xstools Documentation

Release 0.1.31

XESS Corporation

February 07, 2017

Contents

Contents:

CHAPTER 1

XSTOOLs

XSTOOLs is a collection of Python classes for interfacing to XESS FPGA boards through a USB connection.

There are also several examples of command-line utilities that use these classes to perform operations on XESS boards.

- Free software: GPL V3 license
- Documentation: https://xstools.readthedocs.org.

1.1 Features

- Python package for accessing XuLA FPGA boards through a USB link.
- Command-line tools for configuring the FPGA, uploading/downloading the serial flash and SDRAM, and running diagnostics on the board.
- GUI tool that performs the same functions as the command-line tools.

CHAPTER 2

Installation

2.1 Install steps for Ubuntu/Debian

1. `sudo apt-get install python-pip`
2. `sudo pip install xstools --allow-all-external --allow-unverified intelhex`

2.2 Install steps for Windows

1. Install some version of python (e.g., www.activestate.com).
2. `easy_install xstools`, or ...
3. `pip install xstools --allow-all-external --allow-unverified intelhex`

CHAPTER 3

Usage

The Python XSTOOLs utilities are used to:

- Test XESS XuLA FPGA boards.
- Download configuration bitstream files to their FPGAs.
- Upload and download the onboard SDRAM.
- Program and read the serial flash.
- Program and verify the code in the USB interface microcontroller.
- Set flags that control behavior.

3.1 Command-Line Tools

3.1.1 xstest

usage: xstest.py [-h] [-u N] [-b BOARD_NAME] [-m] [-v]

Run self-test on an XESS board.

optional arguments:

-h, --help	show this help message and exit
-u N, --usb N	The USB port number for the XESS board. If you only have one board, then use 0.
-b BOARD_NAME, --board BOARD_NAME	***DEPRECATED*** The XESS board type (e.g., xula-200)
-m, --multiple	Run the self-test each time a board is detected on the USB port.
-v, --version	Print the version number of this program and exit.

Examples

To test a single board attached to a USB port:

```
xstest
```

To test two boards attached to USB ports:

```
xstest -u 0 xstest -u 1
```

To test a whole bunch of boards, one at a time:

xstest -m

(After a board is tested, remove it and attach another and the test will be run again.)

3.1.2 xsload

usage: xsload.py [-h] [–fpga FILE.BIT] [–flash FILE.HEX] [–ram FILE.HEX] [-u LOWER UPPER] [–usb N] [-b BOARD_NAME] [-v]

Program a bitstream file into the FPGA on an XESS board.

optional arguments:

-h, --help show this help message and exit

–fpga FILE.BIT The name of the bitstream file to load into the FPGA. –flash FILE.HEX The name of the file to down/upload to/from the serial

configuration flash.

–ram FILE.HEX The name of the file to down/upload to/from the RAM. -u LOWER UPPER, –upload LOWER UPPER

Upload from RAM or flash the data between the lower and upper addresses.

--usb N The USB port number for the XESS board. If you only have one board, then use 0.

-b BOARD_NAME, --board BOARD_NAME ***DEPRECATED*** The XESS board type (e.g., xula-200)

-v, --version Print the version number of this program and exit.

Examples

To configure the FPGA with a bitstream from a file:

xsload –fpga my_bitstream.bit

To download a file of data in Intel HEX format to the SDRAM:

xsload –ram my_down_data.hex

To upload 512 bytes of data between addresses 256 and 767 (inclusive) of the SDRAM to an Intel HEX file:

xsload –ram my_up_data.hex –upload 256 767

To download an Intel HEX file to the serial flash:

xsload –flash my_down_data.hex

To upload the first 1024 bytes of data from the serial flash to an Intel HEX file:

xsload –flash –upload 0 1023

3.1.3 xsflags

usage: xsflags.py [-h] [-u N] [-b BOARD_NAME] [-j {on,off}] [-f {on,off}] [-r READ] [-v]

Change configuration flags on an XESS board.

optional arguments:

-h, --help show this help message and exit

-u N, --usb N The USB port number for the XESS board. If you only have one board, then use 0.

-b BOARD_NAME, --board BOARD_NAME ***DEPRECATED*** The XESS board type (e.g., xula2-lx9)

-j {on,off}, –jtag {on,off} Turn the auxiliary JTAG port on or off.

-f {on,off}, –flash {on,off} Make the serial flash accessible to the FPGA. (Only applies to the XuLA-50 & XuLA-200 boards.)

-r, --read Read the flag settings from the XESS board.

-v, --version Print the version number of this program and exit.

Examples

To enable the auxiliary JTAG port of a XuLA or XuLA2 board:

xsflags –jtag on

After this, the other XSTOOLs utilities will no longer work because the USB port will no longer have access to the JTAG port of the FPGA. In order to re-enable USB access to the JTAG port, use the command:

xsflags –jtag off

The serial flash on the XuLA boards (but **not** the XuLA2 boards) is normally not accessible by the FPGA. This can be changed with the command:

xsflags –flash on

Once the flash is enabled, the FPGA can no longer reliably use the SDRAM. To disable the flash (and re-enable the SDRAM), use the command:

xsflags –flash off

The setting of the flash flag has no effect on the XuLA2 boards because the serial flash is always accessible to the FPGA.

To read the current settings of the flags from the board:

xsflags –read

3.1.4 xsusbprg

usage: xsusbprg.py [-h] [-f FILE.HEX] [-u N] [-b BOARD_NAME] [-m] [–verify] [-v]

Program a firmware hex file into the microcontroller on an XESS board.

optional arguments:

-h, --help show this help message and exit

-f FILE.HEX, –filename FILE.HEX The name of the firmware hex file.

-u N, --usb N The USB port number for the XESS board. If you only have one board, then use 0.

-b BOARD_NAME, --board BOARD_NAME The XESS board type (e.g., xula-200)

-m, --multiple	Program multiple boards each time a board is detected on the USB port.
--verify	Verify the microcontroller flash against the firmware hex file.
-v, --version	Print the version number of this program and exit.

Examples

To load the microcontroller with the latest firmware:

```
xsusbprg
```

To load the microcontroller with the contents of an Intel HEX file:

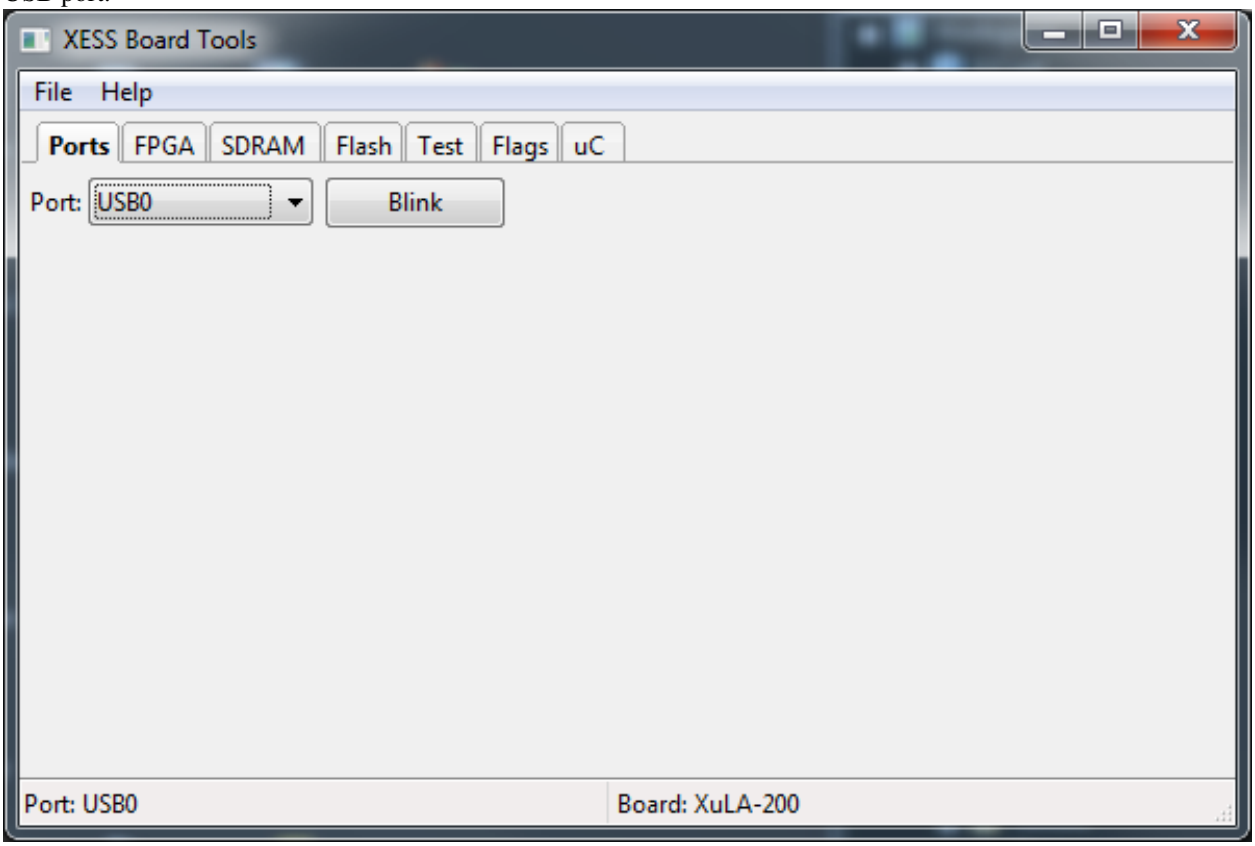
```
xsusbprg -f my_uc_program.hex
```

To verify the stored microcontroller program against a version stored in an Intel HEX file:

```
xsusbprg -f my_uc_program.hex –verify
```

3.2 GUI Tool

gxstools provides the same functions as the command-line tools, but with a GUI wrapper. The *Ports* tab allows you to select the USB port you will be working with. The port and the type of board attached to it are shown in the status bar at the bottom of the tab. Pressing the *Blink* button will cause the LED to blink on the board attached to the selected USB port.

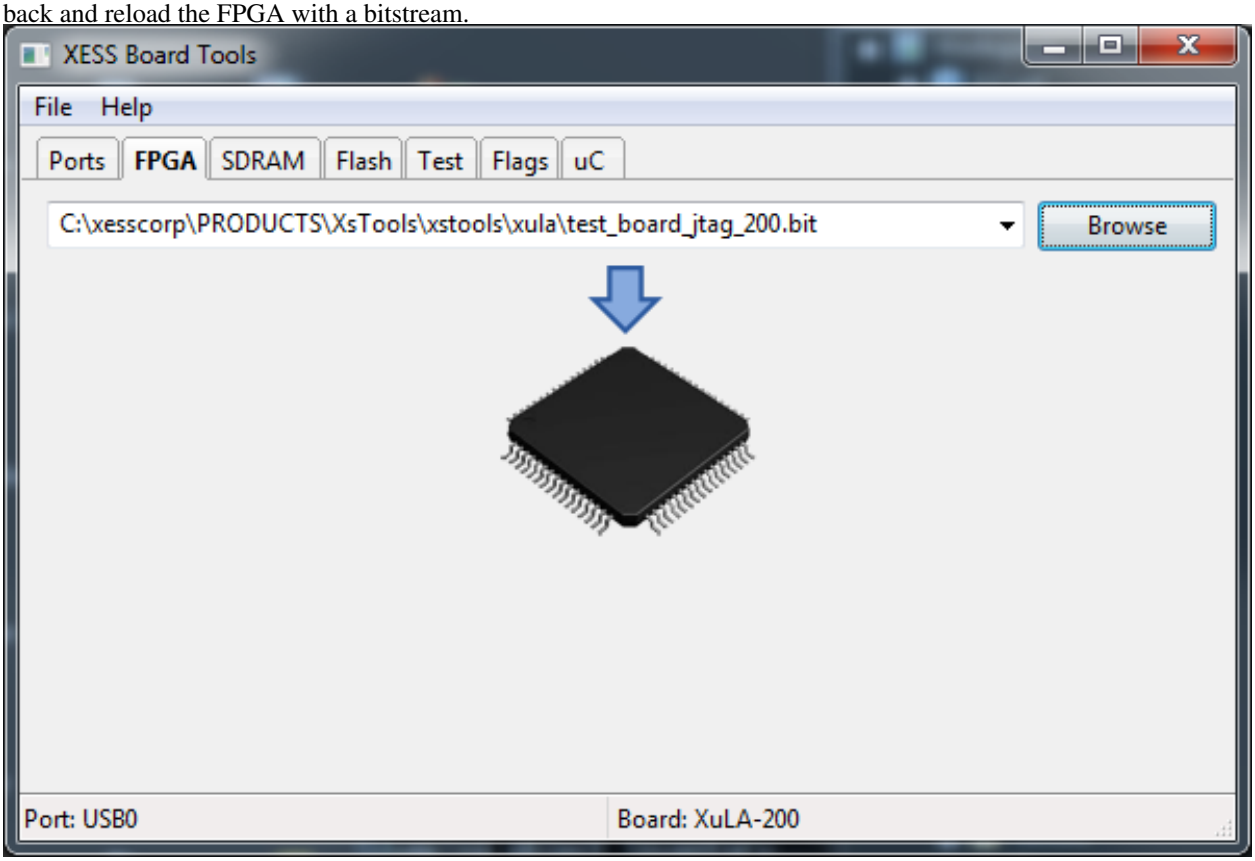


The *FPGA* tab is used to download configuration bitstream files into the FPGA. You can select a bitstream file by:

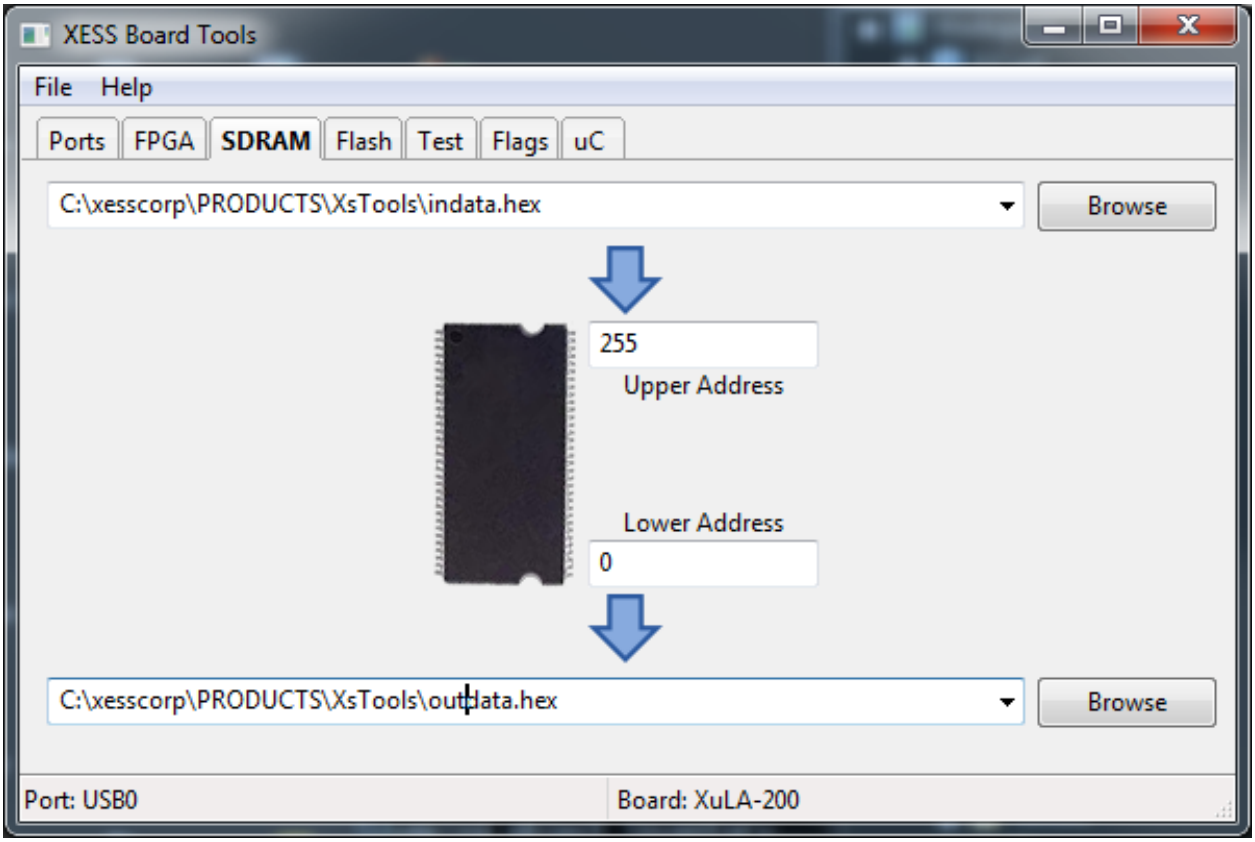
- Typing the filename directly into the text field.
- Using the *Browse* button to go to a directory and select a bitstream file.
- Using drag-and-drop to drop a file into the text field.

Once a file is entered in the text field, clicking on the downward-pointing arrow will load the bitstream in the file into the FPGA.

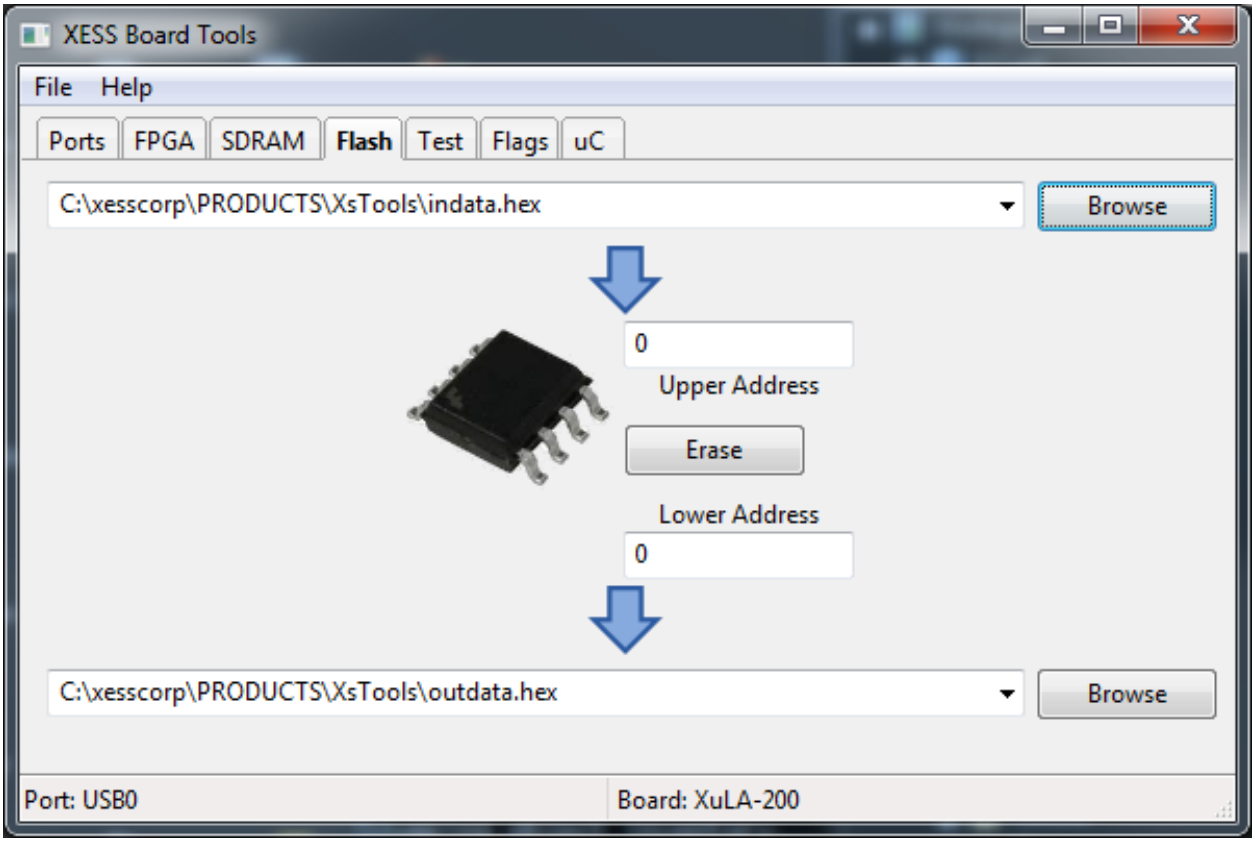
The text field maintains a history of all the files that have been downloaded to the FPGA. This makes it easy to go back and reload the FPGA with a bitstream.



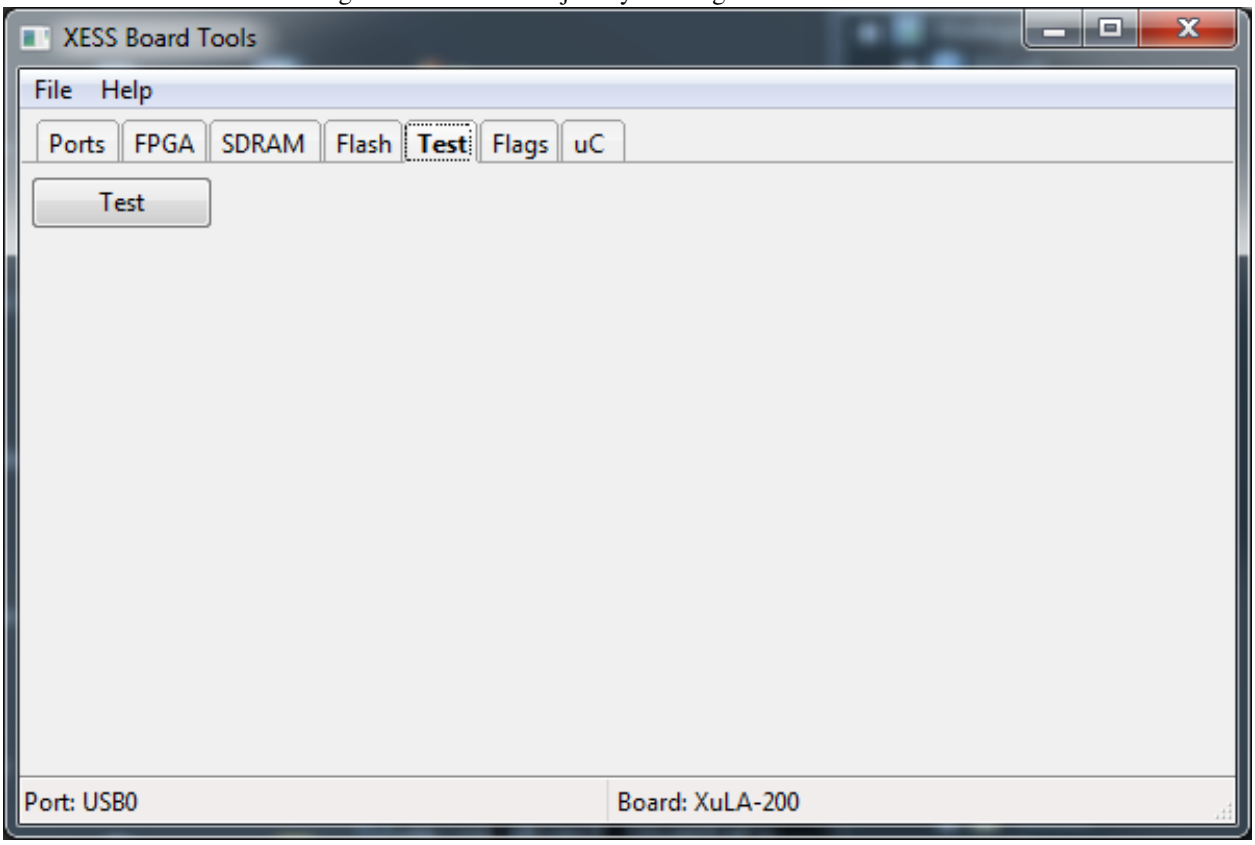
The *SDRAM* tab lets you download and upload data to/from the SDRAM. The data in an Intel HEX file can be downloaded into the SDRAM by entering the file name into the upper text field and then clicking on the upper downward-pointing arrow. Similarly, data between the values typed into the *Upper Address* and *Lower Address* fields will be loaded into the file whose name is entered into the lower text field once the lower downward-pointing arrow is clicked.



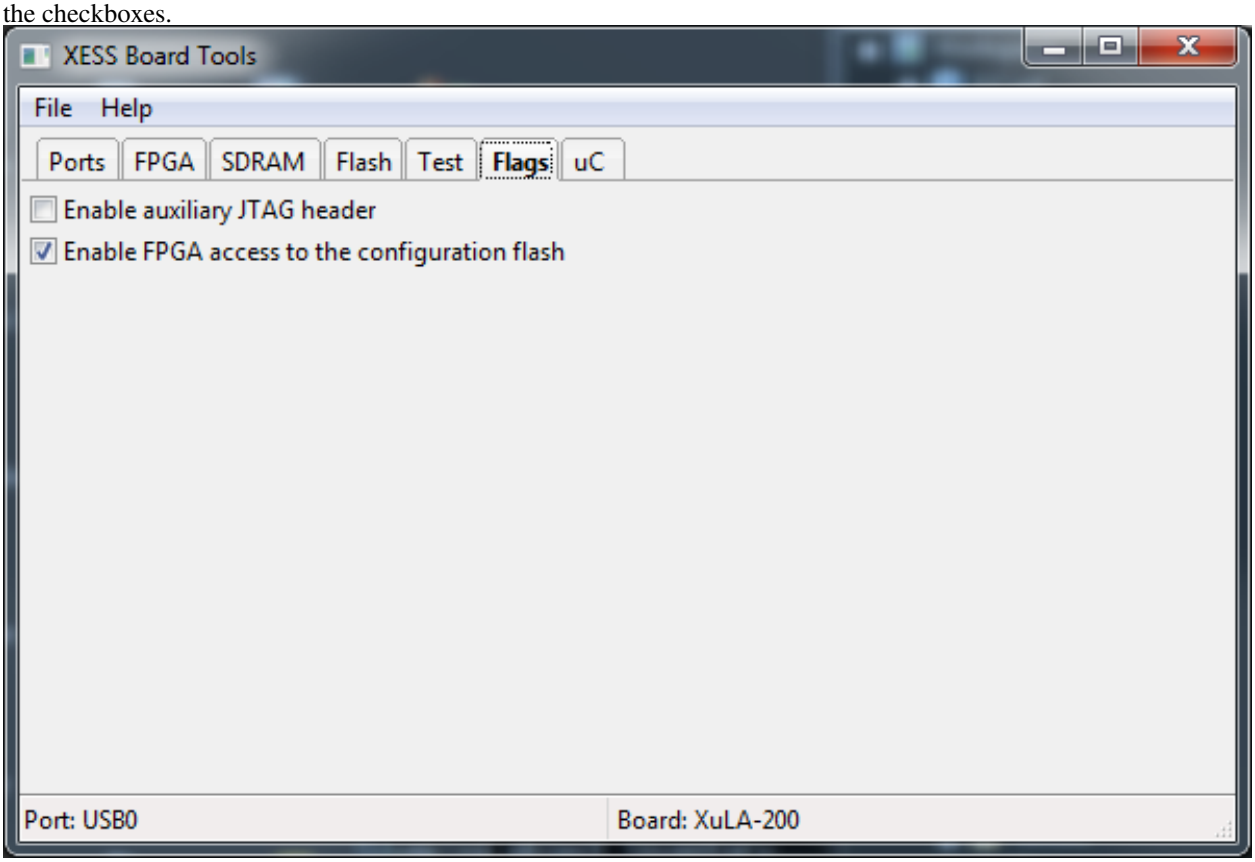
The serial flash can be handled in the same manner as the SDRAM by using the *Flash* tab. The only additional feature is that the flash can be loaded with an FPGA configuration bitstream by entering a *.bit* file into the upper text field.



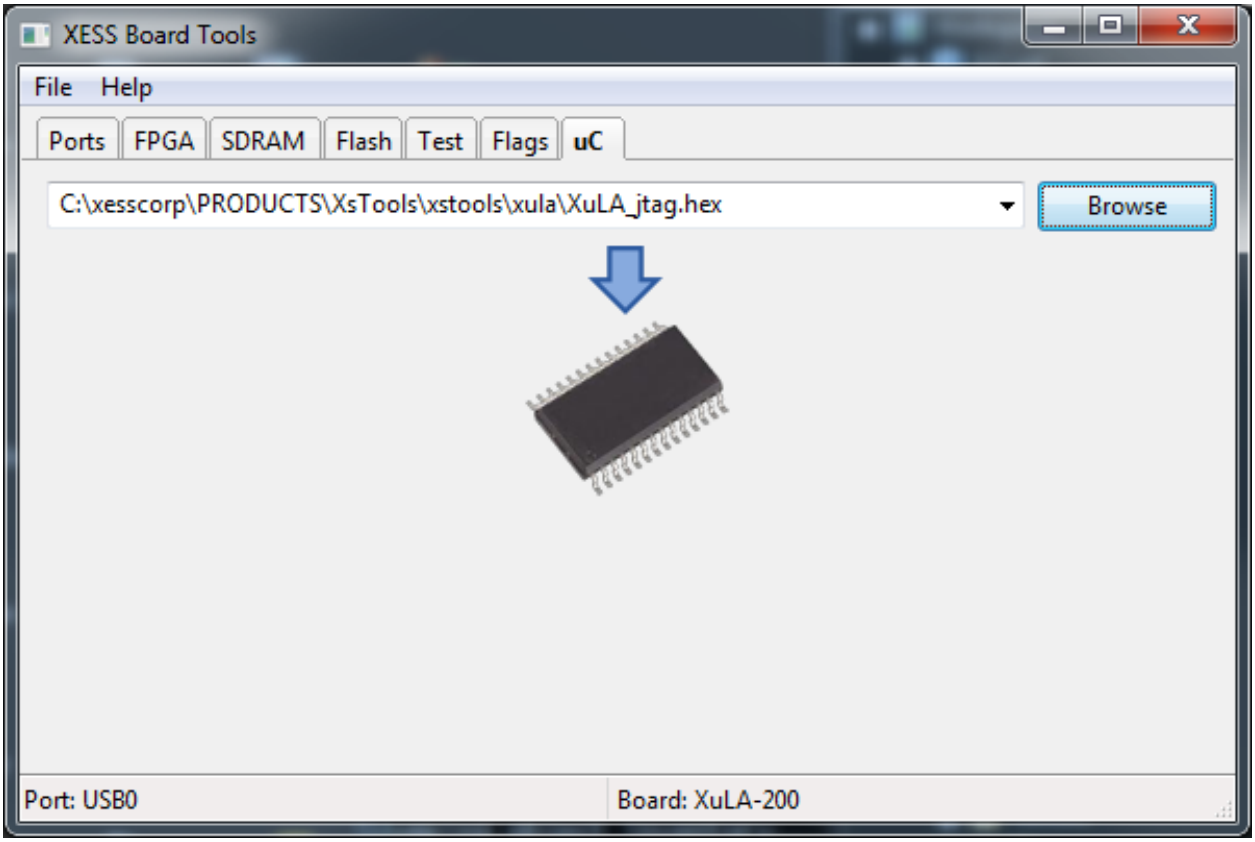
The *Test* tab is used to run a diagnostic on the board just by clicking the *Test* button



The *Flags* tab lets you set the various flags for the board. The current settings for the flags are indicated by the state of the checkboxes.



The microcontroller can be reprogrammed by entering the name of an Intel HEX file into the text field of the *uC* tab and clicking on the downward-pointing arrow.


XESS Board Tools
File
Help
Ports
FPGA
SDRAM
Flash
Test
Flags
uC
C:\xesscorp\PRODUCTS\XsTools\xstools\xula\XuLA_jtag.hex
Browse
Port: USB0
Board: XuLA-200

CHAPTER 4

Contributing

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

You can contribute in many ways:

4.1 Types of Contributions

4.1.1 Report Bugs

Report bugs at https://github.com/xesscorp/xsconnect/issues.

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

4.1.2 Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with "bug" is open to whoever wants to implement it.

4.1.3 Implement Features

Look through the GitHub issues for features. Anything tagged with "feature" is open to whoever wants to implement it.

4.1.4 Write Documentation

xsconnect could always use more documentation, whether as part of the official xsconnect docs, in docstrings, or even on the web in blog posts, articles, and such.

4.1.5 Submit Feedback

The best way to send feedback is to file an issue at https://github.com/xesscorp/xsconnect/issues.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

4.2 Get Started!

Ready to contribute? Here's how to set up *xsconnect* for local development.

1. Fork the *xsconnect* repo on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:your_name_here/xsconnect.git
```

3. Install your local copy into a virtualenv. Assuming you have virtualenvwrapper installed, this is how you set up your fork for local development:

```
$ mkvirtualenv xsconnect
$ cd xsconnect/
$ python setup.py develop
```

4. Create a branch for local development:

```
$ git checkout -b name-of-your-bugfix-or-feature
```

   Now you can make your changes locally.

5. When you're done making changes, check that your changes pass flake8 and the tests, including testing other Python versions with tox:

```
$ flake8 xsconnect tests
$ python setup.py test
$ tox
```

   To get flake8 and tox, just pip install them into your virtualenv.

6. Commit your changes and push your branch to GitHub:

```
$ git add .
$ git commit -m "Your detailed description of your changes."
$ git push origin name-of-your-bugfix-or-feature
```

7. Submit a pull request through the GitHub website.

4.3 Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests.
2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the list in README.rst.
3. The pull request should work for Python 2.6, 2.7, 3.3, and 3.4, and for PyPy. Check https://travis-ci.org/xesscorp/xsconnect/pull_requests and make sure that the tests pass for all supported Python versions.

4.4 Tips

To run a subset of tests:

```
$ python -m unittest tests.test_xsconnect
```

CHAPTER 5

Credits

5.1 Development Lead

- XESS Corporation <info@xess.com>

5.2 Contributors

- Dave Vandenbout wrote the original C++ version of XSTOOLs and the majority of the Python version.
- John Bowman wrote a Python version of xsload. Hector Peraza modified the python code to eliminate some problems and make FPGA configuration via JTAG conform to accepted practice. Dave took ideas and bits from Hector's code and integrated them into this package.
- Al Neissner wrote a Python version of xsusbprg and bits of his code are used in this package.
- Alireza Moini added the methods for reading voltages from the XuLA board analog I/O pins. Dave modified these to output floating-point values.

CHAPTER 6

History

v0.1.31 (2016-03-09)

- Sped-up memory reads & writes from/to the SDRAM.

CHAPTER 8

v0.1.30 (2015-09-02)

- Removed include in README.rst that caused an error generating the documentation.

CHAPTER 9

v0.1.29 (2015-08-10)

- Specifying the board model on the CLI utilities is now case-insensitive.

CHAPTER 10

v0.1.28 (2015-08-09)

- Now supports pyusb versions 1.0.0a and 1.0.0b.
- The utilities only connect to the USB port when they are actively executing some function for the attached XESS board. This allows other utilities to access the board.
- Added drag-and-drop capability for selecting bitstream and hex files.
- A history of bitstream and hex files is maintained.
- Exceptions caused by pyusb on program termination are now caught and filtered out.

CHAPTER 11

v0.1.27 (2015-07-31)

- Removed scripts in bin directory and placed them in the xstools directory.
- Added entrypoints to generate executables for the XSTOOLs scripts.

CHAPTER 12

v0.1.26 (2015-06-06)

- Changed distribution to use readthedocs for documentation.

CHAPTER 13

v0.1.25 (2015-06-05)

- Fixed problem where flash and SDRAM couldn't be accessed in gxstools because I appended 'bitstream' to the attribute names.

CHAPTER 14

v0.1.24 (2015-03-16)

- Modified usb2serial.py to prevent accidental triggering of serial BREAK.

CHAPTER 15

v0.1.23 (2015-02-03)

- Modified usb2serial.py and xscomm.py to support new serial BREAK command.
- Added .cmd file to initiate each XSTOOLs script in a Windows command window.

CHAPTER 16

v0.1.22 (2015-01-26)

- Modified usb2serial.py to support non-ZPUino use of the USB-to-serial server.

CHAPTER 17

v0.1.21 (2015-01-07)

- Modified setup.py so pyusb < 1.0.0b1 is installed as a dependency since the newer 1.0.0bX libraries cause a problem with finding XESS boards on USB ports.

CHAPTER 18

V0.1.20 (2014-12-17)

- Modified setup.py to include microcontroller firmware hex files for the XESS boards.

CHAPTER 19

V0.1.19 (2014-12-12)

- Modified setup.py so pyusb <= 1.0.0b1 is installed as a dependency since the newer 1.0.0b2 library causes a problem with finding XESS boards on USB ports.

CHAPTER 20

v0.1.18 (2014-12-10)

- Fixed query for XESS USB devices which failed for some USB libraries.

CHAPTER 21

v0.1.17 (2014-11-06)

- Fixed handling of XuLA/XuLA2 boards with old firmware or with the USB-to-JTAG path disabled.

CHAPTER 22

v0.1.16 (2014-10-27)

- Added command-line and GUI methods for setting/getting flags in XuLA/XuLA2 boards.
- Enabled loading of .bit files into serial configuration flash via gxstools.
- Updated hex file to newest version of the PIC 18F14K50 firmware.
- Added USB-to-serial bridge server between XuLA board and virtual comm port on PC.

CHAPTER 23

v0.1.15 (2014-05-16)

- Added support for upload/download of signed integers to memio methods.

CHAPTER 24

v0.1.14 (2014-05-05)

- Fixed FPGA bitstreams to remove errors during SDRAM upload/download.

CHAPTER 25

v0.1.13 (2014-04-09)

- Add bidirectional communication channel between host and FPGA: xscomm.py.

CHAPTER 26

v0.1.12 (2014-02-03)

- Added graphical front-end to XSTOOLs: gxstools.py.

CHAPTER 27

v0.1.11 (2014-01-03)

- Fixed bit direction for checking status bits in xsi2c.py.

CHAPTER 28

v0.1.10 (2013-11-20)

- Added support for XuLA2-LX9 board.

CHAPTER 29

v0.1.9 (2013-05-15)

- Added ability to load Xilinx bitstream files directly into serial configuration flash.
- Fixed byte order of addresses sent to the W25X serial flash.

CHAPTER 30

v0.1.8 (2013-05-14)

- Fixed FlashDev class so address bounds could not go outside the min/max addresses for the device.

CHAPTER 31

v0.1.7 (2013-05-11)

- Added FlashDevice class for reading/writing flash memory devices.
- Made Pic18f14k50 class inherit from the FlashDevice class for flash read/write operations.
- Added routines for reading/writing serial configuration flash on the XuLA and XuLA2 boards.
- Extended xsload.py to enable serial flash uploading and downloading.

CHAPTER 32

v0.1.6 (2013-04-30)

- Fixed xsusbprg.py so it works under linux.
- Fixed USB read/write timeouts so they are dependent upon the amount of data transferred.
- Replaced exit() with sys.exit() in scripts.

CHAPTER 33

v0.1.5 (2013-04-19)

- Added XuLA firmware .hex files for use with xsusbprg.py.
- Fixed xsusbprg.py so it would upgrade XuLA board firmware by default.
- All user-accessible scripts now use xstools_defs.py to get a unified version #.
- Added .rules file for USB connections to XESS boards.

CHAPTER 34

v0.1.4 (2013-04-01)

- Replaced bitarray module with pure-Python bitstring module.

CHAPTER 35

v0.1.3 (2013-02-15)

- Fixed so multiple XsUsb objects can share a single USB link to access an XESS board.

CHAPTER 36

v0.1.2 (2013-02-14)

- Changed CR-LF EOL in .py files to LF EOL so linux wouldn't barf.

CHAPTER 37

v0.1.1 (2013-01-23)

- Use pypubsub instead of wxpython for publish/subscribe communications.

CHAPTER 38

v0.1.0 (2013-01-06)

- Initial release.

CHAPTER 39

Indices and tables

- genindex
- modindex
- search