Homework 1

(Due date: January 30th @ 11:59 pm)

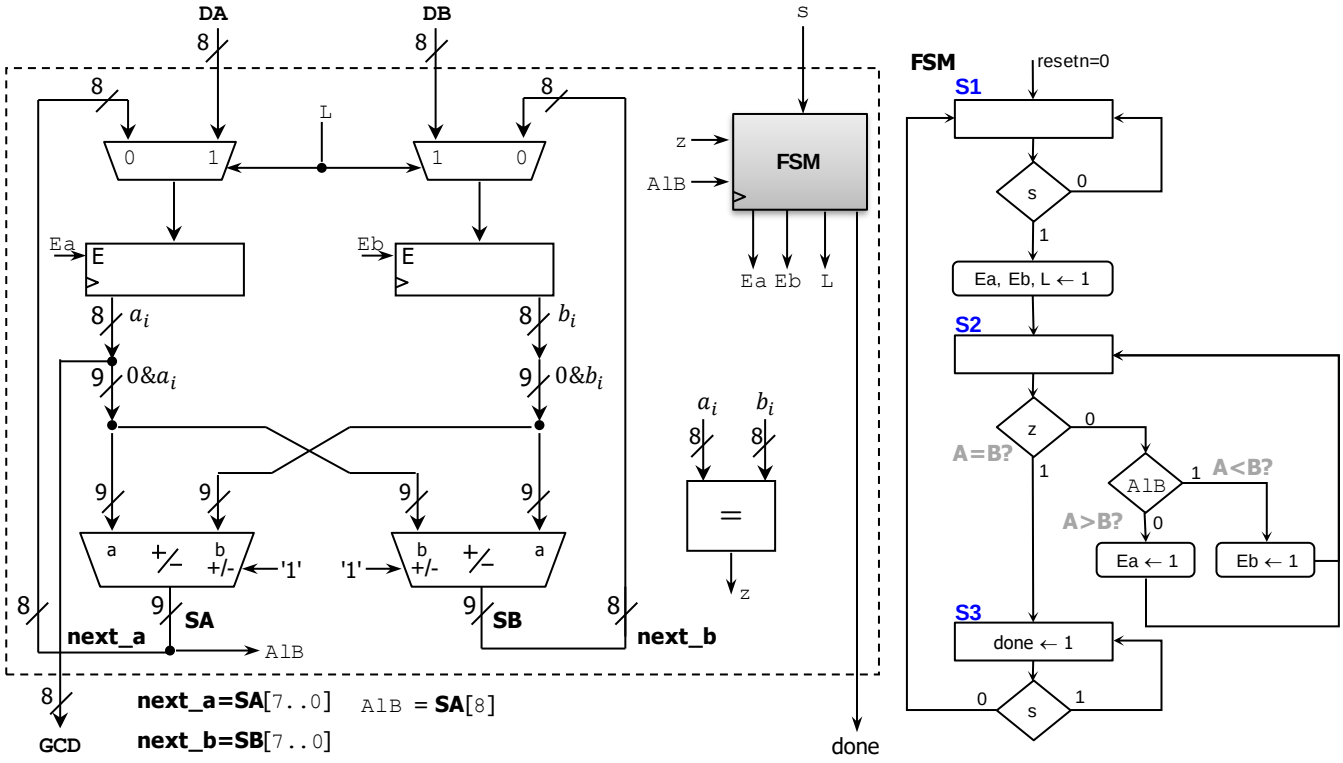
Presentation and clarity are very important! Show your procedure!

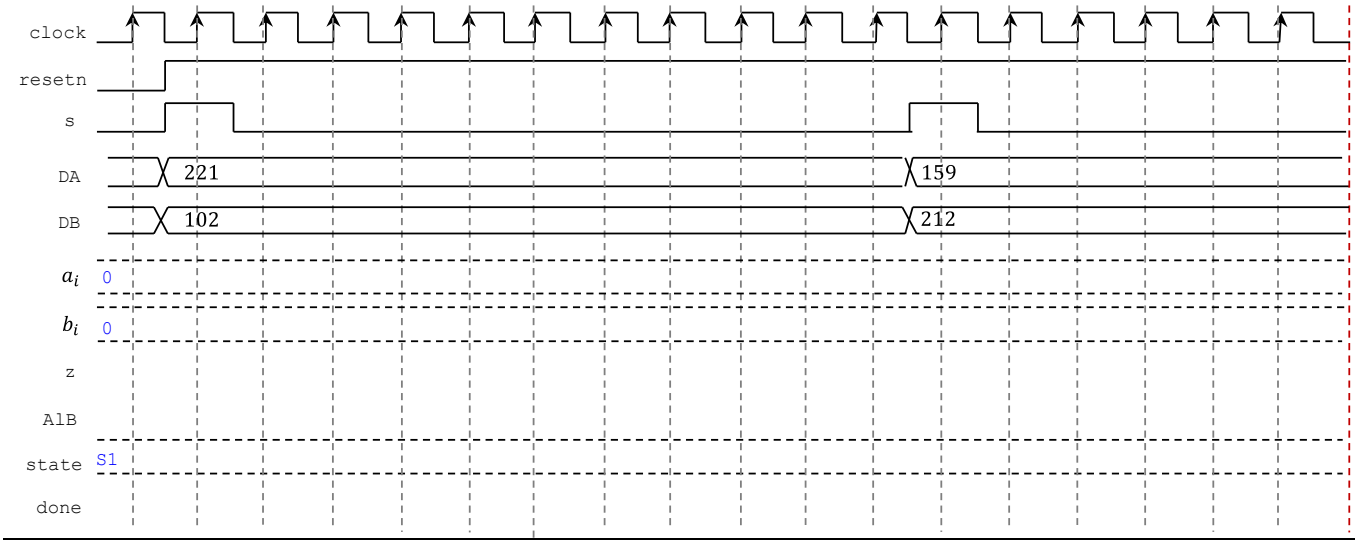
PROBLEM 1 (15 PTS)

- **Greatest Common Divisor (GCD):** This circuit processes two n-bit unsigned numbers (A, B) and generates the GCD of A and B. For example:
  - ✓ If A = 132, B = 72 → GCD = 12.
  - ✓ If A = 216, B = 192 → GCD = 24.
  - ✓ If A = 169, B = 63 → GCD = 1.
- The digital system (based on a sequential algorithm) is depicted in the figure below, and includes an FSM (in ASM) and a datapath circuit.
- Complete the timing diagram of the digital system (DA, DB, a_i, b_i are decimal values)

Sequential Algorithm

```
a,b: unsigned integers
while a ≠ b
    if a > b
        a ← a-b
    else
        b ← b-a
    end
end while
return a
```





PROBLEM 2 (45 PTS)

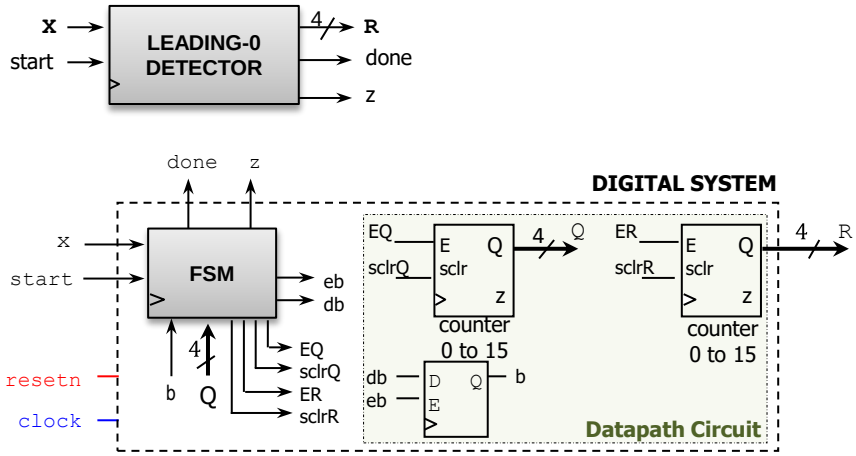
- **Leading Zero Detector (LZD)**: This iterative circuit processes a 15-bit input (MSB first) and generates the number of leading 0's. before the first 1. Example:
  - ✓ If the sequence is: 0000 0001 1000 001 → R = 7
  - ✓ If the sequence is: 0000 0000 0011 010 → R = 10
  - ✓ If the sequence is: 0001 0000 0011 010 → R = 3

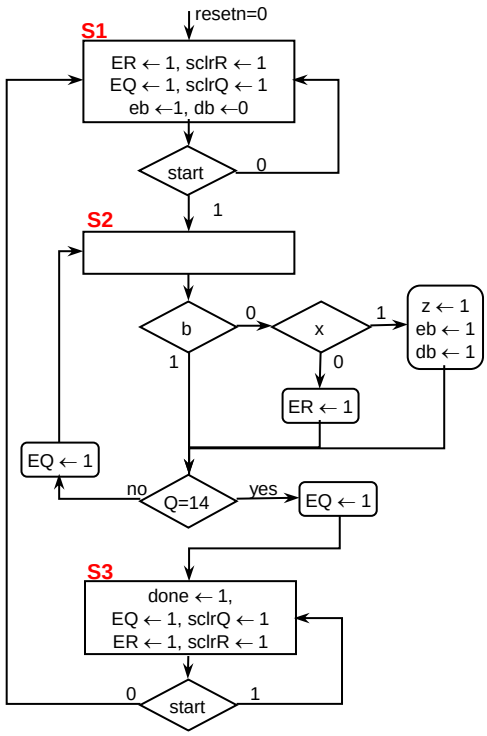
- **Iterative architecture:** The figure depicts the FSM (in ASM form) and a datapath circuit.
  - ✓ Inputs: X (serial data, MSB first), start.
  - ✓ Outputs: z (leading 1 detected), done (N bits processed), and R (number of 0's before the first 1).
  - ✓ Processing begins when start is asserted. When the first '1' is detected on X, then z=1 (for 1 clock cycle) and R is frozen. When all the bits of the sequence have been processed, done=1. We can re-start the process after this.
    - Note: if X is just 0's, z is never '1'.
  - ✓ Counters R and Q: modulo-(N+1): count from 0 to N. (N=15).

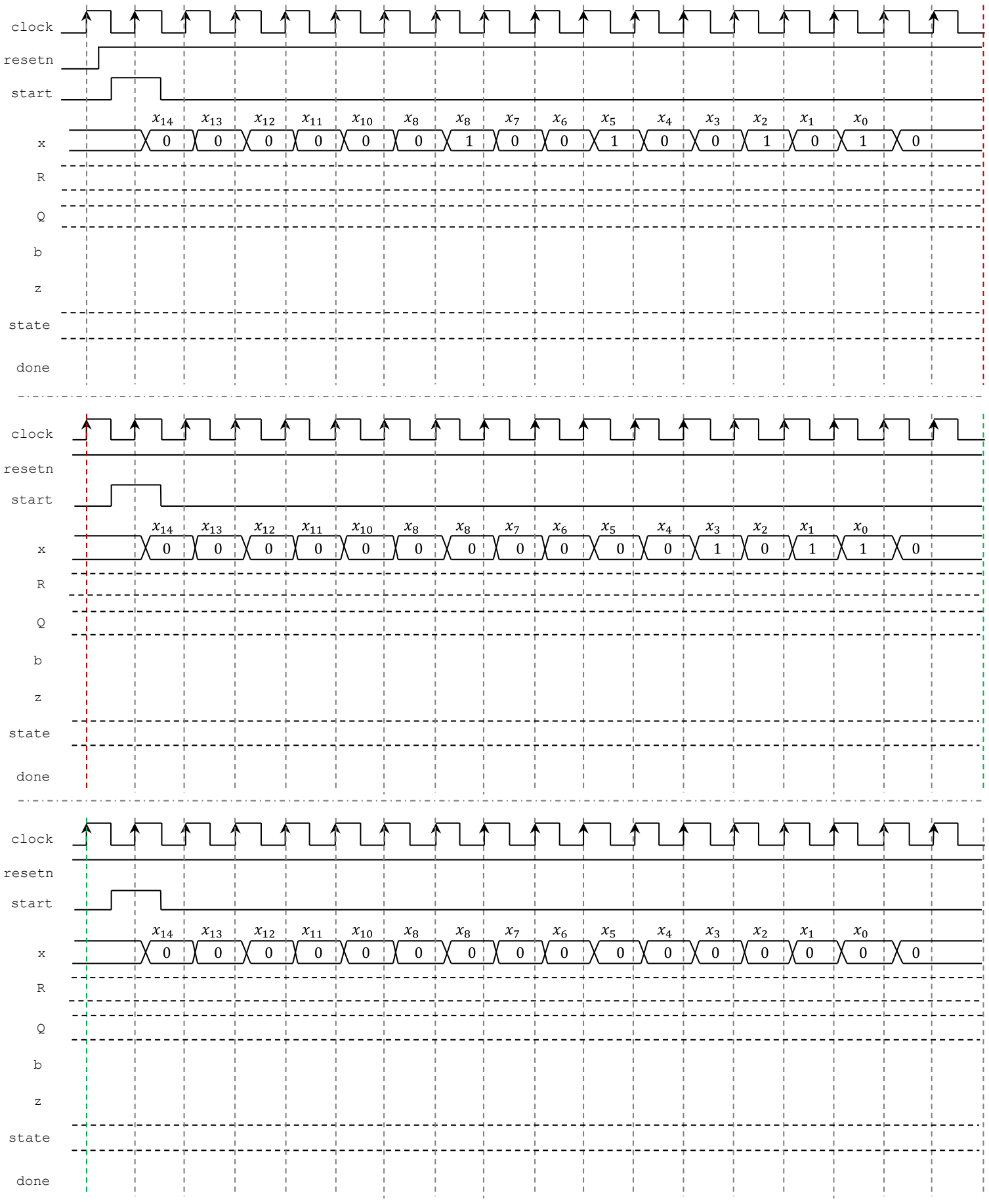
	Counter modulo-N (0 to N+1): If E=0, the count stays.

Generic component - counter (my_genpulse_sclr):
Behavior on the clock tick:
	`if E = 1 then`
	`  if sclr = 1 then`
	`  Q ← 0`
	`  else`
	`  Q ← Q+1`
	`  end if;`
	`end if;`
	`* z=1 if Q = N-1 (max. count)`

- **Procedure:**
  - ✓ Complete the timing diagram of the digital circuit (next page). Note that 3 sequences are evaluated.
  - ✓ Write a structural VHDL code. You MUST create a file for i) modulo-16 counter, ii) flip-flop, iii) 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). Perform Behavioral Simulation. Verify that the simulation is correct by comparing it with the timing diagram you completed manually.
    - R and Q are specified as decimals.
  - ✓ Upload (as a .zip file) the following files to Moodle (an assignment will be created):
    - VHDL design source files
    - VHDL testbench
    - A simulation screenshot, showing the processing of the first data input (000000100100101).






clock
resetn
start
x
R
Q
b
z
state
done
x_{14} x_{13} x_{12} x_{11} x_{10} x_8 x_8 x_7 x_6 x_5 x_4 x_3 x_2 x_1 x_0
0 0 0 0 0 0 1 0 0 1 0 0 1 0 1 0
0 0 0 0 0 0 0 0 0 0 0 1 0 1 1 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

PROBLEM 3 (25 PTS)

- **Fibonacci numbers Computation**: We want to design a circuit that reads an unsigned number ($n > 1$) and generates F_n:

$$F_n = F_{n-1} + F_{n-2},\ F_0 = 0,\ F_1 = 1$$

  - ✓ To compute F_n in an iterative fashion, we can use:

```
n: unsigned integer
F0=0, F1=1
if n > 1
    for i = 2 to n
         Fi = Fi-1 + Fi-2
    end
end
return Fn
```

- Operation: The circuit reads n when the s signal (usually a one-clock cycle) is asserted. When the result (F_n) is ready, the signal done is asserted.
  - ✓ Input: s (start signal), n (input data)
  - ✓ Outputs: Fn (result), done.
  - ✓ Clock frequency: 100 MHz.
  - ✓ We restrict the bitwidth of Fn to 16 bits. As a result, the largest n would be 24 ($F_{24} = 46368$).

- Sketch the circuit: FSM (in ASM form) + 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).
  - ✓ Suggestion: Use two registers (for F_{i-1} and F_{i-2}), initialized with $F_1 = 1$ and $F_0 = 0$ (iterations start at $i = 2$). At every iteration ($i = 2, \dots, n$), F_i is computed, and then the registers (parametric register with enable: my_rege) are updated:
    - Register F_{i-1}: It captures the computed F_i.
    - Register F_{i-2}: It captures F_{i-1}.
  - ✓ Feel free to use any other standard component (e.g.: register, counter, comparator, adder, busmux).
  - ✓ The iteration index i can be implemented with a standard counter (my_genpulse_sclr). Here, you can only initialize the count to 0. To ensure that iterations start at $i = 2$, you can include more states (to increase the count).
  - ✓ To simplify the design, you can assume that $n > 1$.

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.**

01011 × 10010	10010 × 11011	10101 ÷ 1011	1010011 ÷ 0110	011101 ÷ 1101