

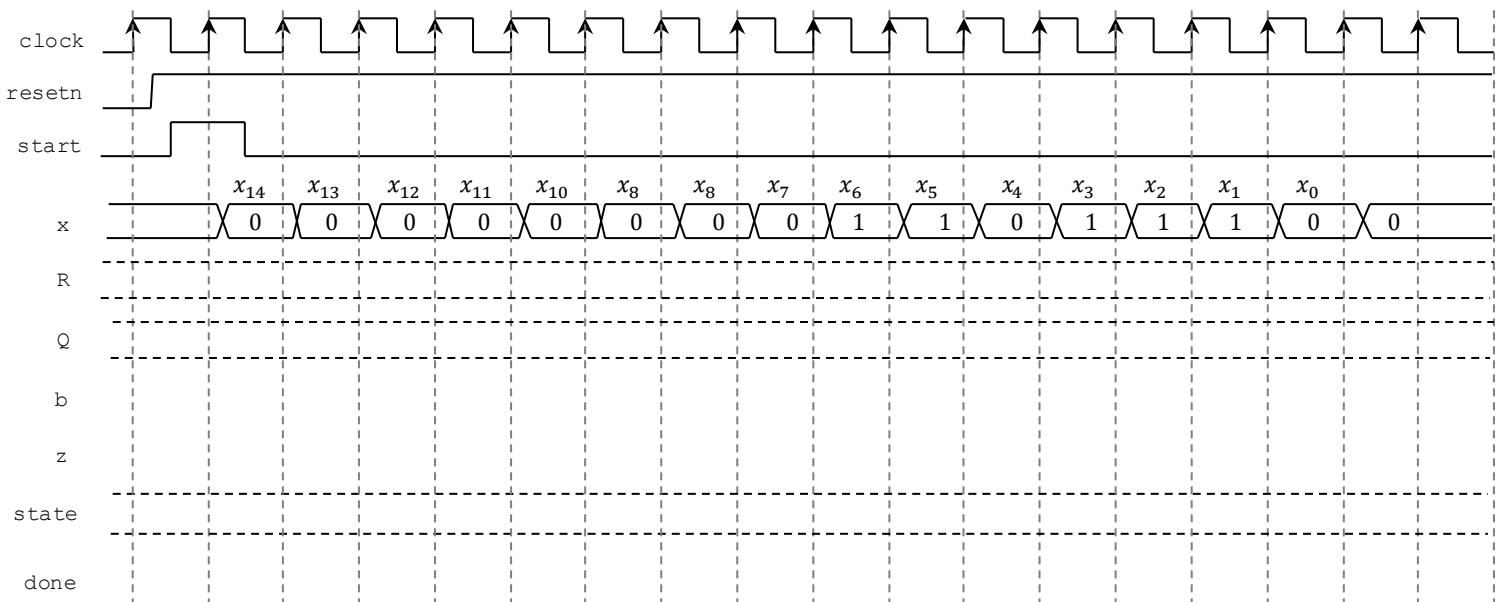
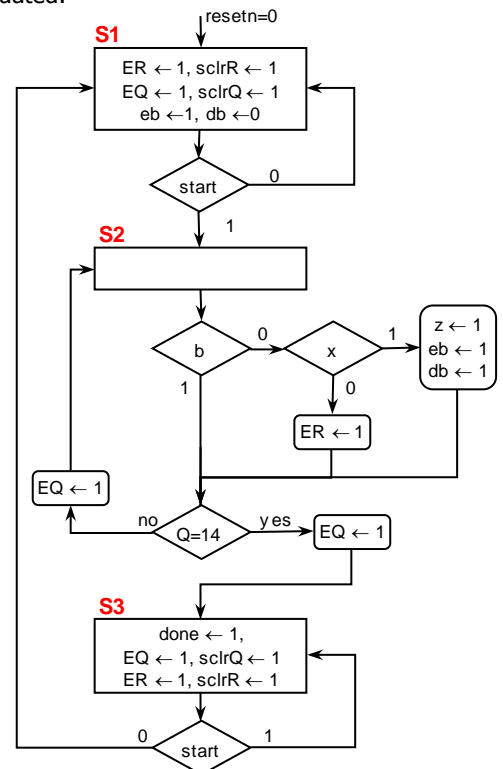
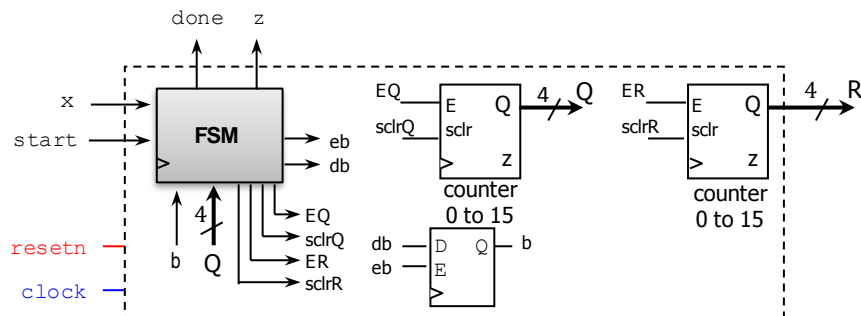
Homework 1

(Due date: February 1st @ 7:30 pm)

Presentation and clarity are very important! Show your procedure!

PROBLEM 1 (10 PTS)

- Leading Zero Detector: This iterative circuit processes a 15-bit input (MSB first) and generates the number of leading 0's. Example:
 ✓ If the sequence is: 0000 0000 0111 010 $\rightarrow R = 9$
 ✓ If the sequence is: 0001 0001 0011 010 $\rightarrow R = 3$
- The figure depicts the (in ASM form) and a datapath circuit. Note: Counters. If $E=sclr=1, \rightarrow Q=0$.
 Input data: x (entered sequentially, MSB first). Output data: R.
 ✓ Complete the timing diagram of the digital circuit where one sequence is evaluated.



PROBLEM 2 (40 PTS)

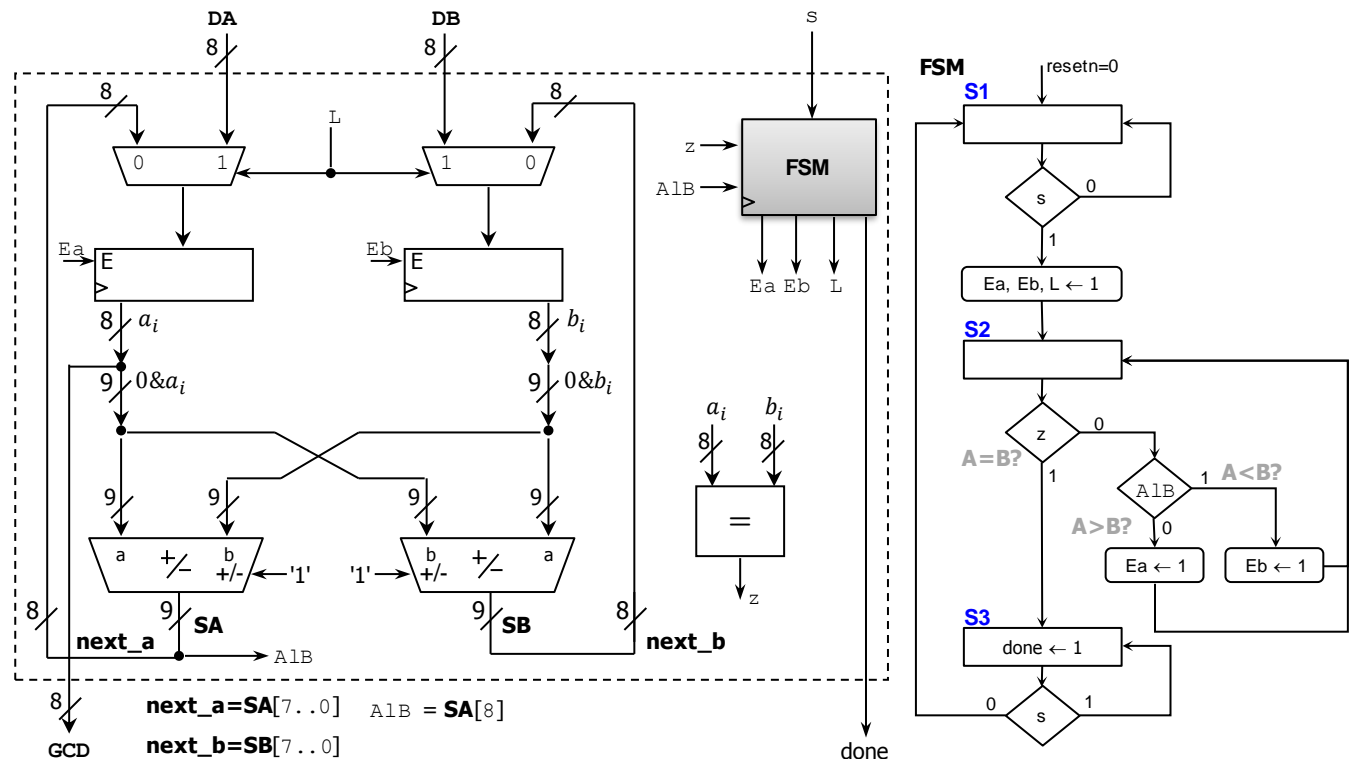
- **Greatest Common Divisor (GCD):** This iterative circuit processes two 8-bit unsigned numbers (A, B) and generates the GCD of A and B. For example:
 - ✓ If $A = 216, B = 192 \rightarrow \text{GCD} = 24$.
 - ✓ If $A = 132, B = 72 \rightarrow \text{GCD} = 12$.
 - ✓ If $A = 169, B = 63 \rightarrow \text{GCD} = 1$.
- The circuit is based on the Euclid's GCD Algorithm:

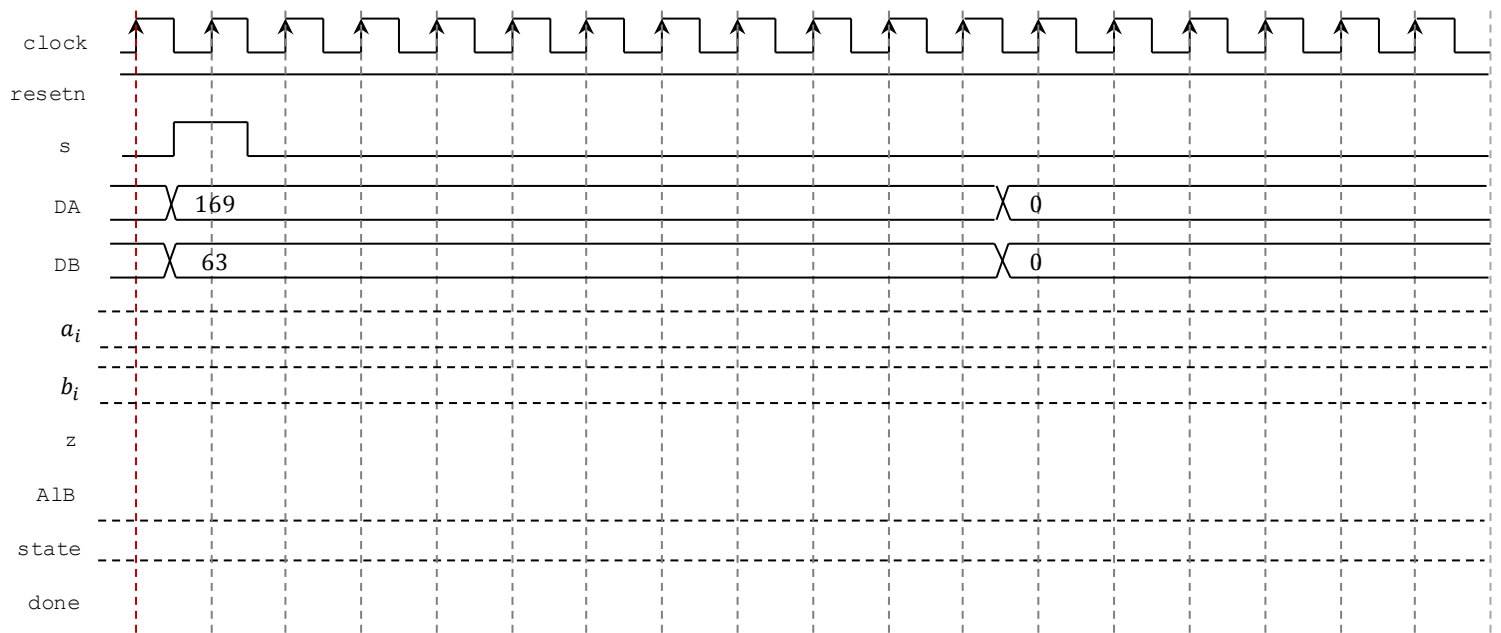
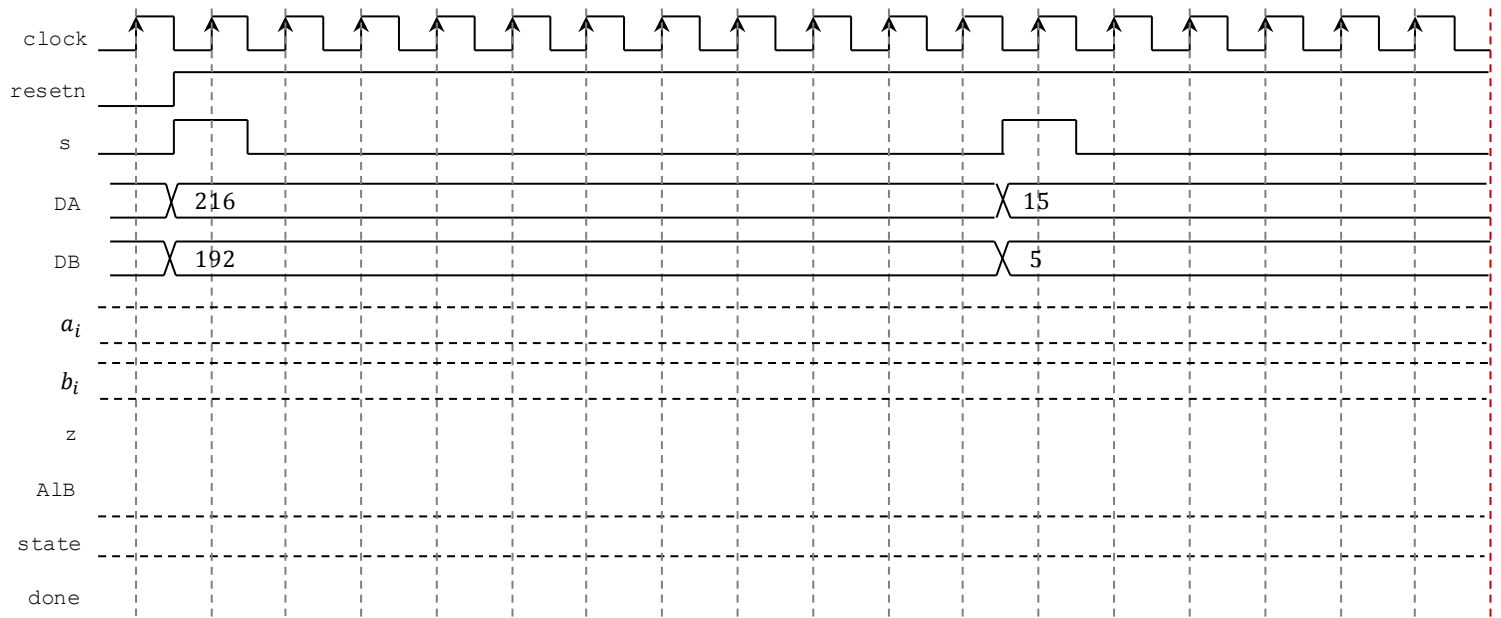

```

a,b: unsigned integers
while a ≠ b
  if a > b
    a ← a-b
  else
    b ← b-a
  end
end
return a
      
```
- The figure depicts the (in ASM form) and a datapath circuit.

Input data: DA, DB. Output data: GCD.

 - ✓ Complete the timing diagram of the digital circuit (next page). Note that 3 pairs of numbers are evaluated.
 - ✓ Write a structural VHDL code. You MUST create (or re-use) a file for i) N-bit register, ii) Adder/Subtractor, iii) Bus MUX 2 to 1, iv) Finite State Machine, and v) Top file (where you will interconnect all the components).
 - ✓ Write a testbench according to the timing diagram shown (next page). Simulate the circuit (Behavioral simulation). Verify that the simulation is correct by comparing it with the timing diagram you completed manually.
 - ✓ Upload (as a .zip file) the following files to Moodle (an assignment will be created):
 - VHDL code files
 - VHDL testbench

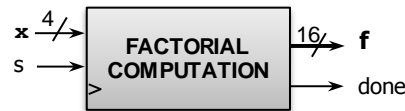




PROBLEM 3 (25 PTS)

- **Factorial Computation:** The following algorithm computes the factorial of an unsigned number.

```
x: unsigned integer
f = 1
if x ≠ 0
  for i = 1 to x
    f ← f×i
  end
end
return f
```



- We want to design a circuit that reads in an unsigned number (x) and generates $f = x!$.
- Operation: The circuit reads data in when the s signal (usually a one-cycle pulse) is asserted. When the result is ready, the signal $done$ is asserted.
 - ✓ Inputs: x (input data), s (start signal).
 - ✓ Outputs: f (factorial), $done$.
 - ✓ We restrict the bitwidth of f to 16 bits. As a result, the largest x is 8 (as $8! = 40320$).
- Sketch the circuit: FSM + Datapath components. Specify all the I/Os of the FSM, as well as the signals connecting the FSM and the Datapath components (as in Problem 2).
 - ✓ You can use an array multiplier as a component. This multiplier will multiply two values: f and i (as per the algorithm).
 - Note: the multiplication only needs 16 bits (even if the sum of the input bitwidths is greater than 16 bits). Thus, some MSBs will need to be discarded (not a problem since the largest f fits with 16 bits).
 - ✓ Feel free to use any other standard component (e.g. counter, register, comparator, busmux).
 - ✓ Your circuit should compute any factorials from $x = 0$ to $x = 8$.
 - ✓ Provide the State Diagram (in ASM form) of the FSM.

PROBLEM 4 (15 PTS)

- Calculate the result of the following operations, where the operands are signed integers. For the division, calculate both the quotient and the residue. **No procedure = zero points.**

10011 × 11011	10010 × 01001	100101 ÷ 1101	01111010 ÷ 100	100010 ÷ 0101
------------------	------------------	------------------	-------------------	------------------

PROBLEM 5 (10 PTS)

- Compute the result of the additions and subtractions for the following fixed-point numbers.

UNSIGNED (1 pt. each)		SIGNED	
0.101010 + 1.0110101	1.00101 - 0.0000111	10.001 + 1.001101	0.0101 - 1.0101101
	100.1 + 0.1000101	1000.0101 - 111.01001	101.0001 + 1.0111101