Problem 1 (5 points)

Draw a block diagram of the datapath unit of a circuit capable of executing the pseudocode given below.

In this code, MEM_A and MEM_B represent two single-port memories of the size of 128 x 8. These memories are initialized during the period when s=0, using external ports (dina, addra, wra) and (dinb, addrb, wrb), respectively. After the calculations are over, the results are read using ports (douta, addra, rda), (doubt, addrb, rdb), and (sel_out, min_avr).

begin:

done = 0
initialization of memories MEM_A, MEM_B
wait for s=1
SUM_ABS = 0x00
MIN_ABS = 0xFF

for i=0 to 63 do
    A = MEM_A[i]
    B = MEM_B[i]
    if (A > B) then
        ABS = A - B
        GT = A
    else
        ABS = B - A
        GT = B
    end if

    SUM_ABS = SUM_ABS + ABS
    if (ABS < MIN_ABS) then
        MIN_ABS = ABS
    end if

    MEM_A[64+i] = GT
    MEM_B[64+i] = ABS
end for

AVR_ABS = SUM_ABS/64

done = 1
wait for s=0
reading the contents of memories MEM_A, MEM_B
go to begin

Please clearly mark the widths of all buses in your circuit.
Assume the following interface to your circuit:

<table>
<thead>
<tr>
<th>Port</th>
<th>Width</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>clk</td>
<td>1</td>
<td>System clock.</td>
</tr>
<tr>
<td>reset</td>
<td>1</td>
<td>System reset – clears all internal registers and counters. Active high.</td>
</tr>
<tr>
<td>s</td>
<td>1</td>
<td>Operating mode: 0 = initialization/reading results, 1 = processing.</td>
</tr>
<tr>
<td>dina</td>
<td>8</td>
<td>Input data bus for MEM_A</td>
</tr>
<tr>
<td>dinb</td>
<td>8</td>
<td>Input data bus for MEM_B</td>
</tr>
<tr>
<td>addra</td>
<td>7</td>
<td>Address bus for MEM_A</td>
</tr>
<tr>
<td>addrb</td>
<td>7</td>
<td>Address bus for MEM_B</td>
</tr>
<tr>
<td>wra</td>
<td>1</td>
<td>External write enable for MEM_A</td>
</tr>
<tr>
<td>wrb</td>
<td>1</td>
<td>External write enable for MEM_B</td>
</tr>
<tr>
<td>douta</td>
<td>8</td>
<td>Output data bus for MEM_A</td>
</tr>
<tr>
<td>doutb</td>
<td>8</td>
<td>Output data bus for MEM_B</td>
</tr>
<tr>
<td>rda</td>
<td>1</td>
<td>External read enable for output douta. 0 = high impedance on douta, 1 = valid output douta</td>
</tr>
<tr>
<td>rdb</td>
<td>1</td>
<td>External read enable for output doutb. 0 = high impedance on doutb, 1 = valid output doutb</td>
</tr>
<tr>
<td>min_avr</td>
<td>8</td>
<td>One of the two results calculated by the circuit.</td>
</tr>
<tr>
<td>sel_out</td>
<td>1</td>
<td>Selection between the two calculated results: 0 = avr, 1 = min.</td>
</tr>
<tr>
<td>done</td>
<td>1</td>
<td>Asserted when all results are ready, zero otherwise</td>
</tr>
</tbody>
</table>

**Problem 2 (5 points)**

1. Draw an ASM chart corresponding to the pseudocode from Problem 1.
2. Express all operations in your ASM chart in terms of active values of control signals generated as outputs of the Control unit and used as inputs in the Datapath.

**Problem 3**

**Task 1 (5 points)**

Draw a detailed block diagram of the digital system including

1. the PicoBlaze 6 core, KCPSM6
2. the instruction ROM required for the basic operation of the PicoBlaze core
3. 64 x 8 external data RAM visible under addresses 0x00-0x3F
4. 64 x 8 external data ROM visible under addresses 0x40-0x7F
5. two input registers with the virtual addresses 0x80 and 0xC0
6. two output registers with the virtual addresses 0x80 and 0xC0
7. a D flip-flop with the output Q connected to the interrupt input of the PicoBlaze core, input SET connected to the external port INT, and input CLR connected to an appropriate output of the PicoBlaze core.

Assume that

- input register with the address 0x80 is the same as the output register with the address 0x80
- the input and output registers, data RAM, and data ROM specified above are the only I/O devices that the PicoBlaze core is communicating with
• your system needs to be able to allow the PicoBlaze core to write to all aforementioned output registers and data RAM, and read from all the aforementioned input registers, data RAM, and data ROM, using instructions OUTPUT and INPUT, respectively
• you need to provide all details of the address decoder, and build it out of basic logic components you are familiar with
• all registers and flip-flops have a reset input connected to the external port RESET.

Please clearly mark on your schematic:
• sizes of all memories and registers
• sizes and directions of all buses.

**Task 2 (5 points)**

Determine the contents of
1. PC
2. Stack
3. Flags I, preserved C, and preserved Z
4. All output registers
5. Internal registers s0, s1, s2, s3
6. Flags C, Z

at the time of the execution of the instruction **RETURNI ENABLE** before this instruction takes effect assuming that
• at the time = 0 a short pulse is generated at the input RESET
• at the time = 5 seconds, a short pulse is generated at the input INT
• the contents of the instruction memory is given by the following program:

```assembly
CONSTANT BIT0_COMP, FE
CONSTANT BIT6, 40

ADDRESS 000
JUMP START

ADDRESS 100

START:
LOAD s2, 40
LOAD s3, 81

INIT:
SR1 s3
ADD s2, 40
OUTPUT s3, (s2)
JUMP NZ, INIT
ENABLE INTERRUPT

STORE s3, 00
FETCH s2, 00
INPUT s3, 00

CALL SUB
JUMP START
```
SUB:
  CALL LOOP
  JUMP START

LOOP:
  JUMP LOOP

ISR:
  ADDRESS 300
  INPUT  s0, 80
  LOAD   s1, BB
  XOR    s1, BIT6
  AND    s1, BIT0_COMP
  TEST   s0, s1
  SRA    s1
  RETURNI ENABLE

  ADDRESS 3FF
  JUMP ISR