EE457

Lab 7: Washing Machine Simulator – Spring 2025

The EE457 students have banded together to get into the high-margin home appliance market.

Our class has been tasked with writing the firmware for the Model 42 Washing machine. We are going to use the DE10-lite board to simulate the washing machine I/O and create a low-cost washing machine that every college student will want to own.

Design Requirements:

1) Key0 is your design reset (like lab6)
2) Your washing machine shall have 2 different wash cycles
    a. Standard wash: (SW0 on/up) Fill, Wash, Drain, Fill, Rinse (acts like wash) spin, and Drain.
    b. Soak'n Spin: (SW0 off/down) Fill, Wash, Spin, Drain
    c. Cycle Timing:
        i. Fill: 3 seconds
        ii. Wash: 6 seconds
        iii. Spin: 6 seconds
        iv. Drain: 3 seconds
    d. Indicator Timing
        i. Cycle "Digit", changes animation at ¼ second. Pattern sequence depends on spin or wash
        ii. Fill "Digit", changes 1 second, holds at empty/full.

(See animation notes below)

3) SW1 on/up transition will start the sequence selected by SW0, and once started, the sequence will run to completion or until estop (see next requirement) regardless of SW1's state
4) Each wash setting shares a common "Emergency Stop" button (Key 1) which will drain if there is water in the machine immediately and stop. IF no water exists, it shall stop immediately. When the Emergency stop is pushed you must create a sequence that drains and stops as quickly as possible, respecting the fact that it takes 3 seconds to drain (if needed).
5) We're going to simulate the sequences by The Simulation shall include the following uses of the 7 segment LED's (These may be smaller State machines operating within a larger one)
6) You must properly synchronize all your inputs and resets
7) When the wash is done, the display should leave the "Water" Digit (Hex5) showing empty, blank the "Cycle" digit(Hex4), and display "d0nE" the 4 right-most digits (Hex3-Hex0).
8) Only if the machine is not running: a transition high-to-low on SW1 should blank the display simulating the machine being turned off. A Low-high transition in this state will take you back to starting (requirement 2).
9) You must have a Modelsim test bench to test this design including all operational modes. The design must simulate using a generic so that the simulation time is reasonable.
10) Your design must compile in Quartus with no inferred latches.

11) You must use enumerated types for your statemachine(s)
12) To make grading in simulation understandable, in addition to driving the seven segment pins, you must provide enumerated type(s) representing your LEDs at your top-level (any LED patterns that don't match the animation should be decoded into the respective ERR state)

```
type spin_led_t is (BOT, RGHT, LFT, MID, SPIN_OFF, SPIN_ERR);
signal spin_led : spin_led_t; --You must decode your spin LED 7seg output into this enumerated type.
type fill_led_t is (EMPTY, FILLING, FULL, FILL_OFF, FILL_ERR);
signal fill_led: fill_led_t; --You must decode your fill LED 7seg output into this enumerated type.
type done_leds_t is (D, O, N, E, DONE_OFF, FILL_ERR);
signal let1, let2, let3, let4: done_leds_t; --You must decode your DONE 7seg outputs into this enumerated type.
```

If your design, as submitted, does not compile in quartus and/or does not run in modelsim **you will be docked points**. BEFORE YOU SUBMIT: Extract the zip you intend to submit to a fresh location on your computer and verify both quartus and modelsim run successfully on this extracted zip.

Implementation Notes:

Use the de10-lite base project (as you did in lab6). You may use files from a previous lab if they are useful to you, but it is suggested that you re-extract a fresh base project from the supplied course material.

It may make sense to have multiple state machines especially for controlling the output.

Like the prior lab, you should provide a generic to speed up the "human-scale" timing so that your simulations run quickly.

Design failures may lead to flooded houses, so make sure you have a good test plan and test accordingly in simulation. Explain your testing rationale in your report. What should happen when the system is running and the toddler pushes buttons? What does your design do and why? What would happen if someone left the start switch (sw1) on? What does your design do and why? Your boss comes to you and says marketing has changed desired sequence timing. How complicated a change is this in your code? What did you/might you have done to make changing requirements easier to implement? All of these kinds of things go into your lab report as well as your testbench tests where applicable.

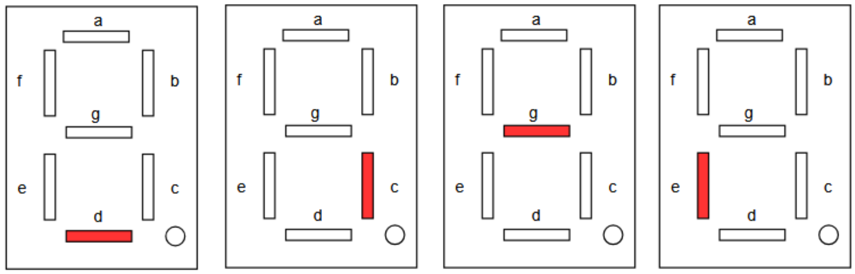
Follow the lab report guidelines in the syllabus/prior lab.

Zip up your design directory and your lab report and submit to Canvas.

Appendix: Animation Notes

"Cycle" Digit Spin Animation
(Loops, Seq #1->Seq #2->Seq #3->Seq #4->Seq #1...)
1full sequence (1 to 4) per second

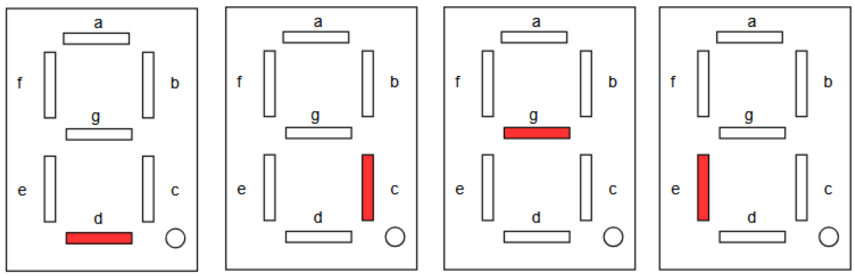
blank if not in spin or wash



Seq #1 Seq #2 Seq #3 Seq #4

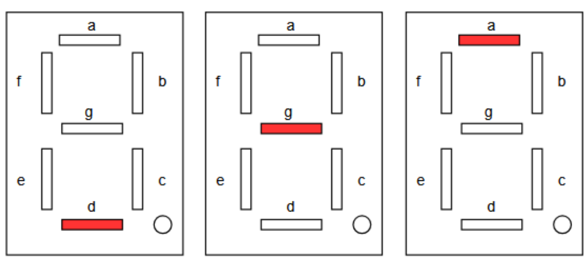
"Cycle" Digit Spin Animation
(Loops, Seq #1->Seq #2->Seq #3->Seq #4->Seq #1...)
1full sequence (1 to 4 to 1) in 2 seconds

blank if not in spin or wash

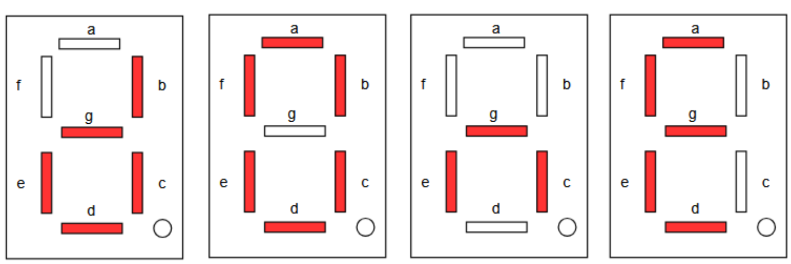


Seq #1 Seq #2 Seq #3 Seq #4

"Water" Digit Fill/Drain Animation
Fill: A->B->C Holds while water present (3 seconds A to C)
Drain: C->B->A Holds while water empty (3 seconds C to A)



Seq #1 Seq #2 Seq #3



Seq #1 Seq #2 Seq #3 Seq #4