Design of Digital Circuits and Systems, HW3

Buffers and ASM

Solution Outlines

Problem 1:

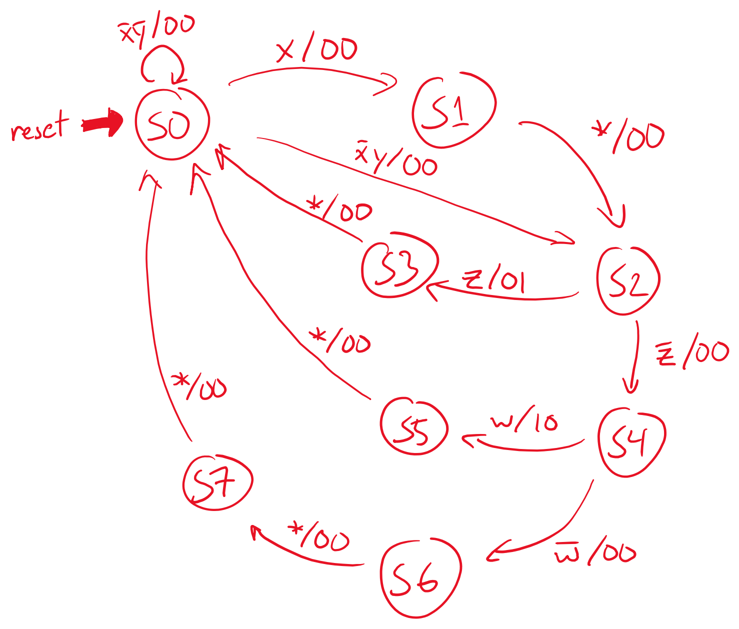
Sample solution code not provided and there are *many* different ways to achieve valid solutions. Some notes:

- The most common solution to handling the imbalanced widths is to make the memory width equal to the width of the read port and increment the write pointer by two each time. Another possible solution would be to make the memory width equal to the width of the write port, but add a "half-word" state bit to the read pointer and read out a halfword.
- Because the read width is smaller, nothing needs to change with the notion of emptiness.
- The notion of fullness needs to be updated – either consider the buffer full if there are 0-1 words free (*i.e.*, not enough space to add the whole write data) by changing your state update table or add another status output with something along the lines of "nearly full" (1 word of space free).
- Simultaneous read and write actions also get trickier when the buffer is nearly full. You should avoid overwriting valid data in the buffer and partial writes. From a usability standpoint, it might have even been better to add an "error" status output for when actions fail.

Problem 2:

There are many different acceptable state diagrams in terms of indicating the inputs and outputs, only one of which is shown below. We did, however, expect a Mealy machine, as implied by the ASM chart.

- Inputs: `wxyz`
- Outputs: AB



Problem 3:

Sample solution code not provided, but it should resemble the standard code outline for an FSM. Some notes:

- Moore outputs will likely be handled by statements that look like `assign Ya = (ps == S0);`
- Mealy outputs could be handled by statements that look like `assign Z2 = (ps == Yc) & X;` or as assignments within the next state `always_comb` block.

In terms of timing differences, Moore outputs update synchronously on clock triggers, while Mealy outputs update both on clock triggers AND input changes. This means that Mealy outputs can appear to react “faster” or “early” if an input value changes before the next clock trigger.

The ASM blocks show that Moore outputs change only when you *enter* a block, whereas the Mealy outputs will have changed before you *exit* your current block.

There is no right answer in the trace comparison for ASM and FSM; we were just looking for your opinion and thoughts.