Continue on in this fashion, watching the lines execute only as they are called, while the yellow arrow shows exactly what line is up next in execution.
4. Click the “Remove all breakpoints” icon.
5. Press “Go” again and notice that the program loop is infinite.
6. Press the “Halt” icon to stop execution.
7. Close the debugger and cycle power to the board.

B.7.5 Stepping Through the Code

The “led.c” file contains a function that is called from code in “blink.h”. The following procedure describes the steps through the code and utilizes a few of the most common step tools:

1. Launch the debugger, open Test80331, and open the “blink.h” and “led.c” files.
2. Set a breakpoint on the following line in “blink.c”: displayLED(leds[8],leds[0]); /* LED display '80'*/
3. Press Go.

   Program execution sit on the first breakpoint.
4. Press the “Step Over” icon and notice how execution jumps over the function call to the next line of execution.
5. Now try the “Step Into” icon and note that the pointer has now jumped into the function “displayLED”, which is located in the “led.c” file.
6. Press the “Step Over” icon again and watch the pointer advance within the function to the next executable line.
7. Now press the “Step Out of” icon and notice how execution leaves the called function and waits on the next executable line in “blink.h”.
8. The animate icon can also be used to provide a “Step Into” effect that occurs at a specified time interval (default of 1 second). This can be modified in the “Settings” section of the “View/Options” menu. Experiment with this as desired.
9. Use Halt to stop the animate mode before the next breakpoint.
10. Also note that Go can be pressed at any time to continue execution from the current line to the next breakpoint or program end.

B.7.6 Setting Code|Lab Debug Options

Besides the Animate debug time interval setting briefly mentioned in Step 8 of the previous exercise, many useful options can be accessed from the “View/Options” menu.

1. Experiment here by bringing up the Registers window (click and change the view options between binary and decimal; for example).

Hint: Settings tab, Interface, Radix

2. Also try bringing up the Memory window (click) and change the number of columns between 4 and 2 and notice the changes.

Hint: Settings tab, Memory Window, Number of Columns

Note: Press window icons a second time to remove them from view.

Again, there are many features of the debug environment not discussed here. Please see the Code|Lab manuals for a full description of debug features.

B.8 Exploring the Code|Lab Debug Windows

This section discusses some basics of the debug environment. Some of these windows and concepts have been dealt with during previous exercises in this manual. However, many new windows are also discussed and basic interaction exercises are given. Begin this section by launching the Code|Lab Debugger environment and connection via the JTAG port.

B.8.1 Toolbar Icons

Placing the mouse arrow on any icon displays the text function of that icon. When the icon launches a special window (i.e., Watch, Memory, Call Trace, etc.), the icon brings that window up on the first click and removes the window when pressed again.

B.8.2 Workspace Window

Click on the Workspace icon. Click on the Files and Browse tabs and examine the contents. Note that there are more files than the original source files. When double-clicking on the source files, blink.c and led.c, the source window appears for that file. When double-clicking on an included file, the debugger is not be able to find the file.

B.8.3 Source Code

The source code windows are opened by double-clicking on the source files in the Workspace window under the files tab. Viewing of mixed Assembly and C code or C code only, is controlled by the tabs at the bottom of these windows.

B.8.4 Four Debug and Console Windows

The Debug window displays debugger activity messages while the Debug tab is displayed. Script commands can be entered manually at the top of the window. Serial output is displayed while the Console tab is active. Commands for the running application can be entered at the top of this window.

B.8.5 Memory Window

Click on the Memory window icon. Change the address at the top of the window to 0xffffe100 and click on the green arrow to the right (or press Enter). This changes the viewable starting address of the Memory window. The ATU header begins at 0xffffe100 and contains a known number (8086). Also look at the base and limit registers for the memory and Flash devices, at 0xffffe508 and ffffe688 respectively, since they were initialized by RedBoot. Use the *Intel® 80331 I/O Processor Developer's Manual*, to see what the values mean.

Note: The tabs at the bottom allow the selection of two memory regions to observe.

B.8.6 Registers Window

Close all the active windows, then bring up the Registers window. Resize the this window and its columns to get a good view of all the registers. Notice that there is a Flags tab at the bottom of this window. This is useful for seeing the system flags defined by the CPSR. These are important especially during conditional code execution (see the *ARM Architecture Reference Manual* for more detail), but the flags are not changed during this exercise.

Click on the registers tab of the registers window and click the Animate icon. Notice how the register values change during program execution (red values are those that were modified during the last execution cycle). Click the Halt icon at any time, then try right clicking a register row and selecting "Go To Memory". Notice how the Memory window is brought up and the address contained in that register is shown.

Click on the registers tab. Red means that the register value changed since the last fetch as opposed to black which represents no change. Register values can be manually changed in this window.

B.8.7 Watch Window

It is often useful during the debugging process to keep an eye on a few select program variables.

1. Open the Tester1LED Program and bring up "led.c".
2. Click the "Watch" icon to bring up the Watch window.
3. Now add the "left" and "right" variables from "led.c" to the watch window.

Note: For each variable double click the variable name to highlight it, then drag it to the watch window.

4. Click the "Animate" icon and observe the changes.

Note: When focus goes back to the Assembly window during this process, try putting a breakpoint in led.c, then hit Go.

B.8.8 Variables Window

The Variables behaves very similarly to the Watch window, except that it shows all active variables. Bring up the Variables window, click Animate, and watch the changes.

B.9 Debugging Basics

B.9.1 Overview

Debuggers allow developers to interrogate application code by allowing program flow control, data observation, and data manipulation. The flow control functions include the ability to single-step through the code, step into functions, step over functions, and run to breakpoint (hardware or software). The data observation and manipulation functions include access to memory, registers, and variables. The combination of the flow control and data functions allows the developer to debug problems as they occur or to validate the application code. As the size of an application grows, the need to be able to narrow down the cause of a problem to a few lines of code is imperative.

Debuggers have a finite set of capabilities and limitations. Debuggers can give insight that is difficult to obtain without them, but they can fail when they are not used within the limits of their functionality. They are intrusive by definition. They are software programs that interact with software monitors or hardware (JTAG) to control a target program. Ultimately, the debugger works best when the developer understands what it can and can not do and uses it within those constraints.

B.9.2 Hardware and Software Breakpoints

The following section provides a brief overview of breakpoints. See the *Intel® 80331 I/O Processor Developer's Manual*, for more detailed information.

B.9.2.1 Software Breakpoints

Software breakpoints are setup and utilized via debugger utilities (such as Code|Lab). The abilities of software breakpoints were seen in Section B.7 of this Guide. Program execution can be halted at a particular line of code, stepped through, and executed again to the next breakpoint via debuggers.

During this process, register values, memory address contents, variable contents, and many other useful pieces of information can be monitored.

B.9.2.2 Hardware Breakpoints

Hardware breakpoints step and breakpoint in code in either ROM or RAM without altering the code, stacks, or other target information. Hardware breakpoints handle difficult issues, by providing the ability to set the processor conditions that cause the program to halt. Use hardware breakpoints to locate problems such as reentrance, obscure timing, etc.

The 80331 contains two instruction breakpoint address registers (IBCR0 and IBCR1), one data breakpoint address register (DBR0), one configurable data mask/address register (DBR1), and one data breakpoint control register (DBCON). The 80331 also supports a 256 entry, trace buffer, that records program execution information. The registers to control the trace buffer are located in CP14.

B.9.3 C.9.3 Exceptions/Trapping

A debug exception causes the processor to re-direct execution to a debug event handling routine. The Intel® 80200 processor debug architecture defines the following debug exceptions:

- instruction breakpoint
- data breakpoint
- software breakpoint
- external debug break
- exception vector trap
- trace-buffer full break

When a debug exception occurs, the processor actions depend on whether the debug unit is configured for Halt mode or Monitor mode.