Digital System Design [ECE437]

Lab 3: Adventure Game [INLAB ASSIGNMENT]

0. Introduction

Total points assigned to all lab 3 assignments- 10 points

Once you have completed the prelab portion of Lab 3, you can start working on your digital logic equations for Lab 3 and eventually the block design for Lab 3.
Please read and follow the steps of this lab closely. Start early and ask questions if parts are confusing. It is much easier to get your design right the first time around than to make a mistake and spend large amounts of time hunting down the bug. As always, don't forget to read the entire lab and refer to the "What to Turn In" section at the end of this lab before you begin.

You will design your FSM using the systematic design approach described in Section 3.4.5 of the textbook.

1. Design

[Picking up from the green highlighted part of prelab assignment]

Using your tables from the prelab report, you should be able to write down a separate Boolean logic equation for each output and for each bit of the next state (do this separately for each FSM). In your equations, you can represent the different bits of the state encoding using subscripts: S_1, S_2, etc. Depending on which state encoding you chose, a different number of bits will be required to represent the state of the FSM, and thus you will have a different number of equations. Simplify your equations where possible.

As you know, you can translate these equations into logic gates to implement your FSMs directly in hardware. That is what you will do in the next section. We will use these logic equations that we obtained to build the block design for each of the two FSMs- Room FSM and Sword FSM.

2. Schematics

Start Quartus and create a new project named "lab03_xx" (where xx are your initials).

By now, you are familiar with the Schematic Editor. In this lab, however, you will learn how to create **hierarchical** schematic designs. In the same way that you can add symbols such as AND and OR gates to your schematic, you can add sub-components that are themselves specified by schematics. This creates a hierarchy of schematics.

Note that the flip-flop in the schematic editor is called DFF. It has asynchronous active-low set and reset inputs named PRN and CLRN. Both of these should be connected to VCC so that the element behaves as an ordinary flip-flop. (Yes, this gets pretty annoying…)

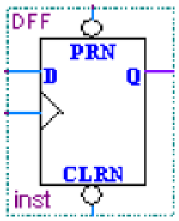

Fig. 1- *DFF flip-flop from the schematic editor*

You will use a **hierarchical design** for your adventure game. Just like in Python or C programming language you can create user-defined functions and then call them in some other function, a similar type of modular design is also supported in Quartus. This is what we call **hierarchical design**.

1. In Quartus we design circuits, whereas we define functions in Python/C language. These circuits are also called modules. You can use one module in another high-level module. In this lab, the top-level schematic file will use the modules for Room FSM and the Sword FSM. So, first create the top-level schematic file and name it "Game.bdf". You can leave it empty for now and move to next step.
2. Now, create another schematic file (block design file) and name it "sword.bdf". We start our design of the game by first making a schematic (ie the block design) for the Sword FSM. Use the logic equations that you obtained for the Sword FSM in the prelab to make this schematic.
3. Once you are satisfied with the block design of the Sword FSM, you need to verify it works as expected. To do this we will use simulation waveforms. But first we need to set this schematic as the top-level entity. To do this first go to the files tab in the project navigator pane (it is the leftmost pane in Quartus). Then right click on the schematic file for Sword FSM and select "Set as top-level entity". This is shown in picture below-

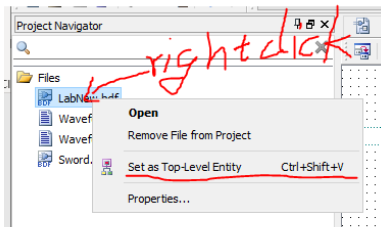

Fig. 2- *How to set top-level entity*

Once you have set it as the top-level entity, you can go ahead and use simulation waveforms to verify your design for the Room FSM.

4. Now we have to create the symbol for our schematic of the Sword FSM. To do this- Use *File -> Create/Update -> Create Symbol Files for Current File* to make a symbol for your schematic of the Sword FSM. Name it “sword.bsf”. This is shown in the figure below-

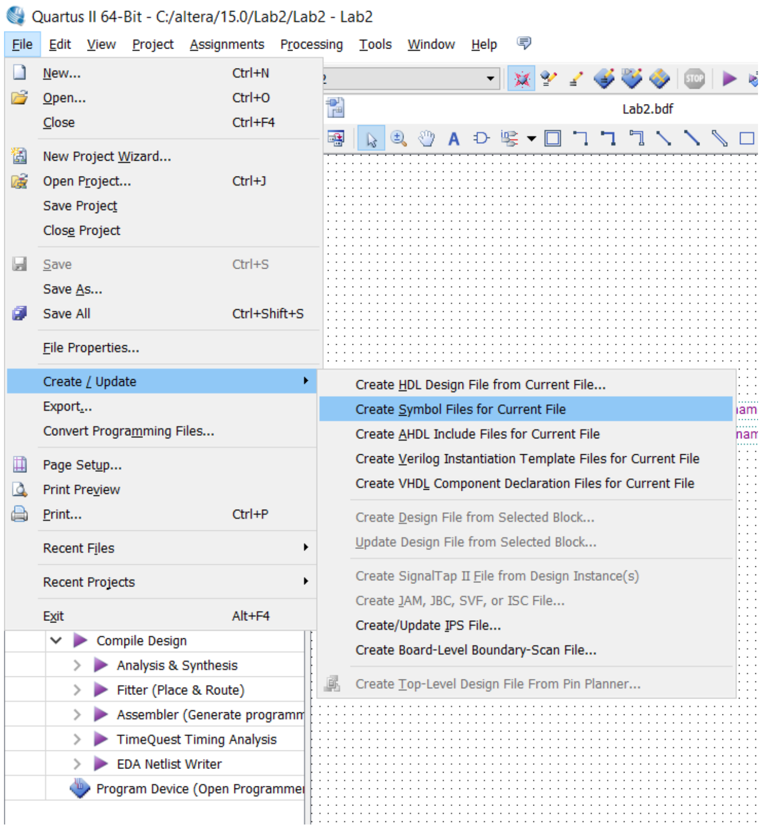
Quartus II 64-Bit - C:/altera/15.0/Lab2/Lab2 - Lab2
File Edit View Project Assignments Processing Tools Window Help
New... Ctrl+N
Open... Ctrl+O
Close Ctrl+F4
New Project Wizard...
Open Project... Ctrl+J
Save Project
Close Project
Save Ctrl+S
Save As...
Save All Ctrl+Shift+S
File Properties...
Create / Update
Export...
Convert Programming Files...
Page Setup...
Print Preview
Print... Ctrl+P
Recent Files
Recent Projects
Exit Alt+F4
Create HDL Design File from Current File...
Create Symbol Files for Current File
Create AHDL Include Files for Current File
Create Verilog Instantiation Template Files for Current File
Create VHDL Component Declaration Files for Current File
Create Design File from Selected Block...
Update Design File from Selected Block...
Create SignalTap II File from Design Instance(s)
Create JAM, JBC, SVF, or ISC File...
Create/Update IPS File...
Create Board-Level Boundary-Scan File...
Create Top-Level Design File From Pin Planner...
Lab2.bdf
Compile Design
Analysis & Synthesis
Fitter (Place & Route)
Assembler (Generate programm
TimeQuest Timing Analysis
EDA Netlist Writer
Program Device (Open Programme

Fig. 3- *How to create a symbol for your schematic*

5. We repeat steps 2, 3 and 4 for the Room FSM. First create a new schematic file and name it room.bdf. Use your logic equations from prelab to complete the block design for Room FSM. Then verify your design using simulation waveforms. Once verified, create a symbol for your schematic of Room FSM and name it “room.bsf”.

6. Now open the schematic for “Game.bdf” and add the two symbols we just created as described in picture below-

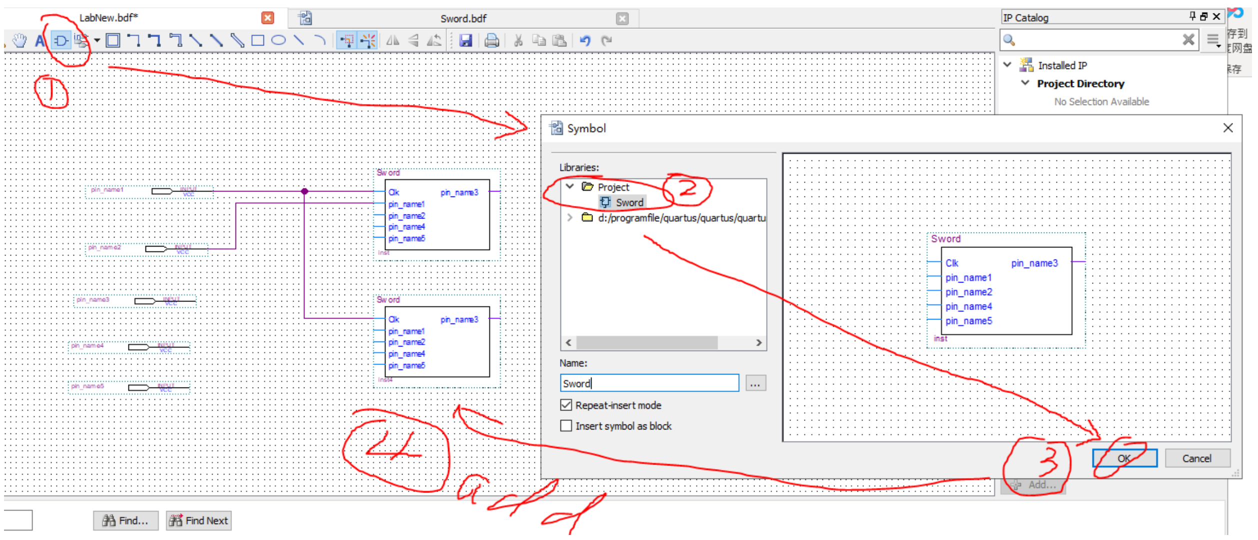

Fig. 4- *How to add symbols to the "Game.bdf"*

7. Now connect the two FSMs and complete the design of your adventure game. **Do not forget to make "Game.bdf" the top-level entity once you are satisfied with your design for the two FSMs.**
8. **If you modify the inputs or outputs for a block** that you have already created, regenerate the symbol file for the block. If the symbol has already been used in a higher level of the hierarchy, right click on the symbol and choose Update Symbol or Block… to update it with the modified symbol.
9. The inputs and outputs of your top-level schematic will determine which signals will be available in the simulator when you play the game, so you should make sure to include at least `clk`, `reset`, `n`, `s`, `e`, and `w`, as inputs and the current room `s0`–`s6` as an output.
10. Now we move onto verifying the functionality of our Adventure Game using simulation waveforms.

3. Simulation

Now it is time to fire up the simulation waveform and play your game. In this lab, you can manually set the clock and the input waveform. You will assume that you can provide only one valid input for each clock cycle.

Hints for generating the clock-

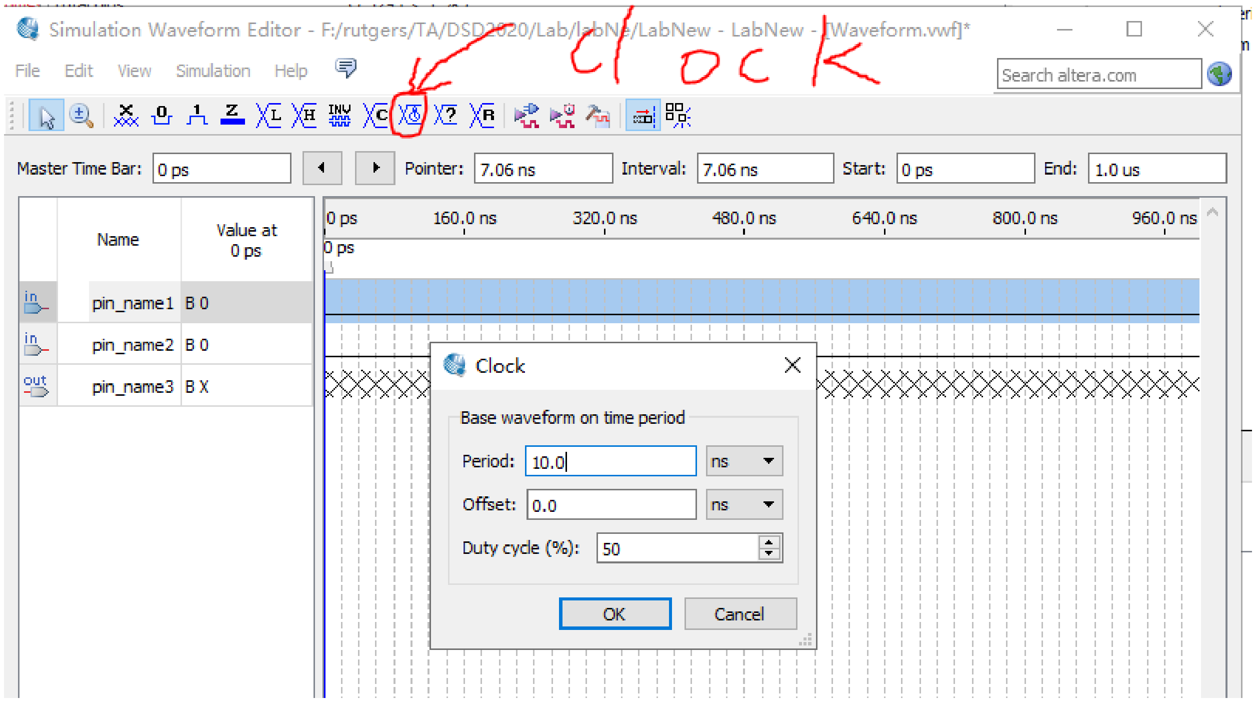

Fig. 5- *Hint for generating clock (picture 1)*

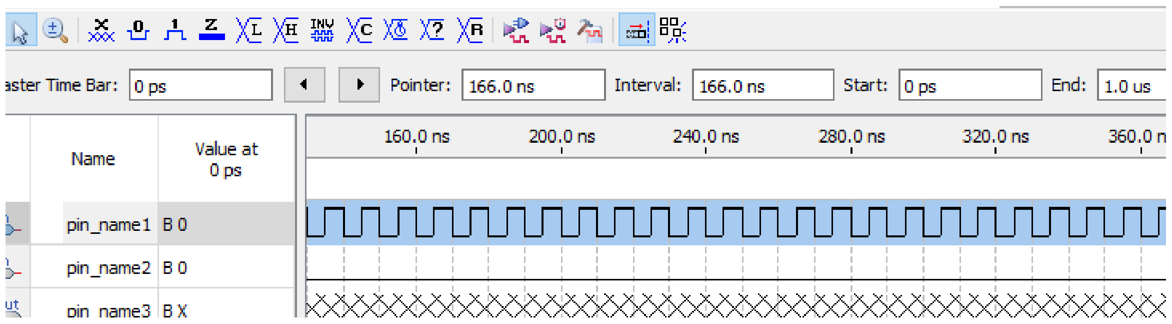

Fig. 6- *Hint for generating clock (picture 2)*

What to Turn In

Due Date- <u>One week from when the experiment is assigned</u> (that means week of September 29)

Points assigned to in-lab Assignment- <u>8 points for Lab 3 come from inlab assignment report submission.</u>

You must submit an electronic copy of the following items (in the correct order, and each part clearly labeled) via Canvas. Submit to the Lab 3 In-Lab Assignment. These should all be included in a single file (.pdf).

1. To the pdf document from your prelab assignment, just add a section for the in-lab assignment.
2. Please indicate how many hours you spent on this lab**.** This will be helpful for calibrating the workload for the upcoming labs.
3. An image of your schematic/block design for the Room FSM.
4. An image of your schematic/block design for the Sword FSM.
5. An image of your schematic for "Game.bdf" (built by connecting symbols for both FSMs).
6. Two images of your simulation waveforms: one that shows you playing the game and winning (entering "Victory Vault"), and another that shows an example of losing the game (entering the "Grievous Graveyard"). **Your signals must be printed in the following order: `clk`, `n`, `s`, `e`, `w`, `r`, `win`, `d`, $s_6 \ldots s_0$, `sw`, `v`.** Please place the images in a landscape orientation so that they fit better on the page.