ADM-XRC

PCI Mezzanine Card

User Guide

Version 1.3

Alpha Data Parallel Systems Ltd.
58 Timber Bush
Edinburgh EH6 6QH
Scotland
UK

Phone: +44 (0) 131 555 0303
Fax: +44 (0) 131 555 0728
Email: support@alphadata.co.uk

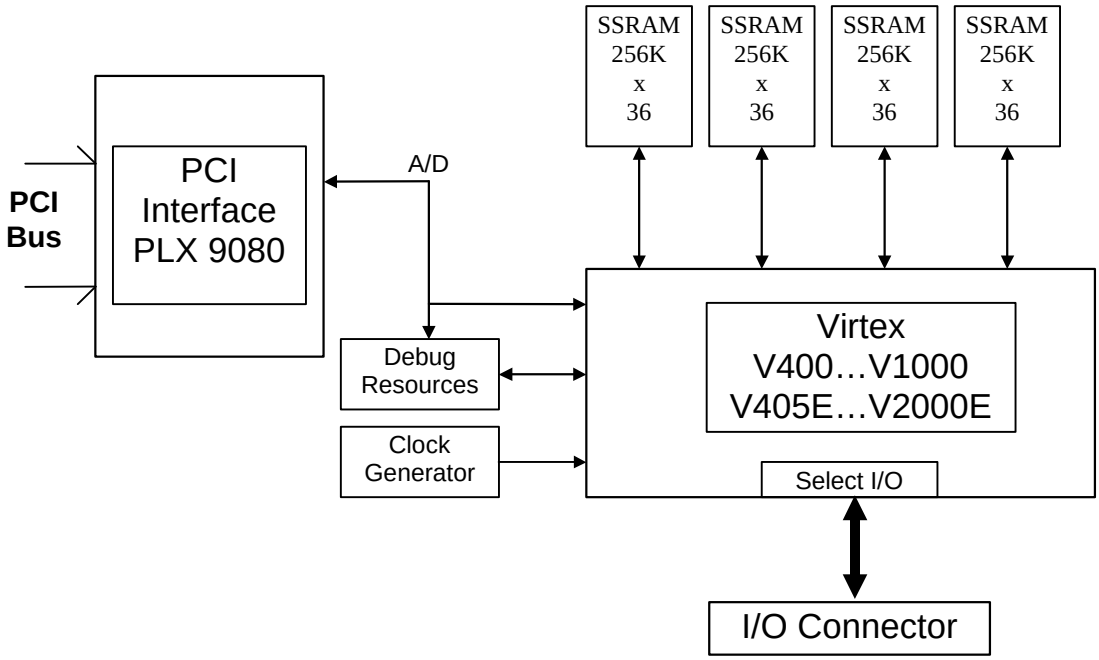
Copyright © 2001 Alpha Data Parallel Systems Ltd. All rights reserved.

This publication is protected by Copyright Law, with all rights reserved. No part of this publication may be reproduced, in any shape or form, without prior written consent from Alpha Data Parallel Systems Limited.

Table of Contents

1. Introduction

The ADM-XRC is a high performance PCI Mezzanine Card (PMC) format device designed for supporting development of applications using the Virtex series of FPGA's from Xilinx.



1.1. Specifications

The ADM-XRC supports high performance PCI operation without the need to integrate proprietary cores into the FPGA. A PLX PCI9080 provides a rich set of PCI resources including two high speed DMA controllers that can achieve rates of 120Mbytes/sec.

- Physically conformant to IEEE P1386 Common Mezzanine Card standard
- High performance PCI and DMA controllers
- Local bus speeds of up to 40MHz
- Four banks of 256kx36 SSRAM, Synchronous burst or ZBT
- Up to 512kx36 bits ZBT supported
- User clock programmable between 0.5MHz and 100MHz
- User 68 pin front panel connector with 34 free I/O
- Supports 3.3V and 5V PCI signalling levels (VI/O)

2. Installation

This chapter explains how to install the ADM-XRC onto a PMC motherboard.

2.1. Motherboard requirements

The ADM-XRC must be installed in a PMC motherboard which supplies 3.3V power to the PMC connectors. Ensure that the motherboard satisfies this requirement before powering it up.
The ADM-XRC supports both 3.3V and 5V PCI signalling levels (VI/O).

2.2. Handling instructions

Observe precautions for preventing damage to components by electrostatic discharge. Personnel handling the board should take SSD precautions.
Avoid flexing the board.

2.3. Installing the ADM-XRC onto a PMC motherboard

Note: This operation should not be performed while the PMC motherboard is powered up.
The ADM-XRC must be secured to the PMC motherboard using M2.5 screws in the four holes provided. The PMC bezel through which the I/O connector protrudes should be flush with the front panel of the PMC motherboard.

2.4. Installing the ADM-XRC if fitted to an ADC-PMC

The ADM-XRC can be supplied for use in standard PC systems fitted to an ADC-PMC carrier board. The ADC-PMC can support up to two ADC-PMC cards whilst maintaining host PC PCI compatibility. There are no links or jumpers to set on the ADC-PMC. All that is required for installation is a 5V PCI slot that has enough space to accommodate the full-length card.
It should be noted that the ADC-PMC uses a standard bridge to provide a secondary PCI bus for the ADM-XRC and that some older BIOS code does not set up these devices correctly. Please ensure you have the latest version of BIOS appropriate for your machine. The ADC-PMC requires only 5V from the host PC.

3. PCI Bus Interface

The PCI bus is implemented in a PLX PCI9080 and is configured with settings as described later in this document to simplify the integration of user applications in the FPGA.

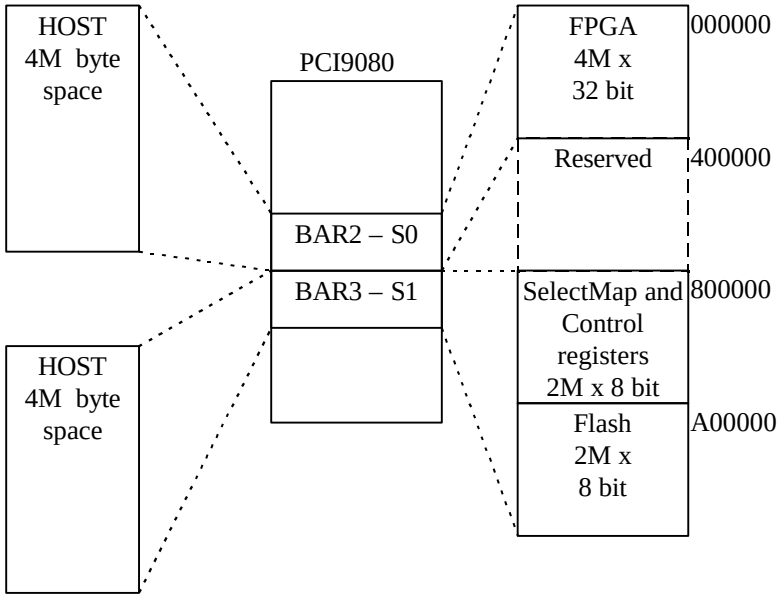
The PCI configuration space of the ADM-XRC is shown below.

Config. Offset	31	24	23	16	15	8	7	0
00	Device ID (0040) (9080)				Vendor ID (4144) (10B5)			
04	Status				Command			
08	Class Code							RevisionID
0C	BIST		HeaderType		Lat. Timer			Cache Line
10	PCI BAR0 (PLX Internal Registers/Memory)							
14	PCI BAR1 (PLX Internal Registers/IO)							
18	PCI BAR2 (Local Bus FPGA)							
1C	PCI BAR3 (Local Bus Control/Flash/SelectMap)							
20	PCI BAR4 (Not used)							
24	PCI BAR5 (Not used)							
28	Card Bus CIS Pointer(Not used)							
2C	Subsystem ID				Subsystem Vendor ID			
30	PCI Base Address for Local Expansion ROM							
34	Reserved							
38	Reserved							
3C	Max Lat		Min Gnt		Int. Pin			Int. Line

The PCI9080 uses the first two Bar's to provide access to its internal registers both via memory accesses and I/O accesses. Either BAR may be used by the host.
BAR 2 provides access to a 4Mbyte space for use by the FPGA and must be accessed only when a valid FPGA configuration is loaded that can respond correctly to local bus access.
BAR 3 provides access to the local control registers and the flash memory.

4. ADM-XRC Local Bus Architecture

It is useful to refer to the PLX PCI9080 user manual for information on the operation of the local bus and how address spaces map to the BAR's in PCI configuration space. The first two BAR's decode memory and I/O ranges for the PCI9080 internal registers.
There are two BAR's (at offsets 0x18 and 0x1C) that map the two main local bus address spaces, S0 and S1. In the ADM-XRC, S0 is a 4Mbyte memory address space, 32 bits wide and is available for the user to access the FPGA and for local bus control registers. S1 maps in a 4Mbyte memory address space, 8 bits wide for access to the flash and FPGA SelectMap port.



The PCI9080 can be programmed to support 8, 16 or 32 bit local bus widths and this feature is used to match with the device widths fitted on the ADM-XRC. Other than programming the BAR's with values determined by the host system, no other programming of the PCI9080 is required in order to access the local bus registers and devices.

4.1. Characteristics of Address Spaces

Space	Size	Width	Burst	Prefetch	Burst Term	Local Offset
S0	4MB	32	yes	yes	yes	0x00000000
S1	4MB	8	no	no	n/a	0x00800000

From the PCI side of the PCI9080, transfers to either space may be 8, 16 or 32 bit in width and of any length. The PCI9080 breaks up transfers to suit the address space on the local bus whilst respecting the characteristics outlined above.

4.2. Local Control Registers

The local bus control registers are implemented in a CPLD attached to the PCI9080 local bus. The map of these registers is shown below.

S1+offset	Write	Read
0	FCON	FSTAT
1	CCON	CSTAT
2	IMSET	IMSTAT
3	IMCLR	IMSTAT
4	ICON	ISTAT
5	PCON	PSTAT
6	MBZ	RAX
7	MBZ	RAX
8-15	SelectMap	SelectMap

Registers that are not implemented are marked Must Be Zero (MBZ) and Read As Don't Care (RAX). The same applies for bits within implemented registers that are not used or reserved.

4.2.1. FCON Register

The FPGA can be configured by the host system using the SelectMap port on the target device. Before this can be achieved, the FPGA must be initialised to an erased state. Asserting PROG and then releasing it will start the initialisation process. The INIT bit is only valid whilst the device is not configured, indicated by a zero in DONE. After configuration, the INIT pin becomes a user I/O pin and has no further function on the ADM-XRC. In this case the place and route program sets the INIT pin to an input with a weak pull down thus resulting in INIT appearing set.

	7	6	5	4	3	2	1	0	
FCON	MBZ	MBZ	MBZ	MBZ	MBZ	MBZ	INIT	PROG	W
FSTAT	RAX	RAX	RAX	RAX	RAX	DONE	INIT	PROG	R

FCON Function (W0/1 is write value, R is when read)
PROG W0 - Release PROGRAM to the FPGA
W1 - Asserts PROGRAM pin on the target FPGA
RX - Indicates state of FCON[PROG]
INIT W0 - Has no effect on FPGA INIT pin
W1 - Asserts INIT pin to FPGA to postpone configuration
R0 - FPGA has no error
R1 - FPGA has asserted INIT
DONE R0 - FPGA is not configured or is being configured
R1 - FPGA is successfully configured

4.2.2. CCON Register

The CCON register controls access to the ICD2061 clock generator. The range of frequencies supported is between 25 and 40 MHz. Although the ADM-ARC can operate at frequencies less than 25MHz, the CLKDLL circuits in the FPGA will not. The maximum frequency of the PCI9080 local bus is 40MHZ.
Refer to the ICD2061A data sheet for further information on determining clock generator settings.

	7	6	5	4	3	2	1	0	
CCON	MBZ	MBZ	MBZ	MBZ	INTCLK	FEATCLK	DATA/S1	CLK/S0	W
CSTAT	RAX	RAX	RAX	SERERR	INTCLK	FEATCLK	DATA/S1	CLK/S0	R

FCON	Function (W0/1 is write value, R is when read)
CLK	W0/1 Drives clock signal to ICD2061 Determines S0 when static
DATA	W0/1 Drives data signal to ICD2061 Determines S1 when static
FEATCLK	W0 - Do not use
INTCLK	W0 - Do not use

The ICD2061 is programmed using CLK and DATA bits to form manchester encoded sequences. Each sequence consists of a 5 bit unlock preamble followed by a 24 bit data word all of which must be programmed within 10ms from the start of the first bit. If programming is not completed within this time, the ICD2061 will assert SERERR and ignore the remainder of the sequence. When the ICD2061 is not being programmed, S1 and S0 select the internal VCLK programming register.

4.2.3. ICON Registers

The ICON registers consist of a mask register and an interrupt status register. The mask register can be set or cleared by writing to IMSET or IMCLR with a bit mask. The ADM-XRC only supports one interrupt from the local bus and is masked, set or cleared using bit 0. On reset the mask bits are set disabling local bus interrupts.

	7	6	5	4	3	2	1	0	
IMSET	MBZ	MBZ	MBZ	MBZ	MBZ	MBZ	MBZ	FINTM	W
IMCLR	MBZ	MBZ	MBZ	MBZ	MBZ	MBZ	MBZ	FINTM	W
IMSTAT	RAX	RAX	RAX	RAX	RAX	RAX	RAX	FINTM	R

The ICON register contains the status of the interrupt from the FPGA. This bit can be read independently of the state of the FINTM mask bit. To clear the FINT interrupt reported in ISTAT, write the corresponding bit in ICON.

	7	6	5	4	3	2	1	0	
ICON	MBZ	MBZ	MBZ	MBZ	MBZ	MBZ	MBZ	FINT	W
ISTAT	RAX	RAX	RAX	RAX	RAX	RAX	RAX	FINT	R

4.2.4. PSTAT Register

The PSTAT register presents information about the power supply to the Virtex device. The ADM-XRC generates power for the FPGA core from 5V, using a switch mode supply circuit that outputs two signals to indicate over-temperature and accuracy.

	7	6	5	4	3	2	1	0	
PSTAT	RAX	RAX	RAX	RAX	RAX	RAX	PTEMP	PGOOD	R

PGOOD R0 - PSU is out of range
R1 - PSU is within +/- 10%
PTEMP R0 - PSU has shutdown (thermal detect)
R1 - PSU is within operating range

4.2.5. SelectMAP Register

The ADM-XRC supports only SelectMAP download of configuration data to the Virtex FPGA. The SelectMAP register is a write only port (in the current ADM-XRC) that is written with configuration information.

	7	6	5	4	3	2	1	0	
SelectMap	Configuration Data Byte								W

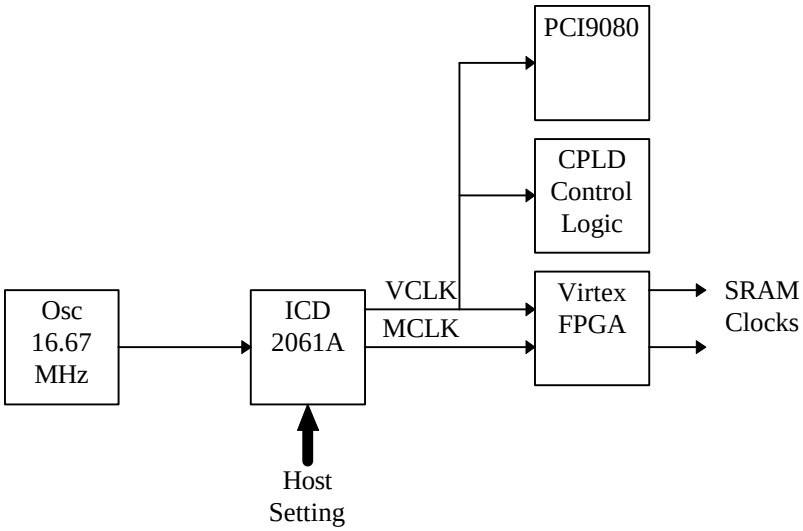
NOTE. Do not write to the SelectMAP register whilst DONE is set.

5. FPGA Operation

The following sections describe the operation of the FPGA in terms of the resources that are available. Where pin numbers are not mentioned, these can be found the constraints file for the V1000, supplied with the example FPGA design. Please refer to the application notes supplied in the installation directory for examples of working designs and guidelines.

5.1. Clock Distribution

The ADM-XRC uses the PLX PCI9080 local bus to provide a synchronous data transfer interface and all devices attached to the PCI9080 run at the local bus clock rate.
The VCLK output from the ICD2061 provides the local bus clock and by default runs at 28.5MHz. The VCLK output can be determined by three registers in the ICD2061, selected by the S1/S0 bits in the CCON register. It is recommended that alterations to VCLK use REG1 or REG2 to set the new value so that on reset the VCLK output selects REG0 when S1/S0 are reset.



5.2. Input Clocks

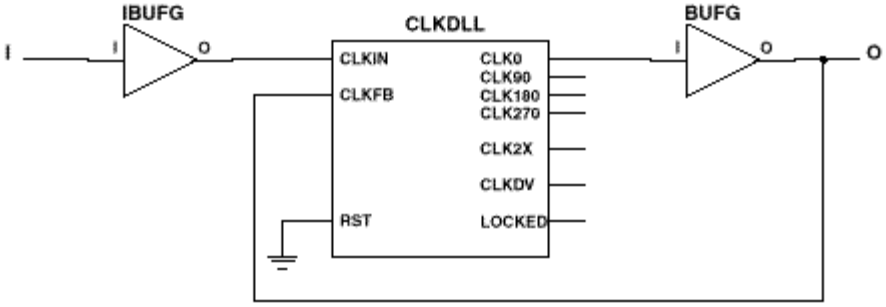
There are two primary clock inputs to the FPGA, both from the programmable clock generator. The MCLK signal from the clock generator is used only by the FPGA. It is therefore free for use by logic in the FPGA.

The VCLK signal from the clock generator is the local bus clock, used by the FPGA, PLX PCI9080 and a support CPLD. In addition, this clock is routed to the FPGA CCLK pin to enable SelectMap configuration at local bus speed.

Both MCLK and VCLK can be programmed between 400kHz and 100MHz.

A restriction on VCLK is that it must not exceed 40MHz,the maximum speed of the PCI9080, and should not be lower than 25MHz for reliable CLKDLL operation.

VCLK is input to the FPGA on GCK3 (pin A17) and should be used to drive a Virtex CLKDLL circuit which aligns the internal FPGA clock to the local bus clock. The circuit below demonstrates how to align the internal clock of the FPGA to the local bus clock. The output of the BUFG is available to all flip-flops in the design.



MCLK is input to the FPGA on GCK1 (pin AJ17) and can be used for any purpose within the FPGA.

5.3. Output Clocks

The FPGA is responsible for providing clocks to the SRAM's and also for aligning its internal global clock with the local bus clock. To do this requires the use of Virtex CLKDLL's that are specifically designed for the purpose of minimising skew between external and internal clock domains.
The R0_R2CLK and R1_R3CLK should each be slaved to the FPGA clock on the output of the BUFG in the above diagram. The feedback for the SRAM CLKDLL should come form R2_CLK. The same applies for R1_R3CLK and R3_CLK.
NOTE. There are issues arising from the use of CLKDLL modules in some service packs for 2.1i software. Please ensure that the latest service pack has been applied to your system.

5.4. Local Bus

The local bus of the ADM-XRC uses the PCI9080 to provide a non-multiplexed address and data capability with synchronous speeds of up to 40MHz, independent of PCI operation. Whilst the local bus is capable of achieving near PCI performance, it is much simpler to interface with than PCI. The ADM-XRC routes most of the local bus signals to the FPGA and devotes an entire address space to the FPGA.
The signals provided are :-

Signal	Active	Direction	Purpose
LA[23:2]	high	IN	Address
LBE[3:0}	low	IN	Address/byte enables
LD[31:0]	high	BIDIR	Data Bus
LWRITE	high	IN	Write cycle when true, read when false
LBLASTL	low	IN	End of burst

LADSL	low	IN	Address / data start
LBTERML	low	OUT/TRI	Burst terminate
LREADYIL	low	OUT/TRI	Accepts/completes data transfer
LDREQL[1:0]	low	OUT	Request DMA transfer
LDACKL[1:0]	low	IN	DMA transfer acknowledge
LEOTL[1:0]	low	OUT	Terminate current DMA transfer
LINTIL	low	OUT	Interrupt (via CPLD)
FHOLD	high	OUT	Reserved for future use
FHOLDA	high	IN	Reserved for future use
LRESETOL	low	IN	Reset from PLX and PCI
LCLKA	high	IN	Local bus clock (VCLK from ICD2061)

The local bus provides 4Mbytes of address space for the FPGA to use for whatever purpose is desired by the application. LA[23:21]=000 should be used to determine when the FPGA is being accessed.

Example Verilog code demonstrates how to interface to the local bus and provide access to the SSRAM's for test purposes. User applications can define how the address space allocated to the FPGA is mapped to the local bus and may or may not provide access to the SRAM memory.

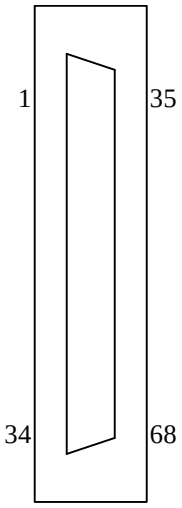
5.5. Synchronous SRAM

The four banks of synchronous SRAM are identical in device type and FPGA interface. The devices fitted as standard are Micron MT55L256L36FT-10 which are 256Kx36 bits flow through synchronous burst type with 10ns access time.
The pins are allocated to support synchronous burst or ZBT SRAM and each bank provides the following interface

FPGA Pin	Active	Type	Function
Rn_D[35:0]	high	Bidir	Data Bus
Rn_A[16:0]	high	OUT	Address Bus
Rn_WEL[3:0]	low	OUT	Byte enables [35,31:24] …. [32,7:0]
Rn_GWEL	low	OUT	Global write enable
Rn_OEL	low	OUT	Output enable
Rn_CEL	low	OUT	Chip enable
Rn_ADVL	low	OUT	SYNC: ADSC function ZBT: ADV (advance)function
Rn_C2L	low	OUT	SYNC: ADV (advance) function ZBT: no function
Rn_CKEL	low	OUT	SYNC: Synchronous write enable ZBT: Clock enable
R0_R2CLK	high	OUT	Clock to SRAM 0 and SRAM 2
R2_CLK	HIGH	IN	Feedback for CLKDLL
R1_R3CLK	high	OUT	Clock to SRAM 1 and SRAM 3
R3_CLK	HIGH	IN	Feedback for CLKDLL

5.6. User I/O Front Panel Variant

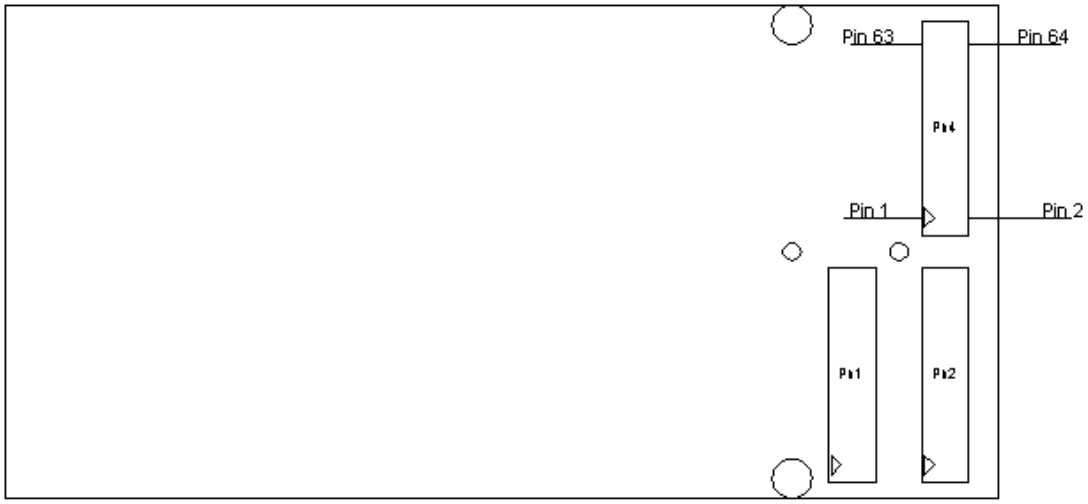
User I/O is presented on a 68 way miniature D connector (of SCSI-2 style) with interleaved signal and ground pairs as shown below.



Signal	Pin	Pin	Signal
All GND	1	35	USER[0]
	2	36	USER[1]
	3	37	USER[2]
	4	38	USER[3]
	5	39	USER[4]
	6	40	USER[5]
	7	41	USER[6]
	8	42	USER[7]
	9	42	USER[8]
	10	44	USER[9]
	11	45	USER[10]
	12	46	USER[11]
	13	47	USER[12]
	14	48	USER[13]
	15	49	USER[14]
	16	50	USER[15]
	17	51	USER[16]
	18	52	USER[17]
	19	53	USER[18]
	20	54	USER[19]
	21	55	USER[20]
	22	56	USER[21]
	23	57	USER[22]
	24	58	USER[23]
	25	59	USER[24]
	26	60	USER[25]
	27	61	USER[26]
	28	62	USER[27]
	29	63	USER[28]
	30	64	USER[29]
	31	65	USER[30]
	32	66	USER[31]
	33	67	USER[32]
	34	68	USER[33]

5.7. User I/O PMC Pn4 Variant

User I/O is presented on the User Connector Pn4 via a standard 64-way PMC connector. This should be routed via a suitable CMC compliant motherboard to an external I/O adapter.



Signal	Pn4 Pin	Pn4 Pin	Signal
USER[0]	1	2	USER[1]
USER[2]	3	4	USER[3]
USER[4]	5	6	GND
USER[5]	7	8	USER[6]
USER[7]	9	10	USER[8]
USER[8]	11	12	USER[10]
USER[11]	13	14	USER[12]
USER[13]	15	16	USER[14]
GND	17	18	USER[15]
USER[16]	19	20	USER[17]
USER[18]	21	22	USER[19]
USER[20]	23	24	USER[21]
USER[22]	25	26	USER[23]
USER[24]	27	28	GND
USER[25]	29	30	USER[26]
GND	31	32	GND
USER[27]	33	34	USER[28]
USER[29]	35	36	USER[30]
GND	37	38	USER[31]
USER[32]	39	40	USER[33]
USER[34]	41	42	USER[35]
USER[36]	43	44	USER[37]
USER[38]	45	46	USER[39]
USER[40]	47	48	USER[41]
USER[42]	49	50	GND
USER[43]	51	52	USER[44]
USER[45]	53	54	USER[46]
USER[47]	55	56	USER[48]
USER[49]	57	58	USER[50]
GND	59	60	GND
USER[51]	61	62	USER[52]
GND	63	64	GND

6. Configuring the FPGA

The Virtex FPGA family support a mode of configuration referred to as SelectMAP. The ADM-XRC uses the local bus clock to synchronise byte loading through the SelectMAp interface and is thus limited to a maximum of 40Mbytes/sec.

6.1. SelectMAP Operation

Before the FPGA can be configured using SelectMap, the FPGA must be in a state where it is ready to accept data. This can be confirmed by the following process :-

1. Assert PROG and INIT, hold for 20 usec.
2. Release PROG and INIT, wait for 50usec
3. Check that INIT is not set.

If INIT is clear then configuration can proceed.

Configuration is a simple process and requires the entire bitstream to be written to the SelectMap register. At the end of the process, DONE should be high. If DONE is not high and INIT is set then an error has occurred and will probably be due to an invalid bitstream. Note that INIT is not valid when DONE is set as it becomes a user I/O after configuration and is pulled low (active) by default.

6.2. Bitstream Issues

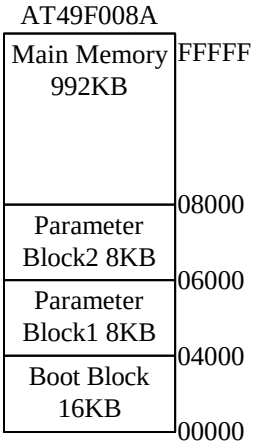
The bitstream produced by the **bitgen** program is stored in a **design.bit** file and is suitable for use with the download cables produced by Xilinx. The driver for the ADM-XRC loads this file and determines the location of the binary bitstream data within it. This data is not suitable for writing directly to the SelectMap registers as it is bit reversed. The driver performs this bit reversal on each byte before download to the FPGA.
Files generated by the **promgen** program and stored in **mcs** format for example are not bit reversed but the resulting file is around three times larger than the binary **design.bit** file.

7. Interrupts

The PCI9080 can provide a number of interrupts from internal sources as well as from the local bus. The FPGA can interrupt the host system by asserting the LINTIL (active low) signal and keeping it asserted until the source of the interrupt is cleared. See the ICON registers on how to mask and respond to local bus interrupts.

8. Flash Memory

The flash memory is implemented by an Atmel AT49F008A device with 1Mbyte capacity. The device is sectored as shown :-



The flash memory is located is Space 1 (S1) at offset 0x200000-0x2FFFFF. The device can be written or read without restriction and is available at all times. The flash device is connected to the local bus reset signal to allow the device to resume normal operation if the system hangs during a programming sequence.
The Atmel device is easy to program and erase and has practically no timing constraints. Each byte must be written individually but with the advantage of 10usec typical programming time.
The following sequences are required for programming the flash device.

	Cycle 1		Cycle 2		Cycle 3		Cycle 4		Cycle 5		Cycle 6	
COMMAND	**Addr**	**Data**	**Addr**	**Data**	**Addr**	**Data**	**Addr**	**Data**	**Addr**	**Data**	**Addr**	**Data**
Chip Erase	5555	AA	2AAA	55	5555	80	5555	AA	2AAA	55	5555	10
Sector Erase	5555	AA	2AAA	55	5555	80	5555	AA	2AAA	55	SA	30
Byte Program	5555	AA	2AAA	55	5555	A0	address	byte				
Boot Lockout	5555	AA	2AAA	55	5555	80	5555	AA	2AAA	55	5555	40
Product ID	5555	AA	2AAA	55	5555	90						
Leave ID	5555	AA	2AAA	55	5555	F0						
Leave ID	xxxx	F0										

The erased state of the device is that all locations contain 0xFF;

The ADM-XRC is capable of loading the FPGA from flash on power up or reset and will load the bitstream from the main memory section starting at 0x8001. This is to avoid any problem with the boot block which, if locked out, cannot be unlocked without removing the flash device from the PCB. The byte at location 0x8000 should be programmed with bit 0 = 0 (all other bits are ignored) to enable the load sequence otherwise the assumption is the flash is blank and should not be used.

The following fragment of ‘C’ code demonstrates how to write a byte with value 0x77 to the flash at location 0x1234.

```
UCHAR *Flash;
```

```
Flash = S1 + 0x200000;  //    S1 points to the start of space 1
Flash[0x5555] = 0xAA;   //    First cycle
Flash[0x2AAA] = 0x55;   //    Second cycle
Flash[0x5555] = 0xA0;   //    Enter programming mode
Flash[0x1234] = 0x77;   //    Write byte
Wait(50);               //    Wait for 50 usec worst case
```

Instead of the wait, the device may be polled to check for programming completion. The simplest method is to read the flash device and compare the result with the data just written. Whilst programming is in progress, D7 is the complement of the data written.

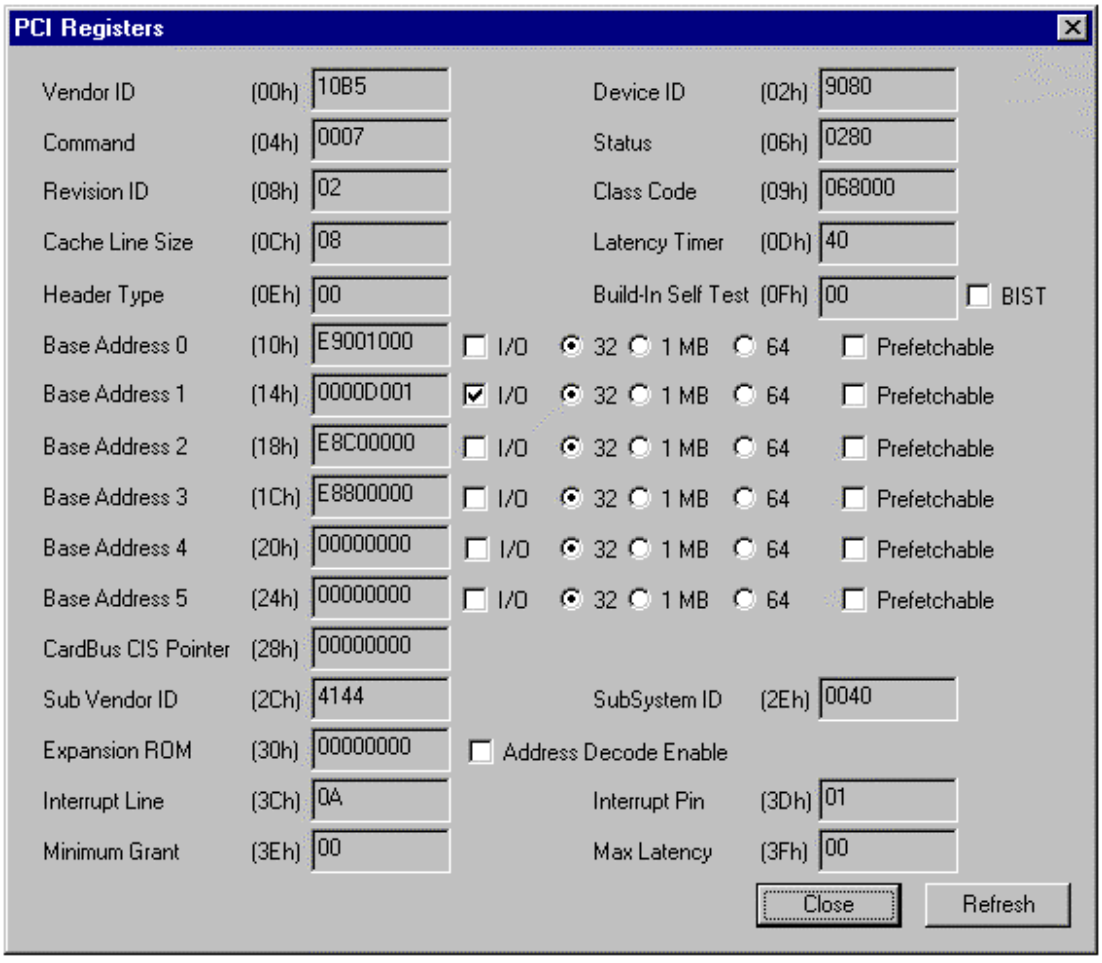
9. PLX PCI9080 Initialisation

The PCI9080 is configured at power-up and reset by a serial EEPROM attached to it. This device is configured at the factory with settings to suit the standard operation of the ADM-XRC. For the values in the EEPROM, see the following chapter.

After reset, the PCI9080 loads initial settings for PCI and other registers from the EEPROM. The following sections show the PCI9080 registers after booting on a system with a PCI BIOS. The screen shots were produced by PLXMON98 on a WIN95 machine. PLXMON98 for WINNT and WIN95 is available from PLX Technology or its distributors.

9.1. PCI Registers

The PCI9080 PCI registers are accessed in PCI configuration space primarily during system boot to configure resources requested by the ADM-XRC.

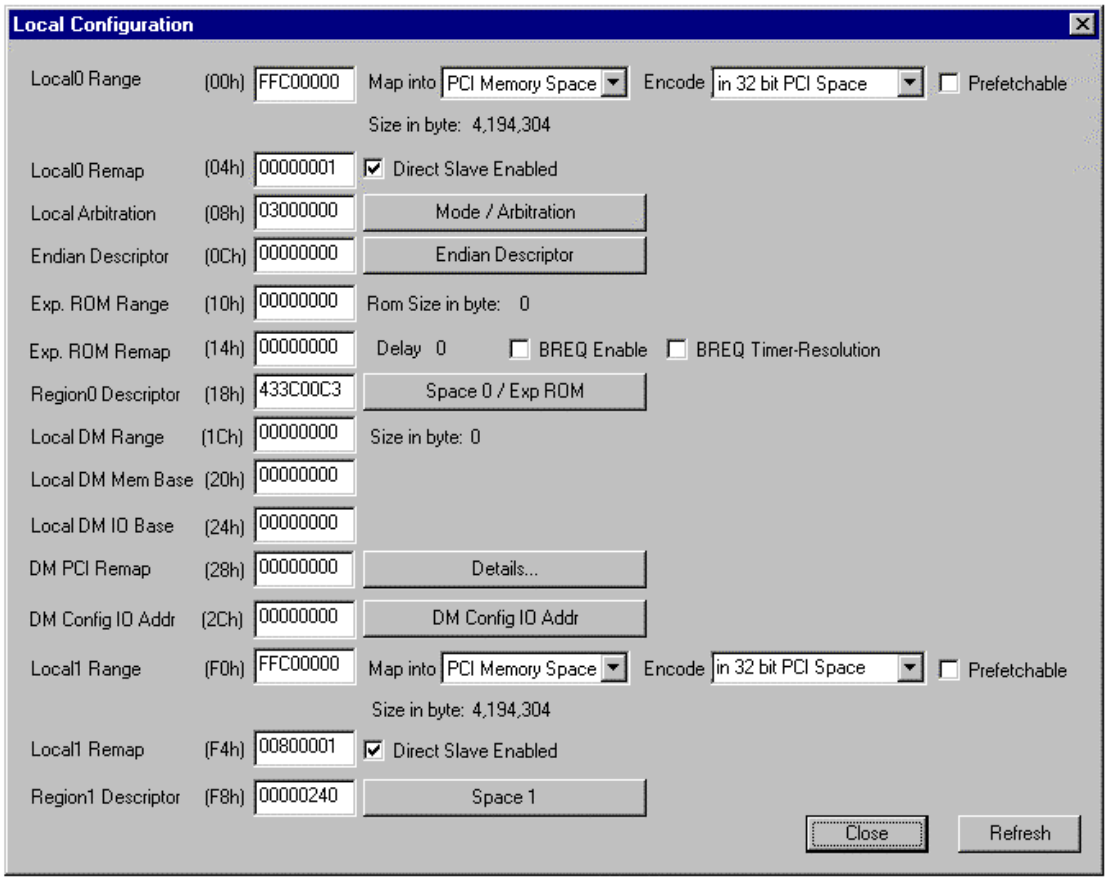


The main points to note are that the device and vendor ID's are 9080/10B5 and the command register is set for memory and I/O access.

BAR0(10h) is allocated to 32 bit memory space and is used for access to all of the PCI9080 control registers. BAR1(14h) is allocated to I/O space and is used for the same purpose as BAR1.
BAR2 defines the PCI9080 Local Bus address range Space 0 (S0) and is allocated to 32 bit memory space. BAR3 maps a similar amount to BAR2 and is used for access to Space 1 (S1).
It is important to note that as far as PCI space is concerned, there is no difference between S0 and S1. The Local Bus registers define the operation of these spaces.

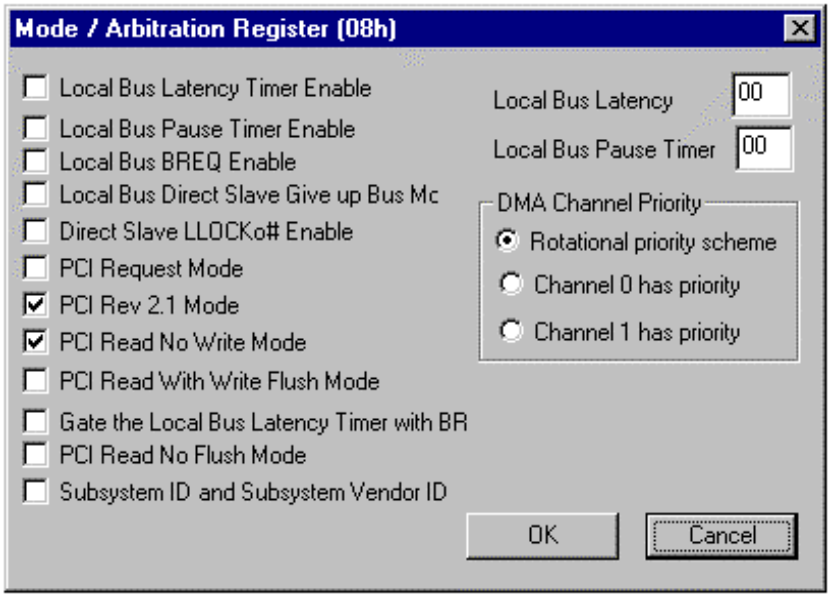
9.2. Local Configuration Registers

The PCI9080 Local Bus provides two main address spaces through which accesses to local bus resources can be made. As described earlier, S0 is a 4Mbyte space that is 32 bits wide, allocated to the FPGA. S1 is also a 4Mbyte space and is allocated to the flash prom and control registers. The Local Configuration registers are shown below in summary with more detailed descriptions following them.



9.2.1. Mode/Arbitration Register

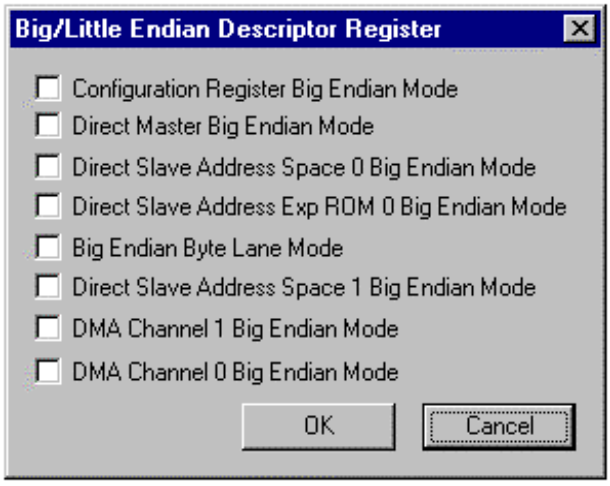
The Mode/Arbitration Register is usually set by the EEPROM initialisation and left unaltered after boot.



The **PCI Rev 2.1 Mode** pin sets the behaviour of the PCI9080 to conform to PCI revision 2.1 with regard to posted transactions.
The **PCI Read No Write Mode** bit is set to cause PCI writes to the PCI9080 to retry whilst a posted read is pending. This is a known workaround in REV 3 silicon.

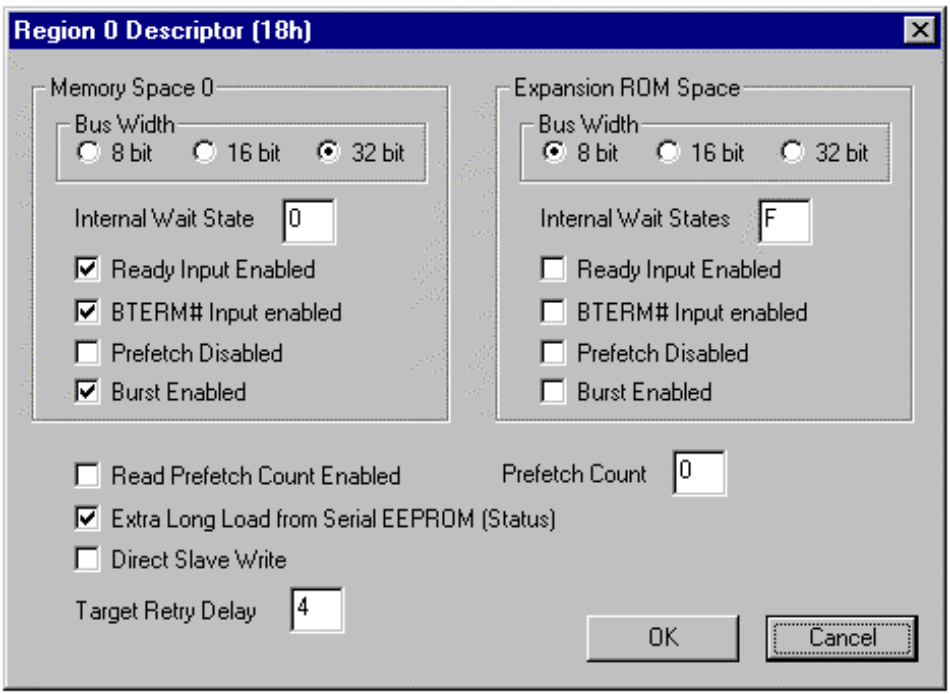
9.2.2. Big/Little Endian Descriptor Register

This register can be used to force big endian mode for various transfers. The ADM-XRC does not perform any endian conversions by default.



9.2.3. Region 0 Descriptor

The Region 0 Descriptor describes the attributes of Local Bus Space 0. It can be seen that 32 bit local bus width is selected and that ready must be generated by the target space. In this case, Space 0 is allocated to the FPGA so all accesses to Space 0 must be acknowledged by the FPGA.
The BTERM bit is set which means that bursts of greater the four LWORD's are permitted. This also means that the FPGA can break a burst transfer into smaller lengths by asserting BTERM. No wait states are generated by the PCI9080 for Space 0 - all wait states are determined by the FPGA.
The Expansion ROM Space is not currently used in the ADM-XRC.
The Extra Long Load from Serial EEPROM bit is set to indicate that a long load did occur during the EEPROM read cycle. The length of the EEPROM load is determined by the contents of the EEPROM.

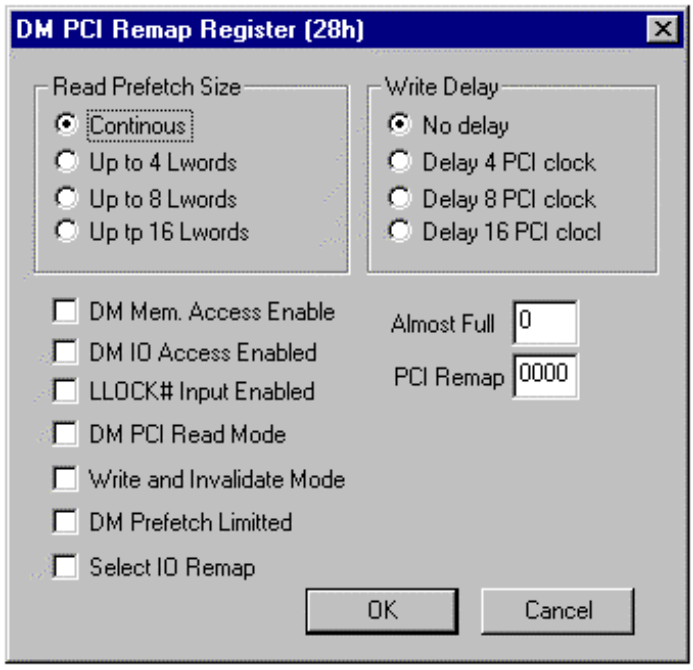


Notes.

1. The bus width of memory space 0 is set to 32 bits by default. As this region is totally under control of the FPGA, it may be changed. It is the responsibility of the FPGA designer to take this into account.
2. The Extra Long Load from Serial EEPROM bit indicates that the extended EEPROM load was performed. This is required to set up Space 1 and some other registers.
3. Expansion ROM Space is not implemented in the ADM-XRC and should not be altered.

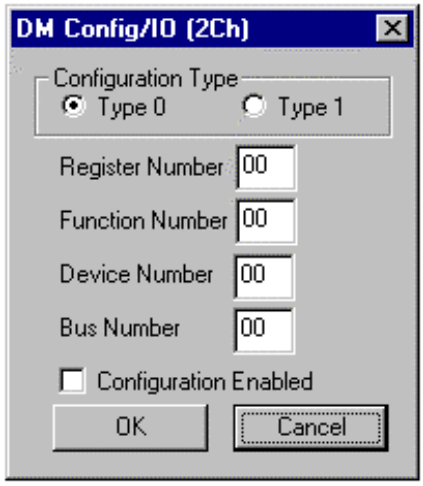
9.2.4. Direct Master PCI Remap Register

This register is for Local Bus initiated PCI Bus transactions (Direct Master) and is not used in the ADM-XRC.



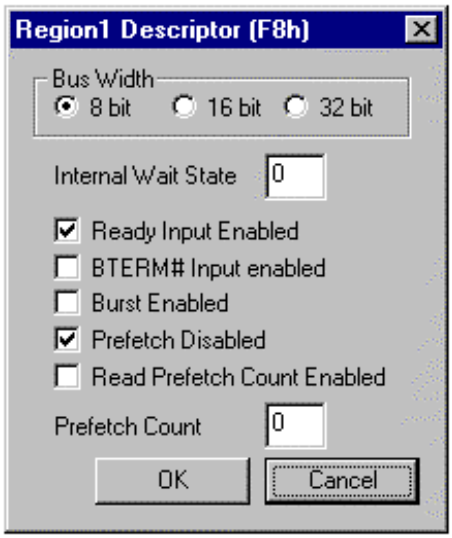
9.2.5. DM Config/IO Register

The DM (Direct Master) Config/IO register is used for controlling configuration cycles on the PCI bus. It is not used in the ADM-XRC.



9.2.6. Region 1 Descriptor

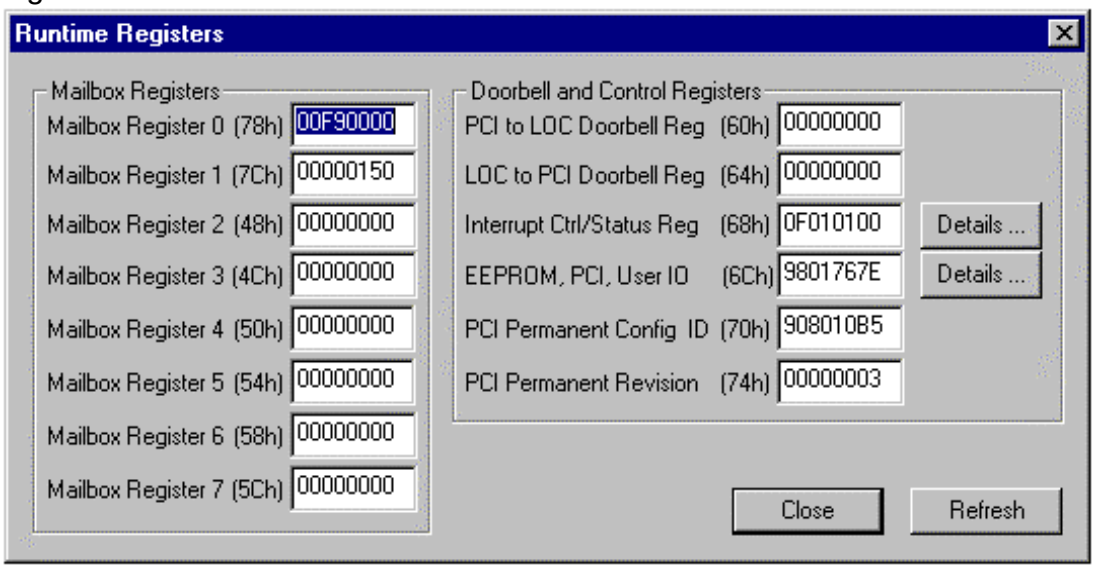
The Region 1 Descriptor describes the attributes of Local Bus Space 1. It can be seen that 8 bit local bus width is selected and that ready must be generated by the target space. Burst transfers are not enabled in this space to avoid any side effects of prefetching from control and /or FPGA configuration registers.
Space 1 maps the flash memory, control registers and FPGA SelectMAP port.



Notes.
1. Do not enable burst for this region as it may cause side effects that will stop FPGA loading and readback.
2. Internal wait states should always be 0.
3. Bus width is always 8 bit.

9.3. Runtime Registers

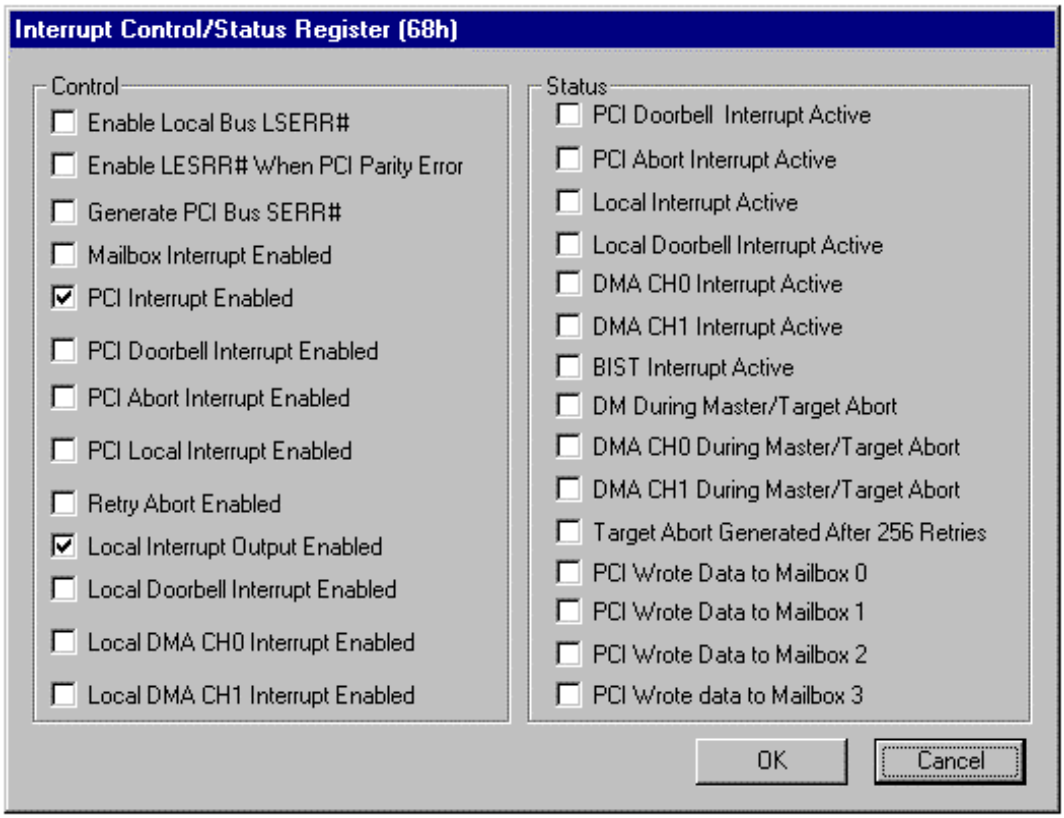
The runtime registers group together mailboxes, control and status registers. The only registers applicable to the ADM-XRC in this group are the INTCSR register at offset 68h and CNTRL at offset 6Ch.



It should be noted that Mailbox 0 and 1 can be set to initial values using the EEPROM.

9.3.1. Interrupt Control/Status Register

In order to for a PCI host processor to receive interrupts from the many sources in the PCI9080, the appropriate enable bits in the INTCSR must be set.

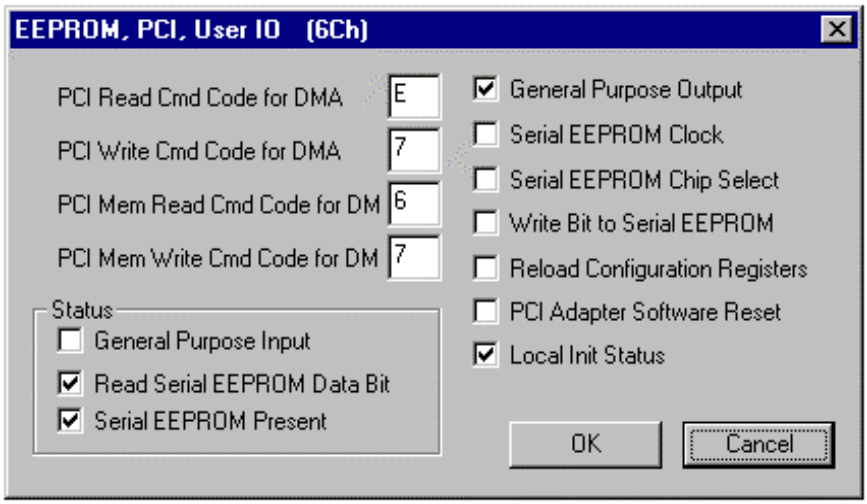


Notes.

1. Refer to the PLX user manual for the actual register format.
2. Clearing a bit in this register does not remove the interrupting source, it simply masks it.

9.3.2. EEPROM, PCI, User IO

This register is known as CNTRL and reports information about the state of the EEPROM interface, DMA transfer codes and general purpose input and output bits.



Notes.

1. A fault with the EEPROM or if the EEPROM is blank will result in the **Serial EEPROM Present** bit being cleared.
2. This register can be used to write and read the EEPROM for initial programming or to perform changes to the initialisation code.

10. EEPROM Contents

Offset	Value	Register	Description
0	10B5		Device ID
2	9080		Vendor ID
4	0680		Class Code
6	0002		Class Code / Revision
8	0000		Maximum Latency / Minimum Grant
A	01FF		Interrupt Pin / Interrupt Line Routing
C	0000		MSW Mailbox 0
E	4144		LSW Mailbox 0
10	0000		MSW Mailbox 1
12	0150		LSW Mailbox 1
14	FFC0	LAS0RR	MSW of Range for PCI to Local Address Space 0
16	0000		LSW of Range for PCI to Local Address Space 0
18	0000	LAS0BA	MSW of Local Base Address (Remap) for PCI to Local Address Space 0
1A	0001		LSW of Local Base Address (Remap) for PCI to Local Address Space 0
1C	0300	MARBR	MSW of Local Arbitration Register
1E	0000		LSW of Local Arbitration Register
20	0000	BIGEND	MSW of Local Bus Big/Little Endian Descriptor Register
22	0000		LSW of Local Bus Big/Little Endian Descriptor Register
24	0000	EROMRR	MSW of Range for PCI to Local Expansion ROM
26	0000		LSW of Range for PCI to Local Expansion ROM
28	0000	EROMBA	MSW of Local Base Address (Remap) for PCI to Local Expansion ROM
2A	0000		LSW of Local Base Address (Remap) for PCI to Local Expansion ROM
2C	433C	LBRD0	MSW of Bus Region Descriptors for PCI to Local Accesses
2E	00C3		LSW of Bus Region Descriptors for PCI to Local Accesses
30	0000	DMRR	MSW of range for Direct Master to PCI
32	0000		LSW of range for Direct Master to PCI
34	0000	DMLBAM	MSW of Local Base Address for Direct Master to PCI Memory
36	0000		LSW of Local Base Address for Direct Master to PCI Memory
38	0000	DMLBAI	MSW of Local Bus Address for Direct Master to PCI I/O
3A	0000		LSW of Local Bus Address for Direct Master to PCI I/O
3C	0000	DMPBAM	MSW of PCI Base Address (Remap) for Direct Master to PCI
3E	0000		LSW of PCI Base Address (Remap) for Direct Master to PCI
40	0000	DMCFGA	MSW of PCI Configuration Address Reg for Direct Master to PCI IO/CFG
42	0000		LSW of PCI Configuration Address Reg for Direct Master to PCI IO/CFG
44	0040		Subsystem ID
46	4144		Subsystem Vendor ID
48	FFC0	LAS1RR	MSW of Range for PCI to Local Address Space 1 (1MB)
4A	0000		LSW of Range for PCI to Local Address Space 1 (1MB)
4C	0080	LAS1BA	MSW of Local Base Address (Remap) for PCI to Local Address Space 1
4E	0001		LSW of Local Base Address (Remap) for PCI to Local Address Space 1
50	0000	LBRD1	MSW of Bus Region Descriptors (Space 1) for PCI to local accesses
52	3400		LSW of Bus Region Descriptors (Space 1) for PCI to local accesses
54	0000	PCIERBAR	MSW of PCI Base Address for local expansion ROM
56	0000		LSW of PCI Base Address for local expansion ROM

11. FPGA Pins

11.1. Clocks

```
#Local bus clock
NET "LCLKA" LOC = "A17";

#SRAM 0 and 2 Clock
NET "R0R2CLK" LOC = "C18";

#SRAM 0 and 2 Feedback
NET "R2_CLK" LOC = "D17";

#SRAM 1 and 3 Clock
NET "R1R3CLK" LOC = "AN17";

#SRAM 1 and 3 Feedback
NET "R3_CLK" LOC = "AL17";
```

11.2. Local Bus

```
NET "LD<31>" LOC = "H31";
NET "LD<30>" LOC = "K29";
NET "LD<29>" LOC = "H32";
NET "LD<28>" LOC = "J31";
NET "LD<27>" LOC = "K30";
NET "LD<26>" LOC = "H33";
NET "LD<25>" LOC = "L29";
NET "LD<24>" LOC = "K31";
NET "LD<23>" LOC = "L30";
NET "LD<22>" LOC = "J33";
NET "LD<21>" LOC = "M29";
NET "LD<20>" LOC = "L31";
NET "LD<19>" LOC = "M30";
NET "LD<18>" LOC = "L32";
NET "LD<17>" LOC = "M31";
NET "LD<16>" LOC = "L33";
NET "LD<15>" LOC = "N30";
NET "LD<14>" LOC = "N31";
NET "LD<13>" LOC = "M32";
NET "LD<12>" LOC = "P29";
NET "LD<11>" LOC = "P30";
NET "LD<10>" LOC = "P31";
NET "LD<9>" LOC = "P32";
NET "LD<8>" LOC = "R29";
NET "LD<7>" LOC = "R30";
NET "LD<6>" LOC = "R31";
```

```
NET "LD<5>" LOC = "R33";
NET "LD<4>" LOC = "T31";
NET "LD<3>" LOC = "T29";
NET "LD<2>" LOC = "T30";
NET "LD<1>" LOC = "T32";
NET "LD<0>" LOC = "U31";

NET "LA<23>" LOC = "D28";
NET "LA<22>" LOC = "C30";
NET "LA<21>" LOC = "D29";
NET "LA<20>" LOC = "E28";
NET "LA<19>" LOC = "D30";
NET "LA<18>" LOC = "F29";
NET "LA<17>" LOC = "D31";
NET "LA<16>" LOC = "F30";
NET "LA<15>" LOC = "C33";
NET "LA<14>" LOC = "G29";
NET "LA<13>" LOC = "E31";
NET "LA<12>" LOC = "D32";
NET "LA<11>" LOC = "G30";
NET "LA<10>" LOC = "F31";
NET "LA<9>" LOC = "H29";
NET "LA<8>" LOC = "E32";
NET "LA<7>" LOC = "E33";
NET "LA<6>" LOC = "G31";
NET "LA<5>" LOC = "J29";
NET "LA<4>" LOC = "F33";
NET "LA<3>" LOC = "G32";
NET "LA<2>" LOC = "J30";

NET "LBE<3>" LOC = "D27";
NET "LBE<2>" LOC = "B30";
NET "LBE<1>" LOC = "C29";
NET "LBE<0>" LOC = "AL18";
NET "LADSL" LOC = "C28";
NET "LWRITE" LOC = "E26";
NET "LDACKL<1>" LOC = "D26";
NET "LDACKL<0>" LOC = "B29";
NET "LBLASTL" LOC = "C27";
NET "LBTERML" LOC = "E25";
NET "LREADYIL" LOC = "A28";
NET "FHOLDA" LOC = "A27";
NET "LINTIL" LOC = "D25";
NET "LDREQL<1>" LOC = "E24";
NET "LDREQL<0>" LOC = "C26";
NET "FHOLD" LOC = "B26";
NET "LRESETOL" LOC = "AM18";
```

11.3. SSRAM Bank 0

```
NET "R0_A<16>" LOC = "D3";
NET "R0_A<15>" LOC = "F4";
NET "R0_A<14>" LOC = "C1";
NET "R0_A<13>" LOC = "G5";
NET "R0_A<12>" LOC = "E3";
NET "R0_A<11>" LOC = "D2";
NET "R0_A<10>" LOC = "G4";
NET "R0_A<9>" LOC = "F3";
NET "R0_A<8>" LOC = "H5";
NET "R0_A<7>" LOC = "E2";
NET "R0_A<6>" LOC = "H4";
NET "R0_A<5>" LOC = "G3";
NET "R0_A<4>" LOC = "J5";
NET "R0_A<3>" LOC = "F1";
NET "R0_A<2>" LOC = "G1";
NET "R0_A<1>" LOC = "J4";
NET "R0_A<0>" LOC = "H3";
NET "R0_D<35>" LOC = "K5";
NET "R0_D<34>" LOC = "H2";
NET "R0_D<33>" LOC = "J3";
NET "R0_D<32>" LOC = "K4";
NET "R0_D<31>" LOC = "J2";
NET "R0_D<30>" LOC = "L5";
NET "R0_D<29>" LOC = "K2";
NET "R0_D<28>" LOC = "M5";
NET "R0_D<27>" LOC = "L3";
NET "R0_D<26>" LOC = "L1";
NET "R0_D<25>" LOC = "M4";
NET "R0_D<24>" LOC = "N5";
NET "R0_D<23>" LOC = "M2";
NET "R0_D<22>" LOC = "N4";
NET "R0_D<21>" LOC = "N3";
NET "R0_D<20>" LOC = "N2";
NET "R0_D<19>" LOC = "P5";
NET "R0_D<18>" LOC = "P4";
NET "R0_D<17>" LOC = "P2";
NET "R0_D<16>" LOC = "R5";
NET "R0_D<15>" LOC = "R4";
NET "R0_D<14>" LOC = "R3";
NET "R0_D<13>" LOC = "R1";
NET "R0_D<12>" LOC = "T4";
NET "R0_D<11>" LOC = "T5";
NET "R0_D<10>" LOC = "T2";
NET "R0_D<9>" LOC = "U3";
NET "R0_D<8>" LOC = "U1";
NET "R0_D<7>" LOC = "U2";
NET "R0_D<6>" LOC = "V2";
```

```
NET "R0_D<5>" LOC = "V4";
NET "R0_D<4>" LOC = "V5";
NET "R0_D<3>" LOC = "V3";
NET "R0_D<2>" LOC = "W1";
NET "R0_D<1>" LOC = "W3";
NET "R0_D<0>" LOC = "W5";
NET "R0_WEL<3>" LOC = "D7";
NET "R0_WEL<2>" LOC = "C5";
NET "R0_WEL<1>" LOC = "E7";
NET "R0_WEL<0>" LOC = "A3";
NET "R0_GWEL" LOC = "D8";
NET "R0_OEL" LOC = "B4";
NET "R0_CEL" LOC = "A5";
NET "R0_ADVL" LOC = "B5";
NET "R0_C2L" LOC = "E8";
NET "R0_CKEL" LOC = "C6";
```

11.4. SSRAM Bank 1

```
NET "R1_A<16>" LOC = "AK9";
NET "R1_A<15>" LOC = "AL8";
NET "R1_A<14>" LOC = "AN7";
NET "R1_A<13>" LOC = "AJ9";
NET "R1_A<12>" LOC = "AL7";
NET "R1_A<11>" LOC = "AN6";
NET "R1_A<10>" LOC = "AM6";
NET "R1_A<9>" LOC = "AJ8";
NET "R1_A<8>" LOC = "AL6";
NET "R1_A<7>" LOC = "AK7";
NET "R1_A<6>" LOC = "AM5";
NET "R1_A<5>" LOC = "AM4";
NET "R1_A<4>" LOC = "AJ7";
NET "R1_A<3>" LOC = "AL5";
NET "R1_A<2>" LOC = "AK6";
NET "R1_A<1>" LOC = "AN3";
NET "R1_A<0>" LOC = "AL4";
NET "R1_D<35>" LOC = "Y3";
NET "R1_D<34>" LOC = "Y4";
NET "R1_D<33>" LOC = "AA1";
NET "R1_D<32>" LOC = "Y5";
NET "R1_D<31>" LOC = "AA3";
NET "R1_D<30>" LOC = "AA4";
NET "R1_D<29>" LOC = "AB3";
NET "R1_D<28>" LOC = "AA5";
NET "R1_D<27>" LOC = "AC1";
NET "R1_D<26>" LOC = "AB4";
NET "R1_D<25>" LOC = "AC3";
NET "R1_D<24>" LOC = "AD3";
NET "R1_D<23>" LOC = "AE1";
NET "R1_D<22>" LOC = "AC5";
NET "R1_D<21>" LOC = "AE3";
NET "R1_D<20>" LOC = "AD4";
NET "R1_D<19>" LOC = "AF1";
NET "R1_D<18>" LOC = "AF2";
NET "R1_D<17>" LOC = "AD5";
NET "R1_D<16>" LOC = "AG2";
NET "R1_D<15>" LOC = "AE4";
NET "R1_D<14>" LOC = "AF3";
NET "R1_D<13>" LOC = "AH1";
NET "R1_D<12>" LOC = "AE5";
NET "R1_D<11>" LOC = "AF4";
NET "R1_D<10>" LOC = "AJ1";
NET "R1_D<9>" LOC = "AJ2";
NET "R1_D<8>" LOC = "AF5";
NET "R1_D<7>" LOC = "AH3";
NET "R1_D<6>" LOC = "AG4";
```

```
NET "R1_D<5>" LOC = "AK2";
NET "R1_D<4>" LOC = "AJ3";
NET "R1_D<3>" LOC = "AG5";
NET "R1_D<2>" LOC = "AL1";
NET "R1_D<1>" LOC = "AH4";
NET "R1_D<0>" LOC = "AK3";
NET "R1_WEL<3>" LOC = "AK10";
NET "R1_WEL<2>" LOC = "AM9";
NET "R1_WEL<1>" LOC = "AL9";
NET "R1_WEL<0>" LOC = "AJ10";
NET "R1_GWEL" LOC = "AK12";
NET "R1_OEL" LOC = "AM10";
NET "R1_CEL" LOC = "AJ12";
NET "R1_ADVL" LOC = "AL11";
NET "R1_C2L" LOC = "AL10";
NET "R1_CKEL" LOC = "AJ11";
```

11.5. SSRAM Bank 2

```
NET "R2_A<16>" LOC = "B11";
NET "R2_A<15>" LOC = "C11";
NET "R2_A<14>" LOC = "B10";
NET "R2_A<13>" LOC = "D11";
NET "R2_A<12>" LOC = "C10";
NET "R2_A<11>" LOC = "A9";
NET "R2_A<10>" LOC = "E11";
NET "R2_A<9>" LOC = "C9";
NET "R2_A<8>" LOC = "D10";
NET "R2_A<7>" LOC = "A8";
NET "R2_A<6>" LOC = "B8";
NET "R2_A<5>" LOC = "E10";
NET "R2_A<4>" LOC = "C8";
NET "R2_A<3>" LOC = "D9";
NET "R2_A<2>" LOC = "B7";
NET "R2_A<1>" LOC = "A6";
NET "R2_A<0>" LOC = "C7";
NET "R2_D<35>" LOC = "C25";
NET "R2_D<34>" LOC = "D24";
NET "R2_D<33>" LOC = "B25";
NET "R2_D<32>" LOC = "E23";
NET "R2_D<31>" LOC = "A25";
NET "R2_D<30>" LOC = "D23";
NET "R2_D<29>" LOC = "B24";
NET "R2_D<28>" LOC = "E22";
NET "R2_D<27>" LOC = "C23";
NET "R2_D<26>" LOC = "A23";
NET "R2_D<25>" LOC = "D22";
NET "R2_D<24>" LOC = "E21";
NET "R2_D<23>" LOC = "B22";
NET "R2_D<22>" LOC = "D21";
NET "R2_D<21>" LOC = "C21";
NET "R2_D<20>" LOC = "B21";
NET "R2_D<19>" LOC = "E20";
NET "R2_D<18>" LOC = "D20";
NET "R2_D<17>" LOC = "C20";
NET "R2_D<16>" LOC = "B20";
NET "R2_D<15>" LOC = "E19";
NET "R2_D<14>" LOC = "D19";
NET "R2_D<13>" LOC = "C19";
NET "R2_D<12>" LOC = "A19";
NET "R2_D<11>" LOC = "D18";
NET "R2_D<10>" LOC = "E18";
NET "R2_D<9>" LOC = "E17";
NET "R2_D<8>" LOC = "B17";
NET "R2_D<7>" LOC = "B16";
NET "R2_D<6>" LOC = "D16";
```

```
NET "R2_D<5>" LOC = "E16";
NET "R2_D<4>" LOC = "C16";
NET "R2_D<3>" LOC = "A15";
NET "R2_D<2>" LOC = "C15";
NET "R2_D<1>" LOC = "D15";
NET "R2_D<0>" LOC = "E15";
NET "R2_WEL<3>" LOC = "D13";
NET "R2_WEL<2>" LOC = "C12";
NET "R2_WEL<1>" LOC = "E13";
NET "R2_WEL<0>" LOC = "A11";
NET "R2_GWEL" LOC = "C14";
NET "R2_OEL" LOC = "D12";
NET "R2_CEL" LOC = "D14";
NET "R2_ADVL" LOC = "A13";
NET "R2_C2L" LOC = "E14";
NET "R2_CKEL" LOC = "C13";
```

11.6. SSRAM Bank 3

```
NET "R3_A<16>" LOC = "AK16";
NET "R3_A<15>" LOC = "AJ16";
NET "R3_A<14>" LOC = "AL16";
NET "R3_A<13>" LOC = "AM16";
NET "R3_A<12>" LOC = "AL15";
NET "R3_A<11>" LOC = "AK15";
NET "R3_A<10>" LOC = "AJ15";
NET "R3_A<9>" LOC = "AN15";
NET "R3_A<8>" LOC = "AM14";
NET "R3_A<7>" LOC = "AK14";
NET "R3_A<6>" LOC = "AJ14";
NET "R3_A<5>" LOC = "AN13";
NET "R3_A<4>" LOC = "AM13";
NET "R3_A<3>" LOC = "AL13";
NET "R3_A<2>" LOC = "AK13";
NET "R3_A<1>" LOC = "AM12";
NET "R3_A<0>" LOC = "AL12";
NET "R3_D<35>" LOC = "AL30";
NET "R3_D<34>" LOC = "AK28";
NET "R3_D<33>" LOC = "AM31";
NET "R3_D<32>" LOC = "AJ27";
NET "R3_D<31>" LOC = "AN31";
NET "R3_D<30>" LOC = "AL29";
NET "R3_D<29>" LOC = "AK27";
NET "R3_D<28>" LOC = "AL28";
NET "R3_D<27>" LOC = "AJ26";
NET "R3_D<26>" LOC = "AM30";
NET "R3_D<25>" LOC = "AM29";
NET "R3_D<24>" LOC = "AK26";
NET "R3_D<23>" LOC = "AJ25";
NET "R3_D<22>" LOC = "AN29";
NET "R3_D<21>" LOC = "AN28";
NET "R3_D<20>" LOC = "AK25";
NET "R3_D<19>" LOC = "AL26";
NET "R3_D<18>" LOC = "AJ24";
NET "R3_D<17>" LOC = "AM27";
NET "R3_D<16>" LOC = "AM26";
NET "R3_D<15>" LOC = "AK24";
NET "R3_D<14>" LOC = "AL25";
NET "R3_D<13>" LOC = "AJ23";
NET "R3_D<12>" LOC = "AN26";
NET "R3_D<11>" LOC = "AL24";
NET "R3_D<10>" LOC = "AK23";
NET "R3_D<9>" LOC = "AJ22";
NET "R3_D<8>" LOC = "AL23";
NET "R3_D<7>" LOC = "AM24";
NET "R3_D<6>" LOC = "AK22";
```

```
NET "R3_D<5>" LOC = "AM23";
NET "R3_D<4>" LOC = "AJ21";
NET "R3_D<3>" LOC = "AN23";
NET "R3_D<2>" LOC = "AK21";
NET "R3_D<1>" LOC = "AM22";
NET "R3_D<0>" LOC = "AJ20";
NET "R3_WEL<3>" LOC = "AL19";
NET "R3_WEL<2>" LOC = "AN19";
NET "R3_WEL<1>" LOC = "AJ18";
NET "R3_WEL<0>" LOC = "AK18";
NET "R3_GWEL" LOC = "AN21";
NET "R3_OEL" LOC = "AL21";
NET "R3_CEL" LOC = "AL20";
NET "R3_ADVL" LOC = "AJ19";
NET "R3_C2L" LOC = "AM20";
NET "R3_CKEL" LOC = "AK19";
```

Revision History

Date	Revision	Nature of Change
July-1999	1.0	Initial release
January-2000	1.1	Improved section on clocks. UCF Naming revised. PCI Configuration guidelines. Added references.
January-2000	1.2	Added I/O Pinout
January-2001	1.3	Added Pn4 Pinout