SANTA CLARA UNIVERSITY	ELEN 21L Fall 2022

Laboratory #9: Clock Divider and Mealy Machines

For lab sections Monday-Friday November 28 – December 2, 2022

I. OBJECTIVES

- To understand and practice the design of a clock divider
- To understand and design a Mealy Finite State Machine (FSM)
- To design, test, and implement a Mealy machine based on a functional specification

PROBLEM STATEMENT

In this lab, you are going to practice a Mealy machine design with a simple counter that supports pause/resume on the Intel/Altera's DE2-115 board.

In addition to reset and clock, the Mealy machine you are to design in this lab session includes two inputs: *PR* (pause/resume, 1-bit) and *Max_Val* (6-bit). As output, it displays the current counter value using two 7-segment displays.

This functionality of this machine is as follows:

1) Upon a reset, the counter value is initialized to 0.

2) Then, at each rising edge of the clock signal, the counter value is incremented by 1 until the value reaches the maximum value (specified by the input *max_CNT*).

3) When *PR* is asserted (by pressing the push button KEY[0]) during this normal run, the counting should be paused until the same button is pressed again.

4) When the counter reaches the maximum value, it stops incrementing the value and keeps displaying the maximum number until reset.

- After reaching the maximum number, adjusting the maximum value (*max_CNT*) to a smaller value does not make any effects.
- After reaching the maximum number, if the maximum value (*max_CNT*) is adjusted to a bigger value, the counter should start increasing the value again until it reaches the new maximum value.

Note that the target board (Intel/Altera's DE2-115) only includes one oscillator that produces 50 MHz clock signal, which is too fast to be used in our counter example. Therefore, as a first step, you will implement a simple counter-based clock divider circuit that transform the 50 MHz clock signal to 1 Hz. Then, you will implement the Mealy machine (you completed for the pre-lab) in Verilong HDL.

The FSM design procedure includes the followings:

1. Describing a state diagram
2. Building a state transition table
3. Write the *Next State* and *Output* logic in Verilog

II. PRE-LAB

(a) [Counter-based Clock divider] See the "1Hz clock generation" part in lab9.v, which is a counter-based clock divider that generates 1Hz clock signal out of the given 50 MHz clock (CLOCK_50). In the code, the counter (*clk_count*) resets to 0 upon reset, then increases by 1 whenever the rising edge of CLOCK_50 arrives. The clock output (*CLK_1Hz*) is '1' until the counter value reaches a half of the threshold (*half_th*). Then, while the counter value is between *half_th* and *th-1*, the clock output is '0'. When it reaches the constant number defined as *th*, the counter value resets to 0. This way, the period of the clock output (*CLK_1Hz*) becomes *th* * *T_CLK*, where *T_CLK* denotes the clock period of the original clock (*CLOCK_50*).

In our case, we aim at 1Hz for *CLK_1Hz* and the *T_CLK* is 1/50MHz = 20ns. Calculate the value of *th* that results in 1Hz.

(b) [State Diagram] We will use a Mealy machine to implement the state machine that controls the counter. The inputs of the state machine to be designed are as follows (clock and reset are omitted for brevity):

a. *PR* (pause/resume): 1-bit push button signal,
b. *max_CNT* (maximum counter value): 6-bit switch signal, and
c. *CNT_out* (current counter value): 6-bit signal from counter.

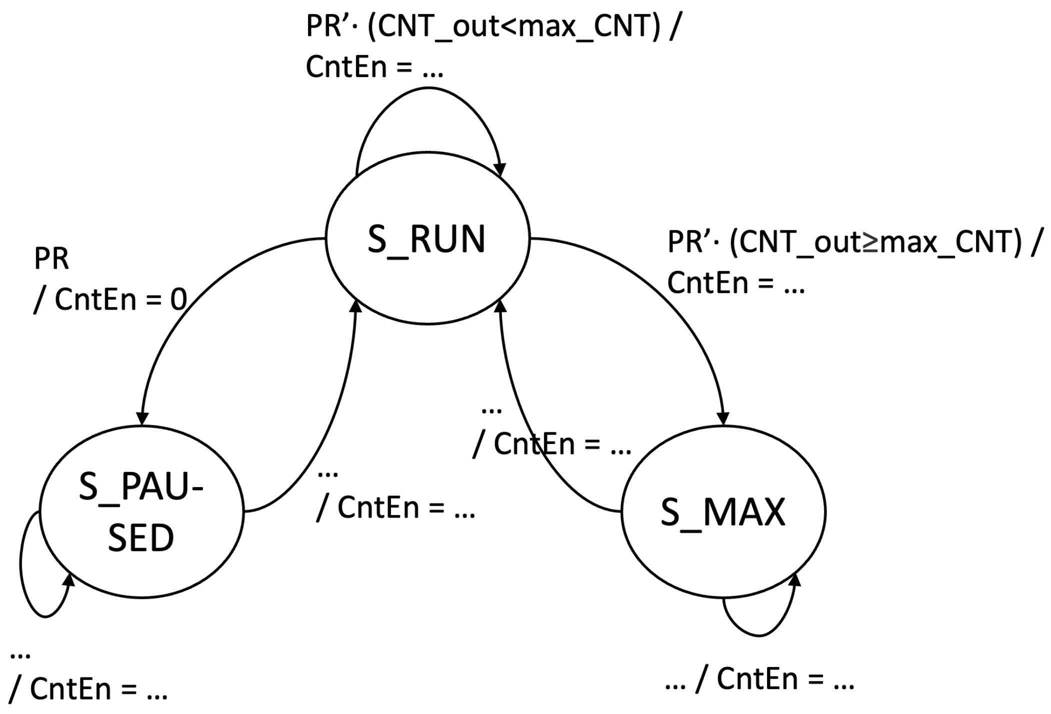
Based on these inputs, the FSM produces a single output diagram, let us define five states as follows:

a. *CntEn* (counter enable): 1-bit signal.

The states are defined in the following table.

States	Meaning
S_RUN	Counter is increasing
S_PAUSED	Counter is paused
S_MAX	Counter reaches the maximum value (i.e., stops increasing)

Based on this definition, complete the state diagram shown on the next page. Note that, unlike a Moore machine, output is determined <u>both by the current **state** and the current **input** value</u> in a Mealy machine.



(c) [State Transition Table] For each of the three states defined above, complete the state transition table.

For *S_RUN*

Tr. #	Transition condition	Next State (*S_star*)	Output (*CntEn*)
1	*PR*	*S_PAUSED*	*CntEn* = 0
2	$PR' \cdot (CNT\_out < max\_CNT)$	*S_RUN*	*CntEn* =
3	$PR' \cdot (CNT\_out \geq max\_CNT)$	*S_MAX*	*CntEn* =

For *S_PAUSED*

Tr. #	Transition condition	Next State (*S_star*)	Output (*CntEn*)
4		*S_PAUSED*	*CntEn* =
5		*S_RUN*	*CntEn* =

For *S_MAX*

Tr. #	Transition condition	Next State (*S_star*)	Output (*CntEn*)
6		*S_MAX*	*CntEn* =
7		*S_RUN*	*CntEn* =

(d) [Implementation] Complete the Verilog implementation of your Mealy machine design (based on lab9_fsm.v.txt).

Submit your Verilog code as your pre-lab assignment

III. LAB PROCEDURE

1. Reconcile your design choices with your lab partner

Compare the state diagram and transition table that you built in your pre-lab with what your partner has done. Decide between yourselves which version you will use.

2. Generate a 1Hz clock signal

a. Create a Quartus project and add the given Verilog files (listed below) in the project (*Project → Add/Remove Files in Project…*)
   i. *lab9.v* (top-level design with clock divider)
   ii. *UpCounter_6bit.v* (6-bit UpCounter)
   iii. *bin_7seg.v* (seven segment display)
   iv. *BCD_encoder_2_digit.v* (2-digit BCD encoder)

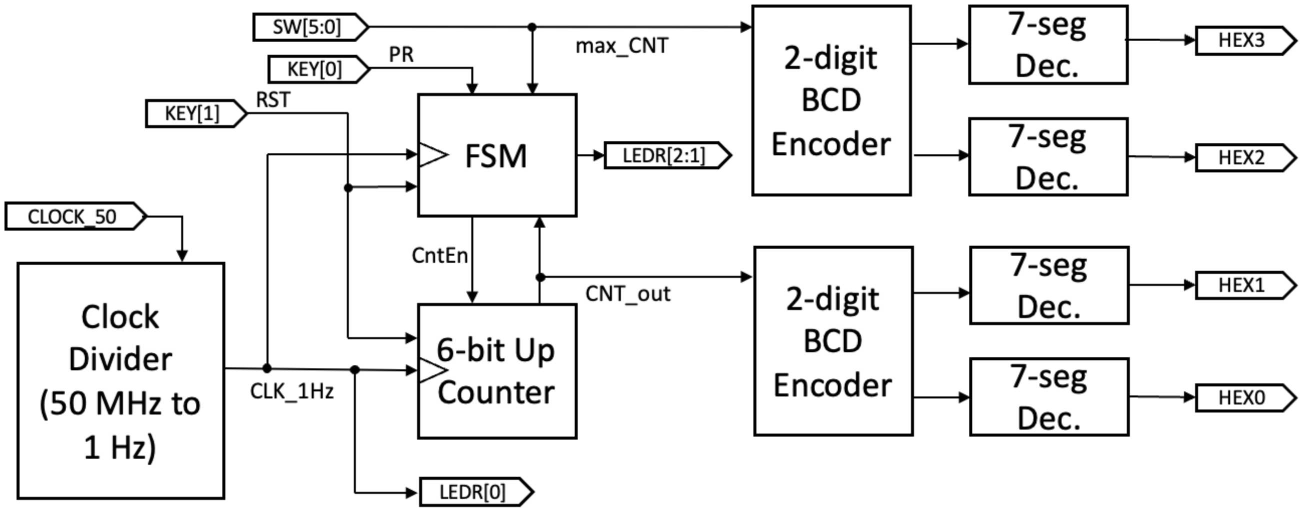
b. Complete the code (*lab9.v*) with your calculated *th* value.

c. Perform pin assignment (Assignments → Import Assignments…) with the given qsf file (*DE2_115.qsf*) and compile your design. Note that you have use the full compilation, i.e., "*Start Compilation*" (▶).

d. **[DEMO]** Program the board and check LEDR[0] blinks at 1Hz.

3. Complete the Mealy machine design

a. Following is the top-level block diagram of your design,



in which you need to complete the FSM part.

b. In order to implement the FSM part, create a Verilog file *lab9_fsm.v* (*File* → *New…* → *Design Files* → *Verilog HDL File*), then, copy and paste what is given in *lab9_fsm.v.txt*.

c. For your information, the input/output port information of the design is summarized in the following table. Do NOT modify the input/output port specifications since the pin assignment is performed based on these port names.

	Port name	Descriptions
Inputs	*CLOCK_50*	Input 50MHz clock
	KEY[1]	Reset (*RST* in *lab9_fsm.v*)
	KEY[0]	Pause/Resume (*PR* in *lab9_fsm.v*)
	SW[5:0]	Max Counter value (*max_CNT* in *lab9_fsm.v*)
Outputs	*HEX3[6:0]*	7-seg display out for tens position (max)
	HEX2[6:0]	7-seg display out for ones position (max)
	HEX1[6:0]	7-seg display out for tens position (counter)
	HEX0[6:0]	7-seg display out for ones position (counter)
	LEDR[0]	1Hz clock
	LEDR[2:1]	Current State (for debugging purpose)

d. Complete the implementation of *lab9_fsm.v* based on your pre-lab.
   i. See pages 37-38 of Verilog Syntax Reference for case and if-else statements examples.
   ii. After completing the FSM implementation,
      i. comment out line number 73 (assign cnt_en = 1'b1;) in *lab9_fsm.v* and
      ii. uncomment line number 72 (instantiation of lab9_fsm).

e. Perform pin assignment (Assignments → Import Assignments…) with the given qsf file (*DE2_115.qsf*) and compile your design. Note that you have use the full compilation, i.e., "*Start Compilation*" (▶).

f. **[DEMO]** Program the board and demonstrate the following scenario.
   i. Set SW[5:0] to "000110" and show that the counter increases from 0 to 6.
   ii. From the above setting, show that pause/resume button works.
      i. When pressing the PR button, make sure that the press lasts more than 1 second in order to be detected at the rising edge of the 1Hz clock cycle.
   iii. Show that the counter eventually stops at 6 (FSM state == S_MAX).
   iv. From the above state, change the SW[5:0] input to "001110" and counter resumes counting up until it reaches the new maximum value.

IV. REPORT

Your report should address the following:

- Were the state diagrams/state tables you created in your pre-lab correct or not? If it was incorrect, in what way was it incorrect? What do you think led you to your incorrect diagram/table?

- If your pre-lab was incorrect, provide a corrected answer.

- Consider the transition table of S_RUN that you completed in your pre-lab. Suppose that the transition condition of transition #2 is changed from "*PR'* · (*CNT_out* < *max_CNT*)" to "(*CNT_out* < *max_CNT*)". What is the problem of this change? In what situations, does this cause problems?

- What would happen if one keeps pressing the *PR* button for a long time (at least for more than 2 seconds)? If you believe that this is a problem, suggest a new state machine that can resolve this issue.