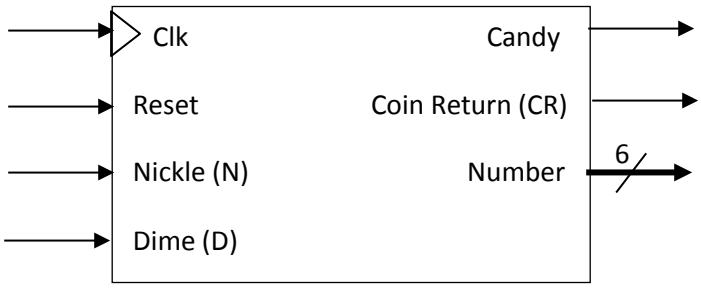
Digital System Design Capsule
Semester Project

Objectives:

- Design and implement **Button Synchronizer FSM**
- Design and implement **Simple Vending machine**
- Practice writing testbench for sequential circuits
- Use Structural VHDL style to connect multiple modules to implement a digital circuit.

Project Overview:

The purpose of the project is to implement a simple and almost practical Vending machine on an FPGA board. We use "Two-Digit Display" in this project, please complete it if you have not done.



You are to design the **electronic vending machine** which sells a ***candy*** for **40 cents**.

- The machine has a clock (Clk) signal and Reset (Rst) button such that when Reset = 1, FSM goes back to an initial state.
- It accepts only N (Nickel = 5 cents) and D (Dime = 10 cents)
- When the sum of the coins inserted in sequence (at the active edge of the clock, **only one coin can be inserted at a time**) is 40 cents or more, the machine releases the candy by setting Candy = 1 and
  - ⇨ if there is no change (a customer pays exactly 40 cents), Coin Return (CR) = 0
  - ⇨ if there is change (a customer enters more than 40 cents), CR = 1
- The machine also has an output (6-bit Number) that displays how many cents the customer has entered (00, 05, 10, 15, 20, 25, 30, 35, or 40 (for 40 cents or more)).

Preliminary design work:

A) (10 pts) Draw a **state diagram of Button Synchronizer** (Page 3). We will write **VHDL code** and will **functionally simulate** Button Synchronizer this week(Use the input waveform on page 3 for your testbench)

B) (10 pts) Draw a **state diagram of Simple Vending Machine** (See below (page 1 and 2) – read it carefully before drawing the state diagram). We will **write VHDL code** for Simple Vending Machine

Note:

1) **For this project, once the machine reaches the state(s) that release the candy, make your machine stay at that state** (this is the almost-practical part since we want to be able to see the outputs, we make the machine stay at that state (state(s) that Candy = 1) until the reset button is pushed. Once the reset button is pushed, the machine goes back to the initial state and waits for the next purchase).
   In practice, the machine will return the change (if a customer enters more than 40 cents), release the candy and then go back to the initial state waiting for the next purchase by a customer.
2) In practice, the coin receptor is a mechanical device and thus very slow compared to an electronic circuit. There will likely be an arbitrary long time between an insertion of 2 coins (=> possible to have both inputs N and D = 0 ➔ when drawing the state diagram, when both inputs = 0, make it stay at the same state).
3) The coin receptor can accept only one coin at a time, not possible to have both D and N set to 1 at one time). For this case, make your state diagram stay at the same state (this is another almost-practical part. In practice, there is another mechanism that prevents a customer to enter two coins at once).

Demo: To make the simple Vending machine work on the FPGA board, we have to use more components as shown in the top-level design below. See Action Items for demo on page 5 and 6. This top-level design (.vhd file) has

4) **4 inputs**: **Clk**, **N**, **D**, and **Reset** (a push button will be used) and
5) **4 outputs**: Candy (LED will be used), CR (LED), 7-bit out7 and 8-bit en_out [out7 and en_out are used in TwodigitDisplay)

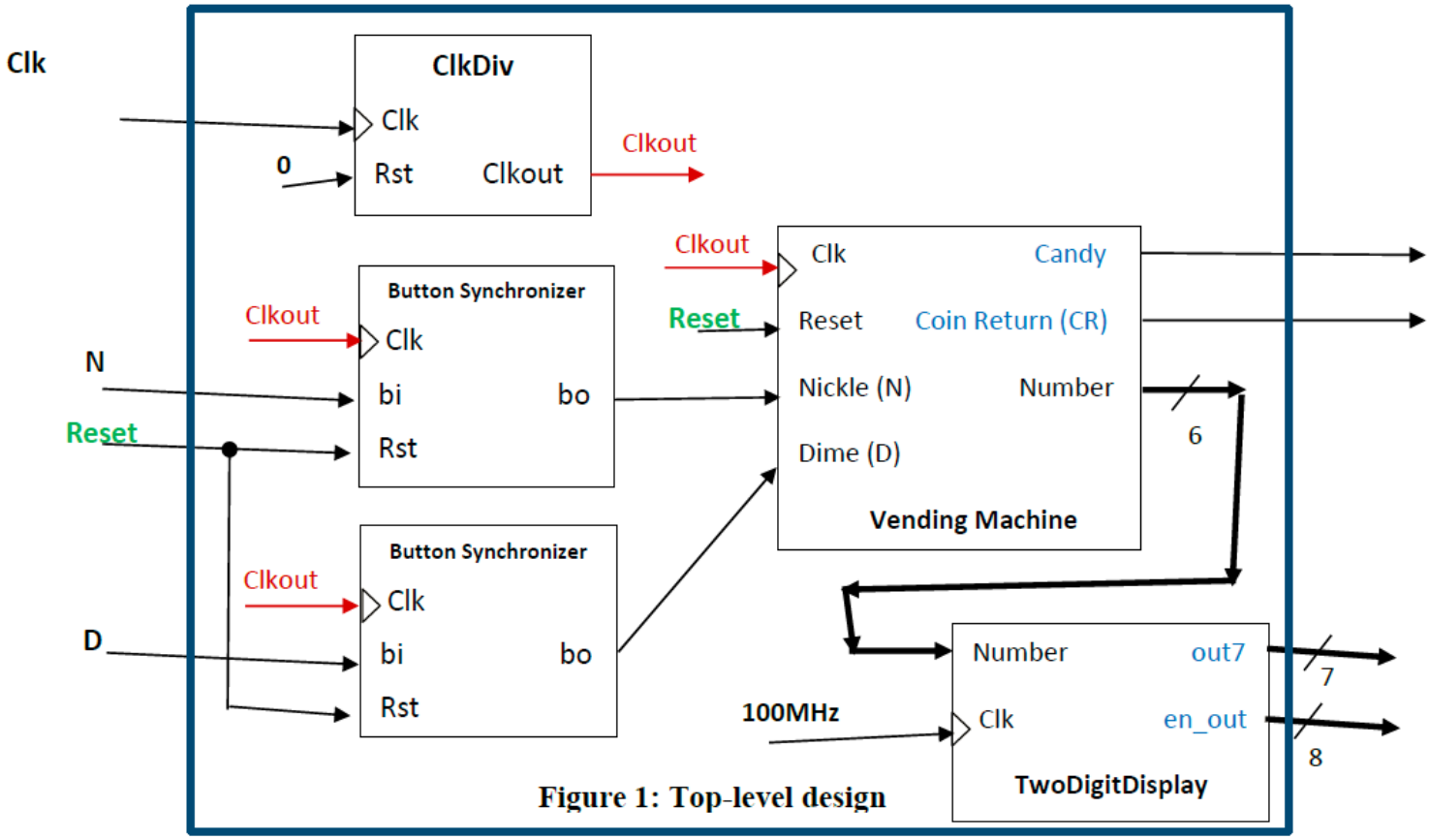


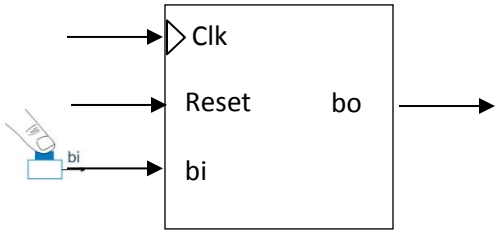
Figure 1: Top-level design

Button Synchronizer

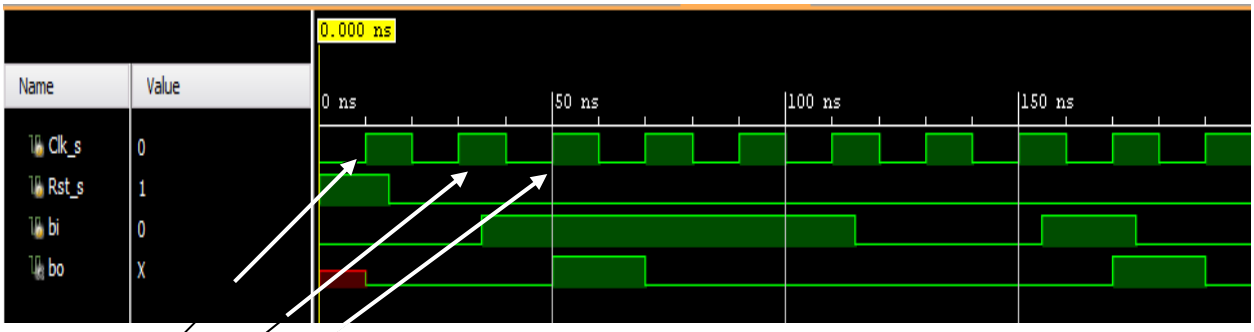
This sequential circuit, Button Synchronizer, converts button press to a **single cycle** duration, ***regardless of length of time that the button is actually pressed*** as shown in the waveform.

Input: bi (1-bit), Reset and Clk

Output: bo (1-bit)



Use the following waveform in your testbench



@ 1st rising edge of the clock (Clk_s), Rst_s = 1, the FSM is at the initial state
@ 2nd rising edge of Clk_s, Rst_s = 0 and bi = 0, b0 = 0
@ 3rd rising edge of Clk_s, bi = 1, now b0 = 1.
@ 4th rising edge of Clk_s, even though bi = 1, b0 is now back to 0. This is what it means by the statement "it converts a button press to a **single cycle** duration, ***regardless of length of time that the button is actually pressed"***

ClkDiv: It has
two inputs: Clk and Rst, and
one output: ClkOut.

Given the 100 MHz clock provided by the FPGA board, the ClkDiv component given on page 4 will generate a 100-Hz clock on its output, ClkOut.

This 100-Hz clock will then be connected to the clock input of several components in the top-level design (including your vending machine) as shown in Figure 1 on page 2 (see Clkout)

Action Items for project demo

Step 1) Write a **testbench** for your simple **Vending machine**.

The idea for the simulation is to test scenarios that can occur in the circuit. For example, you should imagine 3 scenarios like that: 1) when a user enters 4 Dimes; 2) when a user enters 1 Nickel and 4 Dimes; 3) when a user enters 8 Nickels (also show both N and D = 1 at the same time).

Once it releases the candy, it stays at that state until a user hits the reset button (recall that this is the almost practical part that is mentioned before)

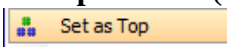
Show the waveform to TA to get credits for this step.

Step 2)

6) Get the following .vhd files and put them in the .srcs folder of this project

- ClkDiv.vhd file
- TwoDigitDisplay.vhd file and sevensegment.vhd file from your previous lab
- the .vhd file of Button Synchronizer

Step 3) Create a new .vhd file to implement the top-level design. **Set this new .vhd file as Top module** (Right-click on the file name under Design Sources and choose "Set as



)

Write ***Structural*** VHDL code to connect the following components together as shown in Figure 1 (page 2)

ClkDiv
Button Synchronizer
Your simple Vending machine (this is your code and after you get the correct simulation waveform from Step 1))
TwoDigitDisplay (from previous lab)

Make sure that your Vivado project (you may need to add these into the project) contains ClkDiv.vhd file, the .vhd file for button synchronizer, the .vhd file of your Vending machine, the
.vhd file of your sevensegment, and the .vhd file of TwoDigitDisplay

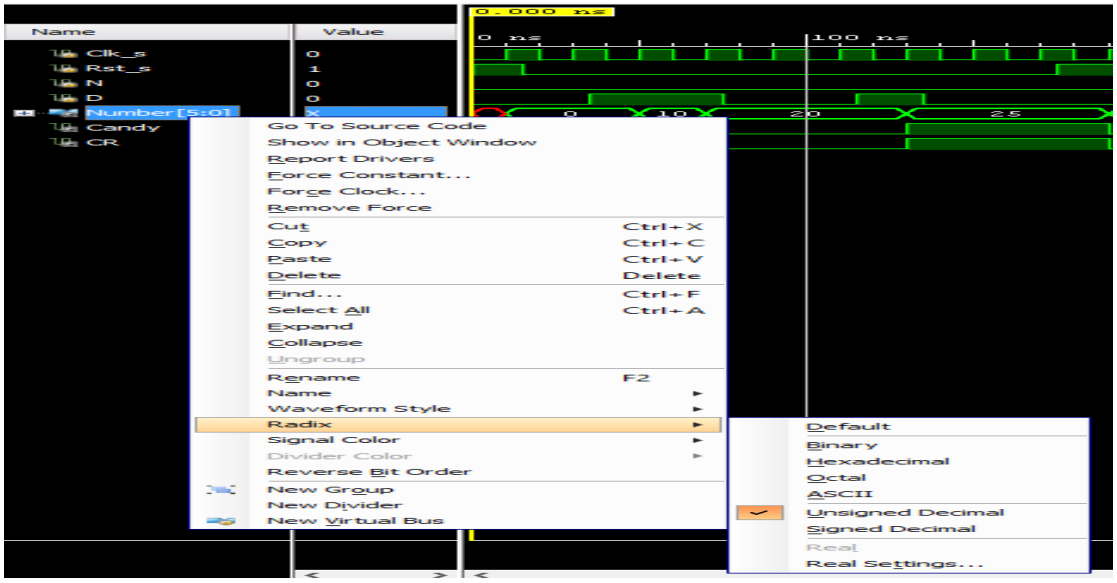
Step 4) Assign the pins to your top-level design circuit (create .xdc file).
From Figure 1, there are **4 inputs**: **Clk**, **N**, **D**, and **Reset** (a push button will be used) and **4 outputs**: Candy (LED will be used), CR (LED), 7-bit out7 and 8-bit en_out [out7 and en_out are used in TwodigitDisplay)

Step 5) Generate Bitstream and be ready to download to the board.

Pushing the reset button should start over after your machine gets to the state that releases the candy.

To change the base number (Radix) for the vector signal

Right-click on the vector signal, choose Radix, then choose the base-number that you want to use. In this project, we will use unsigned decimal since Number is used to represent number of cents that a user enters.



Grading Criteria

Button Synchronizer state diagram, its VHDL and Testbench	10 pts
State diagram of Vending Machine and VHDL	10 pts
Step 1) Testbench for Vending Machine and correct waveform	15 pts
Step 3) Structural code for Top-level design	20 pts
Step 4) Correct .xdc file for top-level circuit	10 pts
Step 5) Circuit ready for working on the FPGA board	10 pts
Demo question and answer	25 pts

Code submission: failure in submission will result in a 40% penalty

1. Submit all the vhd source files (.vhd only) including testbenches on Canvas to the designated dropbox.