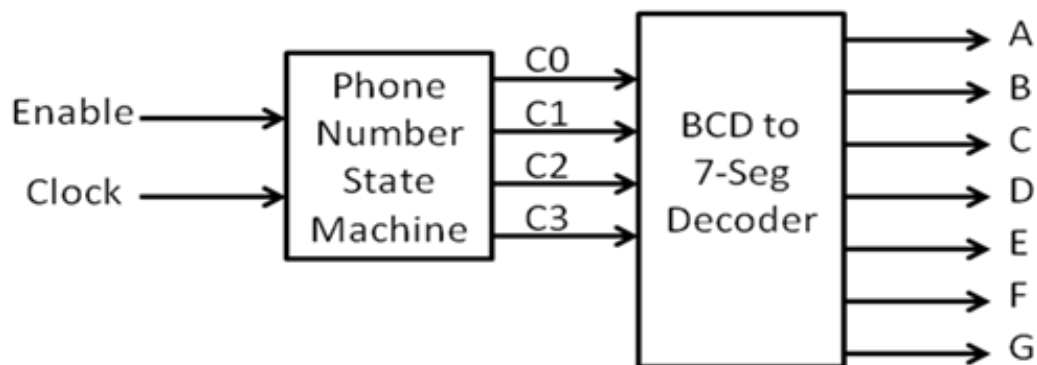


Activity 4.1.2

State Machines: Phone Number Using PLTW S7

INTRODUCTION

The block diagram below is for a simple **state machine** that counts out the last four digits of a phone number. This design has two inputs and seven outputs.



In addition to the clock input required for all state machines, this design's second input is Enable (EN). Whenever the EN input is a logic (1), the outputs will continuously cycle through the four digits of the phone number. Whenever the EN input is a logic (0), the outputs will hold at their current values.

The three (or four) outputs from the state machine are the binary encoding of the phone number digits. You will only need the fourth output if the phone number contains an 8 (1000) or a 9 (1001).

The binary output of the state machine connects to a decoder and a seven-segment display, which will display the decimal equivalent of each binary number.

In this activity you will design, simulate, and program (to the FPGA chip on the PLTW S7) a state machine that counts out the last four digits of *your* phone number.

EQUIPMENT

- Circuit Design Software (CDS)
- Digital MiniSystem (DMS)
 - myDAQ
 - myDigital Protoboard
 - PLTW S7 FPGA Module
- #22-gauge solid wire

RESOURCES



[State Machine Design](#)



Procedure

- 1 Using your phone number, draw the state graph diagram for your state machine design.
- 2 Determine the number of **state variables** required for your design. Assign these variables to the state graph you created in step 1.
- 3 Binary encode the outputs for each of the states in the state graph. Remember, only designs with phone numbers containing an 8 or a 9 require four outputs; all others require only three outputs.
- 4 Using the completed state graph from step 3, create the **state transition table** for your design. Assume that this design will use D flip-flops.
- 5 Using the completed state transition table from step 4, write the logic expressions for the next state variables and the three (or four) outputs. Be sure to simplify the expression using Boolean algebra or the K-Mapping technique.
- 6 Using the simplified design equations, sketch your state machine. Your implementation may use any form of combinational logic, but the sequential logic must be limited to D flip-flops.
- 7 Connect the output from the state machine design to the DEC_BCD_7 component. If there are only three outputs from the state machine, tie the fourth input on the DEC_BCD_7 to a digital low.
- 8 Using the CDS in PLD Design Mode, enter and test your design. Use the interactive digital logic component for the input **EN**, and use a probe for the outputs. You may also consider connecting probes to the Q outputs of the flip-flop and to the outputs of the state machine. This will make it much easier to test the design.

9

Start the simulator and verify that the circuit is working as expected. If the results are not what you expected, review your circuit and make any necessary corrections. You may need to adjust the simulation speed to observe the outputs changing.

10

Prepare your design for transfer to the board:

- a. Map the design's three (or four) outputs to the LEDs on the PLD Module. LED0 through LED2 (LED3).
- b. Map the state variables (the flip-flop Qs) to LEDs on the protoboard. These signals will be useful in the event that your design must be debugged.
- c. Map the design's Enable (EN) input to any of the DMS push-button switches.
- d. Map the output from the seven-segment decoder to one of the DMS seven-segment displays.
- e. Finally, map the design's clock input to any one of the PIO16s. Using a jumper wire, connect this PIO16 pin to the DIO2 of the DMS.

11

Complete the implementation by downloading the design file to the PLD. On the DMS, verify that the circuit is working as expected. If it is not, debug the design to determine your design error. Make the necessary corrections.

CONCLUSION

- 1 The state machine design in this activity has four states and thus requires two state variables. If a design required eight states, how many state variables would be required?
- 2 What about sixteen states?
What is the relationship between the number of states and the number of state variables?
- 3 If you simplified the logic expression for the three (or four) outputs correctly, the final expressions were not a function of the input EN. It will ALWAYS be the case that the outputs are not a function of the inputs. Why?
- 4 List three advantages of implementing sequential logic designs with programmable logic versus traditional discrete logic design (AOI, NAND, or NOR logic).

Going Further (Optional)

- 5 The state machine design in this activity used D flip-flops. This type of flip-flop was selected because of its ease of design; however, J/K flip-flops could have been used. Redo your state machine design using J/K flip-flops or a NAND only approach. Can you add a new feature or option to this design that would be beneficial?

Proceed to project