Fourth Lab Session

Before we start working, here's a quick reminder of the rules in effect while in the Embedded Systems Lab.

In-lab rules:

We have limited space, and you will be working with sensitive electronic components. Therefore:

- ***No food or drinks are allowed inside the lab.*** Be sure to eat properly before your session, you can step out for water if you need to. Notify your TA since he/she is the only person that can grant you access to the lab.
- ***No laptops, notebook computers, tablets, or other electronic gizmos*** allowed during the session.
- ***Bags and backpacks must be stored at the front or back of the room***.
- ***While in the lab, you're responsible for taking good care of all equipment***.
- ***Treat everyone else in the lab with respect and consideration***.
- ***You must listen and follow all instructions provided by your TA***.

How your work will be marked:

Your TA will observe your work during the three hour period.

- 15% of the grade for this lab is given by attending the session and working hard .
- 85% is given for completing your work. This includes the parts of the report due ***before*** the session begins. Your in-lab work (which will be tested by your TA), and this completed report.

You will work in pairs, but each student ***must complete and hand their own lab handout to the TA*** at the end of the session.

If you encounter any problems with the software or hardware, bring this to the attention of your TA immediately. If no solution can be found quickly, your TA may have to have you join another team for the duration of the session.

Lab 4 – Memory elements and sequential circuits

This lab is designed to allow you practice with the design and use of circuits that contain memory elements (flip-flops). Starting from the basic flip-flop circuit, you will build and package circuit modules. These modules will be put together at the end into a simple digital clock that displays hours and minutes using the FPGA's 7-segment displays.

By now, you should have some experience using the Quartus II software and programming the FPGA board. But if you run into problems, do not hesitate to ask your TA for help.

Learning Objectives:

You will put into practice your knowledge of flip-flops and build a working flip-flop module.

You will design a simple counter using memory elements.

You will learn to think about the circuit's behaviour over time. You will be using clock signals and reset/clear signals that change over time and have an effect on the circuit's behaviour.

You will put together the knowledge you acquired during the digital circuits part of the course and use it to build a simple (but very common) digital appliance.

Skills Developed:

Design and use of simple memory elements

Handling of clock signals. Interaction between memory circuits and digital logic

Creation of re-usable circuit modules

Reference material:

This handout.

The Quartus II handbook (*http://www.altera.com/literature/hb/qts/qts_qii51008.pdf*)

The DE2-115 user manual (inside the Altera System CD directory)

Your lecture notes on flip-flops and circuits that use them

Student 1 Name (last, first):

Student 1 Number:

Sturent 2 Name (last, first):

Student 2 Number:

I understand that I am responsible for handling the equipment carefully and preserving it in working order.

Signature 1: _____

Signature 2: _____

You must complete all parts marked as 'prior to the lab session' before your appointed lab section starts.

0) Carefully unpack the FPGA board and place is safely on the desk. Connect the power cable to the board and a desk outlet, connect the USB cable to the board's USB 'blaster' port and to a computer USB terminal. ***DO NOT POWER UP THE BOARD yet.***

Part 1 – Building a memory unit – The J-K flip flop.

Download the starter project from the course website. Unpack it onto the desktop. It will generate a directory called Lab_4.

1) Open the Quartus II software. Close the initial information box. Choose to open an existing project and select the .qpf file from the Lab_4 directory.

   Note that the project already contains 3 ***currently empty*** files. You will be filling-in these files during the lab session.

To be completed prior to the lab session:

Show in the space below the diagram for a ***J-K flip flop.*** All gates that are required to fully implement the flip-flop in the schematic diagram should be included in your drawing.

2) In the ***JK_flip_flop.bdf*** schematic ***implement your design for a J-K flip-flop***. Check every step of your design and implementation. Do not forget to add (and name!) input/output pins, but ***DO NOT CREATE AN OUTPUT FOR Q'.***

When you have completed your implementation, ***choose Processing → Analyze current file*** to check that your design is correct.

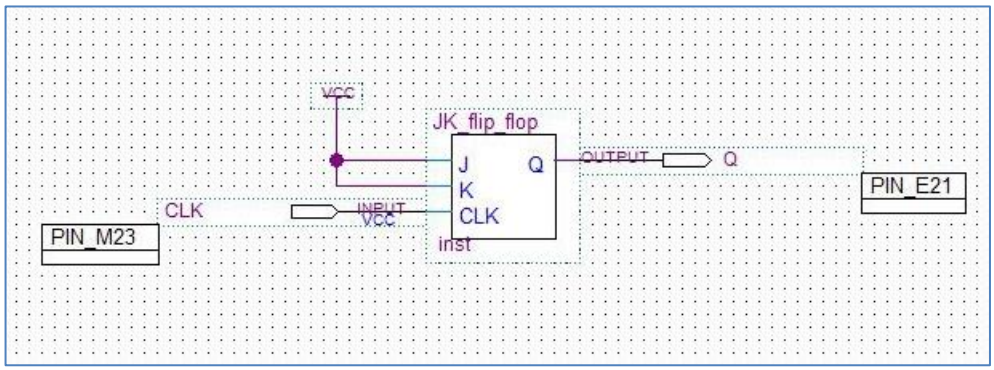
Once you have created a correct design, choose ***File → Create/Update → Create Symbol Files for Current File***.

This will generate a file called ***JK_flip_flop.bsf***. This is a fully packaged module containing your circuit design that can be used as a unit to build more complex circuits!

3) Test the J-K flip flop.

In the file corresponding to ***Lab_4.qpf*** (the main file for your project) implement the following circuit:

You can select your J-K flip-flop circuit just like you would a regular logic gate, except it will be stored in your project directory instead of the Altera libraries!



Note that you are already provided with pin numbers for the single input and output. The ***clock input*** is mapped to ***Key0*** and the ***output*** is mapped to ***LEDG0***. ***Compile your work and program the FPGA board.***

Test the functionality of your circuit:

What type of flip-flop is implemented by the above circuit? _____

Flip-flop functionality verified by TA in lab:

Part 2 – Implementing a 4-bit counter with clear

1) To be completed prior to the lab session:

Show in the space below the diagram for a ***4-bit counter*** which should have the following characteristics:

- It counts up with each clock pulse
- it has 2 inputs: ***clock_in*** and ***clear***
- it has 4 outputs, one for each of the flip-flops in the 4-bit counter
- If the ***clear*** signal is set, the counter must reset to ***0000*** on the next clock pulse

Your notes already contain the diagram for a counter, but that counter does not have a ***clear*** input.

Be sure to show your TA your design if you are not certain that it works properly. You may have to alter your design at the lab, but you must have a prototype ready before the session starts.

Important note:

Implementing a ***clear*** function with flip-flops as the ones we have been working with is difficult. So, ***do not use your own JK Flip-Flops from the previous part.*** Instead, use Altera's ***JKFF*** provided as part of the primitives (look under ***storage***).

These flip-flops have standard ***J,K, and CLK*** inputs, but also have a ***CLRN*** and ***PRN*** inputs. ***CLRN*** is the ***clear*** input we need, and is ***active low*** (i.e. the flip-flop is reset to ***0*** regardless of J and K when ***CLRN=0***). ***PRN*** is a ***preset*** signal which sets the flip-flop to ***1*** regardless of J and K. ***PRN*** is also ***active low***.

Finally, the Altera JKFFs are ***positive edge triggered*** which means you will have to change a bit (and carefully) the design of the counter, else it will count ***downward!***

2) Implement your circuit for the 4-bit counter in the file ***counter_4bit.bdf***

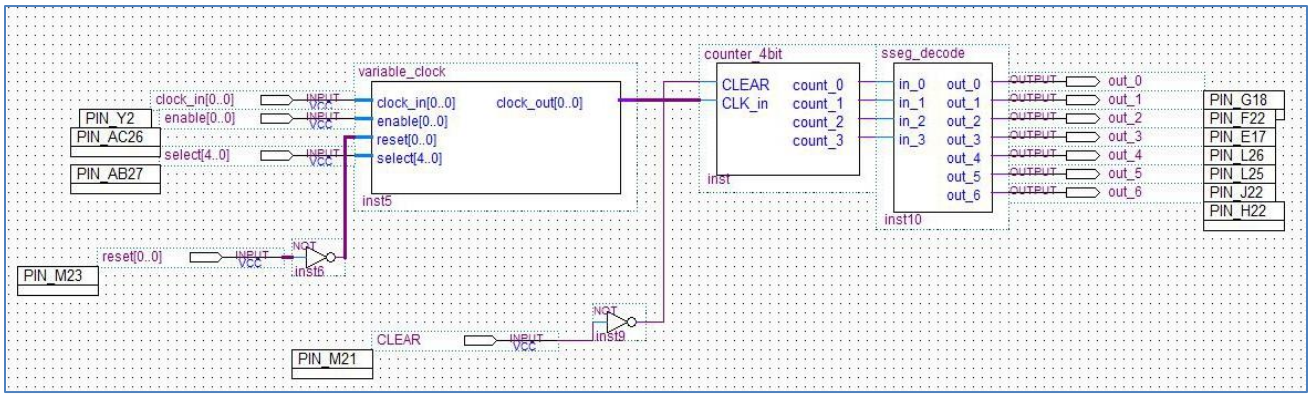
When you are done, check your design with ***Processing → Analyze current file***

If all goes well and you see no errors, go ahead and package the counter by choosing ***File → Create/Update → Create symbol files for current file***

This will create the ***.bsf*** file that makes your counter available as a module in the schematic editor.

3) Test your counter

Go back to the file ***Lab_4.qpf*** (your main project file) ***delete the flip-flop test circuit.*** Then implement the circuit shown below:



Note that: ***Your 4 bit counter should be available from your project's directory***
The variable clock and 7-segment decoder are already included with the starter lab package, you can find them also in the project's directory.

Pay close attention to the wiring: **Thick** wires are ***buses***, and thin wires are simple wires. Save everything, compile your design, and once you have successful compilation assign I/O pins as shown in the figure above. Note that the push-buttons give a logic '0' when pressed.

clock_in for the ***variable_clock*** is provided by the FPGA's on-board 50MHz clock at ***PIN_Y2***
enable for the ***variable_clock*** is tied to ***SW5 (PIN_AC26)***
reset for the ***variable_clock*** is tied to ***Key0 (PIN_M23)***
The ***5 select bits*** for the ***variable_clock*** are tied to ***SW0-SW4 (Pins: AB28, AC28, AC27, AD27, AB27).***

The ***clear*** input for the ***counter_4bit*** is tied to ***Key1 (PIN_M21)***
The outputs of the ***sseg_decode*** (7-segment decored) are tied to the first digit in the FPGA.

3) Testing… continued

Compile your project – make sure there are no errors

> ***Errors about 'already defined' components? Check the names of your logic components. These should not be repeated!***

Before programming the FPGA:

- Set SW5 to 1 (enable the variable clock!)
- Set SW4-SW0 to 10111 (binary 23)

The ***variable_clock*** gives you a clock with frequency ***50Mhz/(2^(select+1))***. With the settings above, you will get a clock rate of 50Mhz/(2^24) which is approx. 3Hz. You can play with the clock frequency later, for now, use the settings above to verify your design.

Program the FPGA board.

Your circuit should start counting upward on each clock cycle. If you press ***clear*** the counter should go to ***0000*** on the next clock cycle.

Counter verified by TA in lab: ☐

Clear verified by TA in lab: ☐

Part 3 – Building a digital clock

1) To be completed prior to the lab session

Show in the space below the diagram for a four digit (HH : MM) digital clock built using the four bit counters designed in part 2.

- You can include a block to represent the variable clock that feeds the entire system
- You can also use blocks to represent the 7-segment decoders and 7-segment displays

The circuit should be similar to that in the previous page times 4, but note that you need to add ***boolean logic*** to ensure that ***minutes cycle from 59 back to 00, and hours cycle from 23 back to 00***. You should be able to do this with little effort by using the ***clear*** signal of your counters.

However: Note that the ***clear*** signal may have to be active for a full clock-cycle in order for all flip-flops to reset. So, the clear-signal generated by your logic circuits should last for about that long. You should be thinking flip-flops again!

Be sure to check with your TA if you are not sure about your design.

2) Implement your design in the ***Lab_4.qpf*** file (delete the simple 4-bit counter circuit).

Once you are done, compile your design and correct any errors found.

Set the variable clock inputs as in the previous part.

Program the FPGA board and test your digital clock!

Clock verified by TA in lab: ☐

Notes:

3) Wrapping up (10 minutes before the end of the session)

Power-down and carefully store the DE2-115 board, bring the boards and cables to the storage locker your TA will indicate.

On the Desktop. Locate the folder for ***Lab_4*** and create a ***.zip*** file containing ***all the design files (.bsf, .bdf, .qpf) and nothing else***.

e-mail this .zip file to the ***course instructor*** with the following subject:

'***CSCB58, Lab 4'*** (without the quotes!)

Inside the email, note the name and student numbers of each team member.

Don't forget to attach the .zip file!

Complete the section below (there will be a penalty for not completing the feedback!)

4) Post-session feedback. Feel free to use simple yes/no answers, or to add comments where you feel they are relevant.

- Did you complete the sections marked ***prior to lab session*** before the start of the session?

- Did you complete the lab exercise in the 3-hour allotted time span?
(if no, how much more time do you think you would have needed?)

- Did this exercise improve your understanding of flip-flops and time-dependent (sequential) circuits?

- Overall: Do you feel that the lab sessions have enhanced your learning of the digital circuits material in the course? (provide us with any comments you may like to add)

- Any other thoughts or comments (use the back of the page if needed):