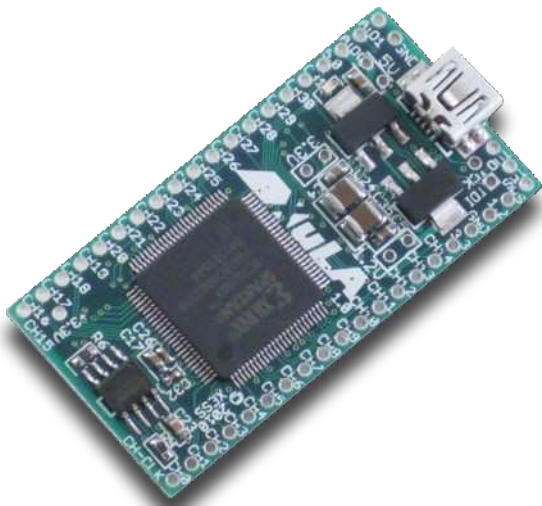


XuLA Manual

*How to install, test and use
your new FPGA board*



XESS is disclosing this Document and Intellectual Property (hereinafter “the Design”) to you for use in the development of designs to operate on, or interface with XESS hardware devices. XESS expressly disclaims any liability arising out of the application or use of the Design. XESS reserves the right to make changes, at any time, to the Design as deemed desirable in the sole discretion of XESS. XESS assumes no obligation to correct any errors contained herein or to advise you of any correction if such be made. XESS will not assume any liability for the accuracy or correctness of any engineering or technical support or assistance provided to you in connection with the Design.

THE DESIGN IS PROVIDED “AS IS” WITH ALL FAULTS, AND THE ENTIRE RISK AS TO ITS FUNCTION AND IMPLEMENTATION IS WITH YOU. YOU ACKNOWLEDGE AND AGREE THAT YOU HAVE NOT RELIED ON ANY ORAL OR WRITTEN INFORMATION OR ADVICE, WHETHER GIVEN BY XESS, OR ITS AGENTS OR EMPLOYEES. XESS MAKES NO OTHER WARRANTIES, WHETHER EXPRESS, IMPLIED, OR STATUTORY, REGARDING THE DESIGN, INCLUDING ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE, AND NONINFRINGEMENT OF THIRD-PARTY RIGHTS.

IN NO EVENT WILL XESS BE LIABLE FOR ANY CONSEQUENTIAL, INDIRECT, EXEMPLARY, SPECIAL, OR INCIDENTAL DAMAGES, INCLUDING ANY LOST DATA AND LOST PROFITS, ARISING FROM OR RELATING TO YOUR USE OF THE DESIGN, EVEN IF YOU HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. THE TOTAL CUMULATIVE LIABILITY OF XESS IN CONNECTION WITH YOUR USE OF THE DESIGN, WHETHER IN CONTRACT OR TORT OR OTHERWISE, WILL IN NO EVENT EXCEED THE AMOUNT OF FEES PAID BY YOU TO XESS HEREUNDER FOR USE OF THE DESIGN. YOU ACKNOWLEDGE THAT THE FEES, IF ANY, REFLECT THE ALLOCATION OF RISK SET FORTH IN THIS AGREEMENT AND THAT XESS WOULD NOT MAKE AVAILABLE THE DESIGN TO YOU WITHOUT THESE LIMITATIONS OF LIABILITY.

The Design is not designed or intended for use in the development of on-line control equipment in hazardous environments requiring fail-safe controls, such as in the operation of nuclear facilities, aircraft navigation or communications systems, air traffic control, life support, or weapons systems (“High-Risk Applications”). XESS specifically disclaims any express or implied warranties of fitness for such High-Risk Applications. You represent that use of the Design in such High-Risk Applications is fully at your risk.

© 2001–2010 XESS, Inc. All rights reserved. XESS, the XESS logo, and other designated brands included herein are trademarks of XESS Corporation. All other trademarks are the property of their respective owners.



This document is licensed under the Attribution-ShareAlike 3.0 Unported license, available at <http://creativecommons.org/licenses/by-sa/3.0/>.

XuLA Manual MAN001 (V1.5)

The following table shows the revision history for this document.

Date	Version	Revision
12/16/2010	1.0	Initial release for XuLA Board V1.0.
08/10/2011	1.1	Enhanced prototyping header picture with pin I/O directions.
08/31/2011	1.2	Added section on 5V-tolerance for prototyping header.
07/25/2012	1.3	Clean-up / polish text.
09/10/2012	1.4	Added notes about connections from the prototyping header pins to FPGA global clock pins in the Pin Connections appendix .
12/28/2012	1.5	Expanded explanation of how to use a XILINX programming cable with the XuLA.

Table of Contents

- C.1 Preliminaries..... 1**
 - Getting Help!..... 1
 - Take Notice!..... 1
- C.2 Installation..... 2**
 - Installing the XSTOOLS Utilities and Documentation..... 2
 - Connecting Your XuLA Board to a PC..... 2
 - Testing Your XuLA Board..... 3
 - Setting the Jumpers on Your XuLA Board..... 3
 - Applying Power to Your XuLA Board..... 4
 - Applying Power Through the USB Port..... 4
 - Applying Power Through the Prototyping Header..... 4
 - Inserting the XuLA Board into a Breadboard..... 5
- C.3 Programming..... 6**
 - Generating Bitstreams for the FPGA..... 6
 - Downloading Bitstreams into the FPGA..... 8
 - Downloading Using GXSLD..... 8
 - Downloading Using a XILINX or Third-Party JTAG Cable..... 10
 - Storing Non-Volatile Bitstreams in the Flash..... 11
 - Transferring Data to/from the SDRAM..... 14
- C.4 Programmer Models..... 16**
 - XuLA Board Components..... 16
 - FPGA..... 17
 - Microcontroller..... 17
 - SDRAM..... 18
 - Flash..... 19
 - Prototyping Header..... 20
 - 5V Tolerance Issues..... 21
 - Auxiliary JTAG Header..... 22
- A.1 Pin Connections..... 23**
- A.2 Schematic..... 25**

C.1 Preliminaries

Here's some helpful information before getting started.

Getting Help!

Here are some places to get help if you encounter problems:

- If you can't get the XuLA Board hardware to work, send an e-mail message describing your problem to help@xess.com or submit a problem report at <http://www.xess.com/help.php>. Our web site also has:
 - answers to frequently-asked-questions,
 - example designs, application notes and tutorials for our FPGA boards,
 - a place to sign-up for our email forum where you can post questions to others.
- If you can't get your XILINX ISE *WebPACK* software tools installed properly, check their web site at <http://www.xilinx.com/support/>.
- If you need help using the XILINX ISE *WebPACK* software to create FPGA designs, then check out this [tutorial](#).

Take Notice!

- The XuLA Board is not 5V-tolerant. **Do not connect 5V logic signals to the prototyping header.**
- The XuLA printed circuit board (PCB) is manufactured such that the terminals of the jumpers labeled "5V", "3.3V" and "1.2V" are connected on the underside of the PCB by short wiring traces. **You must cut these traces if you want to open the jumper connections.**
- Even if you have experience with the XILINX ISE *WebPACK* software, please read this [section on setting the bitstream generation options for the XuLA Board](#).

C.2 Installation

Installing the XSTOOLS Utilities and Documentation

XILINX currently provides the free ISE® *WebPACK*™ software for programming many of their small and mid-size FPGAs and CPLDs. You can download the most current version of ISE *WebPACK* from www.xilinx.com.

In addition, XESS provides the XSTOOLS utilities for interfacing a PC to your XuLA Board. These utilities (along with manuals, design examples and tutorials) are installed automatically when you insert the XSTOOLS CD into your PC. If not, then manually run the SETUP.EXE installation program on the CD. You can also download the XSTOOLS installer from www.xess.com.

Connecting Your XuLA Board to a PC

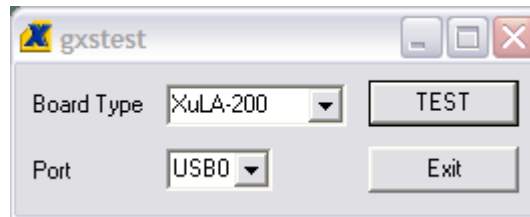
The XuLA Board is a USB peripheral that you can attach to any USB 1.1 or 2.0 port through a cable with a five-pin mini-B connector such as this one:



The LED on your XuLA Board will light up as soon as it establishes a connection with the PC.

Testing Your XuLA Board

Once your XuLA Board is connected to a USB port, you can test it by double-clicking the GXSTEST icon placed on your PC desktop during the XSTOOLS installation. This brings up the window shown below.



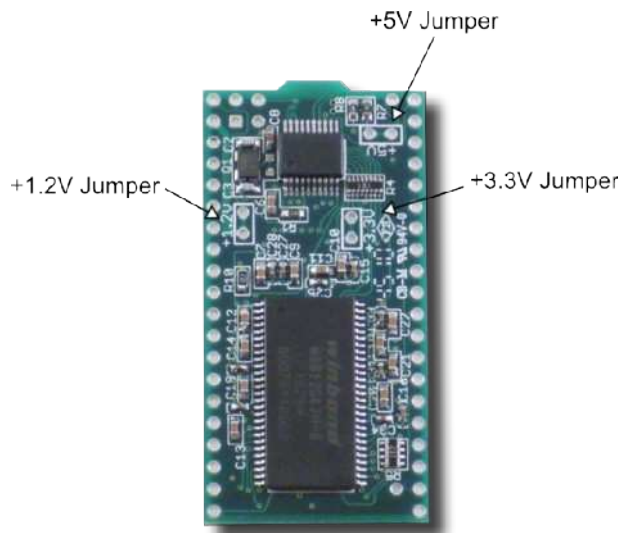
Next, select the type of board you are testing and the port that it's attached to. Then click on the TEST button. GXSTEST will configure the FPGA on your XuLA Board to perform a test procedure. Within a few seconds, a status window will appear informing you of the success or failure of the test.

If your XuLA Board fails the test, you will be shown a checklist of common causes for failure. If none of these applies to your situation, then [contact XESS Corp](#) for further assistance.

Setting the Jumpers on Your XuLA Board

The XuLA Board has three jumpers labeled "5V", "3.3V" and "1.2V" that are used to configure how the board receives power. In their factory-original configuration, the jumpers are unpopulated but the terminals of each jumper are connected on the underside of the board by short wiring traces. **You must cut these traces if you wish to open the jumper connections.** You only need to do this if you are powering your XuLA Board through its prototyping header (not through the USB cable) as described [here](#).

The locations of the shorting traces on the underside of the XuLA Board are shown below.



Applying Power to Your XuLA Board

There are two ways of powering your XuLA Board that can be used alone or in combination:

- receiving power through the USB connector, or
- applying power through the XuLA prototyping header.

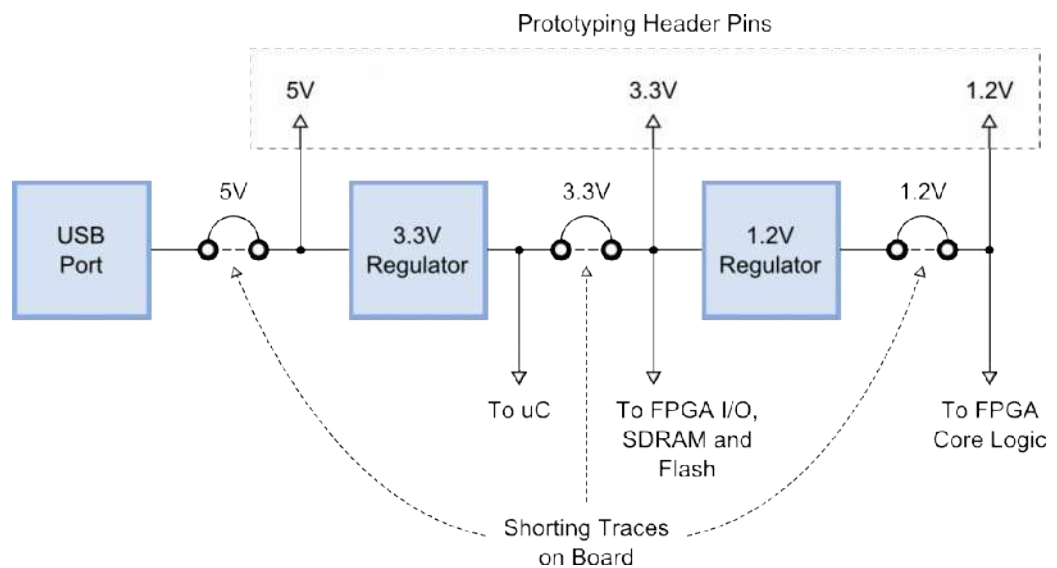
Applying Power Through the USB Port

Connecting the XuLA Board to a PC USB port provides a 5V supply capable of delivering up to 500 mA of current. This is sufficient for many small to medium-sized FPGA designs running at less than 200 Mhz.

Applying Power Through the Prototyping Header

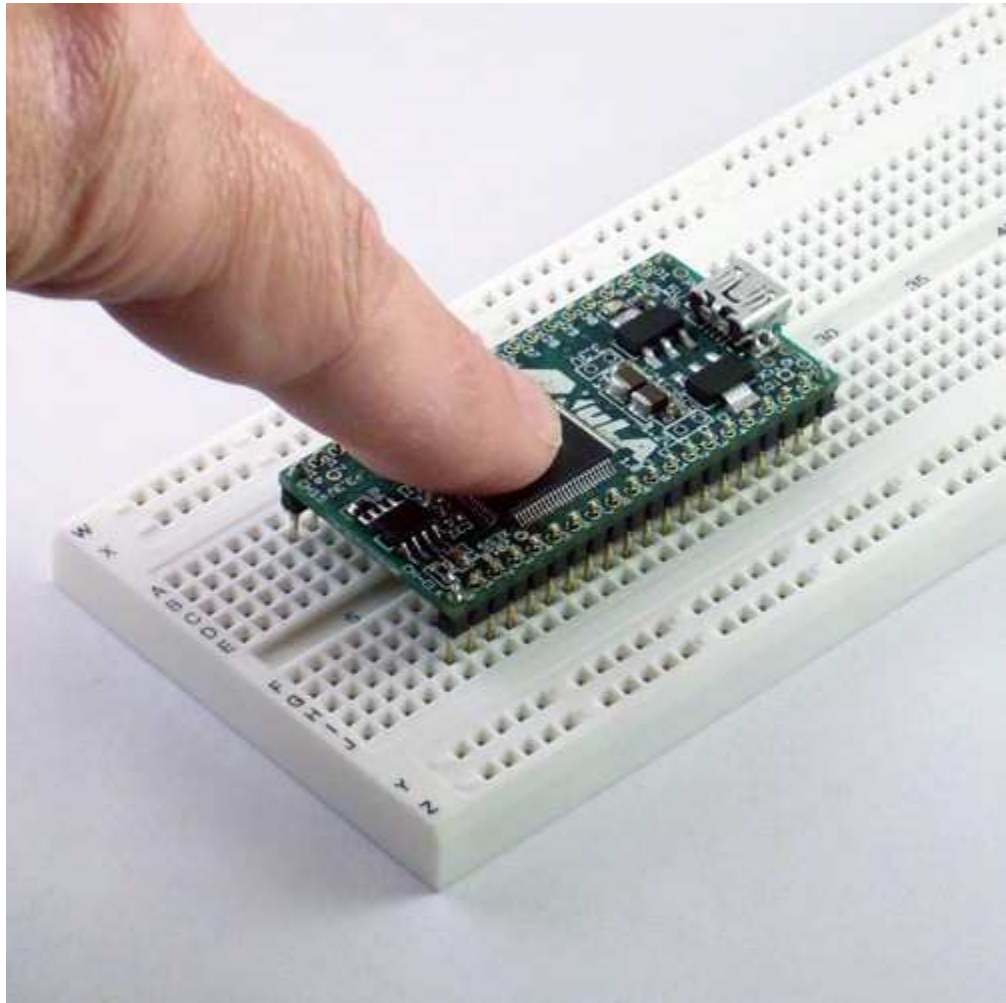
For more power-hungry applications, you can connect one or more voltage supplies directly to the power supply pins of the prototyping header. There are several ways to do this:

- You can attach an 18V–5V supply directly to the 5V pin of the prototyping header. The XuLA's voltage regulators will generate the required 3.3V and 1.2V supplies needed by the microcontroller, FPGA, SDRAM and Flash (see the figure below). **Do not attach a USB cable unless you have removed the shunt from the 5V jumper** or else you will short the PC USB supply to the external voltage supply and cause possible damage.
- You can attach a 3.3V supply directly to the 3.3V pin of the prototyping header. This supply will directly power the microcontroller, FPGA I/O, SDRAM and Flash while the voltage regulator will generate the 1.2V needed by the FPGA core logic. **Do not attach a USB cable unless you have removed the shunt from the 5V jumper and do not attach a supply to the 5V prototyping pin** or else the output of the 3.3V regulator will drive against the external 3.3V supply and cause possible damage.
- You can power the FPGA core logic by attaching a 1.2V supply directly to the 1.2V pin of the prototyping header and then use one of the previous two methods to power the rest of the XuLA board. **You must remove the shunt on the 1.2V jumper to isolate the output of the 1.2V regulator from the external 1.2V supply.**



Inserting the XuLA Board into a Breadboard

In its factory-original configuration, the XuLA's prototyping header is empty. If desired, you can solder in a pair of twenty-pin headers and then insert the XuLA into a standard solderless breadboard as shown below.



The XuLA Board will accept common headers with 0.025"-thick pins at 0.1" spacing, but you may find it difficult to remove the XuLA Board given how tightly the breadboard grips the pins. A header with thinner pins (such as the Aries 20-0600-20) is a better choice.

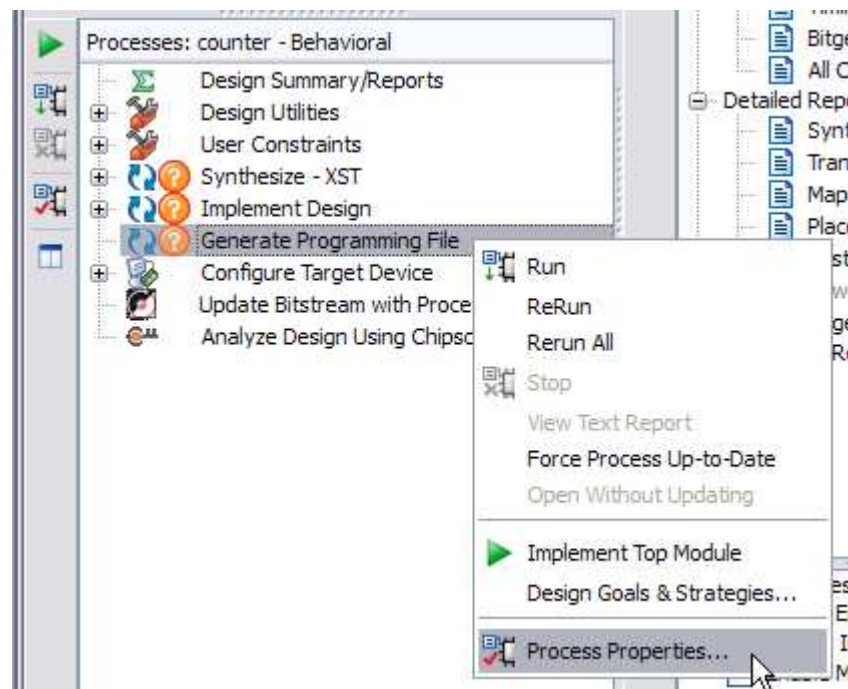
C.3 Programming

This chapter will show you how to download logic designs into the FPGA of your XuLA Board and how to transfer data between the PC and the SDRAM and Flash memories on the board.

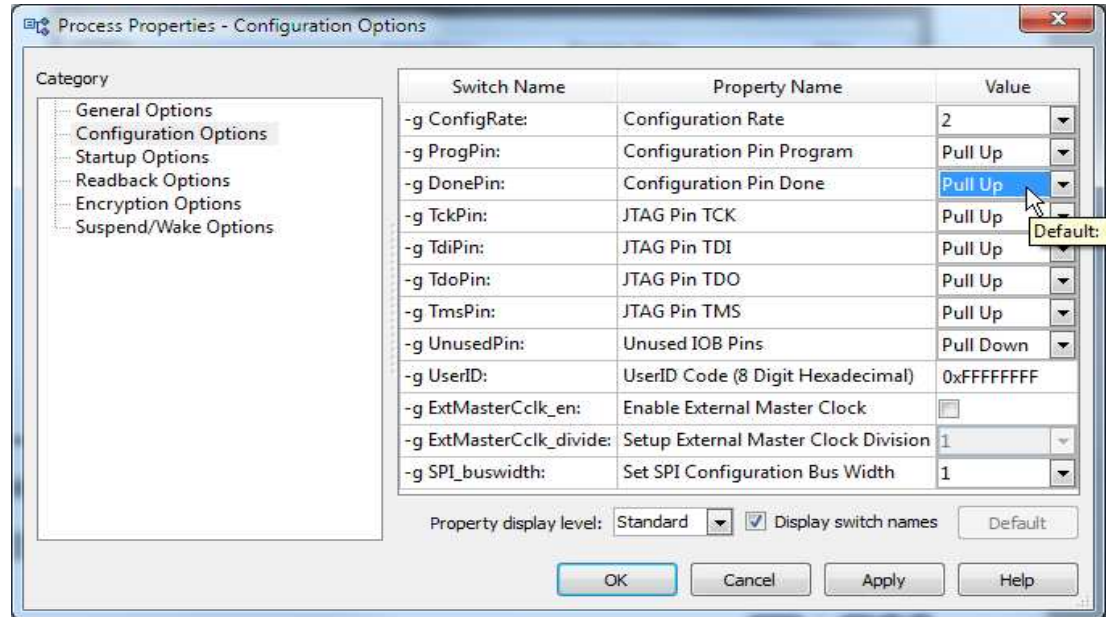
Generating Bitstreams for the FPGA

Before you can download a logic design to the FPGA on your XuLA Board, you need to generate a bitstream (i.e., a .BIT file) with XILINX ISE *WebPACK*. Steps for doing this are given in the XILINX documentation and this [XESS tutorial](#), but there are several details that you have to be aware of when generating the bitstream.

After creating your logic design in ISE *WebPACK*, right-click on the **Generate Programming File** item in the **Process** window and select **Process Properties...** from the context menu as shown below.

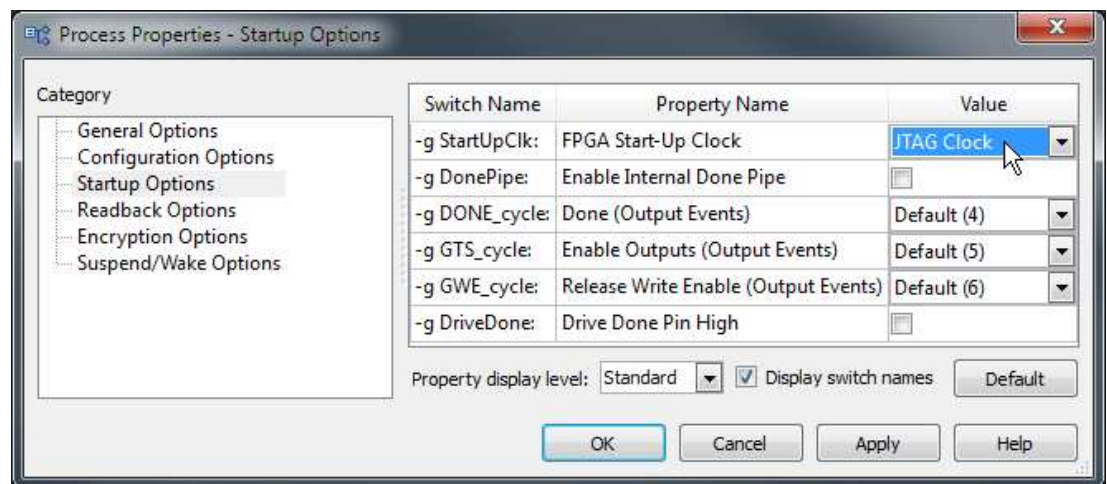


When the **Process Properties** window appears, select the **Configuration Options** category. Make sure the **Configuration Pin Done** property is set to the value **Pull Up**. This is necessary for the microcontroller on the XuLA to detect when the FPGA has successfully been configured by the bitstream.



Next, select the **Startup Options** category and set the **FPGA Start-Up Clock** to one of the following values:

FPGA Start-Up Clock Settings	
JTAG Clock	Use this setting when your bitstream will be downloaded directly into the XuLA's FPGA through the USB cable.
CCLK	Use this setting when your bitstream will be downloaded into the XuLA's serial configuration Flash for eventual transfer to the FPGA.

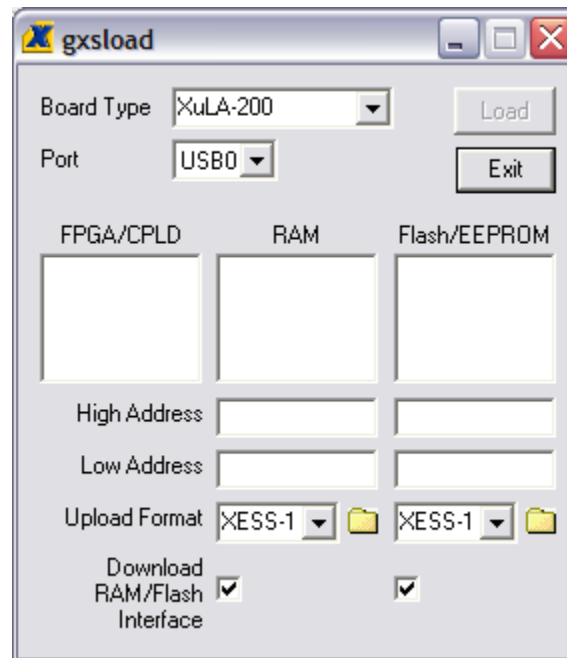


Downloading Bitstreams into the FPGA

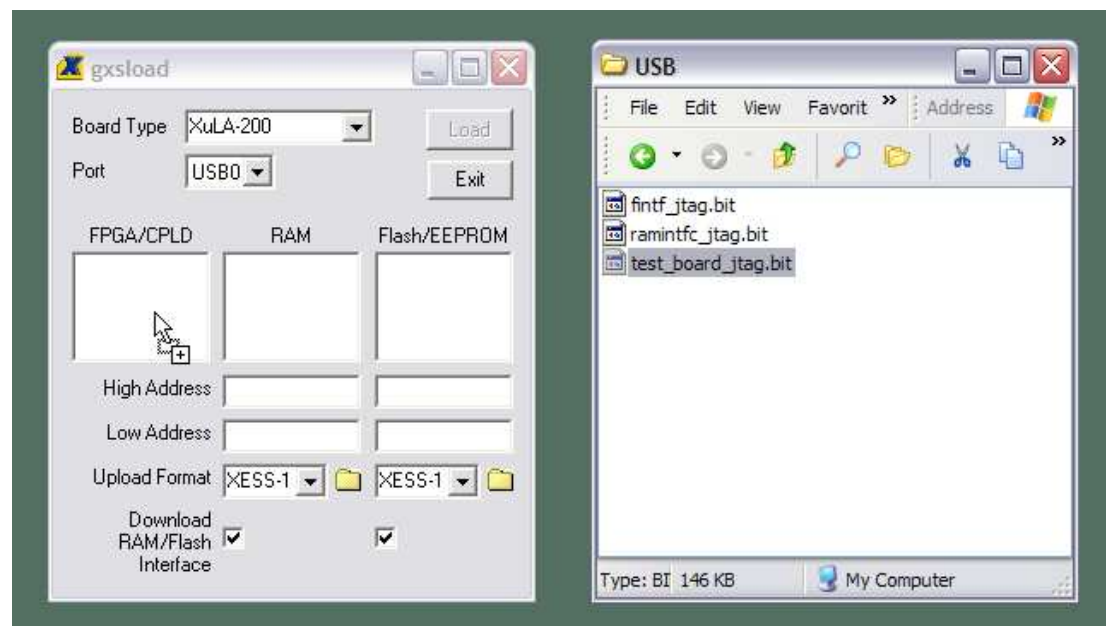
Downloading Using GXSLD

As you work on a logic design, you will usually download the bitstream from the PC to the XuLA to test your modifications. The GXSLD utility is used for downloading bitstreams over the USB link.

Start GXSLD by double-clicking the icon placed on the desktop during the XSTOOLS installation. This brings up the window shown below.

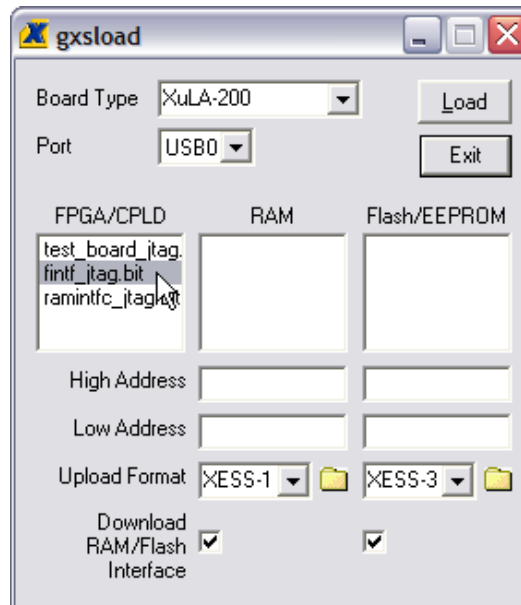


Now you can download bitstream files to the FPGA or CPLD simply by dragging them from their folder and dropping them into the **FPGA/CPLD** pane as shown below.

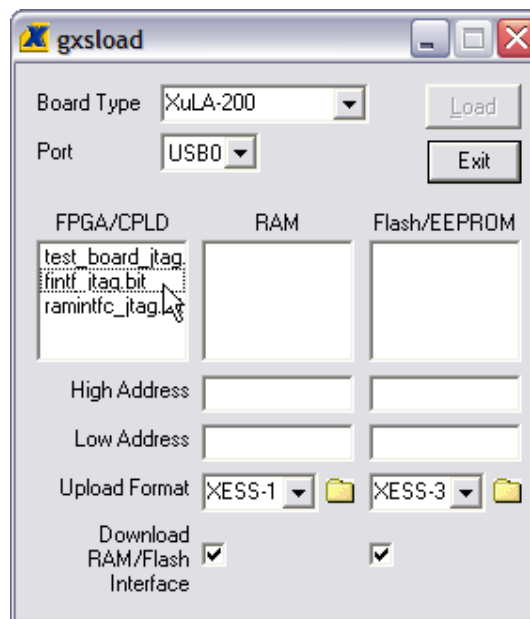


Once you drop the file, the highlighted file name appears in the **FPGA/CPLD** pane and the **Load** button is enabled. Clicking on the **Load** button will begin sending the bitstream to the XuLA Board through the USB cable. GXSLD will reject any non-downloadable files (ones with a suffix other than .BIT). During the downloading process, GXSLD will display the name of the bitstream file and the progress of the current download. The LED on the XuLA Board will blink as the bitstream is transferred.

You can drag & drop multiple files into the **FPGA/CPLD** pane. Clicking your mouse on a file name will highlight the name and select it for downloading. Only one file at a time can be selected for downloading.

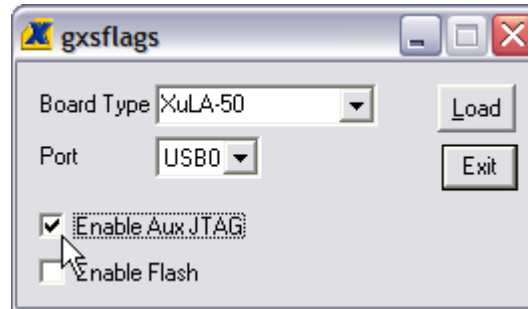


Double-clicking the highlighted file will deselect it so no file will be downloaded. Doing this disables the **Load** button.



Downloading Using a XILINX or Third-Party JTAG Cable

As an alternative to using GXSLoad and a simple USB cable, you can use a XILINX or third-party JTAG cable to configure the FPGA on the XuLA. But first, you have to set a non-volatile flag to give priority to the auxiliary JTAG header using the GXSFLAGS utility. Connect the XuLA to your PC with a USB cable and then double-click the GXSFLAGS icon so the following window appears.

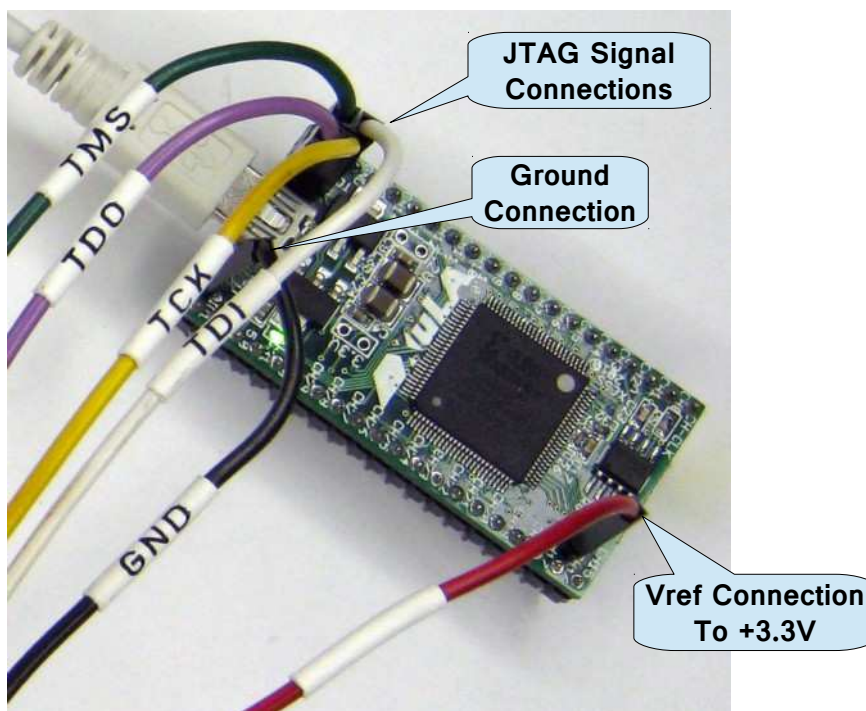


Click on the **Enable Aux JTAG** checkbox and then click on the **Load** button to set the flag. Note that the flag only needs to be set once to support the JTAG cable because it retains its value even when power is removed from the XuLA. (Be aware that enabling the auxiliary JTAG header disables the functions of the other XSTOOLS utilities such as GXSTEST and GXSLoad. To restore their functions, you'll have to use GXSFLAGS again to disable the auxiliary JTAG header.)

Now attach your JTAG cable to the auxiliary JTAG header as shown below. (The USB cable remains attached only to provide the XuLA with power. You can detach it if the board is getting power from another source, such as through its prototyping header.)



Here's a close-up of the connections from the XILINX cable to the XuLA:



Now start the ISE *WebPACK* iMPACT tool and follow the XILINX instructions to download bitstreams to the FPGA in boundary-scan mode.

Storing Non-Volatile Bitstreams in the Flash

The FPGA on the XuLA stores its configuration in an on-chip SRAM which is erased whenever power is removed. Once you complete a design, you may want to store the bitstream in the serial configuration Flash on the XuLA. After that, the FPGA will configure itself from the Flash each time power is applied.

Loading a bitstream into the Flash is easy: just drag the .BIT file into the **Flash/EEPROM** pane of GXSLD and click on the **Load** button. This activates the following sequence of events:

1. The FPGA on the XuLA is configured with an interface to transfer data from the USB port to the Flash.
2. The entire Flash is erased.
3. The contents of the .BIT file are downloaded into the Flash through the USB port.

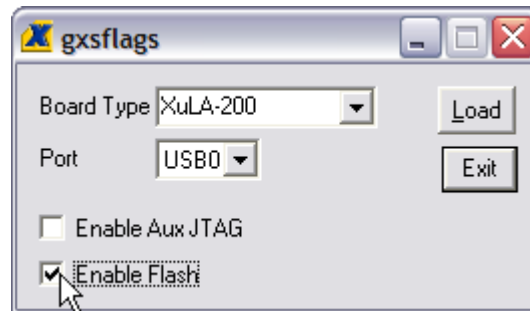
Once the Flash download is completed, the FPGA will be configured with the stored bitstream whenever power is applied to the XuLA. *(Make sure that you selected CCLK as the start-up clock when you generated the bitstream or else the FPGA will fail to configure from the Flash.)*

You can also use the XILINX iMPACT software to convert FPGA bitstreams into .MCS or .EXO data files. These file types can be programmed into the XuLA's Flash using GXSLD in the same way as with a .BIT file.


Multiple files can be stored in the Flash device just by dragging them into the

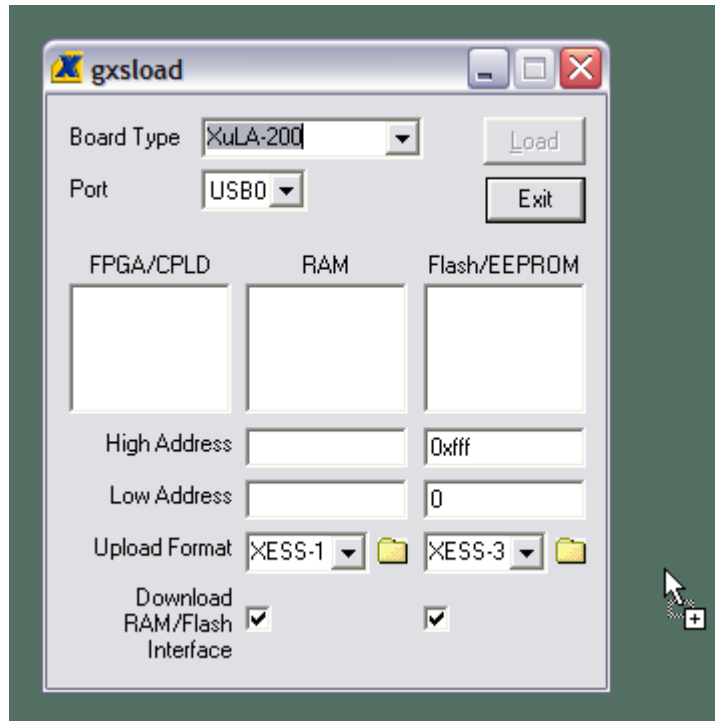
Flash/EEPROM area, highlighting the files to be downloaded and clicking the **Load** button. (Note that anything previously stored in the Flash will be erased by each new download.) This is useful if you need to store information in the Flash in addition to the FPGA bitstream. Files are selected and de-selected for downloading just by clicking on their names in the Flash/EEPROM area. The address ranges of the data in each file should not overlap or this will corrupt the data stored in the Flash device!

Because of the FPGA's limited number of pins, only the SDRAM or Flash (not both) is accessible by the FPGA. In order to access the data that follows the bitstream in the Flash, you have to set a non-volatile flag to allow Flash access. First, connect the XuLA Board to your PC with a USB cable. Then double-click the GXSFflags icon so the following window appears.



Click on the **Enable Flash** radio button and then click on the **Load** button to set the flag in the microcontroller. Now, any FPGA logic design you download to the XuLA Board can access the Flash (but not the SDRAM).

You can also examine the contents of the Flash by uploading it to the PC. To upload data from an address range in the Flash, type the upper and lower bounds of the range into the **High Address** and **Low Address** fields located below the **Flash/EEPROM** pane, and select the format for the uploaded data from the **Upload Format** pulldown list. Then click on the file icon () and drag & drop it into any folder.



This activates the following sequence of steps:

1. The FPGA on the XuLA Board is configured with an interface between the Flash device and the PC USB port.
2. The Flash data between the high and low addresses (inclusive) is uploaded through the USB port.
3. The uploaded data is stored in a file named FLSHUPLD with an extension that reflects the selected upload file format.


The uploaded data can be stored in the following formats:

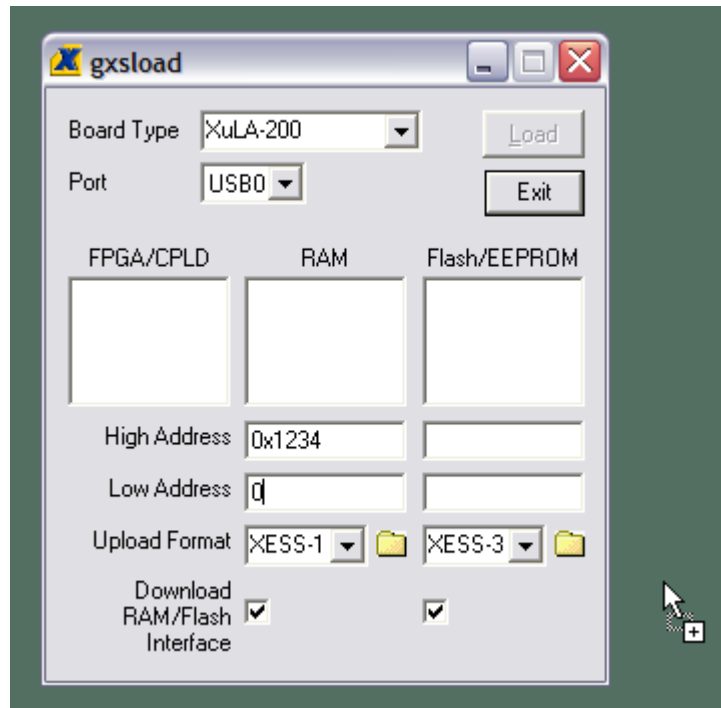
- MCS:** Intel hexadecimal file format.
- HEX:** Identical to MCS format.
- EXO-16:** Motorola S-record format with 16-bit addresses (suitable for 64 KByte uploads only).
- EXO-24:** Motorola S-record format with 24-bit addresses.
- EXO-32:** Motorola S-record format with 32-bit addresses.
- XESS-16:** XESS hexadecimal format with 16-bit addresses. (This is a simplified file format that does not use checksums.)
- XESS-24:** XESS hexadecimal format with 24-bit addresses.
- XESS-32:** XESS hexadecimal format with 32-bit addresses.

Transferring Data to/from the SDRAM

The XuLA Board contains a synchronous DRAM (SDRAM) whose contents can be downloaded and uploaded by GXLOAD. This is useful for initializing the SDRAM with data for use by the FPGA and then reading the SDRAM contents after the FPGA has operated upon it. The SDRAM is loaded with data by dragging & dropping one or more .EXO, .MCS, .HEX, and/or .XES files into the **RAM** pane of the **gxload** window and then clicking on the **Load** button. This activates the following sequence of steps:

1. The FPGA is configured with an interface to transfer data between the SDRAM device and the USB port.
2. The contents of the .EXO, .MCS, .HEX or .XES file are downloaded into the SDRAM through the USB port.

You can also examine the contents of the SDRAM device by uploading it to the PC. To upload data from an address range in the SDRAM, type the upper and lower bounds of the range into the **High Address** and **Low Address** fields below the **RAM** pane, and select the format for the uploaded data from the **Upload Format** pulldown list. Then click on the file icon () and drag & drop it into any folder.



This activates the following sequence of steps:

1. The FPGA is configured with an interface between the SDRAM device and the PC USB port.
2. The SDRAM data between the high and low addresses (inclusive) is uploaded through the USB port.
3. The uploaded data is stored in a file named RAMUPLD with an extension that reflects the selected upload file format.

The 16-bit data words in the SDRAM are mapped into the eight-bit data format of the .HEX, .MCS, .EXO and .XES files using a Big Endian style. That is, the 16-bit word at location *N* in the SDRAM is stored in the eight-bit file with the upper eight bits at address

$2N$ and the lower eight bits at address $2N+1$. This byte-ordering applies for both RAM uploads and downloads.

C.4 Programmer Models

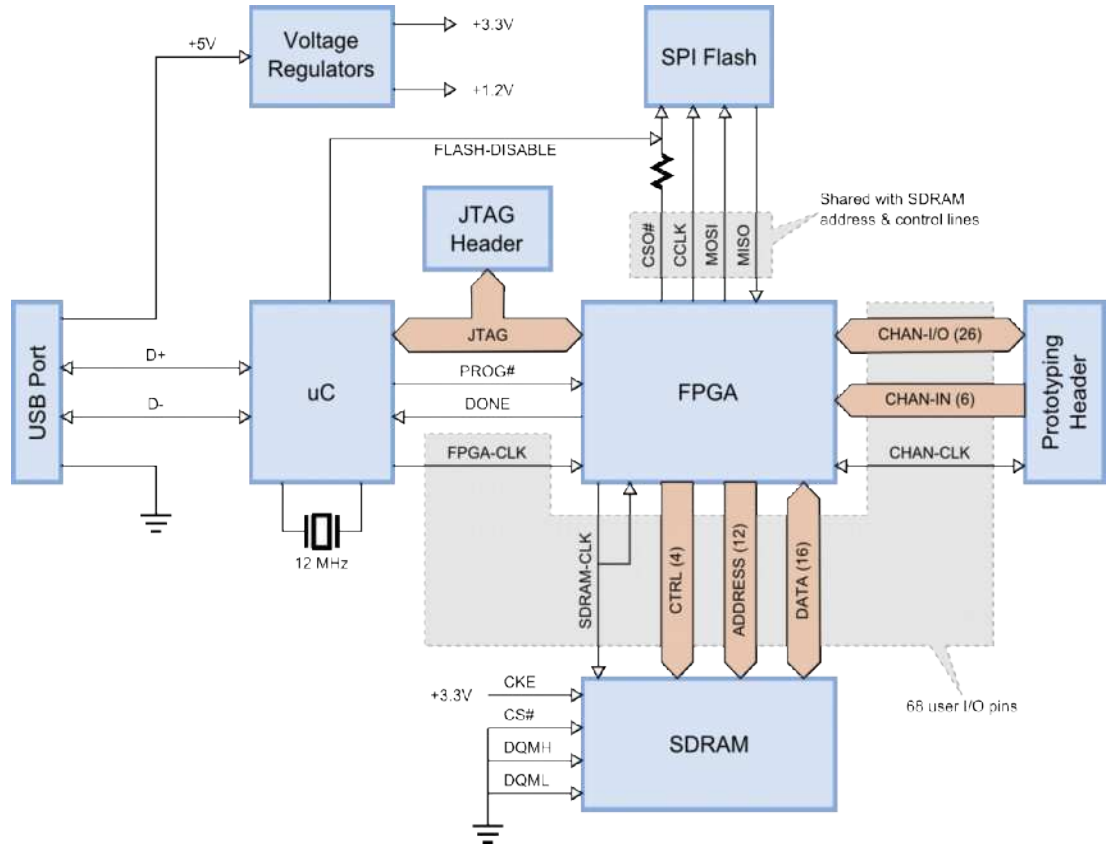
This section describes the various sections of the XuLA Board and shows how the FPGA I/O is connected to the rest of the circuitry. The schematics which follow are less detailed so as to simplify the descriptions. For more information, you can find a table of [pin connections](#) and detailed [schematics](#) at the end of this manual.

XuLA Board Components

The XuLA Board contains the following major components:

- FPGA:** This is the programmable logic device.
- Microcontroller:** The microcontroller (uC) handles initialization, clock generation and USB-to-JTAG communications.
- SDRAM:** A 128-Mbit SDRAM provides volatile data storage accessible by the FPGA.
- Flash:** A 2-Mbit serial Flash device provides non-volatile storage for FPGA configuration bitstreams and user data.
- Prototyping Header:** FPGA I/O pins, several uC pins and power/GND pins are connected to this 40-pin header that is meant to mate with solderless breadboards or other devices with 0.1"-spacing sockets.
- Aux. JTAG Header:** This header provides direct access to the FPGA's JTAG pins.

The interconnection of these components is shown in the following block diagram.



FPGA

The programmable logic device on the XuLA Board is either a [XILINX 200,000-gate XC3S200A Spartan 3A FPGA](#) in a 100-pin VQFP, or a 50,000-gate XC3S50A.

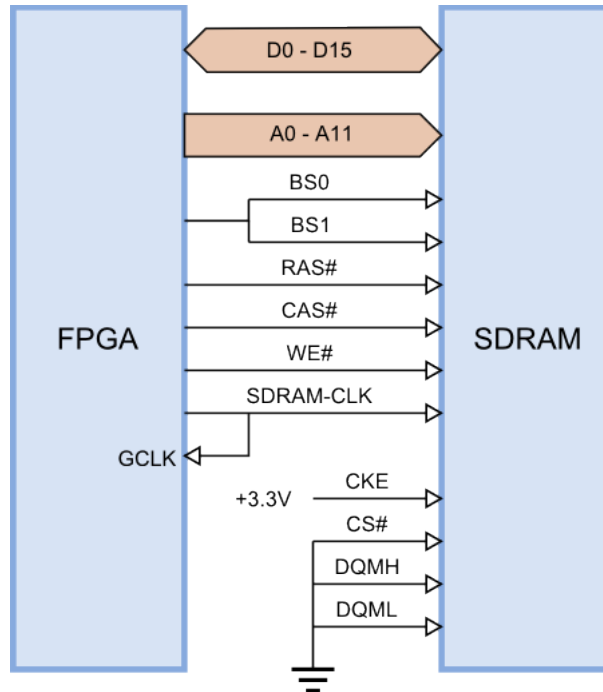
Microcontroller

The XuLA Board uses a [Microchip 18F14K50 PIC](#) to perform the following functions:

- Reset & initialization:** Upon power-up or assertion of the reset, the microcontroller initiates the configuration of the FPGA from the Flash and holds the FPGA in its cleared state if the configuration fails. It instantiates its USB endpoints and participates in the USB enumeration process.
- Clock generation:** The microcontroller uses its pulse-width modulation (PWM) circuitry to generate a 12 MHz square-wave that enters one of the FPGA's global clock inputs.
- USB-to-JTAG communication:** The microcontroller accepts configuration bitstreams and data as packets over the USB link and transforms these into a sequence of transitions upon the FPGA's JTAG pins. It also receives data from the FPGA through the JTAG port which it bundles into packets for return through the USB link.

SDRAM

The XuLA Board incorporates an 8M x 16 SDRAM ([Winbond W9812G6JH](#)) that connects to the FPGA as shown below. To compensate for circuit delays, the clock signal to the SDRAM is re-routed back to a global clock input so the FPGA can synchronize itself with the SDRAM.

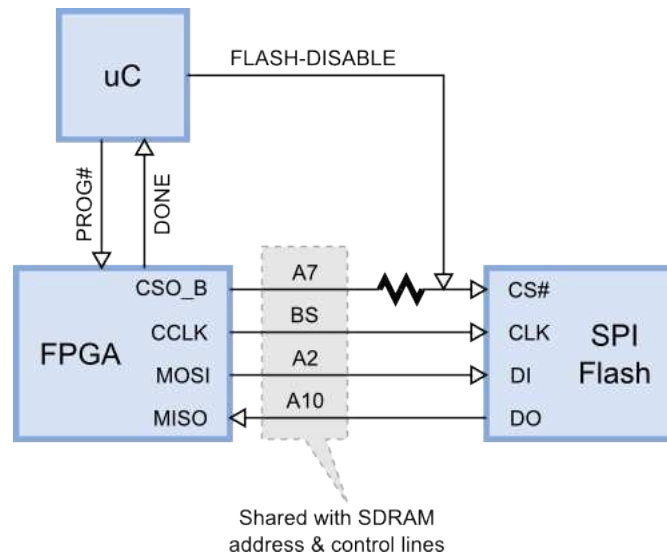


Because of the limited number of pins provided by the FPGA's package, the following modifications were made to the SDRAM interface:

- The chip-select (CS#) is tied low so the SDRAM is always enabled.
- The clock-enable (CKE) is tied high so the SDRAM always requires a clock signal in order to keep its data refreshed.
- Both data qualifier mask enables (DQMH and DQML) are tied low to disable byte-wide access to the SDRAM, so all operations involve the entire sixteen-bit data width.
- A single FPGA pin drives both bank-select inputs (BS0 and BS1). **This reduces the accessible region of the SDRAM by half to 4M x 16.**

Flash

The XuLA Board has a 2-Mbit SPI Flash ([Winbond W25X20BV](#)) that connects to the FPGA as shown below.



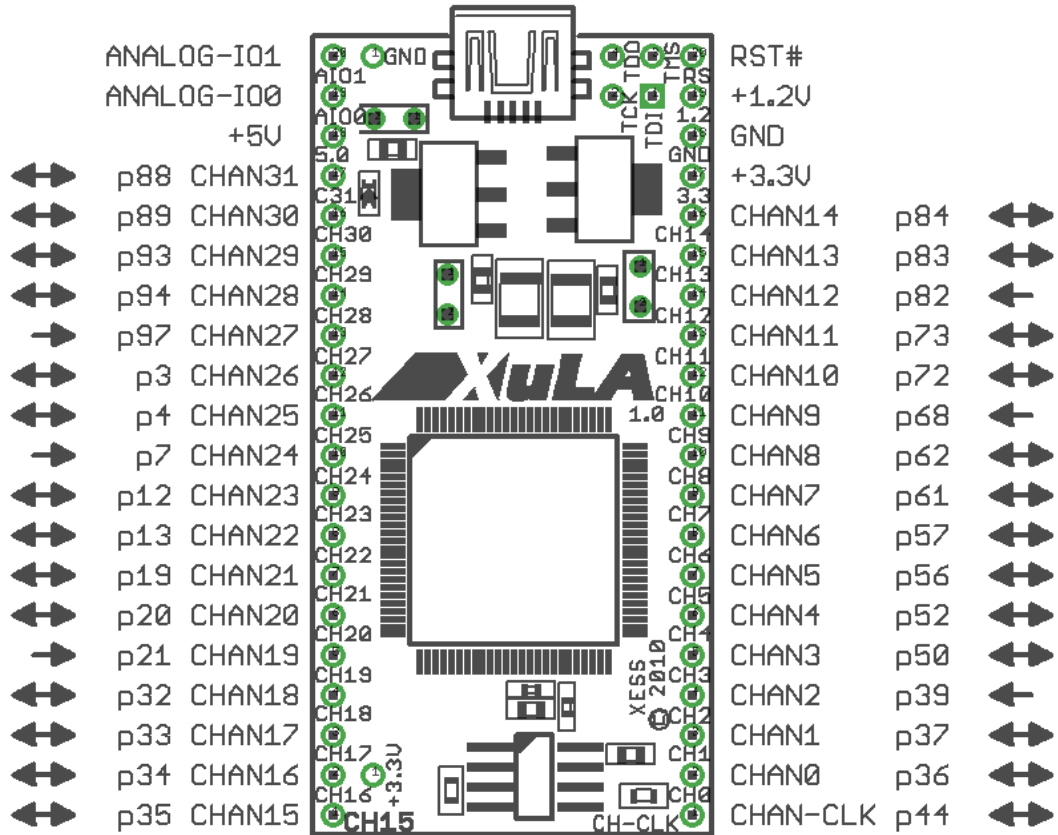
Due to the limited I/O of the FPGA package, the Flash and SDRAM chips share some of the FPGA pins. In combination with the SDRAM being permanently enabled (its chip-select is always asserted), this means that the Flash and SDRAM cannot be used simultaneously. Therefore, the following events occur when the XuLA Board comes on:

1. The microcontroller puts its FLASH-DISABLE output into a high impedance state and pulses the FPGA's PROG# pin to initiate the configuration from the Flash.
2. The FPGA lowers its CSO_B pin to enable the Flash, and then uses its CCLK, MOSI and MISO pins to read the stored bitstream. Because the SDRAM is not active during the FPGA's configuration (SDRAM-CLK is inactive since the FPGA is not configured), it doesn't matter if the SDRAM address and control pins are toggling.
3. Once the Flash delivers its bitstream, the FPGA's DONE pin is asserted. The microcontroller senses this and drives its output pin high to de-assert the FLASH-DISABLE signal, thus turning the Flash off. The FPGA can't access the Flash any more, but it can access the SDRAM without affecting the Flash.

It is possible to set a flag that prevents the microcontroller from disabling the Flash after the FPGA is configured ([as described here](#)). Then the logic design loaded into the FPGA can access the Flash, but it can no longer use the SDRAM (the toggling of the SDRAM address and control pins might inadvertently corrupt the Flash contents).

Prototyping Header

The prototyping header connects the FPGA I/O, microcontroller I/O and the power/GND planes to external circuitry. The signals attached to the header pins are shown below.

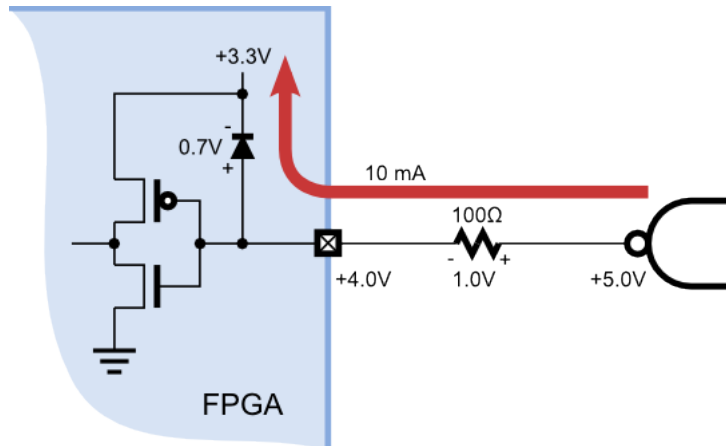


XuLA Header Pin Functions	
CHAN0 – CHAN31	These pins connect directly to the FPGA I/O pins. Channels 2, 9, 12, 19, 24 and 27 are input-only to the FPGA. These pins are not 5V-tolerant (see below).
CHAN-CLK	This is a direct connection to a global clock pin of the FPGA. It can also be used as a general-purpose I/O pin. This pin is not 5V-tolerant (see below).
+5V, +1.2V, +3.3V, GND	These pins connect to the voltage supply planes of the XuLA Board as depicted in this figure .
RST#	This pin connects to the microcontroller reset pin. Pulling it to ground and releasing it will cause the board to reconfigure itself.
ANALOG-IO0, ANALOG-IO1	These pins connect to the analog-to-digital converter in the microcontroller. ANALOG-IO0 can also output an analog voltage with a limited range and precision. Both pins are also useable as standard digital I/O pins.

5V Tolerance Issues

The CHAN* and CHAN-CLK pins connect directly to the pins of the FPGA which typically use an I/O voltage of 3.3V. When driving the inputs of external 5V logic, you should check that their V_{IH} threshold is less than the V_{OH} of the XuLA outputs. (This is true for most 5V logic families.)

You must also take care not to exceed the input voltage rating of the FPGA pins when they are driven by external 5V logic outputs. A common technique for protecting the Xilinx FPGA pins is shown below. When presented with a voltage greater than $3.3V + 0.7V = 4.0V$, the protection diode built into the FPGA pin's circuitry conducts current and the excess voltage is dropped across the 100Ω resistor. This keeps the voltage directly on the FPGA pin from ever exceeding 4V, which is within tolerated limits. The resistor value should be set so the current through the protection diode does not exceed 10 mA.

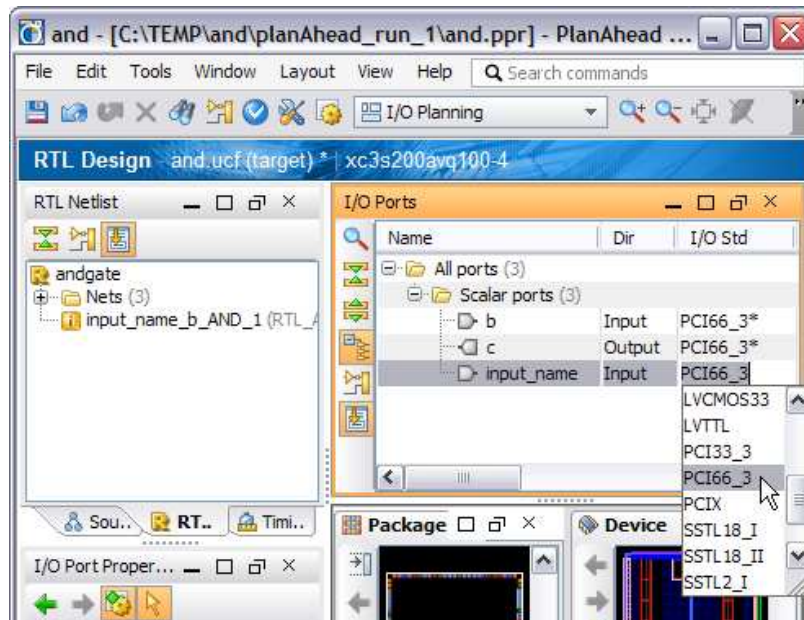


However, by default, the Xilinx Spartan-3A FPGA disables the protection diodes.

In order to enable these diodes, you can place the following text into your Xilinx ISE project constraint file for every I/O pin that will be connected to 5V logic:

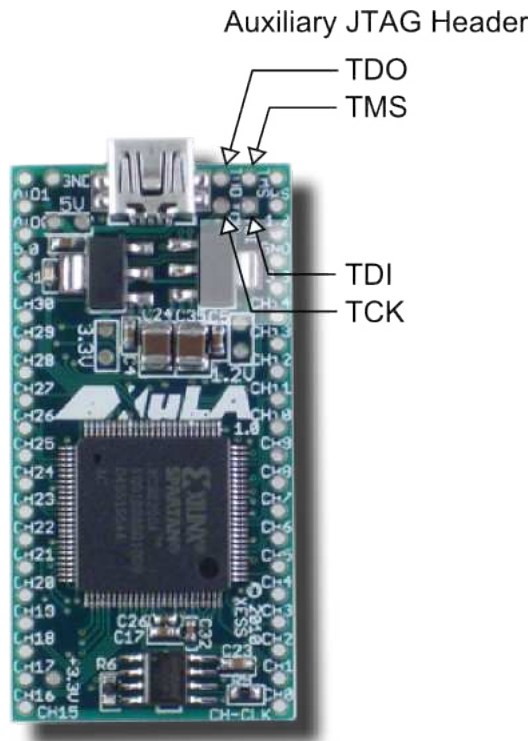
```
NET "input_name" IOSTANDARD = PCI66_3;
```

Or you can use the Xilinx PlanAhead tool to set the I/O standard for the pins like so:

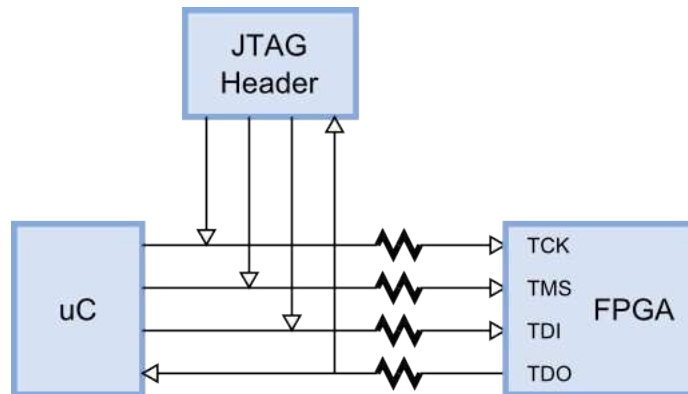


Auxiliary JTAG Header

This four-pin header (shown below) provides third-party JTAG cables with access to the JTAG pins of the FPGA.



The connections of the JTAG header to the rest of the XuLA circuitry are as follows.



The current limiting resistors prevent the microcontroller or JTAG cable from damaging the FPGA if they are using a higher supply voltage than the FPGA.

When using a third-party JTAG cable, the associated I/O pins of the microcontroller must be placed in a high-impedance state. The procedure for doing this is described [here](#).

A.1 Pin Connections

Net Name	FPGA	SDRAM	Prototyping Header	Flash	uC	Notes
A0	49	A0				
A1	48	A1				
A2	46	A2		DI		
A3	31	A3				
A4	30	A4				
A5	29	A5				
A6	28	A6				
A7	27	A7		CS#		
A8	23	A8				
A9	24	A9				
A10	51	A10		DO		
A11	25	A11				
BS	53	BS0, BS1		CLK		SDRAM bank-selects connected.
CAS#	60	CAS#				
CHAN-CLK	44		CHAN-CLK			Goes to FPGA GCLK pin.
CHAN0	36		CHAN0			
CHAN1	37		CHAN1			
CHAN2	39		CHAN2			Input-only to FPGA.
CHAN3	50		CHAN3			
CHAN4	52		CHAN4			
CHAN5	56		CHAN5			
CHAN6	57		CHAN6			
CHAN7	61		CHAN7			Goes to FPGA RHCLK pin.
CHAN8	62		CHAN8			Goes to FPGA RHCLK pin.
CHAN9	68		CHAN9			Input-only to FPGA.
CHAN10	72		CHAN10			
CHAN11	73		CHAN11			
CHAN12	82		CHAN12			Input-only to FPGA.
CHAN13	83		CHAN13			Goes to FPGA GCLK pin.
CHAN14	84		CHAN14			Goes to FPGA GCLK pin.
CHAN15	35		CHAN15			

Net Name	FPGA	SDRAM	Prototyping Header	Flash	uC	Notes
CHAN16	34		CHAN16			
CHAN17	33		CHAN17			
CHAN18	32		CHAN18			
CHAN19	21		CHAN19			Input-only to FPGA.
CHAN20	20		CHAN20			
CHAN21	19		CHAN21			
CHAN22	13		CHAN22			Goes to FPGA LHCLK pin.
CHAN23	12		CHAN23			Goes to FPGA LHCLK pin.
CHAN24	7		CHAN24			Input-only to FPGA.
CHAN25	4		CHAN25			
CHAN26	3		CHAN26			
CHAN27	97		CHAN27			Input-only to FPGA.
CHAN28	94		CHAN28			
CHAN29	93		CHAN29			
CHAN30	89		CHAN30			Goes to FPGA GCLK pin.
CHAN31	88		CHAN31			Goes to FPGA GCLK pin.
D0	90	D0				
D1	77	D1				
D2	78	D2				
D3	85	D3				
D4	86	D4				
D5	71	D5				
D6	70	D6				
D7	65	D7				
D8	16	D8				
D9	15	D9				
D10	10	D10				
D11	9	D11				
D12	6	D12				
D13	5	D13				
D14	99	D14				
D15	98	D15				
DONE	54				I/O	
FPGA-CLK	43				I/O	12MHz from uC.
PROG#	100				I/O	
RAS#	59	RAS#				
SDRAM-CLK	41	SDRAM-CLK				
SDRAM-CLKFB	40					SDRAM clock feedback.
TCK	76				I/O	
TDI	2				I/O	
TDO	75				I/O	
TMS	1				I/O	
WE#	64	WE#				

A.2 *Schematic*

