Synopsis - Week 3: Basic Logic Gates and Finite State Machines (FSMs)

Prof. Jordan C. Hanson

February 26, 2020

1 Introductions to Connect and Disconnect to PYNQ-Z1 System-on-a-Chip (SoC)

1. Connect to the PYNQ-Z1 board in the usual fashion, *ensuring that the system is logged out* before starting a new kernel.
2. If troubleshooting is needed, consult lab partners or the professor.

2 Finite State Machines (FSMs)

1. Navigate to the finite state machine Jupyter notebook in the logictools directory.
2. Load the finite state machine generator according to the steps, and examine the diagram explaining the relationship between the eight states of the FSM.
3. Notice that each state *transition* goes either from state N to $N+1$ or from N to $N-1$. Thus, each state either goes forward or backward.
4. Collect two jumper wires. Connect the pins on the PYNQ-Z1 according to the image shown in the notebook.
5. Once you execute the FSM generator according to the plan in the diagram, the *trace analyzer* will give you the timing diagram of the system.
6. **Verify the that the timing diagram is counting in gray code.** Read off the binary values from the timing diagram, and record them below. Convert the gray code back to binary in a second column of the table.

3 Complex Boolean Logic Functions

1. Close the FSM generator notebook tab, and shut it down using the `Running` tab in the PYNQ-Z1 homescreen.
2. Start a new Jupyter notebook (New → Python3), create a python dictionary entitled *function* and use the boolean generator to perform multiple logic operations:

```
from pynq.overlays.logictools import LogicToolsOverlay
logictools_olay = LogicToolsOverlay('logictools.bit')
function = {'f1': 'LD0 = ~( ~(PB0 & ~(PB0 & PB1)) & ~(PB1 & ~(PB0 & PB1)) )'}
boolean_generator = logictools_olay.boolean_generator
boolean_generator.setup(function)
boolean_generator.run()
```

In a separate cell, you can add the stop function for the boolean generator:

```
boolean_generator.stop()
```

3. Before recording the truth table, draw a diagram below using NAND gates to describe what effect the firmware *should* produce.

4. Now record the truth table, and identify the logic gate by the truth table.

5. Does the logic gate identified match the diagram deduced from reading the code? How can you show that they are equivalent? Construct the proof below.

6. Reset the kernel, and repeat steps 2-5 for the following boolean function:

```
function = {'f1': 'LD0 = ~( ~(PB0 & PB1) & ~(PB0 & PB1) )'}
```

4 Logging Out and Shutting Down

1. Close all Jupyter notebooks, and navigate to the homescreen of the PYNQ-Z1 board.
2. Click New → Terminal.
3. When the terminal pops up, enter `shutdown now` , and wait for the system to disconnect from the laptop. Then close all browser tabs.