CS220A Lab#5
Rotary Shaft Encoder

Mainak Chaudhuri
Indian Institute of Technology Kanpur

Sketch

- Assignment#1: rotary shaft encoder

Lab#5

- Make a new folder Lab5_1 under CS220Labs to do the assignment
- Refer to lab#1 slides for Xilinx ISE instructions
- Finish pending assignments from lab#4 first
- Three marks for this assignment

Assignment#1

- Rotary shaft encoder
  - The rotary shaft can rotate in both directions and while rotating it generates two input signals ROT_A and ROT_B
  - Read pages 18 and 19 of https://www.xilinx.com/support/documentation/boards_and_kits/ug230.pdf
    - Figure 2.8 shows how ROT_A and ROT_B signals change as the shaft makes right rotation i.e., A becomes HIGH first and then B becomes HIGH
    - If you read Figure 2.8 from right hand side, it shows what happens when the shaft rotates left i.e., B goes HIGH first and then A goes HIGH

Assignment#1

- Rotary shaft encoder
  - One rotation event is referred to as rotating the shaft to left or right by one step
    - Several steps constitute one full 360º rotation
    - Figure 2.8 shows that one step rotation to the right from detent position (stationary shaft position) causes ROT_A to go HIGH first and then ROT_B goes HIGH
    - Similarly, one step rotation to the left from detent position causes ROT_B to go HIGH first and then ROT_A goes HIGH
    - The changing voltages on ROT_A and ROT_B arise from closing/opening of two mechanical switches
      - These switches suffer from "chatter" i.e., they bounce for a while before settling; leads to train of narrow pulses in ROT_A and ROT_B before they stabilize

Assignment#1

- Rotary shaft encoder
  - The goal of the assignment is to write a Verilog module that can correctly capture every rotation event and the associated direction
    - At a very high level, a rotation event can be captured by observing the changing levels in ROT_A or ROT_B
      - The challenge is to filter out the chatters so that a false rotation event is not detected (i.e., your module detects a rotation when there is actually no rotation)
    - The direction of a rotation can be detected by understanding the relative order of change in ROT_A and ROT_B
      - The challenge here again is to filter out the chatters

Assignment#1

- Rotary shaft encoder
  - On detecting a rotation event, depending on the direction of the rotation, an LED ripple makes one step progress in the direction of the rotation
  - Structure of the Verilog module
    - Inputs: clk, ROT_A, ROT_B
      - Use the clock generated from pin C9 as you did in lab#3
    - Outputs: rotation_event, rotation_direction (both reg)
    - On posedge clk
      - If ROT_A and ROT_B both are 1, set rotation_event to 1
      - If ROT_A and ROT_B both are 0, set rotation_event to 0
      - If ROT_A is 0 and ROT_B is 1, set rotation_direction to 1
      - If ROT_A is 1 and ROT_B is 0, set rotation_direction to 0

Assignment#1

- Rotary shaft encoder
  - Observe from Figure 2.8 that rotation_direction == 1 means left rotation and rotation_direction == 0 means right rotation
  - Observe that a rising edge in rotation_event signifies a step of rotation
  - Observe that sampling rotation_direction at the time of a rising edge in rotation_event can tell us the direction of rotation
  - Write another Verilog module which exploits these observations to drive LED ripples in appropriate directions

Assignment#1

- Rotary shaft encoder
  - Structure of the second Verilog module
    - Inputs: clk, rotation_event, rotation_direction
    - Outputs: led0, led1, led2, led3, led4, led5, led6, led7
      - Similar to rippling LED assignment from lab#3
      - Initialize at least one LED to 1
    - On posedge clk
      - Copy rotation_event to prev_rotation_event (a local reg initialized to 1) i.e., prev_rotation_event <= rotation_event
      - If prev_rotation_event is 0 and rotation_event is 1 (indicates a rising edge in rotation_event)
        - » If rotation_direction is 0, shift LEDs to right i.e., led0 <= led1, led1 <= led2, ..., led6 <= led7, led7 <= led0
        - » If rotation_direction is 1, shift LEDs to left i.e., led1 <= led0, led2 <= led1, ..., led7 <= led6, led0 <= led7

Assignment#1

- Rotary shaft encoder
  - Write a top-level Verilog module with inputs clk, ROT_A, ROT_B and outputs led0, …, led7
    - Instantiates the previous two Verilog modules and establishes the connection between them
  - Use PlanAhead to assign pins to the inputs and outputs of the top-level module
  - Synthesize the hardware
  - No need to simulate this design in ISim because ROT_A and ROT_B cannot be given exact input behavior as they will receive from the rotary shaft