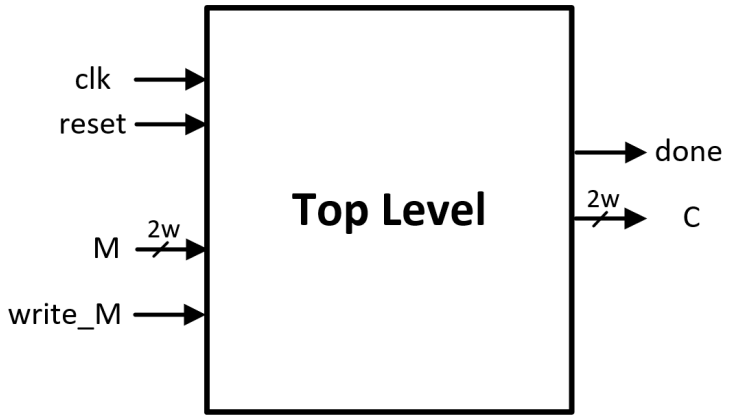
# Midterm Exam ECE 448
# Spring 2019
# Wednesday, March 6
# 15 points

Instructions:
Zip all your deliverables into an archive <last_name>.zip and submit it through Blackboard no later than Wednesday, March 6, 10:00 PM EST.

The EXAM cipher implementation is specified below using its

- Top-level circuit interface
- Interface with the division into the Datapath and Controller
- Table of input/output ports
- Pseudocode
- Block diagram of the Datapath
- ASM chart of the Controller.

**Top-level circuit interface**




**Interface with the division into the Datapath and Controller**

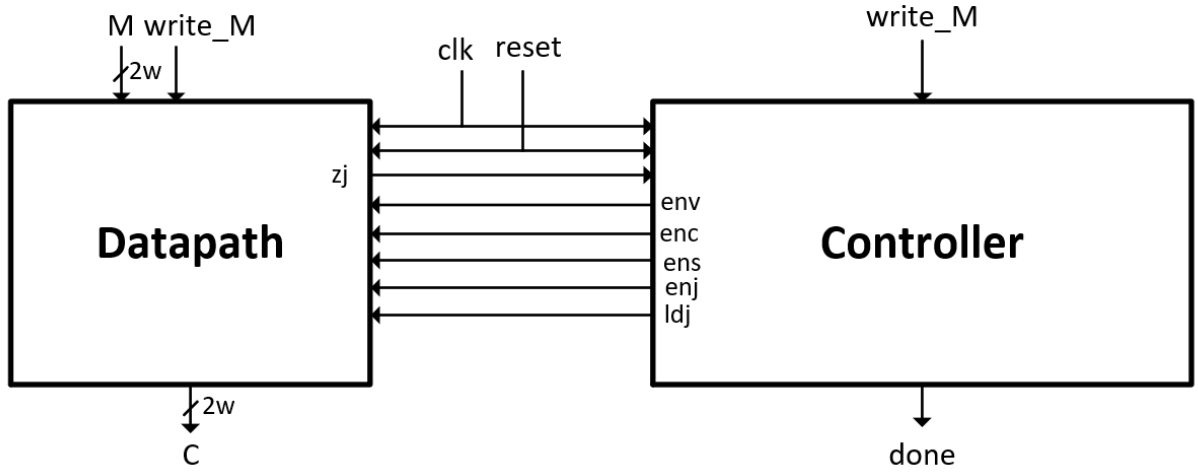



**Table of input/output ports**

| Port | Width | Meaning |
|---|---|---|
| clk | 1 | System Clock |
| reset | 1 | System reset – clears all internal registers. Active high. |
| M | 2w | Message block |
| write_M | 1 | Synchronous write control signal for the message block M. After the block M is written to the EXAM unit, the encryption of M starts automatically |
| C | 2w | Ciphertext block = Encrypted block M. |
| done | 1 | Asserted when ciphertext is ready and available at the output. |

**Pseudocode**

The EXAM cipher is defined below:

An input message block M has 2·w bits (where w is a parameter of the cipher).
The corresponding ciphertext block (i.e., encrypted message block) has also 2·w bits.
In order to encrypt a message block M, the algorithm performs the following operations:

```
Split M into two equal parts V0, V1, each of the size of w bits
SUM = 0
for j= 1 to r do
{
      W00 = ((V1 << 4) xor (V1 >> 5)) + V1
      W01 = SUM + KEY[SUM mod 4]
      T0 = W00 xor W01
      V0' = V0 + T0

      SUM' = SUM + DELTA

      W10 = ((V0' << 4) xor (V0' >> 5)) + V0'
      W11 = SUM' + KEY[(SUM'>>6) mod 4]
      T1 = W10 xor W11
      V1' = V1 + T1

      SUM = SUM'
      V0 = V0'
      V1 = V1'
}
C = V0 || V1
```

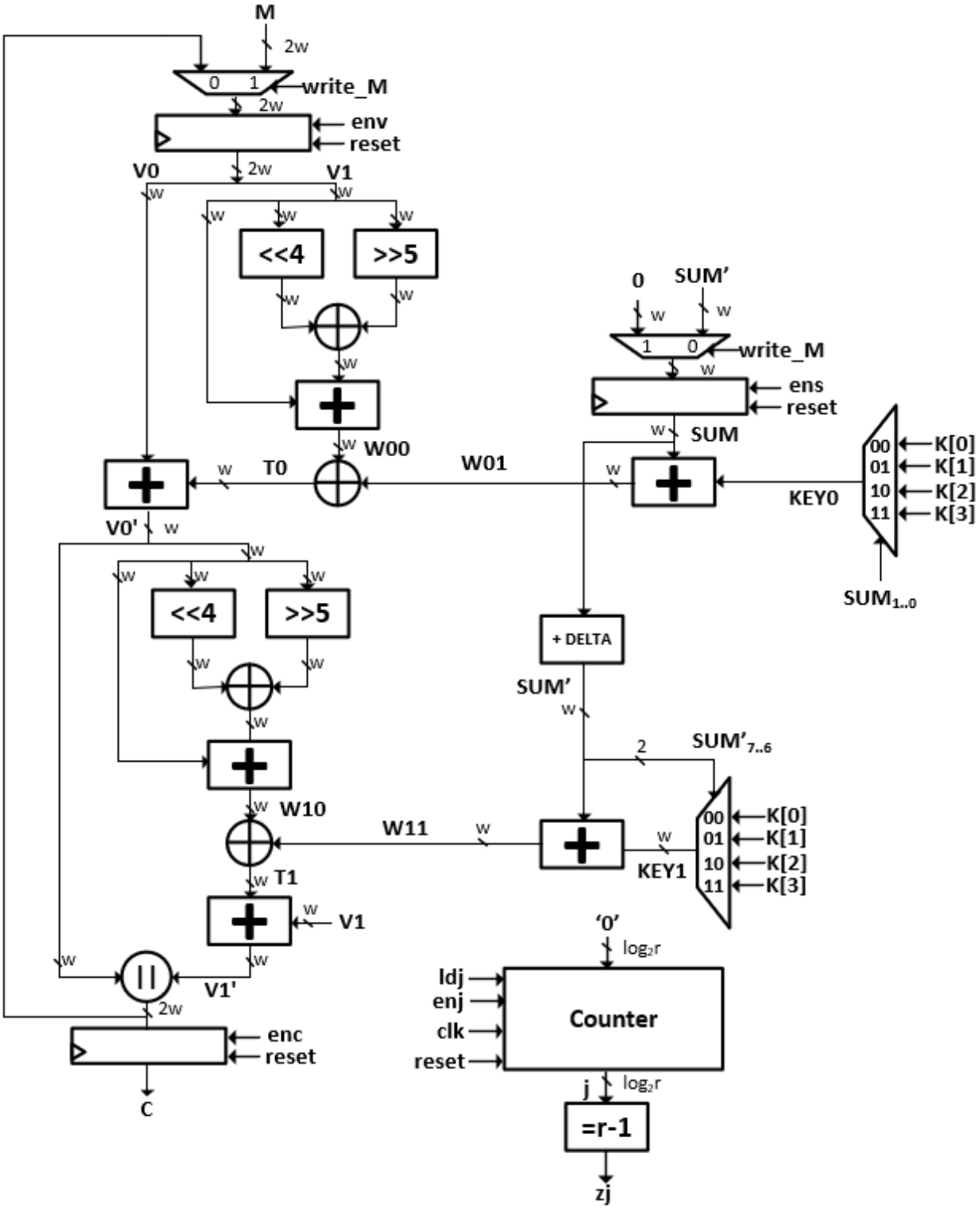
**Notation:**

V0, V1, V0', V1', W00, W01, W10, W11, T0, T1, SUM, SUM': w-bit variables
DELTA: a w-bit constant
K[0], K[1], K[2], K[3]: a set of 4 round keys; each round key is a w-bit constant
xor : an XOR of two w-bit words
+ : unsigned addition mod $2^w$
A << k : logic shift left by k positions
A >> k : logic shift right by k positions
A || B = concatenation of A and B.
Please note that the algorithm has two parameters:

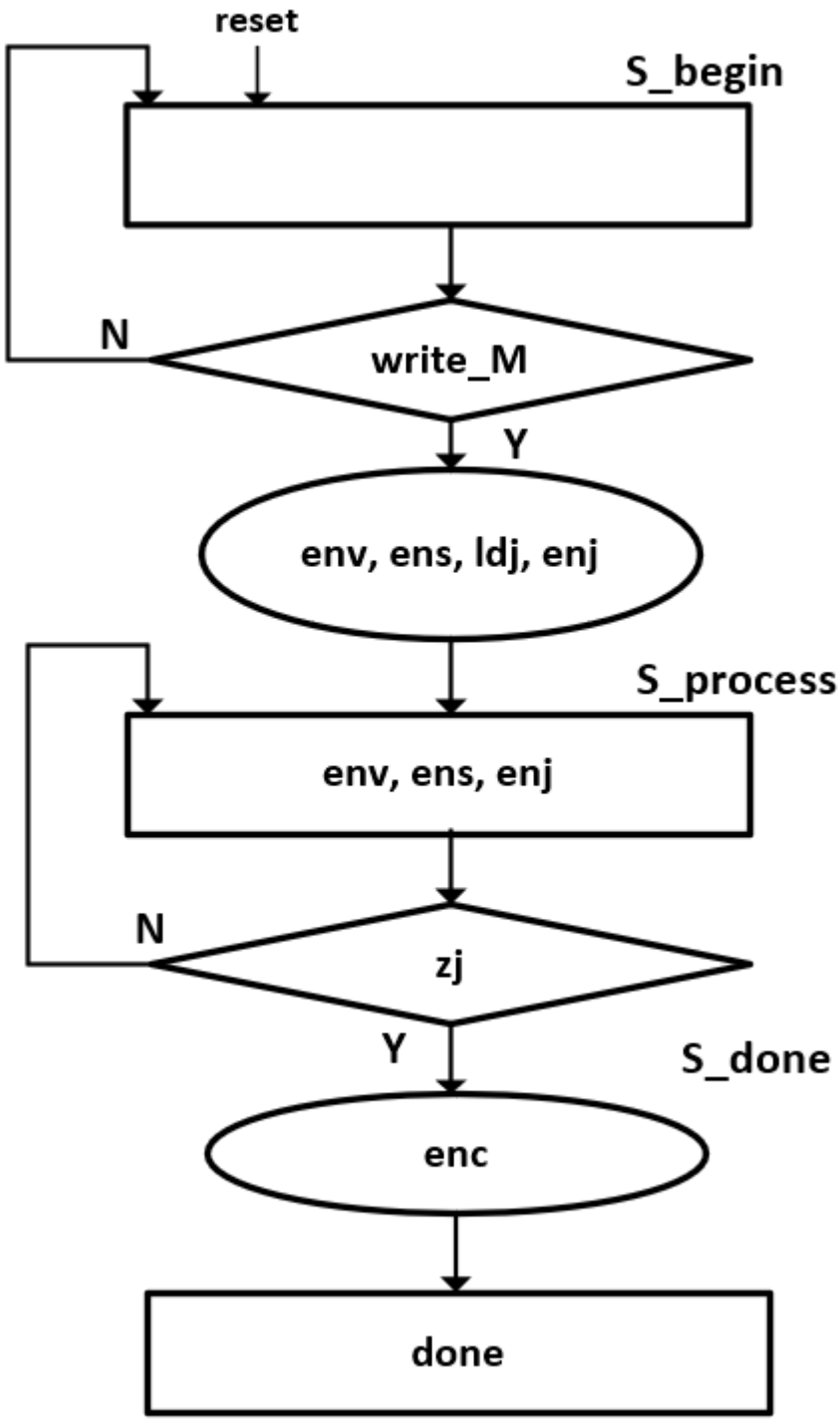
- r = number of rounds (e.g., 4)
- w = word size (always a power of 2, e.g., w = $2^3$ = 8)

These parameters should be treated as generics.

**Block diagram of Datapath**



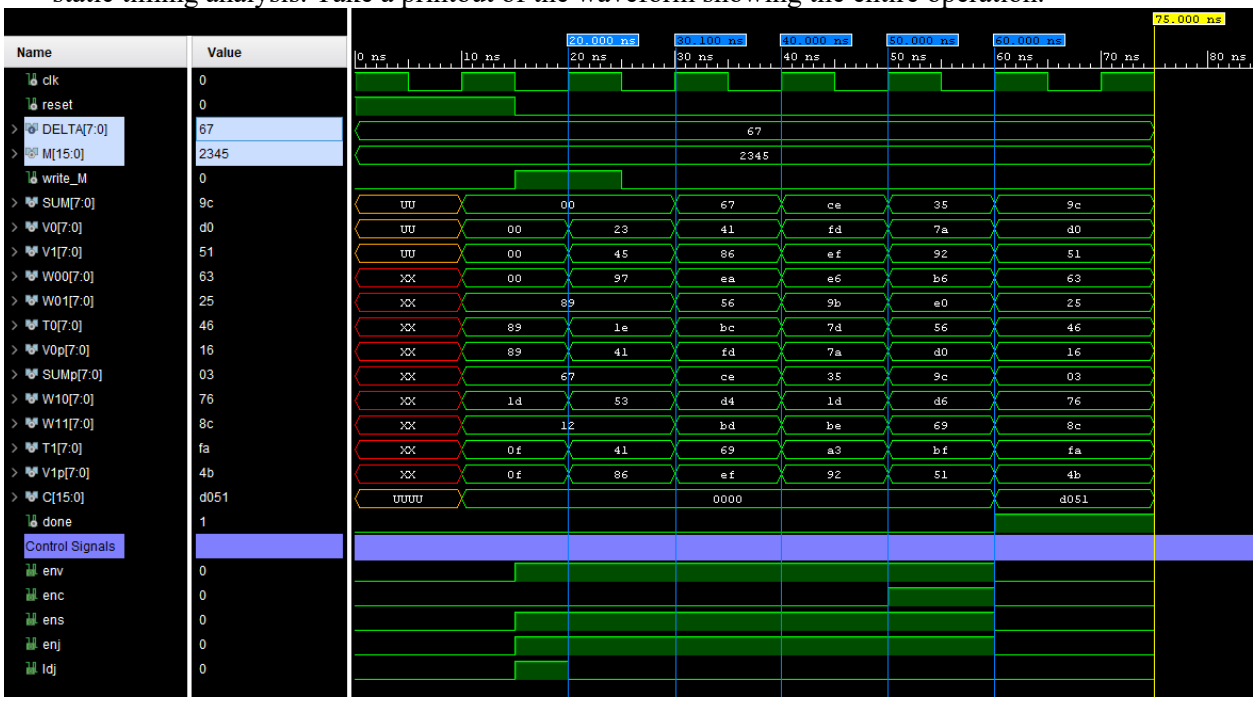
**ASM Chart**


reset
S_begin
N
write_M
Y
env, ens, ldj, enj
S_process
env, ens, enj
N
zj
Y
S_done
enc
done

**Tasks:**

Perform the following tasks:

1. Write a synthesizable VHDL code representing the circuit.
2. Write a testbench verifying the operation of your circuit for the following values of parameters and round keys:
   **w=8, r=4, DELTA = 0x67, KEY[0]=0x89, KEY[1]=0xAB, KEY[2]=0xCD, KEY[3]=0xEF**.
   **Assume that the message M is equal to M=0x2345**.
   **Debug your circuit by comparing its operation with the enclosed timing waveforms**.
3. Perform functional simulation of your circuit and use it to debug your VHDL code. Take a printout of the waveform showing the entire operation.
4. Synthesize your circuit.
5. Implement your circuit using
   a. FPGA family: Artix-7
   b. Part name: XC7A35TCPG236-1
   c. Speed Grade: -1
6. Run the static timing analysis of your circuit. Determine the minimum clock period and the maximum clock frequency.
7. Based on the implementation reports, determine the number of CLB slices, LUTs, flip-flops, and pins used by the circuit.
8. Perform the timing simulation of your circuit at the maximum clock frequency returned by the static timing analysis. Take a printout of the waveform showing the entire operation.




**Deliverables:**

1. VHDL code of your entire circuit.
2. VHDL code of your testbench.
3. Timing waveforms from the functional and timing simulations demonstrating the correct operation of your circuit.
4. FPGA resource utilization (as defined in Task 7 above).
5. Minimum clock period and maximum clock frequency of your circuit.