



## ULTRADMA3 Series

High Performance PCI Bus Data Acquisition Boards  
with Ultra-deep on-board Memory and 64/32-bit DMA

### Models

AD8-1500DMA-16GB	Single 1.5 GSPS 8-bit A/D with 16 Gigabyte Memory
AD8-1500DMA-8GB	Single 1.5 GSPS 8-bit A/D with 8 Gigabyte Memory
DA8-1250DMA-16GB	Single 1.25 GSPS 8-bit D/A with 16 Gigabyte Memory
DA8-1250DMA-8GB	Single 1.25 GSPS 8-bit D/A with 8 Gigabyte Memory

### PRELIMINARY Product Specification

August 15, 2006

Covers AD8-1500DMA Boards with Firmware rev 10/6/05,  
DA8-1250DMA boards with Firmware rev 10/6/05  
and driver release V 2.00B

PRELIMINARY – SUBJECT TO CHANGE

Ultraview Corporation

34 Canyon View, Orinda, CA 94563

(925) 253-2960

Fax (925) 253-4894

e-mail : [support@ultraviewcorp.com](mailto:support@ultraviewcorp.com)

URL : [www.ultraviewcorp.com](http://www.ultraviewcorp.com)

copyright c 2005, 2006 Ultraview Corporation

## TABLE OF CONTENTS

<b>1.WARRANTY.....</b>	<b>4</b>
<b>2.MODEL DESCRIPTIONS.....</b>	<b>5</b>
2.1 MODEL AD8-1500DMA-16GB AND AD8-1500DMA-8GB.....	5
2.2 MODEL DA8-1250DMA-16GB AND DA8-1250DMA-8GB .....	5
<b>3.SPECIFICATIONS (PRELIMINARY).....</b>	<b>6</b>
3.1 A/D CONVERTER (MODELS AD8-1500DMA-16GB AND –8GB).....	6
3.2 D/A CONVERTER (MODELS DA8-1250DMA-16GB AND –8GB).....	7
3.3 GENERAL.....	7
3.4 PHYSICAL.....	8
<b>4.HARDWARE ARCHITECTURE .....</b>	<b>9</b>
4.1 ANALOG INPUT (AD8-1500DMA MODELS ONLY).....	9
4.2 ANALOG OUTPUT (DA8-1250DMA MODELS ONLY).....	9
4.3 EXTERNAL CLOCK INPUT.....	9
4.4 INTERNAL CLOCK.....	10
4.5 TRIGGER INPUT LINE (USE OPTIONAL).....	10
4.6 LED INDICATORS.....	11
4.6.1 ACQR LED.....	11
4.6.2 DMA and D64 (64-Bit DMA Transfer) LEDs.....	11
4.6.3 Int Clk LED.....	11
4.6.4 AD.HOT LED .....	11
4.7 LOW LEVEL SOFTWARE INTERFACE.....	12
4.8 PCI CONFIGURATION HEADER.....	12
4.9 ULTRADMA3 CONTROL REGISTER.....	13
4.9.1 Software_Run (write only).....	14
4.9.2 Buffer_Wrap (write only).....	14
4.9.3 Use_Ext_Trig (write only).....	15
4.9.4 Interrupt_Enable (write only).....	15
4.9.5 Clock_Divide_by_N (write only – for internal clock mode only).....	15
4.9.6 D/A Mode (DA8-1250DMA board only).....	16
4.9.7 OSSTB (write only) .....	16
4.9.8 OSCLK (write only).....	16
4.9.9 OSDAT (write only).....	17
4.9.10 JTAG TMS Bit (write only) .....	17
4.9.11 JTAG TCK Bit (write only).....	17
4.9.12 JTAG TDI Bit (write only).....	17
4.9.13 JTAG TDO Bit (read only).....	18
4.9.14 DMA Block Size (write only).....	18
4.9.15 Board interrupted after A/D or D/A block completion (read only status bit).....	18
4.9.16 Board Interrupting after DMA block completed (read only status bit).....	19
4.9.17 Board Stopped (read only status bit).....	19
4.9.18 DMA in progress (read only status bit).....	19
4.9.19 A/D Converter Overheating (read only status bit).....	19
4.9.20 Buffer RAM Size Bit 1, 0.....	20
4.9.21 Board Type Bits 3,2,1, 0.....	20
4.10 DMA LOW STARTING ADDRESS REGISTER AND READ/WRITE CONTROL.....	20
4.11 DMA HIGH STARTING ADDRESS REGISTER (FOR EXTENDED ADDRESSING ONLY).....	20
4.12 DMA LOCAL MEMORY BLOCK REGISTER.....	20
4.13 DATA REPRESENTATION IN HOST SYSTEM MEMORY .....	21
<b>5.HARDWARE INSTALLATION AND SETUP.....</b>	<b>22</b>
<b>6.SOFTWARE INSTALLATION AND SETUP.....</b>	<b>23</b>

6.1 SOFTWARE INSTALLATION UNDER REDHAT™ ENTERPRISE LINUX WS 4.0 (64-BIT).....	23
6.2 SOFTWARE INSTALLATION FOR WINDOWS 2000™ AND XP™ (TO BE AVAIL. TBD).....	24
6.3 SOFTWARE INSTALLATION FOR SOLARIS 9/10 (SPARC PLATFORM EDITION)™.....	24
6.4 RUNNING THE EXAMPLE PROGRAMS UNDER REDHAT™ LINUX WS 4.0 64-BIT.....	26
6.4.1 <i>acquire_data.c</i> (Acquire analog data and store it in board's 16GB or 8GB buffer), and then store the buffer's data to disk.....	26
6.4.2 <i>acquire_data2.c</i> (Acquire analog data and store it in board's 16GB or 8GB buffer), with concurrent dumping of the buffer's data to disk.....	27
6.4.3 <i>pre_post_trigger.c</i> (Acquire data continuously into board's buffer, stop acquiring after user "trigger" and store both pre and post trigger data.....	28
6.4.4 <i>digosc</i> .....	29
6.5 RUNNING THE EXAMPLE PROGRAMS UNDER SOLARIS 9/10 (SPARC PLATFORM)™.....	30
6.5.1 <i>acquire_data.c</i> (Acquire analog data and store it in board's 16GB or 8GB buffer), and then store the buffer's data to disk.....	30
6.5.2 <i>acquire_data2.c</i> (Acquire analog data and store it in board's 16GB or 8GB buffer), with concurrent dumping of the buffer's data to disk.....	31
6.5.3 <i>pre_post_trigger.c</i> (Acquire data continuously into board's buffer, stop acquiring after user "trigger" and store data pre and post trigger data.....	32
6.5.4 Two-board concurrent operation using <i>acquire_data.c</i> .....	33
6.5.5 <i>digosc</i> (digital oscilloscope).....	35
6.5.6 <i>digosc dual1250_1500</i> (digital oscilloscope for two-board AD8-1500DMA ganged installations) .....	35
<b>7.APPENDIX 1 – INSTALLING AD8-SPLIT2/4 CLOCK/TRIGGER SPLITTER.....</b>	<b>37</b>

## 1. Warranty

Ultraview Corporation hardware, software and firmware products are warranted against defects in materials and workmanship for a period of two (2) years from the date of shipment of the product. During the warranty period, Ultraview Corporation shall, at its option, either repair or replace hardware, software or firmware products which prove to be defective. This limited warranty does not cover damage caused by misuse or abuse by customer, and specifically excludes damage caused by the application of excessive voltages to the inputs and/or outputs of data acquisition boards.

While Ultraview Corporation hardware, software and firmware products are designed to function in a reliable manner, Ultraview Corporation does not warrant that the operation of the hardware, software or firmware will be uninterrupted or error free. Ultraview products are not intended for use as critical components in life support systems, aircraft, military systems or other systems whose failure to perform can reasonably be expected to cause significant injury to humans. Ultraview expressly disclaims liability for loss of profits and other consequential damages caused by the failure of any product, and recommends that customer purchase spare units for applications in which the failure of any product would cause interruption of work or loss of profits, such as industrial, shipboard or military equipment. In no way will Ultraview Corporation's liability exceed the amount paid by the customer for the product.

THIS LIMITED WARRANTY IS IN LIEU OF ALL OTHER WARRANTIES EXPRESSED OR IMPLIED. THE WARRANTIES PROVIDED HEREIN ARE BUYER'S SOLE REMEDIES. IN NO EVENT SHALL ULTRAVIEW CORPORATION BE LIABLE FOR DIRECT, SPECIAL, INDIRECT, INCIDENTAL OR CONSEQUENTIAL DAMAGES SUFFERED OR INCURRED AS A RESULT OF THE USE OF, OR INABILITY TO USE THESE PRODUCTS. THIS LIMITATION OF LIABILITY REMAINS IN FORCE EVEN IF ULTRAVIEW CORPORATION IS INFORMED OF THE POSSIBILITY OF SUCH DAMAGES.

Some states do not allow the exclusion or limitation of incidental or consequential damages, so the above limitation and exclusion may not apply to you. This warranty gives you specific legal rights, and you may also have other rights which vary from state to state.

**WARNING! To avoid overheating, the ULTRADMA3 must be installed in a well-cooled workstation or server chassis, with 64-bit slots, or alternatively in an industrial chassis PC. Installation in a PC or workstation without fans at the front end of the card cage will cause the ULTRADMA3 to overheat, and resulting damage is not covered by warranty. Evaluation boards damaged by overheating will be also billed to evaluator at full value. If after 5 minutes of operation the copper heatsink on the board feels too hot to comfortably touch, or if either of the RED LEDs at the top of the board illuminate, a system with better cooling is required.**

**WARNING! Due to the Xilinx XC2VP30 FPGA in the AD8-1500DMA and DA8-1250DMA using a volatile RAM-based configuration system, and requiring reloading of the board firmware after each system power-up, the program jtagp (in the example\_programs directory) must be run before running any user software which operates the board. This program must be run after every power-up, before first using the board to acquire any data. jtagp does not need to be re-run between board operations, however, so long as the system is not powered down.**

## 2. Model Descriptions

The ULTRADMA3 series of data acquisition and control boards are complete high-speed A/D systems on a single full-size PCI bus card. Due to the height of large on-board 16GB or 8GB acquisition memory DDR DIMMs, the adjacent PCI slot must be left empty, causing each ULTRADMA3 board to occupy a total of two PCI slots. Designed for low jitter operation for military, scientific, medical and industrial applications these boards function in 64-bit PCI bus systems using supplied drivers for Linux (RedHat Enterprise WS 4.0 64-bit) or Solaris 9 or 10 Sparc Platform Edition™. Two, three or four ULTRADMA3 boards may be ganged together to run concurrently and start in sync when triggered a common TTL trigger, thereby acquiring multiple channels simultaneously, using the AD8-SPLIT2 (2 boards) or AD8-SPLIT4 (up to 4 boards) clock/trigger splitters.

### 2.1 MODEL AD8-1500DMA-16GB and AD8-1500DMA-8GB

Model AD8-1500DMA-16GB and –8GB contain a 1.5 gigasample/second 8 bit A/D, 16 GB or 8 GB, respectively, of on-board low-power DDR DRAM memory and the ability to transfer data directly into the computer system's memory at up to 320 MB/s on 64-bit PCI systems. **A/D Sampling may either be controlled by an external clock input between 20 MHz and 1500 MHz or the on-board internal clock that is software selectable to allow sampling at 640MSPS, 320MSPS, 160MSPS, 106.66MSPS, 80MSPS, 64MSPS, 53.33MSPS, 45.714MSPS, 40MSPS, 35.55MSPS, 32MSPS, ... , 20MSPS, ..., 10MSPS or 5MSPS.** Multiple boards may be configured to acquire either concurrently, for more simultaneous acquisition channels, or sequentially, for longer record length.

### 2.2 MODEL DA8-1250DMA-16GB and DA8-1250DMA-8GB

Model DA8-1250DMA-16GB and –8GB contain a 1.25 gigasample/second 8 bit D/A converter, 16 GB or 8 GB, respectively, of on-board DDR SDRAM memory and the ability to transfer data directly from the computer system's memory at up to 320 MB/s on 64-bit PCI systems. **Sampling on D/A operation may only be controlled by an external clock input between 1000 MHz and 1250MHz.** Multiple boards may be configured to output analog streams concurrently, for more simultaneous analog output channels. The board's RAMs are first filled with data by the host computer and then, in response to an external trigger, the two boards begin outputting data via their D/A converters. When two or more DA8-1250DMA boards are run concurrently, up to an 8 sample skew may exist between boards (this skew is not present on AD8-1500DMA A/D models, just on DA8-1250DMA models).

### 3.Specifications (PRELIMINARY)

#### 3.1 A/D Converter (models AD8-1500DMA-16GB and –8GB)

Number of Input Channels:	1
A/D converter resolution:	8 Bits
Signal-to-noise Ratio	42 dB
Analog input range:	-350mV to +350mV - DO NOT EXCEED 800mV!
Analog Input impedance:	50 ohms    4pF
Analog Input bandwidth	DC to 1.6GHz minimum (-3dB BW)
Input connectors:	
Analog Input:	SMA connector
Clock and Trigger Inputs:	Two SMA connectors, one for 0dBm clock, one for positive edge for trigger (ECL or TTL)
Sampling rate into on-board RAM:	
Maximum:	1500 MSPS
Minimum:	20 MSPS (operation as low as 2MHz typically achievable but not guaranteed)
Clock Input AC Voltage Range	
Minimum:	0.2V Peak-to-Peak
Maximum:	1.0V Peak-to-Peak
Clock Input Impedance:	50 ohms in series with 0.01uF
Optional External Trigger Input Signal (AC coupled):	
Signal Type (Selectable via Jumper JP1):	
ECL/PECL (AC coupled):	Positive going pulse, rise time must be < 2ns Minimum Pulse Signal voltage: 0.7V p-p Maximum Pulse Signal voltage: 1.2V p-p Max. DC voltage range: -5V to +5V Input Impedance: 50 ohms in series with 0.01uF
LVTTTL/CMOS	Positive going pulse, rise time must be <4ns Vil: 0V min, 0.4V max, Vih: 2.4V min, 3.3V max.
Minimum trigger pulse width:	200 nanoseconds
Maximum Continuous DMA Transfer Rate into host system RAM: Typical systems with 64-bit PCI bus (eg. Sun Ultra 80, E420, E450, SunBlade1000/2000, Dell Precision 670 Workstation):	
Board in 66MHz 64-bit PCI Slot:	310 Million bytes/sec
Board in 33MHz 64-bit PCI Slot:	170 Million bytes/sec
On-board Memory Depth:	16 Gigasamples for AD8-1500DMA-16GB Or 8 Gigasamples for AD8-1500DMA-8GB

### 3.2 D/A Converter (models DA8-1250DMA-16GB and -8GB)

Number of Output Channels:	1
D/A converter resolution:	8 Bits
Full scale analog output range:	-1000mV to +1000mV into 50 ohms
Analog Output impedance:	50 ohms +/- 10% from DC to 100MHz
Analog Output bandwidth	DC to 600 MHz typical (-3dB BW)
Input/output connectors:	
Analog Output:	SMA connector
Clock and Trigger Inputs:	Two SMA connectors, one for 0dBm clock, one for positive edge for trigger (ECL or TTL)
Output sampling rate from on-board RAM:	
Range1:	800 to 1250 MSPS
Range2:	300 to 505 MSPS
Clock Input AC Voltage Range	
Minimum:	0.2V Peak-to-Peak
Maximum:	1.0V Peak-to-Peak
Clock Input Impedance:	50 ohms in series with 0.01uF
Optional External Trigger Input Signal (AC coupled):	
Signal Type (Selectable via Jumper JP1):	
ECL/PECL (AC coupled):	Positive going pulse, rise time must be < 2ns Minimum Pulse Signal voltage: 0.7V p-p Maximum Pulse Signal voltage: 1.2V p-p Max. DC voltage range: -5V to +5V Input Impedance: 50 ohms in series with 0.01uF
LVTTTL/CMOS	Positive going pulse, rise time must be <4ns Vil: 0V min, 0.4V max, Vih: 2.4V min, 3.3V max.
Minimum Trigger pulse width:	200 nanoseconds
Maximum Continuous DMA Transfer Rate from host system RAM: Typical systems with 64-bit PCI bus (eg. Sun Ultra 80, E420, E450, SunBlade1000/2000, Dell Precision 670):	
Board in 66MHz 64-bit PCI Slot:	250 Million bytes/sec
Board in 33MHz 64-bit PCI Slot:	170 Million bytes/sec
On-board Memory Depth:	16 Gigasamples for DA8-1250DMA-16GB Or 8 Gigasamples for DA8-1250DMA-8GB

### 3.3 General

Operating Temperature Range:	0 to +50 Degrees Celsius
Storage Temperature Range:	-25 to +85 Degrees Celsius
Power Requirements (board occupies 2 slots):	+5V +/-5% at 1.5A Max (AD8-1500DMA models)

(0.5A Max for DA8-1250DMA models)  
 +3.3V +/-5% at 12A Max (AD8-1500DMA-16GB  
 or DA8-1250DMA-16GB models)  
 (9A Max for AD8-1500DMA-8GB or DA8-  
 1250DMA-8GB)

Downloading time for Xilinx XC2VP30 firmware  
 (Required before board is first used after each power-up)      15 sec. typ., system dependent

### 3.4 Physical

All ULTRADMA3 boards are full-size 64-bit PCI bus boards which will operate in 64-bit PCI slots with 3.3V signaling environment. These boards must **not** be installed in PCI slots having 5V signaling environment. They may be installed in 66MHz slots or 33MHz slots with 3.3V signaling environment. **Due to the excess height of the SDRAM DIMMS on the right side of the board, and the power dissipation of the ULTRADMA3 exceeding the maximum allowance for a single slot, it will not be possible to install another PCI board in the slot adjacent to the component side of the ULTRADMA3 board, and therefore each ULTRADMA3 board occupies the space of two slots, rather than a single slot.**

**If installing two ULTRADMA3 boards in a single system, both should be installed in a 66 MHz slots, wherever possible, as most 66MHz slots are separate PCI buses in the system, allowing higher total DMA throughput from the two ULTRADMA3 boards.** The figure below shows the locations of the analog input, clock, and trigger SMA input connectors, and LED indicators.

To avoid overheating, all ULTRADMA3 boards **must be installed either in well-cooled workstation or server chassis with 64-bit slots**. Installation in a standard workstation chassis with only 32-bit slots, is not feasible, as it will not allow sufficient power to reach the board.

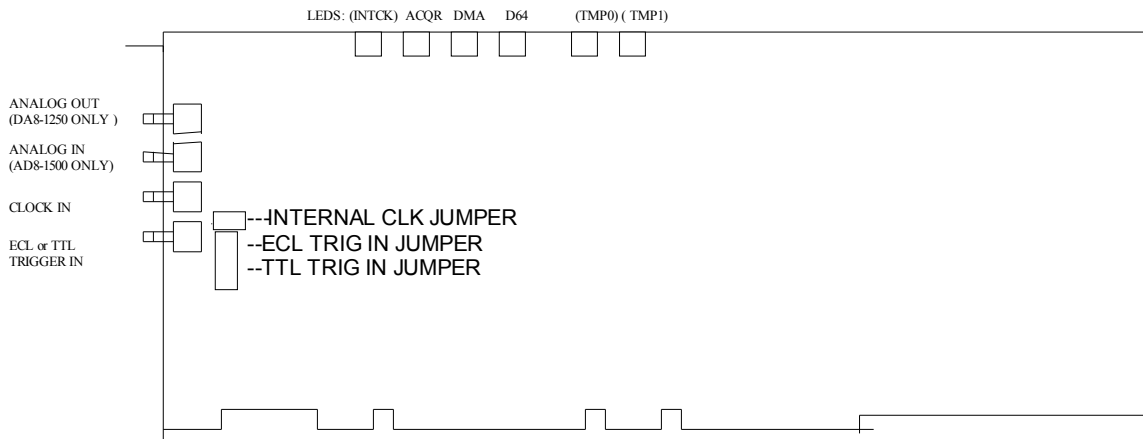


Figure 1. Board layout for AD8-1500DMA and DA8-1250DMA. Items in parenthesis are present only in applicable models.

## 4. Hardware Architecture

ULTRADMA3 series boards are comprised of a digital section and an analog section. The analog section contains a high speed 8-bit A/D converter (AD8-1500DMA models), an optional internal clock, and input-conditioning circuitry for an externally-supplied sampling clock. The output of the A/D converter(s) consists of two eight-bit streams of interleaved data, each at up to 750 MWPS.

The digital section of the AD8-1500DMA includes an ECL data double-buffer, which fans the 16-bit 750 MWPS data from the A/D converter into two interleaved 16-bit ECL data streams, each at up to 375 MWPS, that are fed into four sets of input registers inside a high speed FPGA (Field Programmable Gate Array), which together output bursts of eight 128-bit wide LVTTTL data vectors at up to 200 MWPS that are fed into eight 2GB DDR DIMM modules capable of storing up to 16 gigabytes of A/D input data (eight 1GB DDR DIMMs are instead used for AD8-1500DMA-8GB, for an 8GB total storage capacity). Between these forward bursts of A/D data, the above-mentioned FPGA provides a 64-bit burst of buffered DRAM data to a second FPGA which implements the bus interface and the fast DMA data transfer engine and PCI master interface.

The digital section of DA8-1250DMA models consists of the same FPGA as in the AD8-1500DMA models, plus eight ECL 8:1 serializing registers. These registers convert a 64-bit data stream at up to 156.25 MWPS into an 8-bit data stream at up to 1250 MWPS, which is fed to the D/A converter IC.

**As the Xilinx XC2VP30 FPGA in the AD8-1500DMA and DA8-1250DMA use a volatile RAM-based configuration system, and require reloading of firmware after each system power-up, the program jtagp must be run before running any software which operates the board. This program must be run once after every power-up, before first using the board. jtagp does not need to be re-run between board operations, unless the system is powered down.**

### 4.1 Analog Input (AD8-1500DMA models only)

The SMA analog input connector on the model AD8-1500DMA can accept single-ended analog data with a voltage range from -350 to +350 millivolts into 50Ω. **Under no circumstances should the signal supplied to the analog input of the AD8-1500DMA ever exceed +/-800 mV, or damage may occur that is not covered by the warranty.**

### 4.2 Analog Output (DA8-1250DMA models only)

The SMA analog output connector on DA8-1250DMA models can output single-ended analog data with a voltage range from -1000 to +1000 millivolts into 50Ω. Under no circumstances may any external signal be applied to this output terminal for the board.

### 4.3 External Clock Input

The Clock Input must be fed with an RF signal having a peak-to-peak voltage of between 300 mV and 900 mV, and a frequency of between 20 and 1500 MHz (1000 to 1250 MHz for DA8-1250 models). It must not have a DC voltage component outside the range of +/-5V. This clock voltage must be continuously supplied, and must be present at least 100 milliseconds before sampling is started. The impedance of the clock input is 50 ohms, AC coupled by an internal 0.01μF capacitor. This external clock is useful if acquisition is to be synchronized to an external source, or if multiple boards are ganged together to sample multiple channels concurrently or

(using external signal splitters for clock, trigger and data inputs) to sample a single channel for a longer period of time.

When the external clock input is used, jumper JP2 must be **removed**, and the internal clock bit in the control register must be set to 0.

## 4.4 Internal Clock

For applications in which an external clock is not available the model AD8-1500DMA can use its on-board internal clock. The DA8-1250DMA must only use an external clock. If the internal clock is desired instead of the external clock, **the software must select internal clock mode, jumper JP2 must be installed, and the external clock input must be left unconnected. Also, the internal clock bit in the control register must be set to 1.** The internal clock on the board will sample at the sampling rate specified by the driver (640MSPS, 320MSPS, 160MSPS, 106.666MSPS, 80MSPS, 64MSPS, 40MSPS .... 1.25MSPS).

**Note: EXTREME CARE MUST BE TAKEN TO ENSURE THAT NO VOLTAGES GREATER THAN +/-2V ARE EVER CONNECTED, EVEN MOMENTARLY, TO ANY ANALOG INPUT.**

**D.C. INPUT VOLTAGES EXCEEDING +/- 2 VOLTS ON THE ANALOG INPUTS WILL CAUSE OVERHEATING DAMAGE TO THE A/D CONVERTER IF PRESENT FOR OVER 10 SECONDS.**

## 4.5 Trigger Input Line (use optional)

The **Trigger** input signal on CN4, which may be jumpered (via jumper JP1) for either ECL/PECL or TTL voltage, permits acquisition to be started by an external devices, if the USE\_EXT\_TRIG bit in the control register is set to 1. In this case, upon the application of a rising edge on the **Trigger** signal, the ULTRADMA3 will begin acquiring A/D samples (See also **Software\_Run** bit in ULTRADMA3 Control Register). This external trigger is useful if acquisition is to be synchronized to an external event, or if multiple boards are to be ganged together to sample multiple channels concurrently or, in conjunction with an external signal splitter, to sample a single channel for a longer time period. An Ultraview AD8-SPLIT2 clock/trigger splitter boardlet may optionally be used to convert a standard TTL trigger input signal into the concurrent ECL triggers required by multiple AD8-1500DMA boards. If using an AD8-SPLIT2 clock/trigger splitter, the board must be jumpered for ECL triggering by installing a shunt on the lower position on JP1.

If a shunt is installed on the **upper** position of JP1, trigger input jack CN4 must be driven by a fast rising **LVTTTL** (0V min, 3.3V max) signal. If a shunt is installed on the **lower** position of JP1, trigger input jack CN4 must instead be driven by a fast-rising **ECL or PECL** signal, such as that from a MC100EL11 driver (with pulldown resistor at source), or equivalent. To ensure reliable triggering, the rise time of the trigger pulse must be under 1ns, and preferably under 500ps. If minimal sample uncertainty is desired using an external clock, the rising edge of trigger must not occur between 300ps before the rising edge of the external clock and 200ps after the rising edge of the external clock. **The minimum pulse width of the Trigger Input is 200 nanoseconds. This means that once the rising edge of trigger signal has been generated, the signal must not be brought low again for this time, or the board may experience a glitch on the acquired data.** After this minimum width time, the trigger signal may be brought low any time.

If it is desired that sampling start immediately by the driver, without the need for an external trigger input, then the USE\_EXT\_TRIG bit in the control register must be set to 0 (default case).

## 4.6 LED Indicators

The board status LEDs on the board, described below, are useful during system integration.

### 4.6.1 ACQR LED

The ACQR LED is used to indicate an that actual A/D sampling is occurring. For configurations which use an external trigger, this LED can be used to indicate a successful trigger.

### 4.6.2 DMA and D64 (64-Bit DMA Transfer) LEDs

The DMA LED is illuminated when the ULTRADMA3 board is currently performing 32 or 64-bit DMA transfers. The D64 LED will be additionally illuminated during 64-bit transfers.

### 4.6.3 Int Clk LED

The INT CLK LED is illuminated when the board is configured to run from its internal clock, rather than an externally supplied clock. Note that to invoke internal clock, the internal clock bit must be programmed in the control register, and **additionally jumper JP2 must be installed. Also, when using internal clock, no cable may be connected to external clock connector CN3.**

### 4.6.4 AD.HOT LED

The red AD.HOT LED is illuminated brightly when the MAX108 A/D converter is overheating. If this occurs, the board needs to be installed in a system with increased airflow, or an added fan must be fastened in the system, so as to blow air directly over the heatsinked A/D converter IC. This LED may be ignored if it is glowing only dimly, **but if it is at comparable brightness to the DMA or D64 LEDs, then the board is not receiving sufficient airflow, and must be installed in a system with better cooling, or an additional fan must be installed. Damage due to overheating is not covered by the warranty.**

## 4.7 LOW LEVEL SOFTWARE INTERFACE

The ULTRADMA3 board is easy to communicate with. In most cases, this section may be skipped, as the drivers supplied with the board automatically handle all communication with the board registers. **The best way to develop your own custom software is simply to modify the appropriate sample programs included with the board, and then recompile.** However, the following section gives an overview of how the driver calls control the board.

The software interface consists of a PCI type-00 Configuration Header and four board-specific registers - a Control Register and LOW DMA Address Register. Additionally, there is DMA High Start Address Register, only for use in applications in which dual address cycles are required. Each of these groups of registers is outlined in the following sections. Finally, ULTRADMA3 boards have a local memory block register, which specifies the starting block in the board's 16GB or 8GB buffer from which the data is to be transferred (to/from the PCI bus) via the DMA engine.

Accesses to CONTROL, LOCAL MEMORY BLOCK, BOARD NUMBER / HIGHEST BOARD and DMA START ADDRESS registers **must** be made as 32-bit transfers.

## 4.8 PCI Configuration Header

ULTRADMA3 series boards support a PCI Configuration Header, whose map is shown below..

Double Word Address	byte 3	byte 2	byte 1	byte 0
00 H	Device ID		Vendor ID	
04 H	Status		Command	
08 H	Class Code			Revision ID
0C H		Header Type	Latency Timer	
10 H	Base Address for ULTRADMA3 control register memory			
14 H				
18 H				
1C H				
20 H				
24 H				
28 H				
2C H				
30 H				
34 H				
38 H				
3C H	Max Lat (=01 H)	Min Gnt (=01 H)	Interrupt Pin	Interrupt Line

Double Word Address	byte 3	byte 2	byte 1	byte 0
80 H	ULTRADMA3 Control Register			
84 H	DMA High Start Address Register (for dual address cycles only)			
88 H	---	Overlap Value	Board Number	Highest Board
8C H	DMA Low Start Address Register			WRT
90 H	DMA Local Memory Block Register			

The ULTRADMA3 Control Register is mapped at configuration space address 80H, as well as in **memory space at Base Address + 1FFFFFC**. The DMA Low Start Address register is mapped

at configuration space address 8CH, and **in memory space at Base Address + 1FFFFFF4**. Accessing either place will read or write these register. The optional DMA High Start Address register (which, if used must be written BEFORE the DMA Low Start Address register) is at configuration space address 84H and also in **memory space at Base Address + 1FFFFFF0**. The DMA Local Memory Block Register is mapped at configuration space address 90H, and also in **memory space at Base Address + 1FFFEC**. The Board Number / Highest Board register is mapped at configuration space address 88H, and also in **memory at Base Address + 1FFFFFF8**.

## 4.9 ULTRADMA3 Control Register

The ULTRADMA3 Control Register is used to configure the board and to start and stop the data acquisition process. The table below shows the usage of the ULTRADMA3 Control Register, and these bits' functions are outlined in the sections which follow. The first table shows the function of the Control Register during a **write**. During a **read, these bits will not be read back, but instead, 12 of the bit positions may be read**, as shown in the second table, below. The control register is never directly written or read by user programs, but is modified by calls to the driver, which are each summarized in the discussion of the respective bit.

Bit	Function
31	DMA Blocksize 3
30	DMA Blocksize 2
29	DMA Blocksize 1
28	DMA Blocksize 0
27	Software_Run
26	Buffer_Wrap
25	Use_Ext_Trig
24	/Interrupt_Enable
21	Clock_Divide_by_N
20	D/A mode (DA8-1250 only)
19	Internal Clock mode
18	OSSTB
17	OSCLK
16	OSDAT
15	(not used)
14	JTAG TMS Bit
13	JTAG TCK Bit
12	JTAG TDI Bit
11-0	(reserved for future use)

Function of Control Register bits during **write**.

Bit	Function
31	(reserved for future use)
30	Board Interrupting (DMA completion)
29	A/D or D/A block completion
28	Board Stopped
27	DMA in progress
26	A/D Overheated
25	Buffer RAM Size Bit 1
24	Buffer RAM Size Bit 0
23-13	(reserved for future use)
12	JTAG TDO Bit
11	Board type bit 3
10	Board type bit 2
9	Board type bit 1
8	Board type bit 0
7-0	(reserved for future use)

Function of Control Register bits during read.

#### 4.9.1 Software\_Run (write only)

**Software\_Run** is used to start and stop data acquisition. If no connection is made to the TRIGGER inputs, and the **Use\_Ext\_Trig** bit (see below) is set to 0, data acquisition begins when **Software\_Run** is set to 1, and ends when it is set to 0. If the **Use\_Ext\_Trig** bit is set to 1, a rising-edge TRIGGER signal must be used, **Software\_Run** must be set to 1 before TRIGGER can start data acquisition. Acquisition will always stop when **Software\_Run** is set to 0.

The **Software\_Run** bit is automatically set to 1, turning on the board, using the **start\_ultrad\_io()** driver call under Windows 2000/XP or the **uvdma\_set\_go()** call under the Solaris (UNIX) OS.

The **Software\_Run** bit is automatically set to 0, stopping the board, using the **end\_ultrad\_io()** driver call under Windows 2000/XP or the **uvpci\_stop()** (or indirectly by the **uvdma\_stop\_at\_n\_blocks()**) calls under the Solaris operating system.

#### 4.9.2 Buffer\_Wrap (write only)

**Buffer\_Wrap** is used to specify if the RAM buffer will be filled with A/D conversion samples a single time (storage stops when memory is filled) or if it will be treated as a cyclic buffer and filled continuously, wrapping around to the start of the buffer after the end is reached.

When set to 0, the **Buffer\_Wrap** control bit will cause data acquisition to end when the 16GB or 8GB RAM buffer has been filled once completely.

When **Buffer\_Wrap** is set to 1, A/D data will be stored to RAM continuously. When the buffer is filled, the next A/D sample will be stored in the first RAM data location and the buffer will be overwritten with new data.

The **Buffer\_Wrap** bit is automatically set to 1, enabling the wraparound mode, using the **setup\_ultrad\_io(...,ULTRAD\_FILE\_IO)** driver call under Windows 2000 or the **uvdma\_set\_wrap(board\_fd)** call under the Solaris operating system.

The **Buffer\_Wrap** bit is automatically set to 0, disabling the wraparound mode, using the

**setup\_ultrad\_io(..., ULTRAD\_INTERACTIVE\_IO)** driver call under Windows 2000 or the **uvdma\_unset\_wrap(board\_fd )** call under the Solaris operating system.

### 4.9.3 Use\_Ext\_Trig (write only)

The **Use\_Ext\_Trig** bit, when programmed to 1, causes the ULTRADMA3 board to hold off on starting data acquisition after the **Software\_Run** bit is set, until a rising edge is fed into external trigger input connector CN4. If a shunt is installed on the **upper** position of JP1, trigger input jack CN4 must be driven by a fast rising **LVTTL** (0V min, 3.3V max) signal. If a shunt is installed on the **lower** position of JP1, CN4 must instead be driven by an **ECL or PECL** signal, such as that from a MC100EL11 driver (with pulldown resistor at source), or equivalent. If the **Use\_Ext\_Trig** bit is set to zero, the Trigger input is ignored, and does not need to be connected to a trigger source, and data acquisition will start immediately after the **Software\_Run** bit is set to 1.

The **Use\_Ext\_Trig** bit is set to zero by default, enabling immediate starting of acquisition without waiting for a trigger or, by using the **set\_ultrad\_board\_register(ultrad\_board\_handle, ULTRAD\_USE\_EXT\_TRIG,FALSE)** driver call under Windows 2000/XP, or the **uvdma\_unset\_use\_ext\_trig(board\_fd)** call under the Solaris OS.

The **Use\_Ext\_Trig** bit is set to one, preventing acquisition until a trigger edge is seen, by using the **set\_ultrad\_board\_register(ultrad\_board\_handle, ULTRAD\_USE\_EXT\_TRIG,TRUE)** driver call under Windows 2000/XP or the **uvdma\_set\_use\_ext\_trig(board\_fd)** call under Solaris.

### 4.9.4 Interrupt\_Enable (write only)

The **Interrupt\_Enable** bit is used to specify whether the ULTRADMA3 board will issue A/D interrupts as it fills its on-board RAM. The ULTRADMA3 always uses PCibus interrupt line INTA#. Once acquisition begins and A/D samples are written to the RAM buffer, the ULTRADMA3 can issue an interrupt as each block of the RAM buffer is filled. When **Interrupt\_Enable** is set to 0, interrupts will be generated. When set to 1, interrupts will not be generated.

The **Interrupt\_Enable** bit is automatically set to 0, enabling interrupts, any time the board is started under Windows 2000/XP or by using **uvdma\_set\_int(board\_fd)** under the Solaris OS. The **Interrupt\_Enable** bit is automatically set to 1, disabling interrupts, when the board is stopped under Windows 2000/XP or by using the **uvdma\_unset\_int(board\_fd )** call under the Solaris OS.

### 4.9.5 Clock\_Divide\_by\_N (write only – for internal clock mode only)

**Clock\_Divide\_by\_N**, when set to 1, and when internal clock mode is enabled, causes the board to sample at a rate that is 1/N of its using its 640MHz internal clock. When set to 0 (default, divide-by-1 mode), conversions occur at the actual 640MHz internal clock frequency and not 1/N of this frequency. The value N is set serially using the serial bits OSDAT and OSCLK, described below, using the **uvdma\_set\_serial(... )** command.

The **Clock\_Divide\_by\_N** bit is automatically set to 0, enabling divide-by-1 mode, using the **setup\_ultrad\_io(INTERNAL\_CLOCK,...)** driver call under Windows 2000/XP or the **uvdma\_unset\_double\_speed(board\_fd)** call under the Solaris operating system.

The **Clock\_Divide\_by\_N** bit is set to 1, selecting divide-by-N clock operation, using the **setup\_ultrad\_io(ULTRAD\_EXTERNAL\_CLOCK,...)** driver call under Windows 2000/XP or the **uvdma\_set\_double\_speed(board\_fd )** call under the Solaris operating system.

The clock period may be either be set to the same period as the 640MHz internal clock reference, or this clock can be divided by N, where N is an even number (2,4,6,8... 512). The routine "setperiod(board\_fd, N)", used in various user programs in the software release, can be used to set N. If N is set to 1, then the board samples at the same rate as the external clock or the 640MHz internal clock. If N = 2, then the board samples at half this rate. If N = 4, then the board samples at 1/4 this rate, if N = 6, the board samples at 1/6 the rate, etc., all the way to sampling at 1/512 of this rate if period is specified as 512. The example program "acquire\_data2.c" is a good example, which illustrates the setting of the divide by N function. For example, the command "uvdma\_set\_period(board\_fd, 4)" in acquire\_data2.c causes the 640 MHz internal clock to be divided by 4, thereby sampling both channels simultaneously at 160 MSPS.

If N is specified as 1, this routine chooses the internal divide-by-1 mode, in which the input clock is directly used as the A/D clock. However, if period >= 2, the board is switched to the "divide-clock-by-N" mode (this is done by calling `uvdma_set_double_speed(brd_fd)`, which sets bit 21 of the control register to 1). In this case the bit-banger routine sends a serial stream to the control register bits, specifying sample clock divide ratio, N, using bits RSCLK and RSDAT in the control register (see OSCLK and OSDAT below) to send data to clock divider in these boards.

## 4.9.6 D/A Mode (DA8-1250DMA board only)

**D/A Mode**, when set to 1, specifies that the board will operate as a D/A rather than as an A/D. This bit is always set to 0 when operating an AD8-1500DMA and is always set to 1 when operating a DA8-1250DMA board. If a dual-mode board is developed at a future date, this bit will specify whether the board is being told to digitize incoming analog data, or output analog data.

The D/A Mode bit is set to 1, specifying D/A operation, using the `setup_ultrad_io (ULTRAD_WRITING_TO_BOARD..)` driver call under Windows 2000/XP or the `uvdma_set_adda_mode(board_fd)` call under the Solaris operating system.

The D/A Mode bit is set to 0, specifying A/D operation, using the `setup_ultrad_io (ULTRAD_READING_FROM_BOARD..)` driver call under Windows 2000/XP or the `uvdma_unset_adda_mode(board_fd)` call under the Solaris operating system.

## 4.9.7 OSSTB (write only)

The OSSTB, OSDAT, and OSCLK bits are intended to be used together to serially program the internal clock sampling rate, via the software driver.

The OSSTB bit is automatically set to 1, driving the OSSTB output to a TTL logic 1, using the `set_ultrad_board_register(ultrad_board_handle,ULTRAD_RSSTB, TRUE)` driver call under Windows 2000/XP or the `uvdma_set_serial(...)` call under the Solaris operating system.

The OSSTB bit is automatically set to 0, driving the OSSTB output to a TTL logic 0, using the `set_ultrad_board_register(ultrad_board_handle, ULTRAD_RSSTB,FALSE)` driver call under Windows 2000/XP or the `uvdma_set_serial( ...)` call under the Solaris operating system.

## 4.9.8 OSCLK (write only)

OSCLK is a registered control bit connected used as the clock in the two-bit port to serially program the internal clock sampling rate, via the software driver.

The OSCLK bit is automatically set to 1, driving the OSCLK output to a TTL logic 1, using the `set_ultrad_board_register(ultrad_board_handle,ULTRAD_RSCLK, TRUE)` driver call under

Windows 2000/XP or the `uvdma_set_serial(...)` call under the Solaris operating system.

The OSCLK bit is automatically set to 0, driving the OSCLK output to a TTL logic 0, using the `set_ultrad_board_register(ultrad_board_handle, ULTRAD_RSCLK, FALSE)` driver call under Windows 2000/XP or the `uvdma_set_serial(...)` call under the Solaris operating system.

## 4.9.9 OSDAT (write only)

OSDAT is a registered control bit used as the data bit in the two-bit port to serially program the internal clock sampling rate, via the software driver.

The OSDAT bit is automatically set to 1, driving the OSDAT output to a TTL logic 1, using the `set_ultrad_board_register(ultrad_board_handle, ULTRAD_RSDAT, TRUE)` driver call under Windows 2000/XP or the `uvdma_set_serial(...)` call under the Solaris operating system.

The OSDAT bit is automatically set to 0, driving the OSDAT output to a TTL logic 0, using the `set_ultrad_board_register(ultrad_board_handle, ULTRAD_RSDAT, FALSE)` driver call under Windows 2000/XP or the `uvdma_set_serial(...)` call under the Solaris operating system.

## 4.9.10 JTAG TMS Bit (write only)

The JTAG TMS, TCK, and TDI bits are intended to be used together to provide a TTL serial communication path for field programming of the firmware in the large Xilinx XC2VP30-FF1152 FPGA, which controls most of the operation of the board. Experienced Xilinx users may purchase the RCFIRMWARE-30 package and modify the firmware to perform customized signal processing operations on the data. JTAG TMS is a registered control bit connected to the TMS pin on the XC2VP30 FPGA.

The JTAG TMS bit is automatically set to 1, driving the XC2VP30's TMS signal to a TTL logic 1, using the `set_ultrad_board_register(ultrad_board_handle, ULTRAD_TMS, TRUE)` driver call under Windows 2000/XP or the `uvdma_set_jtag(...)` call under the Solaris operating system.

The JTAG TMS bit is automatically set to 0, driving the TMS output to a TTL logic 0, using the `set_ultrad_board_register(ultrad_board_handle, ULTRAD_TMS, FALSE)` driver call under Windows 2000/XP or the `uvdma_set_jtag(...)` call under the Solaris operating system.

## 4.9.11 JTAG TCK Bit (write only)

JTAG TCK is a registered control bit connected to the TCK pin on the XC2VP30 FPGA.

The JTAG TCK bit is automatically set to 1, driving the XC2VP30's TCK output to a TTL logic 1, using the `set_ultrad_board_register(ultrad_board_handle, ULTRAD_TCK, TRUE)` driver call under Windows 2000/XP or the `uvdma_set_jtag(...)` call under the Solaris operating system.

The JTAG TCK bit is automatically set to 0, driving the TCK output to a TTL logic 0, using the `set_ultrad_board_register(ultrad_board_handle, ULTRAD_TCK, FALSE)` driver call under Windows 2000/XP or the `uvdma_set_jtag(...)` call under the Solaris operating system.

## 4.9.12 JTAG TDI Bit (write only)

The JTAG TDI bit (bit 12 in the control register, during a write) is a registered control bit driving the

TDI pin on the XC2VP30 FPGA.

The JTAG TDI bit is automatically set to 1, driving the TDI pin to a TTL logic 1, using the **set\_ultrad\_board\_register(ultrad\_board\_handle,ULTRAD\_TDI, TRUE)** driver call under Windows 2000/XP or the **uvdma\_set\_jtag(...)** call under the Solaris operating system.

The JTAG TDI bit is automatically set to 0, driving the TDI pin to a TTL logic 0, using the **set\_ultrad\_board\_register(ultrad\_board\_handle, ULTRAD\_TDI,FALSE)** driver call under Windows 2000/XP or the **uvdma\_set\_jtag(...)** call under the Solaris operating system

#### 4.9.13 JTAG TDO Bit (read only)

The JTAG TDO bit (bit 12 in the control register, during a read) allows reading of the state of the TDO pin on the XC2VP30 FPGA.

The JTAG TDO bit is may be read using the **get\_ultrad\_board\_register(ultrad\_board\_handle,ULTRAD\_TDI, TRUE)** driver call under Windows 2000/XP or the **uvdma\_get\_tdo(...)** call under the Solaris operating system.

#### 4.9.14 DMA Block Size (write only)

**DMABS[3..0]** specify the length of the DMA burst that will be started upon a write to the DMA Starting Address Register. Burst lengths of 4KB to 128KB and 512KB and 2048KB are supported, to facilitate operation in host systems with Scatter/Gather requirements, and varying MMU page sizes. At the end of each such burst, the ULTRADMA3 issues an interrupt on PCI signal INTA#. The following table shows all values of **DMABS[3..0]** and the corresponding **DMA Block Size**. Note that the ADC block size (number of A/D bytes sampled per A/D interrupt) is 512KB for all DMA block sizes, except it is 2048KB when the DMA block size is set to 2048KB.

DMABS[3,2,1,0]	DMA Block Size (& ADC Block Size)
0,0,0,0	NO DMA
0,0,0,1	4KB (ADC Size 512KB)
0,0,1,0	8KB (ADC Size 512KB)
0,0,1,1	16KB (ADC Size 512KB)
0,1,0,0	32KB (ADC Size 512KB)
0,1,0,1	64KB (ADC Size 512KB)
0,1,1,0	128KB (ADC Size 512KB)
0,1,1,1	512KB (ADC Size 512KB)
1,0,0,1	2,048KB (ADC Size 2048KB)

#### 4.9.15 Board interrupted after A/D or D/A block completion (read only status bit)

This bit (bit 29), when 1, indicates that the ULTRADMA3 has issued an interrupt on PCI bus line INTA# that is pending, signifying that the ULTRADMA3 has transferred 512Kbytes of A/D or D/A data to/from its on-board RAM (or, alternatively, 2048KB of A/D data if **DMABS[3,2,1,0] = 1000**). This bit is automatically cleared by any write to the board's control register. It should be checked every time the driver's interrupt service routine is entered, to be sure that it was the ULTRADMA3 that actually issued the interrupt presently being serviced. This interrupt does not signify that

data has been transferred from the on-board RAM to host system RAM; completion of such DMA is signified by the DMA completion interrupt described directly below.

#### 4.9.16 Board Interrupting after DMA block completed (read only status bit)

This bit (bit 30), when 1, indicates that the ULTRADMA3 has issued an interrupt on PCI bus line INTA# that is pending, signifying that it has transferred to/from host RAM 4K to 2048KB (depending on the value written earlier to control register bits DMABS[3,2,1,0] ). This bit may be cleared by a write to the board's **DMA Low Starting Address Register** in which bit D1 is 1. Bit 30 must be checked each time the driver's interrupt service routine is entered, to determine if it was the ULTRADMA3 that actually issued the interrupt presently being serviced. This interrupt does not signify that data has been transferred between the ULTRADMA3's A/D or D/A converters and its local RAM; completion of such converter bursts is signified by the A/D or D/A block completion interrupt described above.

#### 4.9.17 Board Stopped (read only status bit)

This bit (bit 28), indicates whether the ULTRADMA3 has stopped acquiring data to its on-board RAM, in the case where the Wrap bit is not set, and the board has finished acquiring an entire 16GB or 8GB of data. This bit is a 1 in this case, and 0 in all other cases, including when the board is stopped due to the Run bit being set to 0. Its function is superfluous, as the driver will also know that the board has finished acquiring a board full of data by the fact that 32,768 (or 16,384 in the case of an 8GB board) A/D interrupts have been received from the board. This bit does not in any way signify that data has been transferred from the on-board RAM to host system memory. Completion of such DMA bursts is signified by the DMA completion interrupts described above. Therefore, the driver must not shut down the board merely because the Board Stopped bit is 1, but must also wait for all DMA bursts to be completed, to ensure that data has made the full journey to the host system memory.

#### 4.9.18 DMA in progress (read only status bit)

This bit (bit 27), if 1 indicates that the ULTRADMA3 is still performing a DMA burst to or from system RAM. Its function is superfluous, as the driver should know that the board has finished a DMA transfer by the fact that a DMA interrupt has been received from the board and the driver has not asked for a new DMA burst to begin. It should be used as a "sanity check" only.

#### 4.9.19 A/D Converter Overheating (read only status bit)

This bit (bit 26), if 1 indicates that, due to inadequate airflow provided by the host system, the A/D converter's internal temperature has reached a temperature which will compromise its operation or reliability. If this bit is ever seen to be high, this indicates that the ULTRADMA3 is installed in a system or slot in which there is inadequate cooling for this board. **If the system does not have large fans (4" or larger) blowing directly through the PCI board stack, then the ULTRADMA3 should be removed from the system, and installed in a larger system with better cooling, or a small fan must be added to blow air directly across the heat sink of the A/D IC.** If, in the new system, the **A/D Converter Overheated** bit still goes to 1, cooling may be improved by being sure that the filler panel for the (empty) PCI slot adjacent to the component (front) side of the ULTRADMA3 board is removed.

#### 4.9.20 Buffer RAM Size Bit 1, 0

These bits (bit 25 and 24) contain a two-bit binary code that tells the driver the size of the on-board buffer RAM, enabling drivers that written for the present 8GB and 16GB boards to be upwards compatible with possible future board versions that may have 32GB or 64GB RAM. The present 8GB board has {Bit 25, Bit24} = (0,0) and the 16GB board has {Bit 25, Bit 24} = (0,1). Future 32GB and 64GB boards would have {Bit 25, Bit24} = (1,0) and (1,1) respectively.

#### 4.9.21 Board Type Bits 3,2,1, 0

These bits (bits 11, 10, 9 and 8) contain a four bit binary code that tells the driver the size of the on-board buffer RAM, enabling drivers written for the present AD8-1250DMA, AD8-650x2DMA, and AD8/DA8-1500DMA boards to be upwards compatible with possible future board models that use the same driver. The present AD8-1250DMA and AD8-650x2DMA boards have {Bit 11 through 8} = (0,0,0,0). The present AD8-1500DMA has these bits equal to (0,0,1,0).

### 4.10 DMA Low Starting Address Register and Read/Write control

The ULTRADMA3 Low Starting Address Register is used to start a DMA burst, in which the ULTRADMA3 board directly transfers samples into host system RAM. This register is never directly written nor read by user programs, but is modified by calls to the driver. Bits 31 through 3 specify DMA address bits 31 through 3 (Bits 2 and 1 are always zero as addresses must be on doubleword boundaries). Bit 0 is also always zero for longword addresses, and is therefore available for specifying that the DMA transfer is a DMA Write (Bit 0 =1), as DMA Reads are not possible by the ULTRADMA3 boards, which contain no D/A converters. For example, to tell the board to write a 16KB burst of data starting at PCI address 0x8456e728, we would write the number 0x8456e729 (which is 0x8456e729 | 1). The 16K burst length would have had to have been specified earlier by writing 011 to bits DMABS[2..0].

When the ULTRADMA3 board has finished writing the burst of data to host RAM, it will issue an interrupt on PCI line INTA#. The fact that this interrupt was issued by the ULTRADMA3, and not another board, can be determined by reading bit 30 of the control register, as discussed above.

### 4.11 DMA High Starting Address Register (For extended addressing only)

The ULTRADMA3 High Starting Address Register specifies the upper 11 bits in optional 64-bit addressing mode. This register, if used, must be written just before the DMA Low Starting Address Register (described above) is written. Bits 31-0 specify DMA address bits 63-32, although only the first 11 bits are actually used internally, giving the board an address range of  $2^{43}$  bytes (8 TB). This register must not be written to for standard 32-bit addressing mode.

### 4.12 DMA Local Memory Block Register

The ULTRADMA3 DMA Local Memory Block Register specifies where in the board's 16GB or 8GB **local** RAM the board should access for the DMA transfer. The number in this register is the block number in the board's local memory, in 256 byte increments. For example, if the number is 0x0000000h, then the DMA will be starting at the very beginning of local memory. If the number is 0x0000010h, then the DMA will start at byte 8192 (16 x 256) of the local memory.

**The DMA Local Memory Block Register, if used, must be written before writing to the DMA Low Starting Address Register** (see above). Also, due to hardware limitations, a **software**

**delay of at least 5 microseconds must occur between the time at which the DMA Local Memory Block Register is written, and the time that the DMA Low Starting Address Register is written.** If the DMA Local Memory Block Register register is not used, then subsequently started DMA bursts will simply be done using sequential blocks of data from the local memory; the first DMA block will be data from the first 512K (or 2048K if DMABS(3..0) =1000) of local memory, the next DMA block will be data from the second 512K (or 2048K) of local memory, etc.

Regardless of the address written to the DMA Local Memory Block Register, the number of bytes transferred will be that specified by the DMABS{3,2,1 and 0} bits in the control register. It is not necessary for the address written to the DMA Local Memory Block Register to be a multiple of the block size specified by DMABS{3,2,1 and 0}. For example, a 512KB transfer may be made with data starting at local memory block 20h (byte 16384), in which it would transfer blocks 32 (20h) through block 160 (128 block in total) of the board's 8GB local memory buffer.

### 4.13 Data Representation in Host System Memory

**For model AD8-1500DMA or DA8-1250DMA, four sequential A/D values are stored in each longword. The format below is also the identical to the format in which A/D data is stored to disk (or D/A data must be stored on disk) in the example programs in all Ultraview-supplied software packages**

Address	D31..24 (Byte 0)	D2..16 (Byte 1)	D15..8 (Byte 2)	D7..0 (Byte 3)
BA+\$0000000	Sample 0	Sample 1	Sample 2	Sample 3
BA+\$0000004	Sample 4	Sample 5	Sample 6	Sample 7
BA+\$0000008	Sample 8	Sample 9	Sample 10	Sample 11
BA+\$000000C	Sample 12	Sample 13	Sample 14	Sample 15
BA+\$0000010	Sample 16	Sample 17	Sample 18	Sample 19
BA+\$0000014	Sample 20	Sample 21	Sample 22	Sample 23
BA+\$0000018	Sample 24	Sample 25	Sample 26	Sample 27
BA+\$000001C	Sample 28	Sample 29	Sample 30	Sample 31
.	.	.	.	.
End of array				

## 5. Hardware Installation and Setup

Be sure your system has at least 1GB of installed RAM. **To avoid overheating, the ULTRADMA3 must be installed in a well-cooled workstation or server chassis, with 64-bit slots, or alternatively in an industrial chassis PC.** Installation in a PC or workstation without fans at the front end of the card cage will cause the ULTRADMA3 to overheat, and resulting damage is not covered by warranty. Evaluation boards damaged by overheating will be also billed to evaluator at full value. If after 5 minutes of operation the copper heatsink on the board feels too hot to comfortably touch, a system with better cooling is required.

1. Use the shutdown command, turn off the system power, and disconnect the power cord.

**BEFORE REMOVING THE COMPUTER SYSTEM COVER OR REMOVING ANY BOARD, BE SURE THAT THE POWER TO THE COMPUTER, AS WELL AS TO ALL PERIPHERAL DEVICES IS OFF. WEAR A STATIC-DISSIPATING WRISTBAND WHICH IS GROUNDED TO THE SYSTEM CHASSIS WHILE OPENING OR WORKING ON YOUR SYSTEM.**

2. Remove any screws that attach the computer system cover and remove the cover.
3. Remove the filler bracket from the **two** 64-bit PCI bus slots the ULTRADMA3 board will occupy - including **the filler bracket from the 64-bit PCI slot in front of (on the component side of) the ULTRADMA3 board, to allow adequate air flow across the front of the ULTRADMA3 board.** If a mixture of 3.3V 66MHz and 33MHz slots are available in the system, **choose a 64-bit 66MHz slot. Do not install in a 5V signaling slot.** If installing two ULTRADMA3s in a system, install both in 66MHz slots. For details, refer your system's manual.
4. Hold the ULTRADMA3 board by the top of the metal PCI bracket and the top midpoint of the board (Do **not** hold by the DIMMS). Carefully slide the board in so its PCI bus connector mates with the PCI bus connector on the motherboard. **Do not apply any force to the DIMM modules on the ULTRADMA3 when sliding the board into the system. Do not force the board if its DIMM sockets snag on the adjacent slot card guide – instead rock the board very slightly when inserting it.** Be sure that the ULTRADMA3 is seated firmly into the motherboard PCI bus connector. Check that no other PCI boards have become unseated when the ULTRADMA3 was installed, as motherboards may flex slightly when installing PCI boards.
5. Plug coaxial I/O cables for the analog inputs and/or outputs into the appropriate SMA connectors on the ULTRADMA3's rear bracket at the rear of the system. Please refer to the diagram on page 8 of this manual. Connect the free ends of the analog input cable to the signal sources to be digitized, and connect the clock input cable to a suitable clock source.
6. We recommend that ANALOG INPUT (AD8-1500DMA models) initially be connected to a signal generator set for a 300mV peak sine wave at approximately 10MHz. The Clock input (3<sup>rd</sup> SMA connector from the top of the bracket) should be initially be connected to an RF synthesizer set for a 600 mV peak-to-peak wave at between 20 MHz and 1500 MHz (1000 and 1250MHz for DA8-1250DMA). This will allow a quick initial test of the ULTRADMA's functionality using the demonstration software supplied with the board. For DA8-1250DMA models, the ANALOG OUTPUT jack should be connected to an oscilloscope, whose vertical scale is set for 200mV/division and input impedance set to 50 ohms.
6. Replace the computer system cover, installing all screws you had removed. Reconnect the power cables to the system and peripherals.
7. Power up and reboot the system. The system will then be ready for software installation.

## 6. Software Installation and Setup

### 6.1 Software Installation under RedHat™ Enterprise Linux WS 4.0 (64-bit)

Before installing the software, be sure the board is installed in the system. **RedHat Enterprise Linux 4.0 is the preferred OS, although Fedora Core5 may be used if the driver and user programs are recompiled. While other versions of Linux may be usable, Ultraview only supports installations in which RedHat Linux WS4.0 is used.**

To avoid data overruns, interruptions in data acquisition, and hanging of user programs, **permanently turn off all power management options, screen savers, etc.** The system must always remain in full-power mode when running the data acquisition board, and must not be allowed to go into sleep mode when the board is running. **Also the system must have at least 1GB (2-4GB preferred) of installed RAM** to realistically acquire data records in the multi-hundred megabyte size, at full speed, unless a suitably fast RAID system is present.

To begin the software installation insert the diskette titled **AD8-1500DMA/AD8-1250DMA/AD8-650x2DMA DRIVER PKG WITH USER-DEMO SOFTWARE-FOR RedHat™ ENTERPRISE LINUX 4.0 64 BIT.**

Log in as **root**, copy the file **src-uvdma-1-2.x86\_64.rpm** to any directory, and type in the following two lines at the prompt (shown here as #):

```
# rpm -ivh src-uvdma-1-2.x86_64.rpm
```

The installation script will automatically create a directory **/uvdma**, and install the software in this directory. To run the example programs, go to **/uvdma/example\_programs**, and first recompile them by typing:

```
# make clean  
# make all
```

Follow the directions in the section “Running the Example Programs under Linux64”, below, for running the example programs. The example program **acquire\_data.c**, **acquire\_data2.c** and **pre\_post\_trigger.c** acquire data via the A/D converters.

To uninstall the Linux64 software release for the UVPCI board, as required prior to installing a newer version of UVPCI software, the following command is used:

```
# rpm -e uvdma-1-2
```

## 6.2 Software Installation for Windows 2000™ and XP™ (To be avail. TBD).

The software release for Windows 2000/XP will not be available until approximately 3Q 06. Before installing the software, be sure the board is installed in the system. Then, insert the diskette titled **AD8-1500 DRIVER PKG WITH USER DEMO SOFTWARE - FOR Win2000/XP™**. Create a directory for installing the ULTRADMA software, as follows:

```
C:> mkdir ultrad
```

Then, copy the entire contents of the diskette to the new directory as follows:

```
C:> xcopy a: C:\ultrad /s /e /v
```

To run the software, please refer to the document **Quick Setup for AD8-1500 Under Windows 2000/XP™**

## 6.3 Software Installation for Solaris 9/10 (Sparc Platform Edition)™

This section describes the installation and use of the current version (V2.00B) of the Solaris 9/10 software package for the AD8-1500DMA boards. Before installing the software, be sure the board is installed in the system. Versions of Solaris 8 OS prior to the 10/00 release are not supported.

To avoid data overruns, interruptions in data acquisition, and hanging of user programs, **permanently turn off all power management options, screen savers, etc.** The system must always remain in full-power mode when running data acquisition boards, and must not be allowed to go into sleep mode, or even screen saving mode when the board is running.

Prior to installing the software, be sure prior versions of Ultraview software (if any) have been uninstalled from the system, using the command `# pkgrm UVPCI` and/or `# pkgrm UVDMA`.

To begin the software installation insert the diskette titled **AD8-650x2 / AD8-1500 DRIVER PKG WITH USER-DEMO SOFTWARE-FOR SOLARIS 9,10 SPARC Disk 1 of 1 V2.00B 5/24/05**.

Log in as root, and type in the following two lines at the prompt (shown here as #):

```
# volcheck  
# pkgadd -a none -d /floppy/floppy0/uvdma
```

You will be shown a list of packages on the diskette (should be UVDMA only) and asked which you wish to install. You can just take the default (press Return) to install the package.

Next, you will be asked which directory should be the base directory for the package. Choose **/opt/uvdma**, or some other place on your system. Always choose an empty directory, or one that has not yet been created, for installation.

The pkgadd program may issue a warning about `/etc/devlink.tab`; ignore the warning, as it is just going to modify the file, not overwrite it. You may also see a question about running programs with superuser permissions – you should just answer yes to this.

Once all the files have been copied to the base directory, some installation scripts are automatically run, giving a usable binary distribution of the package.

You are asked by the post-installation script is whether you wish to recompile the package. If you

have the Gnu C compiler or the SunPro or Sun Studio9 C compiler, you can recompile. You will be asked where the compiler resides (eg. /opt/gnu/bin/ or /opt/SUNWspro/bin/). Due to a bug in the installation script, if the programs fail to recompile, continue with the installation, but later follow the instructions at the bottom of this page regarding running get\_config, first with a “no” response and then with a typed “yes” response, regarding whether you want to compile.

Just before returning to the prompt, pkgadd will warn you that you need to reboot your system, since a new driver has been added. Now it is necessary to reboot the system as shown below :

```
# /usr/bin/shutdown -y -i6 -g0
```

When the system reboots, the driver will be installed and operational. Due to the large 16GB or 8GB size of the board’s internal RAM, you will generally not need large host system memory buffers to efficiently stored data to disk. Normally 100 MB of user memory buffer is sufficient. If you will only need to user memory buffers that are smaller in size than half of the installed RAM in the system, then your driver installation is complete. If, however, you will need a larger memory buffer (such as if you want to place 3 gigabytes of data into host memory all at one time, and your system has 4 gigabytes of RAM DIMMS installed), then you will need to temporarily become superuser and **add the following line to the /etc/system** file on your system, and then reboot:

```
set max_page_get=0x7ffffff (note that this is 0x7 followed by seven f’s)
```

After the driver has been installed as described above, it is now possible to immediately run the demonstration programs. Go to the directory dig\_osc, to run an oscilloscope demo. Connect a signal generator set for a 300 millivolt peak signal at 10 Megahertz to the ANALOG INPUT + connector (the closest SMA connector to the top of the bracket). If you are running an AD8-1500DMA board and wish to use an external clock you must connect an RF clock source of between 300MHz and 1.5GHz, with a voltage level of 300mV to 900mV peak-to-peak, to the CLOCK INPUT connector (be sure to remove jumper JP2). Run the program digosc. Each time you click on the “Digitize” button on the screen, the ULTRADMA3 board should digitize and display on the screen the data from the signal generator. To change to using the internal clock, add the statement `uvdma_set_ext_clk(dig_osc_fd)` just after you open the board, in the program `dig_osc_controls_uvdma_stubs.c` (be sure jumper JP2 is installed in that case).

You may next want to go to the example\_programs directory, and run some of the examples, such as **acquire\_data**, **acquire\_data2**, or **pre\_post\_trigger** which store A/D data to disk. These examples, which can be invoked with command line arguments that specify the amount of data to store, etc, are described in the next chapter.

These example programs are clearly commented, and are a good starting point for users who are developing their own code. They can easily be modified and recompiled by suitably modifying the makefile and then typing “make”. **If you have trouble recompiling the programs (a known bug in the installation procedure), merely run “getconfig” in the /opt/uvdma directory, answer “no” when it asks if you want to recompile, and then run “getconfig” again, this time answering “yes” (you need to actually type “yes”, rather than merely hitting return in response to the “yes” default response). Then you should be able to recompile user example programs. This minor bug will be fixed in an upcoming release.**

The pamphlet “Misc Reference Manual Pages”, included with your board, is also helpful in understanding the usage of the various library calls used in setting up board parameters, starting and stopping the board, etc.

## 6.4 Running the Example Programs under RedHat™ Linux WS 4.0 64-bit

The directory /uvdma has both source and executables for the driver and various example programs, which can immediately be run to demonstrate the use of the board, and form an excellent basis for developing your own custom software. Full source and a makefile are provided for all sample user programs (in the subdirectory example\_programs), allowing for easy modification and recompilation.

**WARNING! Due to the Xilinx XC2VP30 FPGA in the AD8-1500DMA and DA8-1250DMA using a volatile RAM-based configuration system, and requiring reloading of the board firmware after each system power-up, the program jtagp must be run before running any user software which operates the board. This program must be run after every power-up, before first using the board. jtagp does not need to be re-run between board operations, unless the system is powered down.**

To run the AD8-1500 board, connect a clock to the SMA input labeled CLK, if you wish to use an external clock. The external clock, must be connected to the CLK input and must have an amplitude of between 0.2VRMS (0dBm) and 0.6VRMS, into 50 ohms. The clock is AC-coupled inside the board, so any DC voltage on the clock is harmless, as long as it is less than +/-10V. Allowable clock frequencies are from 20MHz to 1.5GHz for an AD8-1500DMA or 1000MHz to 1250MHz for a DA8-1250DMA board. After connecting the clock connect a signal source (+/-350mV peak full scale into 50 ohms) of a few MHz or so to the SMA input labeled INP. Do not exceed +/-0.8V into this input, or the board may be damaged. To quickly test the board, go the directory /uvdma/example\_programs. **Be sure that you have run jtagp, if you have just powered up the system.** Then run the program "digosc". Click the "digitize" button, and you should see a few cycles of your input waveform, each time you press the "digitize" button

The example software for running the board includes user programs "acquire\_data.c", acquire\_data2.c, pre\_post\_trigger.c and digosc\_uvdma.c.

### 6.4.1 acquire\_data.c (Acquire analog data and store it in board's 16GB or 8GB buffer), and then store the buffer's data to disk.

**NOTE: Before running acquire\_data.c (or before the first use of any other user program) the first time after each system power-up, the program jtagp MUST be run, so as to load the firmware in the Xilinx FPGA on the board. Failure to run jtagp (in the example\_programs directory) after a power-up will cause all data acquired by the board to be meaningless.**

The program acquire\_data.c acquires a selectable number of 512KB blocks of data (up to 32,768/16,384 into the board's 16GB/8GB of internal memory, after which the board will stop acquiring and store the data to a file on disk). Number of blocks to acquire, disk file name, internal vs external clock and sampling rate divider (for internal clock only) may all be specified.

The program **acquire\_data.c** can be used to acquire any number of 512K blocks of A/D data, up to a total of 16GB or 8GB, to the board's on-board RAM, followed by an automatic storage of the board's buffer to a disk file. The usage of acquire\_data.c is as follows:

```
# ./acquire_data /dev/uvdma0 3000 ad.dat
```

Here, the first AD8-1500 board is told to acquire 3000 blocks (3000 x 512KB) of A/D data, and store it to disk in a file named ad.dat.

For example, to acquire A/D data to fill the entire 8GB (16384 x 512KB) of on-board RAM on an 8GB board, and then store it to a disk file named “uvdma.dat”, type:

```
# ./acquire_data /dev/uvdma0 16384 uvdma.dat
```

The program will acquire 16384 blocks (8GB) of A/D data at the sample rate of the supplied external clock, indicate progress along the way, and then stop approximately 8 seconds later, (assuming a 1GHz clock frequency for an AD8-1500DMA), and begin storing the 8GB of data to the file it will name as ad.dat. If you wish to acquire fewer blocks, specify a smaller block count. Similarly, to acquire A/D data to fill the entire 16GB (32768 x 512KB) of on-board RAM on a 16GB board model, and store it to a disk file named “uvdma.dat”, type:

```
# ./acquire_data /dev/uvdma0 32768 ad.dat
```

By editing the appropriate lines in acquire\_data.c, it is possible to specify external clock, or if an internal clock is used, to specify a divide ratio other than the default of 1. The divide ratio is just the ratio by which the internal 640MHz is divided, to obtain the sampling rate (default is a divide ratio of 1). For example, a divide ratio of 4 causes the AD8-1500DMA board to sample at ¼ the 640MHz internal clock frequency, resulting in a sampling rate of 160MSPS. However, if an external clock is specified (**and JP2 removed**), the board will always sample at 1x this external clock frequency, regardless of divide ratio specified. **Divide ratio is only applicable to internal clock operation.**

#### **6.4.2 acquire\_data2.c (Acquire analog data and store it in board’s 16GB or 8GB buffer), with concurrent dumping of the buffer’s data to disk.**

**NOTE: Before running acquire\_data2.c the first time after each system power-up (or before the first use of any other user program), the program “jtagp” MUST be run, to load the firmware in the Xilinx FPGA on the board. Failure to run jtagp (in the example\_programs directory) after a power-up will cause all data acquired by the board to be meaningless, and the board to hang.**

The program acquire\_data2.c is a similar program to acquire\_data.c, but begins storing data from the buffer to disk as soon as data starts coming into the buffer, thereby improving throughput. Also, it allows buffer wraparound, allowing storage of larger amounts of data than will fit into the memory buffer. However, in cases where the desired file size (size of data file stored to disk) exceeds the size of the 16GB or 8GB buffer, acquisition speed is greatly limited, as the average acquisition rate must not exceed the rate at which data can be continuously stored on the disk – typically only 40 to 60MB/sec. The board’s 16GB or 8GB RAM is used as a cyclic storage buffer. As the disk storage is concurrent with the signal acquisition, and as data is being emptied to disk from one end of the buffer, while being filled from the A/D, it is possible to actually store more than the 16GB/8GB of data allowed by acquire\_data.c. For example, at A/D sampling rates slower than approximately 50 – 150 MB/sec (the disk throughput of most high-end systems), one can store almost indefinitely, limited only by the capacity of the file system size on the disk.

The program **acquire\_data2.c** can be used to acquire any number of 512K blocks of A/D data, up to a total of 16GB/8GB (even more than this at slower acquisition speeds, at which the disk can partially or fully keep up with the data input rate). The usage of acquire\_data2.c is as follows:

```
# ./acquire_data2 /dev/uvdma0 1000 ad.dat
```

Here, the first AD8-1500 board is told to acquire 1000 blocks (1000 x 512KB) of A/D data, and store it to disk in a file named ad.dat.

For example, to acquire A/D data to fill the entire 16GB (16384 x 512KB) of on-board RAM on a

16GB board, type:

```
# ./acquire_data2 /dev/uvdma0 32768 uvdma.dat
```

The program will acquire 32768 blocks (16GB) of A/D data at the sample rate of the supplied external clock, indicate progress along the way, and then stop approximately 17 seconds later, (assuming a 1GHz clock frequency for an AD8-1500DMA). Storage of the 16GB of data to a file named ad.dat begins almost immediately after starting the acquisition and acquiring the first few blocks of data. If you wish to acquire fewer blocks or more blocks than 32768, specify a smaller (or larger, if your disk can keep up) block count.

By editing the appropriate lines in acquire\_data2.c, it is possible to specify external clock or specify an internal clock with divide ratio other than the default of 1. The divide ratio is just the ratio by which the internal 640MHz is divided, to obtain the sampling rate (default is a divide ratio of 1). For example, a divide ratio of 4 causes the board to sample at  $\frac{1}{4}$  the 640MHz internal clock frequency, resulting in a sampling rate of 160MSPS. If an external clock is specified, however (**also requiring JP2 NOT be installed**), the divide ratio has no effect, and sampling always occurs at 1x the external clock frequency.

### 6.4.3 pre\_post\_trigger.c (Acquire data continuously into board's buffer, stop acquiring after user "trigger" and store both pre and post trigger data.

**NOTE: Before running pre\_post\_trigger.c for the first time after each system power-up, the program jtagp MUST be run, so as to load the firmware in the Xilinx FPGA on the board. Failure to run jtagp (in the example\_programs directory) after a power-up will cause all data acquired by the board to be meaningless.**

The program **pre\_post\_trigger.c** is a powerful program which allows the board to continuously acquire data to its large 8GB or 16GB cyclic buffer and then, upon receiving a user signal, to store only a prescribed further amount of data, thereby allowing the user to capture the N blocks acquired before the trigger as well as the M blocks after the trigger.

The program **pre\_post\_trigger.c** can be used to save a specified number of 512KB blocks of A/D data leading up to a user trigger, and then continue to acquire a specified number of blocks of data after the trigger. The data from these two adjacent time periods will appear in two separate files, which can have a collective size of up to 16GB (assuming an AD8-1500DMA-16GB model board). Usage of pre\_post\_trigger.c is as follows:

```
# ./pre_post_trigger /dev/uvdma0 1000 pre.dat 2000 post.dat
```

Here, the first AD8-1500 board is told to run continuously (storing data continuously in the board's 8GB cyclic buffer), and when the user hits the RETURN key on the keyboard, to record 2000 more blocks. The 1000 blocks recorded before the keyboard "trigger" are stored in a file named "pre.dat", and the 2000 blocks recorded after the trigger are in a file named "post.dat".

For example, to acquire A/D data to fill the entire 16GB (32768 x 512KB) of on-board RAM on an 16GB board model, and save the 12GB before the trigger and the 4GB after the trigger, type:

```
# ./pre_post_trigger /dev/uvdma0 24576 pre.dat 8192 post.dat.
```

By editing the appropriate lines in pre\_post\_trigger.c, it is possible to specify external clock or an internal clock with divide ratio other than the default of 1. The divide ratio is just the ratio by which the internal 640MHz is divided, to obtain the sampling rate (default is a divide ratio of 1). For example, a divide ratio of 4 causes the AD8-1500 board to sample at  $\frac{1}{4}$  the 640MHz internal clock frequency, resulting in a sampling rate of 160MSPS. However, if an external clock is specified

(and JP2 removed), it will always sample at 1x the external clock frequency.

#### 6.4.4 digosc

**digosc\_uvdma.c** displays, in waveform format, the A/D data acquired by an AD8-1500DMA board. To acquire, for example, 2000 blocks of data and display it, type:

```
# ./digosc -c 2000
```

The program will show a running display of its progress in acquiring data blocks and then, once the 2000 blocks (or other requested number of blocks) have been acquired to the board's memory the program will open a window ready to display the acquired data. To display the contents of any desired block type the block number into the text field and then click "Plot". The scroll bar then allows any portion of the block to be displayed; note that the number displayed on *the scroll bar* indicates the offset in the block of the first sample plotted on the screen. To display a different block, hit the up arrow (or type in the new desired block number) and click the "Plot" button. The data shown is not valid until "Plot" has been pressed. Then, you may scroll through the new block using the scroll bar.

This example program may be modified and recompiled by typing, "make -f make\_dig". As mentioned in the include readme.txt file, the GNOME software development package, Specifically the GTK+ library files, which provide the graphical interface, must be installed.

## 6.5 Running the Example Programs Under Solaris 9/10 (Sparc Platform)<sup>TM</sup>

**WARNING!** Due to the Xilinx XC2VP30 FPGA in the AD8-1500DMA and DA8-1250DMA using a volatile RAM-based configuration system, and requiring reloading of the board firmware after each system power-up, the program `jtagp` must be run the very first time before running any user software which operates the board. This program must be run after every power-up, before first using the board. `jtagp` does not need to be re-run between board operations, unless the system is powered down.

There are two directories with both source and executables for various example programs, which can immediately be run to demonstrate the use of the board, and which form an excellent basis for developing your own custom software for the board. Full source and makefiles are provided, allowing for easy modification and recompilation. To run the AD8-1500 board, connect a clock to the SMA input labeled CLK, if you wish to use an external clock. The external clock, if used, must be connect to the CLK input and must have an amplitude of between 0.2VRMS and 0.6VRMS, into 50 ohms. The clock is AC-coupled inside the board, so any DC voltage on the clock is harmless, as long as it is less than +/-10V. Allowable clock frequencies are from 20MHz to 1.5GHz for an AD8-1500DMA or 1000MHz to 1250MHz for a DA8-1250DMA board. Then connect a signal source (+/-350mV peak full scale into 50 ohms) of a few MHz or so to the SMA input labeled INP. Do not exceed +/-0.8V into this input, or the board may be damaged. To quickly test the board, go the directory `/opt/uvdma/dig_osc` and run the program "digosc". Click the "digitize" button, and you should see a few cycles of your input waveform, each time you press the "digitize" button. If you do not see a waveform, be sure that the system's environment is correctly set. The lines present in `/opt/uvpci/environment/cshrc` should be contained in your `.cshrc` file. Once these lines are copied into your `.cshrc`, try invoking a new C shell and rerunning digosc. If a `.cshrc` file does not exist in your system, you can make one by merely copying `/opt/uvdma/environment/cshrc` to `./cshrc`, as shown below:

```
# cd /opt/uvdma/environment
# cp cshrc ./cshrc
```

The example software for running the board includes user programs "acquire\_data.c", `acquire_data2.c` and `pre_post_trigger.c`.

### 6.5.1 `acquire_data.c` (Acquire analog data and store it in board's 16GB or 8GB buffer), and then store the buffer's data to disk.

**NOTE:** Before running `acquire_data.c` the first time after each system power-up, the program `jtagp` **MUST** be run, so as to load the firmware in the Xilinx FPGA on the board. Failure to run `jtagp` (in the `example_programs` directory) after a power-up will cause all data acquired by the board to be meaningless.

The program `acquire_data.c` acquires a selectable number of 512KB blocks of data (up to 16,384 into the board's 16GB/8GB of internal memory, after which the board will stop acquiring and store the data to a file on disk). Number of blocks to acquire, disk file name, internal vs external clock and sampling rate divider (for internal clock only) may all be specified.

The program `acquire_data.c` can be used to acquire any number of 512K blocks of A/D data, up to a total of 16GB or 8GB, to the board's on-board RAM, followed by an automatic storage of the board's buffer to a disk file. The usage of `acquire_data.c` is as follows:

```
# acquire_data /dev/uvdma/0 1000 ad.dat
```

Here, the first AD8-1500 or AD8-650x2DMA board is told to acquire 1000 blocks (1000 x 512KB)

of A/D data, and store it to disk in a file named ad.dat.

For example, to acquire A/D data to fill the entire 8GB (16384 x 512KB) of on-board RAM on an 8GB board, and then store it to a disk file named “ad.dat”, type:

```
# acquire_data /dev/uvdma/0 16384 ad.dat
```

The program will acquire 16384 blocks (8GB) of A/D data at the sample rate of the supplied external clock, indicate progress along the way, and then stop approximately 8 seconds later, (assuming a 1GHz clock frequency for an AD8-1500DMA), and begin storing the 8GB of data to the file it will name as ad.dat. If you wish to acquire fewer blocks, specify a smaller block count. Similarly, to acquire A/D data to fill the entire 16GB (32768 x 512KB) of on-board RAM on a 16GB board, and store it to a disk file named “ad.dat”, type:

```
# acquire_data /dev/uvdma/0 32768 ad.dat
```

By editing the appropriate lines in acquire\_data.c, it is possible to specify external clock, or if an internal clock is used, to specify a divide ratio other than the default of 1. The divide ratio is just the ratio by which the internal 640MHz is divided, to obtain the sampling rate (default is a divide ratio of 1). For example, a divide ratio of 4 causes the AD8-1500DMA board to sample at ¼ the 640MHz internal clock frequency, resulting in a sampling rate of 160MSPS. However, if an external clock is specified (and JP2 removed), the board will always sample at 1x this external clock frequency, regardless of divide ratio specified. Divide ratio is only applicable to internal clock operation.

## 6.5.2 acquire\_data2.c (Acquire analog data and store it in board’s 16GB or 8GB buffer), with concurrent dumping of the buffer’s data to disk.

**NOTE: Before running acquire\_data2.c the first time after each system power-up, the program jtagp MUST be run, so as to load the firmware in the Xilinx FPGA on the board. Failure to run jtagp (in the example\_programs directory) after a power-up will cause all data acquired by the board to be meaningless.**

The program acquire\_data2.c is a similar program to acquire\_data.c, but begins storing data from the buffer to disk as soon as data starts coming into the buffer, thereby improving throughput. Also, it allows buffer wraparound, allowing storage of larger amounts of data than will fit into the memory buffer. However, in cases where the desired file size (size of data file stored to disk) exceeds the size of the 16GB or 8GB buffer, acquisition speed is greatly limited, as the average acquisition rate must not exceed the rate at which data can be continuously stored on the disk – typically only 40 to 60MB/sec. The board’s 16GB or 8GB RAM is used as a cyclic storage buffer. As the disk storage is concurrent with the signal acquisition, and as data is being emptied to disk from one end of the buffer, while being filled from the A/D, it is possible to actually store more than the 16GB/8GB of data allowed by acquire\_data.c. For example, at A/D sampling rates slower than approximately 50 MB/sec (the disk throughput on some of the faster Suns), one can store almost indefinitely, limited only by the capacity of the file system size on the disk.

The program **acquire\_data2.c** can be used to acquire any number of 512K blocks of A/D data, up to a total of 16GB/8GB (even more than this at slower acquisition speeds, at which the disk can partially or fully keep up with the data input rate). The usage of acquire\_data2.c is as follows:

```
# acquire_data2 /dev/uvdma/0 1000 ad.dat
```

Here, the first AD8-1500 board is told to acquire 1000 blocks (1000 x 512KB) of A/D data, and store it to disk in a file named ad.dat.

For example, to acquire A/D data to fill the entire 8GB (16384 x 512KB) of on-board RAM on an 8GB board, type:

```
# acquire_data2 /dev/uvdma/0 16384 ad.dat
```

The program will acquire 16384 blocks (8GB) of A/D data at the sample rate of the supplied external clock, indicate progress along the way, and then stop approximately 8 seconds later, (assuming a 1GHz clock frequency for an AD8-1500DMA). Storage of the 8GB of data to a file named ad.dat begins almost immediately after starting the acquisition and acquiring the first few blocks of data. If you wish to acquire fewer blocks or more blocks than 16384, specify a smaller (or larger, if your disk can keep up) block count.

By editing the appropriate lines in acquire\_data2.c, it is possible to specify external clock or specify an internal clock with divide ratio other than the default of 1. The divide ratio is just the ratio by which the internal 640MHz is divided, to obtain the sampling rate (default is a divide ratio of 1). For example, a divide ratio of 4 causes the board to sample at ¼ the 640MHz internal clock frequency, resulting in a sampling rate of 160MSPS. If an external clock is specified, however (also requiring JP2 be removed), the divide ratio has no effect, and sampling always occurs at 1x the external clock frequency.

### 6.5.3 pre\_post\_trigger.c (Acquire data continuously into board's buffer, stop acquiring after user "trigger" and store data pre and post trigger data.

**NOTE: Before running pre\_post\_trigger.c for the first time after each system power-up, the program jtagp MUST be run, so as to load the firmware in the Xilinx FPGA on the board. Failure to run jtagp (in the example\_programs directory) after a power-up will cause all data acquired by the board to be meaningless.**

The program **pre\_post\_trigger.c** is a powerful program which allows the board to continuously acquire data to its large 8GB or 16GB cyclic buffer and then, upon receiving a user signal, to store only a prescribed further amount of data, thereby allowing user to capture the N blocks acquired before the trigger as well as the M blocks after the trigger.

The program **pre\_post\_trigger.c** can be used to save a specified number of 512KB blocks of A/D data leading up to a user trigger, and then continue to acquire a specified number of blocks of data after the trigger. The data from these two adjacent time periods will appear in two separate files, which can have a collective size of up to 16GB (for an AD8-1500DMA-16GB). Usage of pre\_post\_trigger.c is as follows:

```
# pre_post_trigger /dev/uvdma/0 1000 pre.dat 2000 post.dat
```

Here, the first AD8-1500 board is told to run continuously (storing data continuously in the boards 8GB cyclic buffer), and when the user hits the return key on the keyboard, to record 2000 more blocks. The 1000 blocks recorded before the keyboard "trigger" are stored in a file named "pre.dat", and the 2000 blocks recorded after the trigger are in a file named "post.dat".

For example, to acquire A/D data to fill the entire 16GB (32768 x 512KB) of on-board RAM on an 8GB board model, and save the 12GB before the trigger and the 4GB after the trigger, type:

```
# pre_post_trigger /dev/uvdma/0 24576 pre.dat 8192 post.dat.
```

By editing the appropriate lines in pre\_post\_trigger.c, it is possible to specify external clock or an internal clock with divide ratio other than the default of 1. The divide ratio is just the ratio by which the internal 640MHz is divided, to obtain the sampling rate (default is a divide ratio of 1). For example, a divide ratio of 4 causes the AD8-1500 board to sample at ¼ the 640MHz internal clock

frequency, resulting in a sampling rate of 160MSPS. However, if an external clock is specified (and JP2 removed), it will always sample at 1x the external clock frequency.

## 6.5.4 Two-board concurrent operation using `acquire_data.c`

Two channels may be simultaneously acquired in two-board AD8-1500DMA installations, in which a common clock and trigger are sent to two boards in the same system. If these boards are to acquire in lock-step, a common trigger and clock must be used, and the user program must be recompiled to wait for this external trigger, (eg. by uncommenting the “`uvdma_set_use_ext_trig (board_fd);`” statement in `acquire_data.c`). Then each board will wait until its trigger goes high before acquiring data, and the two boards can be set to acquire time-aligned data, by doing the following:

- 1) A common trigger is connected to both boards, using a resistive (DC) splitter that has better than 30ps delay matching, or by using an Ultraview AD8-SPLIT2/4. The trigger must be a positive going edge with amplitude between 1 and 2 Volts, and rise time under 2ns. **The trigger signal, once it occurs, must remain high for at least 200ns.** If the trigger amplitude or duration is incorrect, or if there is excessive skew between the clock or trigger entering the two boards, the two waveforms displayed will be misaligned.
- 2) A common clock is connected to both boards, using a splitter that has better than 30ps delay matching, or by using an Ultraview AD8-SPLIT2/4. The clock input to the splitter must have an amplitude between 500mV and 1.5 Volts peak-to-peak into 50 ohms. The trigger in (1) above must be synchronous with this clock, to prevent a possible board-to-board skew of up to 16 clock periods.
- 3) Input signals (+/-350mV peak for full scale reading) for the two channels to be acquired should be connected to the “+” inputs of each of the two boards. The “-” inputs may be left unconnected for AD8-1500DMA boards.
- 4) The trigger signal must initially be low before acquisition is started.
- 5) In one window, type “`acquire_data /dev/uvdma/0 16384 ad0.dat`”. In a second window, type “`acquire_data /dev/uvdma/1 ad1.dat`” Note that the “16384” in this example will cause each board to acquire an entire 8GB (16384 x 512KB). Smaller numbers will cause fewer blocks to be acquired. Neither board will start acquiring until the trigger occurs, at which time both will acquire in lock step. The files `ad0.dat` and `ad1.dat` will contain time aligned data recorded by the two respective boards.
- 6) The trigger signal must not go high (active) until 100ms after the second board has been set to go. Once the trigger goes high, both boards will acquire concurrently, and will acquire the two channels of data.
- 7) Once both “`acquire_data`” commands have completed (new prompts appear in each window), then each board may be separately dumped to disk, using the respective `dump_board` commands, as shown below.

The files `file0` and `file1` would then contain the two channels of time-aligned information.

To use N identical ULTRADMA3 boards to store or output multiple **parallel (concurrent)** 8GB or 16GB segments of data from N channels, it is necessary to connect the boards as follows:

- 1) Use a clock splitter or distribution amplifier/buffer to convert a single clock input into N identical clock outputs, all with matching phase (within 30ps), sent along matched-length cables, to the clock input connectors of the N ULTRADMA3 boards being sequenced. The

# ULTRAVIEW

clock splitter or distribution amplifier/buffer must provide N well-matched sine or square waveforms with amplitude between 300 and 900mV peak-to-peak into 50 ohms. An Ultraview AD8-SPLIT2 or AD8-SPLIT4 clock/trigger splitter may be used to perform the above splitting.

- 2) Install the N boards in the host systems. Several boards may fit in some large systems.
- 3) Connect the N separate analog signal inputs or outputs, sent along matched-length cables, to the ANALOG INPUT or ANALOG OUTPUT connectors of the N ULTRADMA3 boards being sequenced.
- 4) Use an ECL or PECL-connected clock buffer chip (or an Ultraview AD8-SPLIT2/4) to provide N trigger outputs, all matched within 30ps, and sent along matched-length cables, to the trigger input connectors of the N ULTRADMA3 boards being sequenced.
- 5) Be sure to run jtagp versions which initialize all N boards (the board number /uvdma0 must be edited in the basic jtagp.c and a version made for the second board (/uvdma1), etc.
- 6) Start all boards, in any sequence using the `uvdma_set_go()` commands.
- 7) When the trigger signal is seen by all boards, they will begin running in step, each filling its RAM concurrently with the others. If the `BUFFER_WRAP` bit in the control register is set to 0, sampling will stop when the board (and its lock-step partner boards) has been filled. If `BUFFER_WRAP` is instead set to 1, sampling will continue by wrapping around back to the beginning of the 8GB buffer. This process will go on indefinitely, until stopped by the driver.
- 8) Data may be read out of each board, when fully or partially filled, using `read(board_fd, buffer, size)` commands, as illustrated in example programs, such as `acquire_data.c`.

### 6.5.5 digosc (digital oscilloscope)

**NOTE: Before running digosc (or before the first use of any other user program) for the first time after each system power-up, the program jtagp MUST be run, so as to load the firmware in the Xilinx FPGA on the board. Failure to run jtagp (in the example\_programs directory) after a power-up will cause all data acquired by the board to be meaningless.**

The first program to run is called “**digosc**”, a crude digital waveform display of the A/D signal(s) input to the board. This program is in the directory `/opt/uvdma/dig_osc`. The programs `digosc_controls_uvdma_stubs.c` and `dig_osc_controls_ui.c` are the source routines. Run the program `digosc` by typing **digosc** at the prompt. When you then get a window with an oscilloscope display, hit the **digitize** button, which will result in the display of the waveforms from your signal generator.

If you do not see waveforms, be sure that you have run `jtagp` (which loads the firmware into the large Xilinx XC3VP30 FPGA on the board), and that the system’s environment is correctly set. The lines present in `/opt/uvdma/environment/cshrc` should be contained in your `.cshrc` file. Once these lines are copied into your `.cshrc`, try invoking a new C shell and rerunning `digosc`. Other Xview-based programs that display waveforms from the board may be written by editing the source files for this program. These programs may be recompiled when using Solaris 8. However, the Xview libraries needed for recompilation are absent under Solaris 9 or Solaris 10. These `.h` files may, if the machine’s binary code license permits, however, be copied from the `/usr/openwin/include/xview` directory of a machine running Solaris 8, to a new directory in the same path on a machine running Solaris 9 or 10, and the digital oscilloscope source programs in this release will then compile.

The program `digosc` can be recompiled to use an external clock or different divide ratio by editing the correspondingly commented lines in `dig_osc_controls_uvdma_stubs.c`, and recompiling.

Other useful programs are `acquire_data.c`, and `acquire_data2.c` in directory `/opt/uvdma/example_programs`.

### 6.5.6 digoscdual1250\_1500 (digital oscilloscope for two-board AD8-1500DMA ganged installations)

**NOTE: Before running digoscdual1250\_1500 for the first time after each system power-up, the program jtagp MUST be run separately for each of the two boards, so as to load the firmware in the Xilinx FPGAs on the two boards. Failure to run jtagp (in the example\_programs directory) after a power-up will cause all data acquired by the boards to be meaningless.**

A simple two-channel oscilloscope program, called “**digoscdual1250\_1500**”, is available for two-board installations, in which a common clock and trigger are sent to two boards in the same system, enabling two channels to be concurrently acquired. The program `digoscdual1250_1500` is in directory `/opt/uvdma/dig_osc`. Programs `digosc_controls_uvdma_stubs_dual1250_1500.c` and `dig_osc_controls_ui.c` are the source files. Run the program by typing **digoscdual1250\_1500**. When you get a window with an oscilloscope display, hit the **digitize** button, which will result in the display of the two waveforms from your signal generators. Adjust your signal generator’s frequency and amplitude to display several cycles of the waveforms. Note that there is no vertical offset between the two waveforms – if the two channels are fed with identical analog signals, the two waveforms will overlap.

For digoscdual1250\_1500 to correctly display two time aligned waveforms, be sure that 1) A common trigger is connected to both boards, using a splitter that has better than 30ps delay matching, 2) A common clock is connected to both boards, using a splitter that has better than 30ps delay matching, and 3) The trigger signal to the splitter has an amplitude of between 1V and 2V and a rise time of less than 2ns, 4) The trigger signal does not go high (active) until after the digitize button has been clicked on, and 5) The trigger signal, once it occurs, remains high for at least 200ns. If the trigger amplitude or duration is incorrect, or if there is excessive skew between the clock or trigger entering the two boards, the two waveforms displayed will be misaligned.

## 7.APPENDIX 1 – Installing AD8-SPLIT2/4 Clock/Trigger Splitter

The AD8-SPLIT2 or AD8-SPLIT4 are optional clock/trigger splitter boards that can, from a single clock and optional trigger input, generate as many as four matched clock and trigger outputs for concurrently triggering and running up to four AD8-1500DMA or DA8-1250DMA boards.

Powered by a “wall-cube” power supply (included), the AD8-SPLIT2/4 boards are connected via short SMA-to-SMA cables to the clock and trigger inputs on the AD8-1500DMA boards that are to be run concurrently

To install the AD8-SPLIT2/4 board, make the following cable connections, as shown in figure 8.1:

- 1) Connect the TRIG0 (and TRIG1, 2 and 3, if used) outputs of the AD8-SPLIT2/4 to the TRIG inputs on the AD8-1500 boards you wish to use. Be sure that the cables used are as short as possible (less than 1 foot recommended) and matched to within  $\frac{1}{4}$ ". Also, check that all SMA cable connectors are firmly hand-tightened onto their respective jacks. Additionally, be sure that JP1 on all AD8-1500 boards is jumpered to its top position (ECL/PECL Trigger).
- 2) Connect the CLK0 (and CLK1, 2 and 3, if used) outputs of the AD8-SPLIT2/4 to the CLK inputs on the AD8-1500 boards you wish to use. Be sure the cables used are short and matched to within  $\frac{1}{4}$ ". Check that all SMA cable connectors are firmly hand-tightened.
- 3) Connect the input clock source (0dBm amplitude) to the AD8-SPLIT2/4's "CLOCK IN" jack.
- 4) Connect the optional TTL trigger input source to the "TTL TRIG INPUT" jack on the AD8-SPLIT2/4. This TTL-only input must never be driven with any signal source that is outside the 0 to +4V TTL voltage range. A TTL 0 is recognized when the TTL TRIG INPUT signal is below 0.8V, and a TTL 1 is recognized when the input is above 2.0V.
- 5) Plug the +5V adapter's cord into the AD8-SPLIT2/4's "DC5V IN" jack. Only the included 5V regulated DC (center + terminal) adapter should be used with the AD8-SPLIT2/4.
- 6) Plug the +5V AC adapter into a standard 120VAC outlet.

After installing the AD8-SPLIT2/4 board, please refer to the appropriate sections in this manual for tips on operating two boards concurrently.



**Figure 7-7.1. Connection of AD8-SPLIT2/4 to two AD8-1500DMA or DA8-1250DMA boards installed in a Sun host system. The left two thick black cables are the TRIG0 and TRIG1 outputs, and the middle two thick black cables are the CLK0 and CLK1 outputs. The black cable going into the top of the AD8-SPLIT2/4 boardlet is from the RF clock source, and the bottom black cable is from the TTL trigger source. The white cable is the +5V DC input from the included wall-cube adapter. The rightmost two thick black cables connected directly to the AD8-1500DMA boards are the signal inputs for the two boards.**