Full name: _____

Circuit design with configurable devices (FPGA)

1 Material

- Computer with Xilinx's ISE software installed.
- Digilent's Basys2 prototype board and documentation.
- Sample design files (lab kit).
- Files and documents can be downloaded from the course's web page.

2 Objectives

Learn how to implement a digital design described with a hardware description language into a configurable FPGA chip.

3 Pre-lab work

1. Get familiar with the Basys2 prototype board.
2. Voter circuit design (as done in the assignments)
3. Binary to 7-segment converter (as done in the assignments)

4 Lab work

4.1 Implementation and test of a three-input voter circuit

ISE project creation

- Download the “lab3_kit.zip” file from the web and extract it in your personal folder or any other folder you like.
- Start Xilinx's ISE software. The application is divided in 3 sections, left area is the project panel (now empty), right area is the main/input panel and below you have the console.
- Click on *File → New Project* and a wizard will open. Enter a name for the project (like “voter”). You may change the project folder, but the default option is OK. Click NEXT.
- Now enter FPGA details. For the BASYS2 board we have:
  - General Purpose
  - Family: Spartan3E
  - Device: XC3S100E
  - Package: CP132
  - Speed grade: -5
  - Design flow options are the defaults: XST synthesis, ISIM simulator, Verilog preferred language, etc.). Click NEXT.
  - Review the project summary and click FINISH.

Full name: _____

Add files to the project

- In order to add your previously created files to the project, right-click anywhere in hierarchy panel and choose "Add Source".
- Select the Verilog source file(s) in the navigation windows by navigating to the folder they were decompressed and OPEN them. You can select many files at a time if you wish (Ctrl+click). Then you have to tell ISE which files are for "implementation" or "simulation" (like test bench files) or both (“all”). Normally, ISE will auto-detect this so you only have to confirm.
- Now the hierarchy tree is showing the files, in the order they will be compiled. The order is automatically detected by ISE but can be changed manually (not recommended). In the hierarchy panel you can find two selectable lists: “Implementation” with all the files used to build the circuit, and “Simulation” with the files used during simulation. In our case, “Implementation” should include only one file with the circuit design and “Simulation” should also include the test-bench.
- If you double-click on a file, a new editor-tab will open and you can view or edit the file within the ISE environment.

Simulate the test bench in ISE (ISIM)

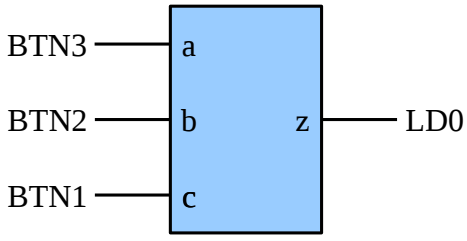
If you have added a test bench you can use the ISIM (ISE SIMulator) to simulate it:

- Click on "Simulation" view and select your test-bench's source file
- In the box below "Processes" you can open the "ISIM SIMulator" tree and execute "Simulate Behavioral Model" by double-clicking on it. In case of error, check the “Console” panel. You may have to check the code's syntax by editing the source files and run the simulation again.
- After a few seconds of processing a new window will open with the ISIM window. Find the zoom selection buttons and use them to plot the whole simulation time range. Check that the results are correct.
- When simulating, ISIM will stop automatically after a few microseconds. If you specified a longer simulation time in your test-bench, you can continue the simulation process by pressing the “Continue” button.

Addition of a UCF file

The FPGA chip in the Basys2 board has its pins connected to the various connectors and peripherals in the board: switches, buttons, LEDs, 7-segment display, etc. In order to implement your design into an FPGA you have to map the top level inputs and output of your design to the right pins in the FPGA device so that the inputs and outputs are connected to useful board peripherals. E.g. connect inputs to board's buttons and outputs to LEDs. The UCF file contains this mapping information and so you always have to include one in your project and edit it according to your needs and the FPGA board you're using.

For the voter circuit we will map inputs a, b and c to three push buttons in the board, and the output



Full name: _____

z to one of the LEDs.

- Add the provided voter UCF file to the project as you did for the Verilog sources.
- Open the UCF file by double-clicking on it in the hierarchy browser panel. This is a generic UCF file for the Basys2 board, with all the lines commented out. We will only uncomment and configure four lines, corresponding to the three inputs of the circuit and the single output:
  - First we will map the output signal “z” to one of the LEDs in the board. Find the line that corresponds to LED 0 in the board (signal name LD0):

    #NET "Led<0>" LOC = "M5" ; # Bank = 2, Signal name = LD0
  - Uncomment the line (delete the “#” at the beginning) and change the net name from “Led<0>” to the name of the output signal in our design (“z”):

    NET "z" LOC = "M5" ; # Bank = 2, Signal name = LD0
  - Now we map the inputs to three push buttons. Find the push buttons lines. E.g. BTN3:

    #NET "btn<3>" LOC = "A7"; # Bank = 1, Signal name = BTN3
  - Map BTN3 to input “a”:

    NET "a" LOC = "A7"; # Bank = 1, Signal name = BTN3
  - In the same way, map BTN2 to input “b” and BTN1 to input “c”.
- Save the UCF file.

Synthesis & Implementation

The “synthesis” and “implementation” processes can be quite complex. “Synthesis” is a process like “compilation” in software design: the Verilog code is converted into a digital circuit. “Implementation” selects the right components to build the circuit (mapping), assign this components to the FPGA processing resources (placing) and finally configures the internal connections in the FPGA so that all the components are wired correctly (routing). Fortunately ISE is a very powerful tool which can do all these steps almost automatically without user intervention in most cases.

- Select the top level source file in the implementation view and take a look at the process panel. There're "Synthetize XST", "Implement Design" and "Generate Programming File" actions. The three actions should be run one after the other but if you click on the last one ISE will execute all of them in order.
- Right-click on "Generate Programming File" and select “Run”. You will see an animated icon rolling which means there's an action going on, just wait for the process to complete. A green icon next to each action indicates everything's okay and a red cross icon that an error has happened.
- Check Console window to find errors and warning and other details about the complete process. In case of any error, solve it and rerun the programming file generation by selecting “Rerun” in the “Generate Programming File” context menu.

Programming the FPGA chip (from Impact -needs Dilent plugin-)

The generated programming file is in the bitstream (.bit) file in the project's folder. The bitstream file must be downloaded to the FPGA chip in the board to configure the device. Once the device is configured the circuit will start to run. The tool inside ISE that manages the transfer of the bitstream

Full name: _____

to the FPGA chip is called “iMPACT”.

- Be sure that Basys2 board is connect to an USB port with the cable included with the board and turned on before going on.
- Right-click on the “Configure Target Device” process and select “Run”. The iMPACT tool will open in a new window.
- In the “iMPACT Flows” panel (top-left) double-click on “Boundary scan”. A new boundary scan process is created in the main panel. Here we are going to configure how we are going to program the device (type of cable) and what is the device to be configured.
- Right-click on the main panel and select "Cable Setup ...". On the window check "Open Cable Plugin". In the selection box select or write down "digilent_plugin". Click OK.
- Right-click on the main panel again and select "Initialize Chain". Impact will now look for the devices connected to the USB cable and show them in a diagram. Click CANCEL on next dialogs until you're back in the IMPACT main window.

The Basys2 board has two programmable devices, the FPGA chip itself and a Flash memory. The bitstream file can be programmed in either of them. Programming the FPGA chip is fast, but the programming will be lost when the board is turn off. To make the programming “permanent” we have to store the configuration in the Flash memory. The FPGA chip will read the configuration from the Flash every time the circuit is turned on. Programming the Flash takes more time and the bitstream has to be converted to the right format before, so it is typically done only when the circuit has already been tested.

To program the FPGA with the bitstream file do:

- Right click on the "FPGA Spartan 3E" xc3s100e chip icon and select "Choose download file"
- Browse to the .bit file. Click OK. Right-Click on the FPGA image again and choose “Program”.
- After a few seconds the FPGA chip should be programmed accordingly to our design. Press the buttons and observe the LED to check that the circuit is working correctly. Congratulations!
- Close the Impact window.

Programming the FPGA chip (from Digilent's Adept command line tools)

Connect the Basys2 board to the computer with the USB cable. Turn on the board.

Open a terminal emulator and use the following commands.

Enumerate (discover) the connected boards:

```
$ djtgcfg enum
...
```

Initialize the board and list target devices:

```
$ djtgcfg init -d Basys2
...
```

Program the FPGA chip (device 0) with the .bit file generated by the Xilinx's tools.

Full name: _____

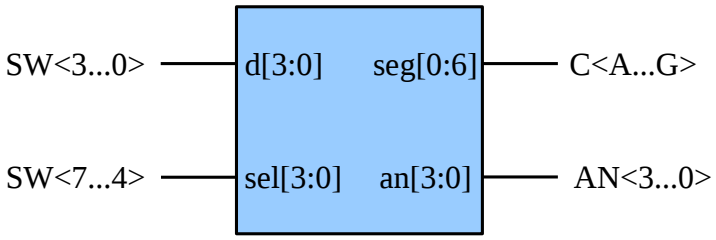
```
$ djtgcfg prog -d Basys2 -i 0 <path_to_bitstream_file>
...
```

Voter modification

Modify the UCF file to map the circuit's inputs to switches instead that buttons. That is, use SW0, SW1 and SW2 instead of BTN3, BTN2 and BTN1:

- Edit de UCF file inside ISE and save it.
- Right-click on the “Generate Programing File” and select “Rerun”.
- Configure the target device with the generated bitstream as explained before.

4.2 Implement and test a 7-segment converter



Project creation and UCF configuration

- Create a new project name “display” and add the files provided with the lab kit, including the UCF file.
- Modify the UCF file to make these mappings:
  - Input number (d) maps to SW3 to SW0. E.g.:

    NET "d<3>" LOC = "B4"; # Bank = 3, Signal name = SW3
  - Display selection bits (sel) map SW7 to SW4. E.g.:

    NET "sel<3>" LOC = "N3"; # Bank = 2, Signal name = SW7
  - Anode control output (an) maps to anode terminals in the board's display (AN3 to AN0). E.g.:

    NET "an<3>" LOC = "K14"; # Bank = 1, Signal name = AN3
  - 7-segment output (seg) maps to display LEDs CA to CG. E.g.:

    NET "seg<0>" LOC = "L14"; # Bank = 1, Signal name = CA

Project simulation (optional)

Simulate the project to check that the converter is working correctly.

Project implementation and test

Implement the project in the FPGA as we did for the voter. Use the switches and check that the display representation is correct.

Full name: _____

Display modification

- Extend the display module description to convert it into a hexadecimal to 7-segment converter.
- Rerun the synthesis process.
- Program the new design and test it. Is it right?