Application Note AN499

Example proFPGA FM-XCVU440-R2 design for a V2M-Juno Motherboard

Non-Confidential

ARM®

Example proFPGA FM-XCVU440-R2 design for a V2M-Juno Motherboard

Release Information

The following changes have been made to this Application Note.

Change History

Date	Issue	Confidentiality	Change
13 April 2016	A	Non-Confidential	First release
08 July 2016	B	Non-Confidential	Memory map fixes and more detail

ARM Limited. Company 02557590 registered in England.

110 Fulbourn Road, Cambridge, England CB1 9NJ.

LES-PRE-20349

Contents

Example proFPGA FM-XCVU440-R2 design for a V2M-Juno Motherboard

1 Conventions and Feedback

The following describes the typographical conventions and how to give feedback:

Typographical conventions

The following typographical conventions are used:

`monospace` denotes text that you can enter at the keyboard, such as commands, file and program names, and source code.

`monospace` denotes a permitted abbreviation for a command or option. You can enter the underlined text instead of the full command or option name.

`monospace italic`
denotes arguments to commands and functions where the argument is to be replaced by a specific value.

`monospace bold`
denotes language keywords when used outside example code.

italic highlights important notes, introduces special terminology, denotes internal cross-references, and citations.

bold highlights interface elements, such as menu names. Denotes signal names. Also used for emphasis in descriptive lists, where appropriate.

Feedback on this product

If you have any comments and suggestions about this product, contact your supplier and give:

- Your name and company.
- The serial number of the product.
- Details of the release you are using.
- Details of the platform you are using, such as the hardware platform, operating system type and version.
- A small standalone sample of code that reproduces the problem.
- A clear explanation of what you expected to happen, and what actually happened.
- The commands you used, including any command-line options.
- Sample output illustrating the problem.
- The version string of the tools, including the version number and build numbers.

Feedback on documentation

If you have comments on the documentation, e-mail `errata@arm.com`. Give:

- The title.
- The number, DAI0499B.
- If viewing online, the topic names to which your comments apply.
- If viewing a PDF version of a document, the page numbers to which your comments apply.
- A concise explanation of your comments.

ARM also welcomes general suggestions for additions and improvements.

ARM periodically provides updates and corrections to its documentation on the ARM Information Center, together with knowledge articles and *Frequently Asked Questions* (FAQs).

Other information

- ARM Information Center, http://infocenter.arm.com/help/index.jsp
- ARM Technical Support Knowledge Articles, http://infocenter.arm.com/help/topic/com.arm.doc.faqs/index.html
- ARM Support and Maintenance, http://www.arm.com/support/services/support-maintenance.php
- ARM Glossary, http://infocenter.arm.com/help/topic/com.arm.doc.aeg0014g/index.html
- proFPGA Technical Support, profpga-service@prodesign-europe.com

The ARM Glossary is a list of terms used in ARM documentation, together with definitions for those terms. The ARM Glossary does not contain terms that are industry standard unless the ARM meaning differs from the generally accepted meaning.

2 Overview

2.1 Purpose of this application note

This application note discusses the operation of an example design for a proDesign FM-XCVU440-R2 UltraScale FPGA module operating with a V2M-Juno Motherboard. It examines the contents of the FPGA Module, the system interconnects, the clock structure, and the specifics of the programmer's model directly relevant to the application note operation.

Reading this application note prepares a user to debug and analyze the operation of the provided example design, make changes to the example design, connect their own AXI or AHB based masters, and AXI, AHB or APB slaves.

2.2 Customer support for ProDesign products

Support for FPGA development using proDesign UltraScale on the ARM Juno platform is provided directly by ProDesign, not ARM. Please contact profpga-service@prodesign-europe.com for assistance.

For general questions on the Juno development platform contact juno-support@arm.com

2.3 References

- *Release notes as supplied with the V2M-Juno motherboard.*
- *ARM DDI 0424 - AMBA® DMA Controller DMA-330 Technical Reference Manual.*
- *ARM 100113_0000_04_en - VersatileTM Express Juno Development Platform (V2M-Juno) Technical Reference Manual.*
- *ARM 100122_0100_00_en - VersatileTM Express Juno r1 Development Platform (V2M-Juno r1) Technical Reference Manual.*
- *ARM 100114_0200_00_en - VersatileTM Express Juno r2 Development Platform (V2M-Juno r2) Technical Reference Manual.*
- *proDesign UD011 - proFPGA ARM Express Adapter User Manual.*
- *proDesign UD001 - proFPGA Hardware User Manual.*

2.4 Terms and abbreviations

DDR	Double Data Rate DRAM.
DMA	Direct Memory Access.
DMC	Dynamic Memory Controller.
DRAM	Dynamic Random Access Memory.
RAM	Random Access Memory.
FPGA	Field Programmable Gate Array.
AXI	Advanced eXtensible Interface.
SCC	Serial Configuration Controller.

DCC	Daughterboard Configuration Controller.
TRM	Technical Reference Manual.
DCM	Digital Clock Manager.
MMCM	Mixed-mode Clock Manager.
APB	Advanced Peripheral Bus.
RTL	Register Transfer Level.
XML	Extensible Markup Language.
TMIF	Thin Links Master Interface.
TSIF	Thin Links Slave Interface.
MMI64	Module Message Interface.
$\mathbf{T_{su}}$	Setup time required by input.
$\mathbf{T_{h}}$	Hold time required by input.
$\mathbf{T_{co\text{-}min}}$	Minimum time between clock and valid output data.
$\mathbf{T_{co\text{-}max}}$	Maximum time between clock and valid output data.

3 Getting Started

The steps below show you how to set up the hardware platform, copy the necessary configuration files to the V2M-Juno motherboard USB Flash disk, and program the proFPGA user FPGA image.

The components required to build this platform are as follows:

- V2M-Juno Motherboard with power supply and USB and RS232 cables
- ProDesign EB-PDS-ARM-EXPRESS-R1-Adapter
- ProDesign UNO Motherboard
- ProDesign FM-XCVU440-R2 Xilinx Ultrascale FPGA board
- ProDesign ATX Power Supply
- USB or Network cable for ProDesign connectivity
- ProDesign EB-PDS-DDR4-R3 DDR4 extention board

Full instructions for assembling the proFPGA + Juno platform are given in the proDesign documents *[UD011] ARM Express Adapter User Manual* and *[UD001] proFPGA Hardware User Manual*.

To configure the platform and your host computer ready to run AN499, proceed as follows:

1. Assemble the platform hardware as described in the ARM Express Adapter User Manual.
2. Install the proFPGA Builder software package on your host computer.
3. Copy the AN499 package of files to a convenient location on your host computer.
4. Connect the ATX power supply to the UNO motherboard, and a USB cable between the host computer and XUSB1 on the UNO motherboard.
5. Connect the USB, UART0 RS232, and power cables to V2M-Juno Motherboard.
6. Start a serial terminal emulator on the host computer for Juno UART0 (115200, 8, n, 1)
7. Apply power to the Juno motherboard.
8. Issue the command `Cmd> flash` on the Juno serial console.
9. Issue the command `Flash> eraseall` on the Juno serial console.
10. Issue the command `Flash> exit` on the Juno serial console
11. Issue the command `Cmd> usb_on` on the Juno serial console.
12. After the host computer recognizes the Juno motherboard Micro-SD card as a USB mass storage device, format the card (FAT16 or FAT32) and then copy the contents of the AN499 recovery directory on to the card.
13. Issue the command `Cmd> reboot` on the Juno serial console. The Juno motherboard will start up and program the various software images into the motherboard NOR flash memory.
14. When the Juno console shows the message `Waiting for Tile Site CB_READY` shut down the Juno motherboard by pressing the hardware reset button on its rear panel.

The above steps only need to be carried out once - the Juno motherboard is now configured for use with the AN499 application note.

To load the AN499 bit file into the FPGA and run the selftest example software, the following steps must be carried out each time the platform is powered up:

1. Open a command prompt window on the host computer and change directory into /AN499/boardfiles/ in the AN499 files directory tree.
2. Apply power to the Juno motherboard.
3. Apply power to the proFPGA board set.
4. Turn on the proFPGA board set by setting switch S7 to the ON position. (The first time this is done, it will be necessary to install the profpga_usb_rndis driver).
5. Issue the command `Cmd> reboot` on the Juno serial console.
6. When the Juno console shows the message `Waiting for Tile Site CB_READY` enter the command `profpga_run mig_arm.cfg -u` in the host computer command prompt window.
7. After the bitfile has been programmed into the FPGA, UEFI will begin running on the Juno motherboard. When the message `The default boot selection will start in 3 seconds` appears on the Juno serial console, break to the UEFI menu by pressing a key.
8. Select menu option `Start: 2` on the Juno serial console.
9. Issue the command `Shell> FS2:` on the Juno serial console.
10. Issue the command `FS2:\> runaxf selftest` on the Juno serial console.

The self test example software will begin running and offer a menu of tests to exercise the AN499 application note running on the FPGA.

To shut down, proceed as follows:

1. Issue the selftest command x to exit from the selftest example software.
2. Enter control-c in the command prompt window on the host computer to shut down the proFPGA board set.
3. Reset the Juno motherboard by pressing the hardware reset button.
4. Turn off the proFPGA board set by setting switch S7 to the OFF position.
5. Remove power from the proFPGA board set.
6. Remove power from the Juno motherboard.

4 System architecture

This system is AXI (AMBA 4) based. The example V2F-Juno image exposes one AXI slave and one AXI master port via Thin Links interfaces to the V2M-Juno motherboard.

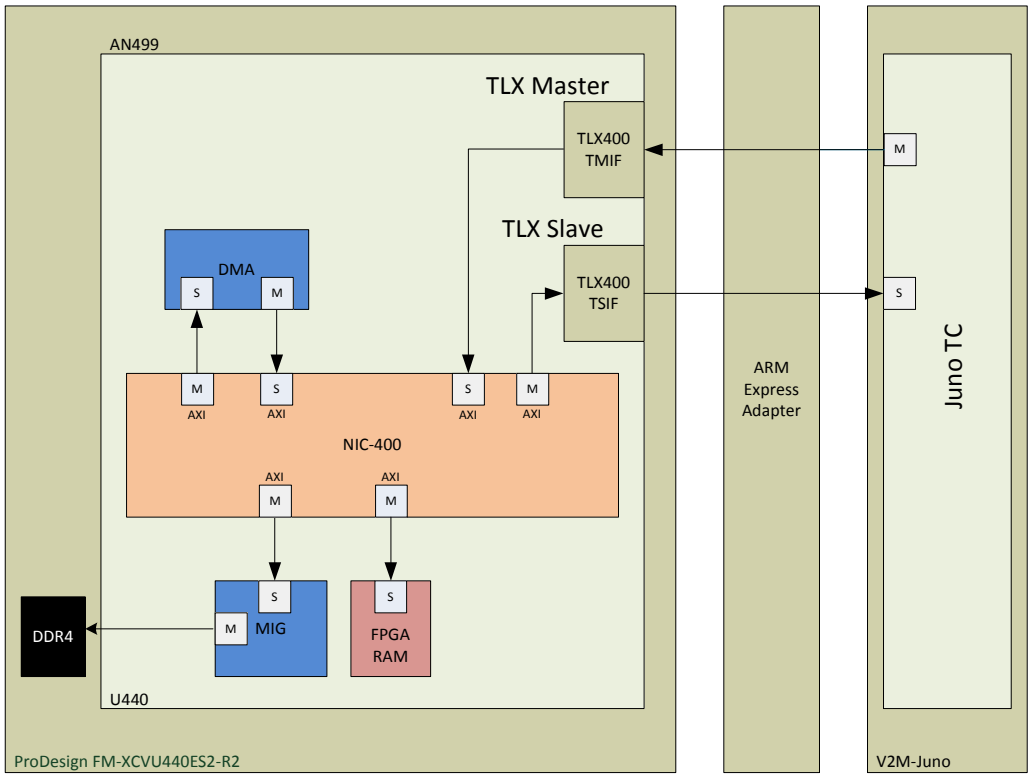

Figure 4-1 System Architecture

Note that the direction of the arrows indicates the direction of control, which points from the Master to the Slave. An AXI bus contains signals going in both directions.

4.1 Module functionality

NIC-400

The NIC-400 is a high performance AMBA-compliant network infrastructure used to connect together the AXI masters and peripherals in AN499.

MIG

The MIG is a DDR4 memory controller configured using Xilinx Vivado tools.

FPGA RAM

Block RAM in the FPGA provides a memory area for general use.

DMA

A PL330 DMA controller is available to move data to and from the TLX400 TSIF.

TLX400 TSIF

The TLX400 is a modified NIC400 Thin Links implementation. It has been split between this application note and the Juno chip providing an efficient high bandwidth link. The TSIF is a slave interface from the point-of-view of the Juno chip and allows AN499 to master into Juno.

TLX400 TMIF

The TLX400 is a modified NIC400 Thin Links implementation. It has been split between this application note and the Juno chip providing an efficient high bandwidth link. The TMIF is a master interface from the point-of-view of the Juno chip and allows Juno to master into AN499.

4.2 Modules revision

Module	Revision
NIC-400 Bus Matrix	r0p2
PL330 DMA	r1p0
DDR4 SDRAM (MIG)	v1.0
PL401 TLX400	r0p0

4.3 Clock architecture

Figure 4-2 describes the clock domains used in AN499.

Section 9.1 Default operating frequencies on page 9-19 describes which oscillator controls which internal clock, along with their default frequencies.

For information how to set up and change the frequencies of the clocks, please refer to V2M-Juno Motherboard TRM and ARM Express Adapter User Manual.

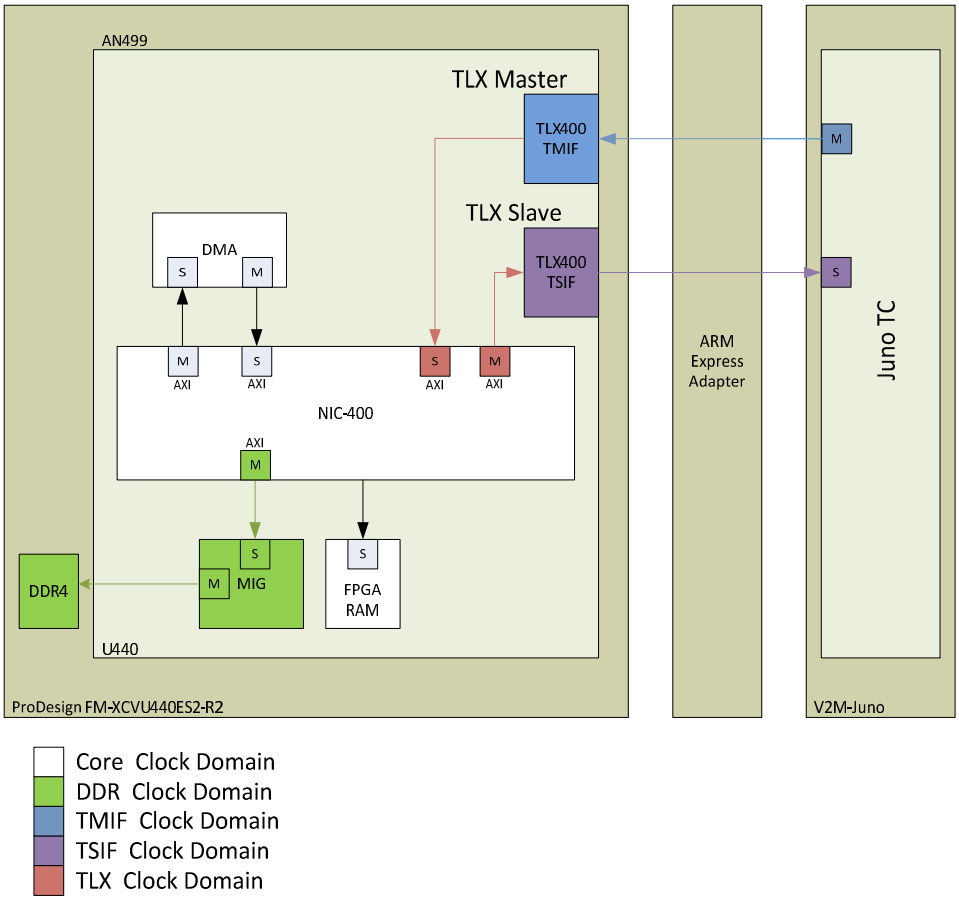

Figure 4-2 Clock Domains

There are 5 clock domains in this design.

Core Clock

DDR, runs the MIG. Created internally.

TMIF, TLX Master, generated by V2M-Juno.

TSIF, TLX Slave, sent to V2M-Juno.

TLX Internal, independent to master and slave clocks.

Table 1 Derived Clocks

Clock	Derived From	Default Frequency
ACK	CLK_1	100.0 MHz
txif_clk_x1	CLK_1	61.5 MHz
txif_clk_x2	CLK_1	123.0 MHz
txif_clkl	CLK_1	61.5 MHz

4.4 Reset architecture

The reset signal CB_nRST from the Juno motherboard is synchronized to the correct clock domains and used to reset all the peripherals in the proFPGA system.

The MMCMs in the User Application Note are reset by the MMI64 infrastructure. The MMI64 infrastructure is provided by proFPGA and is a user fpga clock synchronization module.

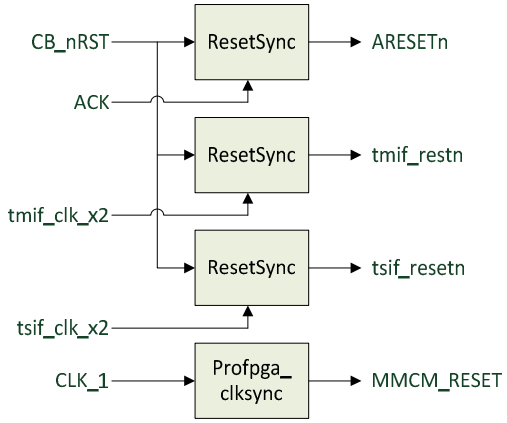

Figure 4-3 Reset Synchronization

The reset signal CB_nPOR is not used in the design.

5 Hardware description

5.1 AN499 wrapper (AN499_wrapper)

This level contains the clocking structure for the application note, static pin tie offs and instantiation of the toplevel module.

5.2 AN499 top level (AN499_toplevel)

This level connects all the major Application Note components together. The top level RTL is provided so modules can be added and removed by the user.

All modules are provided as .DCP netlists.

A Xilinx design constraints file (xdc) is provided with the application note. This provided all FPGA pins used on the User FPGA(s) and timing constraints.

5.3 Thin Links multiplexing scheme

Thin Links is implemented with a multiplexing scheme. This scheme is included in the Thin Links RTL block and is not available for user modification. Further information on Thin Links is available in the Juno Development platform Technical Reference Manual.

5.4 Header HDRX pin allocation

Both Think Link interfaces connect to the HDRX header as shown in the Table 2 X Bus Connectivity, below.

5.5 X Bus connectivity

Table 2 X Bus Connectivity

Thin Links port	Xbus	Thin Links port	Xbus
tmif_clko	XP[110]	tsif_valid_o	XN[70]
tmif_clki	XP[100]	tmif_ctl_i[0]	XP[92]
tmif_ctl_o	XN[126]	tmif_ctl_i[1]	XP[90]
tmif_data_o[0]	XN[159]	tmif_data_i[0]	XN[124]
tmif_data_o[1]	XN[157]	tmif_data_i[1]	XP[123]
tmif_data_o[2]	XN[155]	tmif_data_i[2]	XN[122]
tmif_data_o[3]	XN[153]	tmif_data_i[3]	XP[121]
tmif_data_o[4]	XN[151]	tmif_data_i[4]	XN[120]
tmif_data_o[5]	XP[159]	tmif_data_i[5]	XP[128]
tmif_data_o[6]	XN[158]	tmif_data_i[6]	XP[126]
tmif_data_o[7]	XP[157]	tmif_data_i[7]	XP[124]
tmif_nrst	XP[89]	tmif_valid_i	XN[89]
tsif_nrst	XP[70]	tsif_ctl_i	XN[33]
tmif_valid_o	XP[125]	tsif_data_i[0]	XN[0]
tsif_clko	XP[59]	tsif_data_i[1]	XN[2]
tsif_clki	XP[49]	tsif_data_i[2]	XN[4]
tsif_ctl_o[0]	XP[67]	tsif_data_i[3]	XN[6]
tsif_ctl_o[1]	XP[69]	tsif_data_i[4]	XN[8]
tsif_data_o[0]	XN[35]	tsif_data_i[5]	XP[0]
tsif_data_o[1]	XP[36]	tsif_data_i[6]	XN[1]
tsif_data_o[2]	XN[37]	tsif_data_i[7]	XP[2]
tsif_data_o[3]	XP[38]	tsif_data_i[8]	XN[3]
tsif_data_o[4]	XN[39]	tsif_data_i[9]	XP[4]
tsif_data_o[5]	XP[31]	tsif_data_i[10]	XN[5]
tsif_data_o[6]	XP[33]	tsif_data_i[11]	XP[6]
tsif_data_o[7]	XP[35]	tsif_data_i[12]	XN[7]
tsif_data_o[8]	XP[37]	tsif_data_i[13]	XP[8]
tsif_data_o[9]	XP[39]	tsif_data_i[14]	XN[9]
tsif_data_o[10]	XN[40]	tsif_data_i[15]	XP[1]

Thin Links port	Xbus	Thin Links port	Xbus
tsif_data_o[11]	XN[42]	tsif_data_i[16]	XP[3]
tsif_data_o[12]	XN[44]	tsif_data_i[17]	XP[5]
tsif_data_o[13]	XN[46]	tsif_data_i[18]	XP[7]
tsif_data_o[14]	XN[48]	tsif_data_i[19]	XP[9]
tsif_data_o[15]	XP[40]	tsif_data_i[20]	XN[10]
tsif_data_o[16]	XN[41]	tsif_data_i[21]	XN[12]
tsif_data_o[17]	XP[42]	tsif_data_i[22]	XN[14]
tsif_data_o[18]	XN[43]	tsif_data_i[23]	XN[16]
tsif_data_o[19]	XP[44]	tsif_data_i[24]	XP[10]
tsif_data_o[20]	XN[45]	tsif_data_i[25]	XN[11]
tsif_data_o[21]	XP[46]	tsif_data_i[26]	XP[12]
tsif_data_o[22]	XN[47]	tsif_data_i[27]	XN[13]
tsif_data_o[23]	XP[48]	tsif_valid_i	XP[34]

5.6 Header HDRY pin allocation

Table 3 HDRYConnectivity

Pin	Signal	Pin	Signal
A6	CB_VIO[0]	F6	CB_nRST
A7	CB_SDA	F7	CB_SSPDO
A9	SB1_nDACK0	F16	SB1_IRQ5
A10	SB1_INT2	G5	CB_URXD
A15	SB1_IRQ10	G6	CB_OK
A16	SB1_IRQ0	G7	CB_SSPAD[0]
B5	CB1_TDI	G16	SB1_IRQ6
B6	CB_VIO[1]	H5	CB_RSTREQ
B7	CB_SCL	H6	CB_SSPAD[2]
B10	SB1_INT3	H7	CB_SSPAD[1]
B15	SB1_IRQ11	H16	SB1_IRQ7
B16	SB1_IRQ1	H19	SB1_GCLK2
C5	CB1_TDO_R	J4	CB_SPWR
C6	CB_VIO[2]	J5	CB_VRAMP
C7	CB_SnWC	J6	CB_READY

Pin	Signal	Pin	Signal
C10	SB1_nDRQ0	J7	CB_SSPAD[3]
C15	SB1_IRQ12	J9	SB_nCPUIRQ
C16	SB1_IRQ2	J11	SB1_INT0
D6	CB_CFGnRST	J16	SB1_IRQ8
D7	CB_SSPCK	K1	CB_DET2
D16	SB1_IRQ3	K6	SB1_GCLK1
E6	CB_nPOR	K7	SB1_GCLK0
E7	CB_SSPDI	K8	SB_EVENTI
E16	SB1_IRQ4	K9	SB_nCPUFIQ
E34	CB1_SAD0	K11	SB1_INT1
E38	CB1_SAD1	K16	SB1_IRQ9
F5	CB_UTXD		

6 Programmer’s model

The example design provides

- FPGA SRAM
- DDR4 memory controller (DMC)
- DMA controller
- TLX400 master bus to AN499 slave port.
- TLX400 slave bus from AN499 master port.
- One spare AXI port slave and one spare AXI port master on the NIC400.
- Two spare APB ports on the NIC400.

6.1 AXI memory map

The AXI slave implemented by AN499 exists in the 256MB memory window from 0x00_6000_0000 to 0x00_6FFF_FFFF of V2M-Juno.

Different AXI masters can see different memory areas. The visibility is detailed for each master with Yes indicating access is available and N/C indicating no connection. Juno refers to transfers initiated by V2M-Juno over the TLX interface. DMA refers to accesses from the PL330 DMA controller. Spare refers to accesses from the spare AXI port reserved for user expansion.

Table 4 Memory Map

Start Address	End Address	Description	Bus	Juno AXI-M	DMA AXI-M	Spare AXI-M	Size
0x7000_0000	0xFFFF_FFFF	Access to V2M-Juno	TLX AXI-S	N/C	N/C	Yes	≈ 2.3 GB
0x6800_0000	0x6FFF_FFFF	External SDRAM	AXI	Yes	Yes	Yes	128 MB
0x6400_0000	0x67FF_FFFF	Spare AXI slave	AXI	Yes	Yes	Yes	64 MB
0x6020_0000	0x63FF_FFFF	Reserved	-	N/C	N/C	N/C	63 MB
0x6012_0000	0x601F_FFFF	FPGA Block RAM - Reserverd	AXI	Yes	Yes	Yes	896 KB
0x6010_0000	0x6011_FFFF	FPGA Block RAM - Implemented	AXI	Yes	Yes	Yes	128 KB
0x6006_0000	0x600F_FFFF	Reserved	-	N/C	N/C	N/C	640 KB
0x6005_0000	0x6005_FFFF	Spare APB slave 2	APB	Yes	N/C	Yes	64 KB
0x6004_0000	0x6004_FFFF	Spare APB slave 1	APB	Yes	N/C	Yes	64 KB
0x6002_0000	0x6002_FFFF	PL330 config	APB	Yes	N/C	Yes	64 KB
0x6000_0000	0x6001_FFFF	Reserved	-	N/C	N/C	N/C	128 KB
0x3000_0000	0x5FFF_FFFF	Access to V2M-Juno	TLX AXI-S	N/C	Yes	Yes	768 MB
0x2F00_0000	0x2FFF_FFFF	Reserved	-	N/C	N/C	N/C	16 MB
0x0000_0000	0x2EFF_FFFF	Access to V2M-Juno	TLX AXI-S	N/C	Yes	Yes	752 MB

6.2 Interrupts

This Application Note has two interrupt outputs for the DMA, SB_INT[0] and SB_INT[1].

These are passed over the HDRY connector, connecting to Juno test chip.

Only SB_INT[0] is used by Juno. SB_INT[1] is connected but not available for use.

6.3 Reserved and undefined memory

If reserved memory is accessed, it is caught by the AXI bus matrix and returns a decode error (‘DECERR’) which generates a data abort.

7 RTL

Example files are provided to allow building of the system with Xilinx tools.

The following are included in the application note:

- Top level Verilog RTL
- AXI Compnents as .dcp netlists
- AMBA Designer XML configuration file to allow components to be rebuilt

7.1 Directory structure

The application note has several directories. They are:

- amba_designer: XML configuration files provided to rebuild the NIC400.
- boardfiles: These files are required to program the design into a proFPGA Board.
- docs: Related documents including this document.
- logical: Verilog RTL for this design.
- physical: Synthesis and place and route (P&R) scripts and builds for target board.
- software: ARM executable code to exercise the AN499 application note.

7.2 Logical

The logical directory contains all the verilog supplied with this application note. It also contains AMBA Designer XML configuration files which can be used to regenerate verilog for ARM PrimeCell used in the example design.

7.3 Physical

The physical directory contains synthesized netlists. The function of each block is shown in 4-5 Module functionality.

Each PrimeCell or other large IP block has its own directory (for example `ds703_scc_r0p3`).

7.4 Building the App Note using Linux

AN499 must be built with Xilinx Vivado 2015.3

To build the application note using Linux, run the `make_vivado.scr` batch file in the following directory: `/physical/an499_toplevel/xilinx_revc/scripts/`

This synthesizes the design and runs place and route on the design pulling in pre synthesized components. The memory controller IP will also be generated as part of this process.

A programmable bit file `an499_wrapper.bit` is generated in `/physical/an499_toplevel/xilinx_revc/netlist/`

A prebuilt example of this bit file appears in the boardfiles directory as `a499r0p0.bit`

7.5 Using the bitfile with the proDesign UNO FM-XCVU440-R2 motherboard

To use the new `an499_wrapper.bit` bit file, the filename must be edited into `mig_arm.cfg`

1. Open `mig_arm.cfg` in a text editor.
2. Edit the line `bitstream =` to point to the required bit file
   (for example `bitstream = "C:/application_notes/prodesign_an499/an499_wrapper.bit";`).
3. Run the profpga_run tool to configure the Juno System with the new bitfile.

8 Example software

Example software (Selftest) is provided to verify the example design and the proFPGA system hardware. Selftest is built using ARM Compiler 6.

C, assembly language source files and ARM DS-5.23.0 (or later) project files are included.

After the Versatile Express Juno system is configured you can run the copy of the example software that has been programmed into NOR flash using UEFI, or upload and execute it using a debugger connected to the V2M-Juno motherboard.

The example code communicates with the user via the debugger's console window. It operates as follows:

1. Reads the identification register to ensure that the software is being executed on the correct system.
2. Tests the FPGA SRAM by performing sequential and random write/read accesses of word, half-word and byte lengths.
3. Tests the DDR4 SDRAM by performing sequential and random write/read accesses of word, half-word and byte lengths.
4. Tests the DMA controller and interrupt signal from DMA to the V2M-Juno motherboard.

9 I/O Timing Requirements

All of these specific timing requirements refer to the r0p0 revision of AN499. Units in nano-seconds "ns" and have been rounded to a worst case value.

Signals with setup, hold and clock to data values are bidirectional signals or have been grouped by function in the table.

9.1 Default operating frequencies

Table 5 Operating Frequencies

Clock	Oscillator	Ratio	Default Frequency
ACLK	CLK[1]	x2	100.0
TLX	CLK[1]	x1.23	61.5
DDR SYS CLK	CLK[1]	x2	100.0
DDR REF CLK	CLK[1]	x4	200.0
TMIF_CLKO	Juno - OSC0	x1.23	61.5
TSIF_CLKO	Juno - OSC0	x1.23	61.5

Frequencies above the default frequency are not supported.

9.2 Thin Links timing requirements

The Thin Links interfaces in both directions operates as a DDR interface with clock transition in the center of the data eye.

Input timing requirements

Table 6 Input Timing Requirements

T_{su}	1.2ns	Minimum data setup time before clock edge.
T_h	1.0ns	Minimum data hold time after clock edge.

Output timing requirements

Table 7 Output Timing Requirements

$T_{co\text{-}min}$	2.5ns	Data is available on the bus between $T_{co\text{-}min}$ and $T_{co\text{-}max}$.
$T_{co\text{-}max}$	-2.5ns	Data is available on the bus between $T_{co\text{-}min}$ and $T_{co\text{-}max}$.