Make Xilinx FPGAs from Powerview and XDM for EE151

Kai-Ti Huang and Yucheng Wang
modified by John Board for EE151
updated by Sterling Babcock
Version 1.3

October 11, 1994

Abstract

This document illustrates how to use Viewlogic's Powerview and Xilinx's XACT design system on the EE Sun machines to make Xilinx FPGA files and how to program the FPGA's on the Generic Reusable Modules (GERM's) via the PC's in room 02G. This tutorial is available via ftp at:
`ftp://ftp.ee.duke.edu/users/jdsb/Powerview_to_FPGA.ps/`
or at Duke University at:
`/opt/digital/share/tutorials/Powerview_to_FPGA`.

1 Setting Up Your Working Environment and create new project

This tutorial assumes that you are already familiar with designing and simulating circuits with Viewdraw and Viewsim of ViewLogic's Powerview. If you are not familiar with Powerview, please read the powerview tutorial which is located at `/opt/digital/share/tutorials/powerview_tutorial`. The powerview tutorial is also available via ftp at:
`ftp://ftp.ee.duke.edu/users/jdsb/powerview_tutorial.ps/`

1.1 Setting up environment variables and path

Powerview runs on all departmental Sun4 workstations including new dsil and datc clusters. But XDM (XACT Design Manager) currently runs only on Sun4's with SunOS 4.x.x like `jefferson`. You can design your circuits on local dsil machines and remote login to `jefferson` to use XDM, or you can run Powerview from `jefferson` as well. We are working on getting XDM to work on the Solaris 2 Sun machines.

Before you start a new Powerview–FPGA project, add following lines to your `.cshrc` file.

```
setenv XACT /opt/digital/tools2/xact
setenv WDIR /opt/digital/powerview/standard
set path=($path /opt/digital/bin $XACT/bin/sparc $XACT/workview)
```

After you have done this, type

```
source .cshrc
```

to update your environment. Next time, you will have this new environment ready to use when you login.

1.2 Create a project for Xilinx

Get into your powerview directory, (`cd ~/powerview`). Copy the example directory into your powerview directory:

```
cp -r /opt/digital/share/tutorials/Powerview_to_FPGA/fpga .
```

Note the trailing period indicating your present directory (`~/powerview`).

These are a few files that are important in helping Powerview find the proper libraries and directories to use when designing FPGA circuits. It includes:

- viewdraw.ini - A list of FPGA libraries for Powerview. Advanced users may note that the default viewdraw.ini will work with FPGAs. This viewdraw.ini makes only the necessary libraries available for simplicity.
- xdm.pro - Defaults for the Xilinx software.
- sch/ee151_template - A powerview schematic for EE151
- sch/germ_ht_template - A powerview schematic for using the GERM

2 Design Your Circuit Using Viewdraw

After you set up your environment, you can enter Powerview. As always, you must be in your powerview directory, not your project directory, to use powerview. So make sure you are in your `~/powerview` directory. Create a new project called `~/powerview/fpga`. Note that this will use the directory you created above, and will not harm it.

You design your circuit using the schematic editor (Viewdraw) as usual. You can choose any component from the x3000 or mx3000 libraries, but these are the only libraries you should use to program Xilinx x3000 series FPGAs for this course.

The x3000 library contains basic components like simple combinational gates and D flip flops (DFF in the library, plus a number of variants with buffered inputs and/or outputs). The mx3000 library contains higher level components which have been predefined in terms of the simpler components - of particular interest are JK flip flops, but there are many other predefined components such as counters, registers, mux/demux, etc. The builtin library has pwr and gnd if you need a perminant digital value.

For the EE151 project, you should be able to simply re-enter the schematic you have already produced for the 74xx version of lab 5. The schematic `ee151_template` will get you started.

To use the GERM (Generic Reusable Module), you may want to start with the schematic `germ_ht_template`.

2.1 Adding Pads and Buffers

In Xilinx FPGAs, external inputs must come via an `ipad` and `ibuf` combination. An `ipad` is an input pad; `ibuf` is an input buffer. Any output goes to an `obuf` and then to an `opad`. An `obuf` is an output buffer; `opad` is an output pad. Fig. 1 shows the schematic of a simple combinational circuit with buffers and pads added; you must add these to each of your inputs and outputs.

2.2 Assigning Pin Locations for EE151

The signals from the "gray box" come to and from the FPGA on certain predefined pins. We must tell Viewdraw which pin to connect each input and output (`ipad` and `opad`) to. In the "Yugo" version of the Generic Reusable Modules (GERM's), not all possible signals are available; the signals that are present and their corresponding pins are in Table 1.

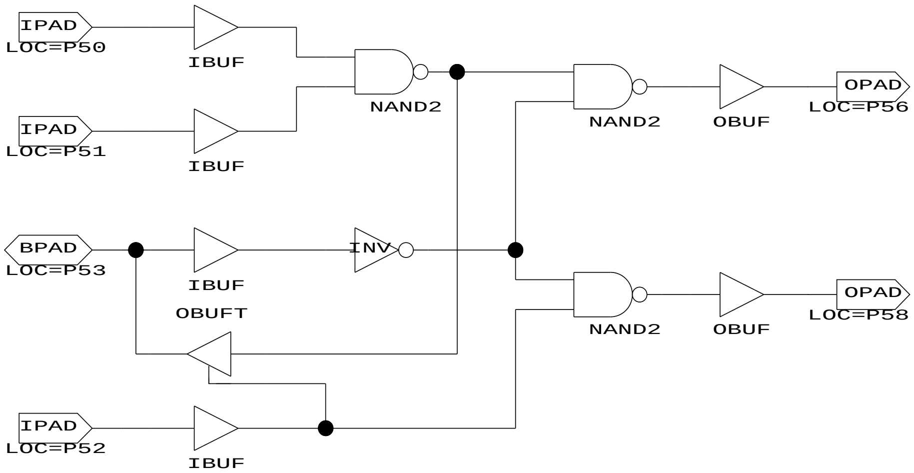

Figure 1: An Example Schematic

Table 1: Pin and signal assignment.

Pin assignment	Gray box signal
P13	B12-CLOCK
P48	C12-X3
P49	C11-X4
P50	C10-X5
P51	C1-Random
P52	B8-KYPDSTB
P53	B13-YCTR03
P59	B14-YCTR02
P63	B15-YCTR01
P68	B16-YCTR00
P69	A1-DATA
P56	A2-DATA
P58	A3-DATA
P60	A4-DATA
P62	A5-DATA
P65	A6-DATA
P67	A7-DATA
P70	A8-DATA
P72	A9-DATA

For this lab, you are certain to need the clock (P13), the random input (P51), and one or more of the output lines (P69, P56, etc.). To assign pin location for a pad, you have to select a pad first. Then go through the menus of **Change** → **Attr** → **Dialog** → **All**, or just push **Ctrl-A**. An Attributes window will popup like Fig. 2.

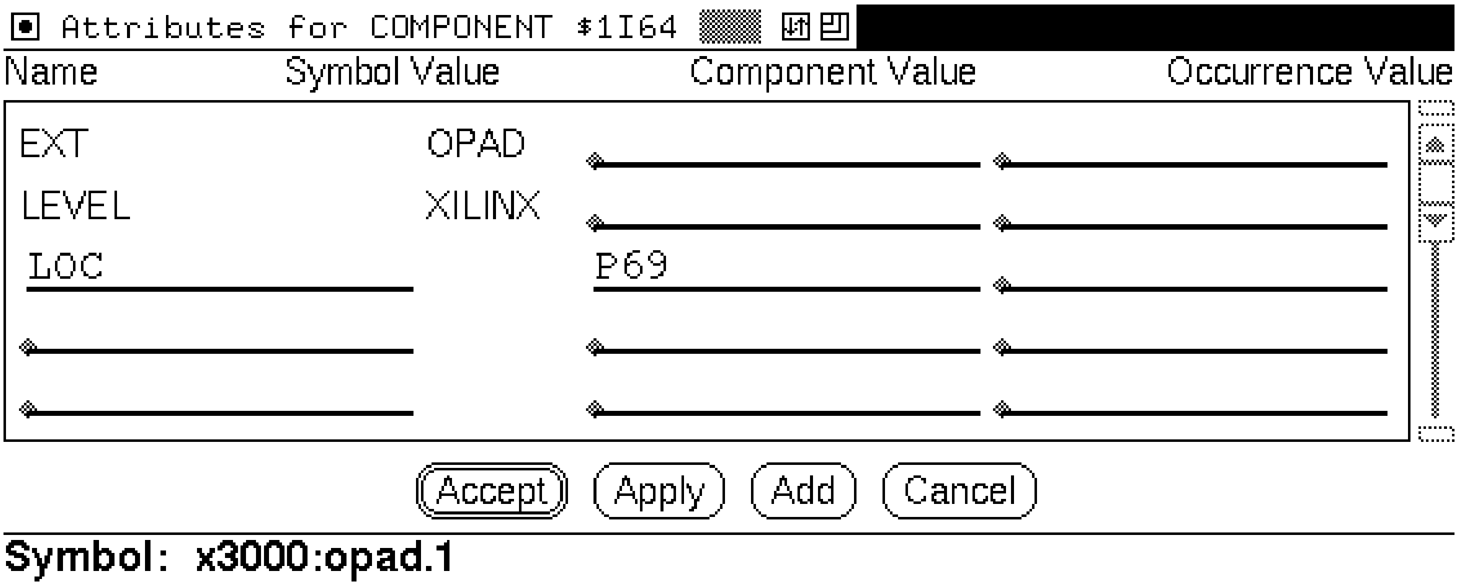

Figure 2: Attrabute Window of an Output Pad.

In all cases, you have to enter `LOC` on the third row of the first column of each pad. The component value of `LOC` is assigned the pin number from `P1` to `P84`. In figure 1, the inputs are assigned `P53`, `P59`, `P63`, and `P68`. The output pin is assigned `P69`. After entering the value of attribute of the pad, click on Accept to save the attribute.

After finish your design, save it before you leave Viewdraw.

3 Converting Powerview design to Xilinx bitmap by XDM

Make sure you are remotely logged on to `jefferson` and have directed X display output back to the machine you're actually sitting in front of. For example, if you are on `dsil4`, allow `jefferson` access by typing

```
xhost jefferson
```

then `rlogin jefferson` and type

```
setenv DISPLAY dsil4.ee.duke.edu:0
```

Go to your project directory (`~/powerview/fpga`) and run `XDM`. Note that `XDM` is different from `xdm` (lowercase, X Windows Manager), a completely different program! On SunOS, XDM is `/opt/digital/xact/bin/sparc/xdm`. On Solaris 2, XDM is `/opt/digital/share/xilinx/bin/sparc/xdm`. An XDM window (XACT Design Manager) will appear.

Choose **Translate** → **XMAKE** menu to use XMAKE to translate your powerview design to a Xilinx bitmap file. Click Done on the popup window. Then choose your design, for example, ee151_template.1. If there are no errors, XMAKE will automatically build a `ee151_template.bit` file for you. This is the file with the crosspoint settings for each possible switch in the FPGA; we will download it to the FPGA's via the PC's in our lab. If your `.bit` file was constructed successfully, log out and move down to the lab for the final steps. If it was not, look at `ee151_template.out` for errors.

4 Programming FPGAs by a logic interface

For EE151, a GERM has been connected to one of the gray boxes in 02G by a special interface cable. There is also a programming cable attached to the GERM which connects it to the PC nearest the gray box.

First, you have to use `ftp` to download the `.bit` file to the PC. Remember to change the file transfer mode to **binary** before you get the file. For example, from the DOS prompt on the PC, type

```
ftp ee
```

Log on with your usual username and password. Type

```
binary
```

to get into binary transfer mode. Use `cd` to get to the directory where your `.bit` file is located (probably `cd powerview/fpga`). Then type

```
get ee151\_template.bit
```

Leave `ftp` with `quit`.

After you get the file, power up the logic analyzer and the gray box; the FPGA gets power from the gray box. Then, for the download cable, on the PC type

```
\xact\download -port lpt1 ee151\_template.bit
```

For the xchecker cable, use:

```
xchecker -port lpt1 ee151\_template.bit
```

Note that `xchecker` is also available on sun machines whose port has been changed to RS-232. See your administrator if you need this. In this case, use `-port /dev/ttya`.
The bitmap file will be transferred to the board and it will begin to work immediately - the output of the logic analyzer should show the desired behavior. Have one of the TA's or me sign off that you have completed this part; there is no writeup due on the FPGA work.

5 References

1. XACT Viewlogic Interface User Guide: Use Viewdraw and Viewsim to design and simulate FPGAs.
2. Xilinx User Guide and Tutorial: Design FPGAs by using various tools.
3. An Experimental Study in Subsystem Reusability Toward Rapid Prototyping of Microelectronic Systems: MS thesis, Brent Ward, 1993.