

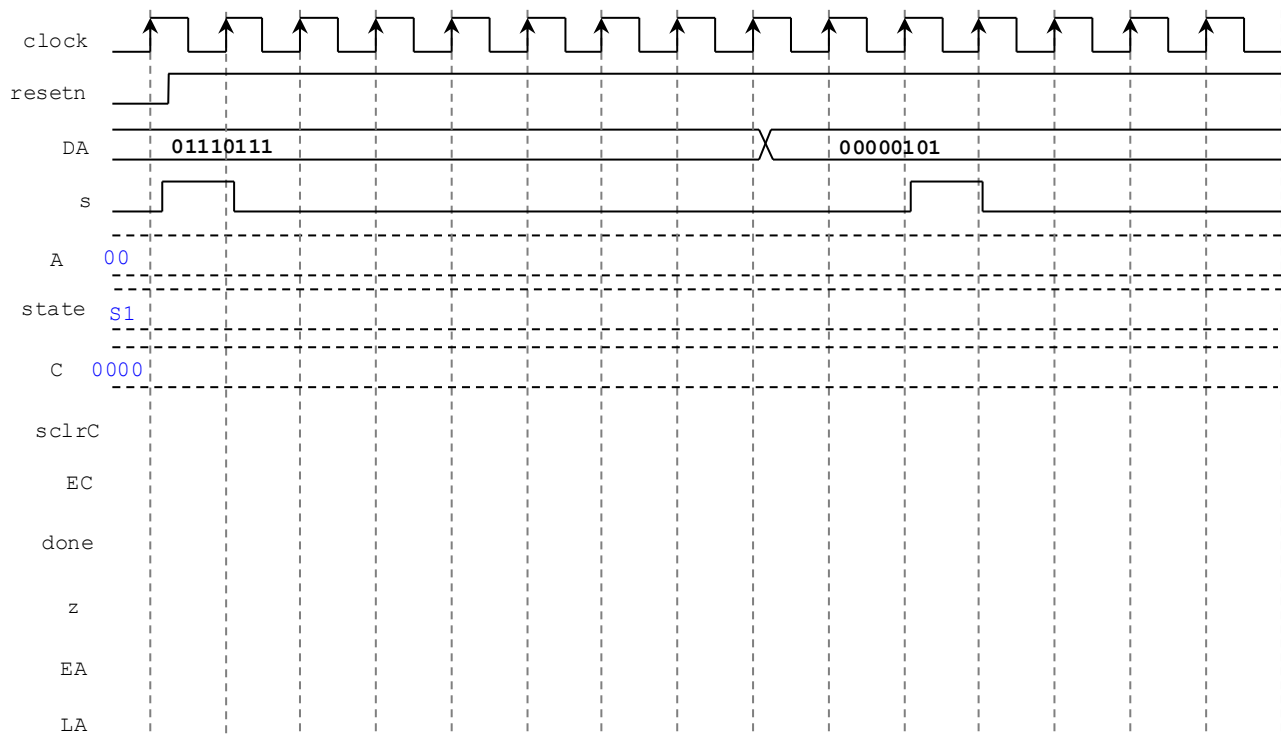
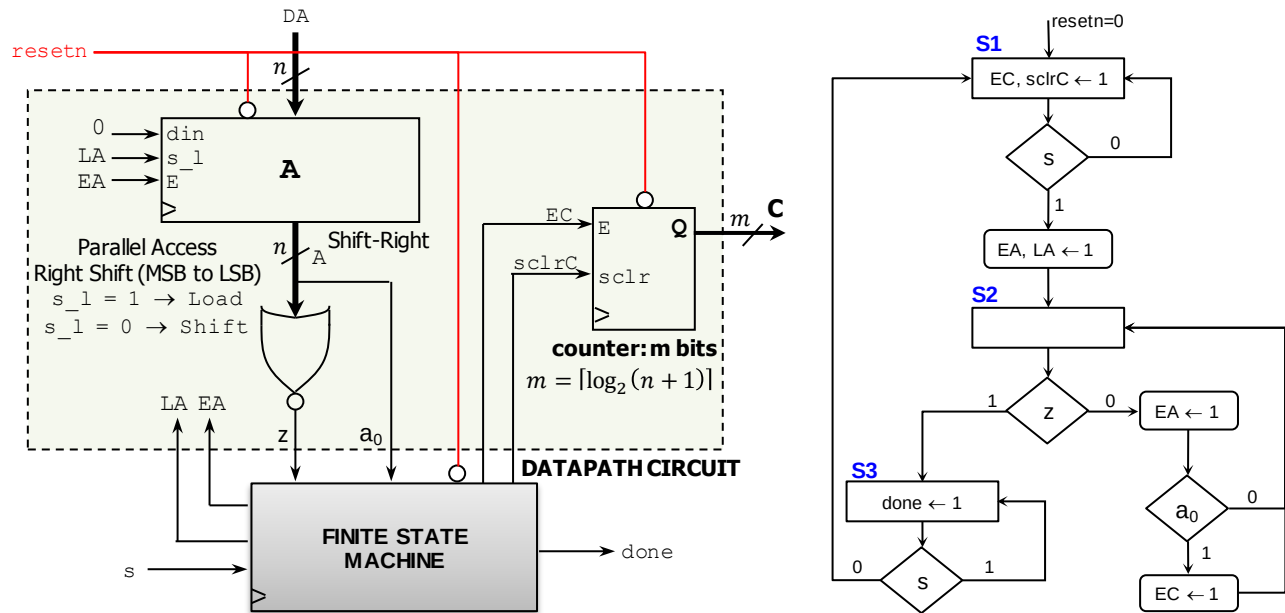
Homework 1

(Due date: May 18th)

Presentation and clarity are very important! Show your procedure!

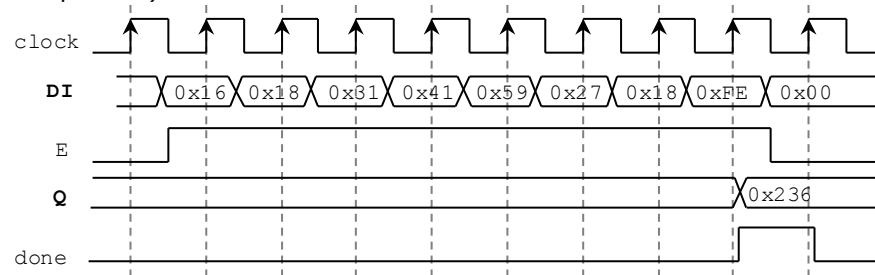
PROBLEM 1 (40 PTS)

- Bit Counting (or "Counting 1's") Circuit: It counts the number of bits in register A that has the value of '1'. The digital system is depicted below: FSM + Datapath. Example: For $n = 8$: if $A = 00110110$, then $C = 0100$.
 - ✓ m-bit counter: If $E = sclr = 1$, then $Q \leftarrow 0$. If $E = 1, sclr = 0$, then $Q \leftarrow Q+1$
 - ✓ Parallel access shift register: If $E = 1, s_l = 1 \rightarrow$ Load. If $E = 1, s_l = 0 \rightarrow$ Shift.
- Complete the timing diagram where $n = 8, m = 4$.



PROBLEM 2 (60 PTS)

- VHDL Description of Accumulator of eight 8-bit unsigned integer numbers.
- Operation:** The circuit starts reading 8-bit data when the *s* signal (usually a one-cycle pulse) is asserted. The values then are read when the enable signal (*E*) is asserted. When 8 values have been read, the *done* signal is asserted to indicate that the accumulated result in 11-bit output *Q* is valid.
 - Inputs: *s* (start signal), *DI* (8-bit input data), *E* (enable for input data)
 - Outputs: *Q* (11-bit output data)



- The figure depicts the digital system: the FSM (in ASM form) and the Datapath circuit.
 - Write a structural VHDL code. You should use the following parametric components (available [here](#)):
 - n*-bit register with enable and synchronous clear: `my_rege`. Assign parameter *N* = 11.
 - Counter modulo-*N* with enable and synchronous clear: `my_genpulse_sclr`. Assign parameter COUNT = 8.
 - n*-bit adder/subtractor: `my_addsub`. Assign parameter *N* = 11.

Then, write code for the FSM. Finally, integrate all components into a top file (named `my_accu.vhd`). Make sure to include the following library declarations: `'use ieee.math_real.log2;'` `'use ieee.math_real.ceil;'`

- Use the provided testbench (`tb_my_accu.vhd`) and simulate the circuit (Behavioral Simulation). The testbench feeds two sets of 8-bit data. Verify that the simulation results are correct.
 - 1st Set: 0x19, 0xFA, 0xCA, 0xDE, 0xFA, 0xCE, 0xB0, 0x0C. The result on *Q* (when *done*=1) should be 0x53F.
 - 2nd Set: 0x16, 0x18, 0x31, 0x41, 0x59, 0x27, 0x18, 0xFE. The result on *Q* (when *done*=1) should be 0x236.
- Upload (as a .zip file) the following files to Moodle (an assignment will be created).
 - VHDL code files and testbench.
 - A screenshot of your simulation showing the proper results on *Q*.

