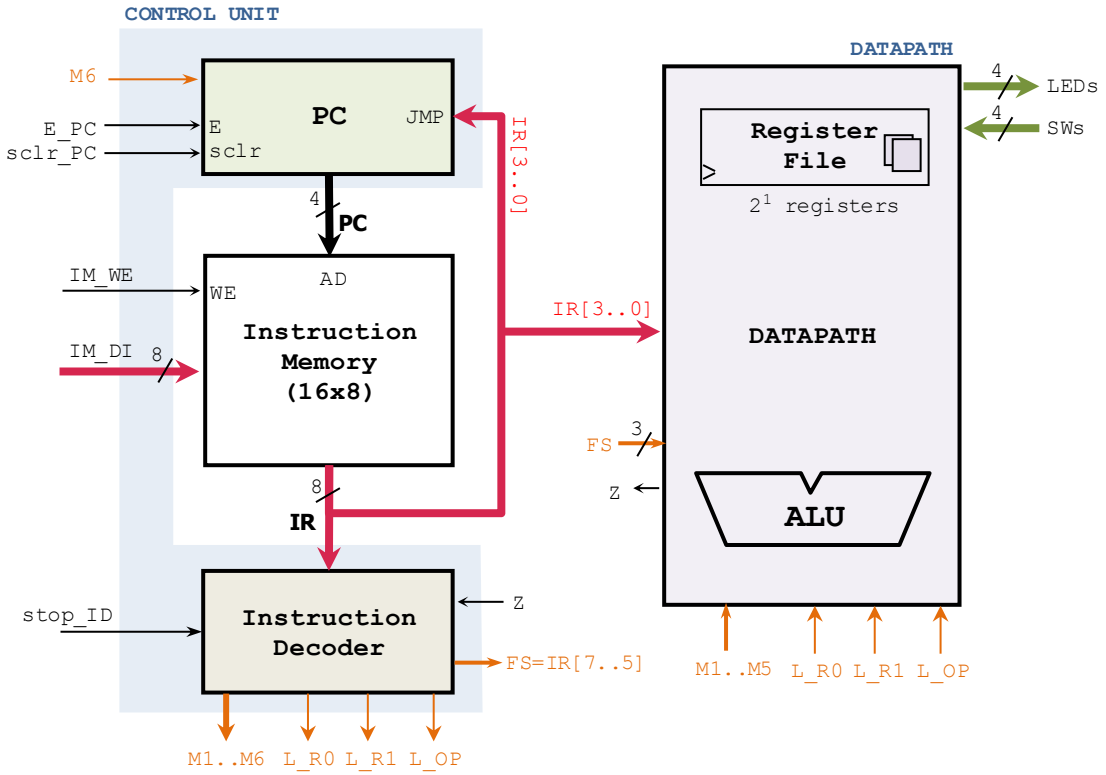
Solutions - Homework 4

(Due date: April 13th @ 11:59 pm)

Presentation and clarity are very important! Show your procedure!

PROBLEM 1 (30 PTS)

- **"VBC (Very Basic Computer)"**: 2 registers, 16-word Instruction Memory (IM), 8 bits per instruction (see Notes – Unit 6).



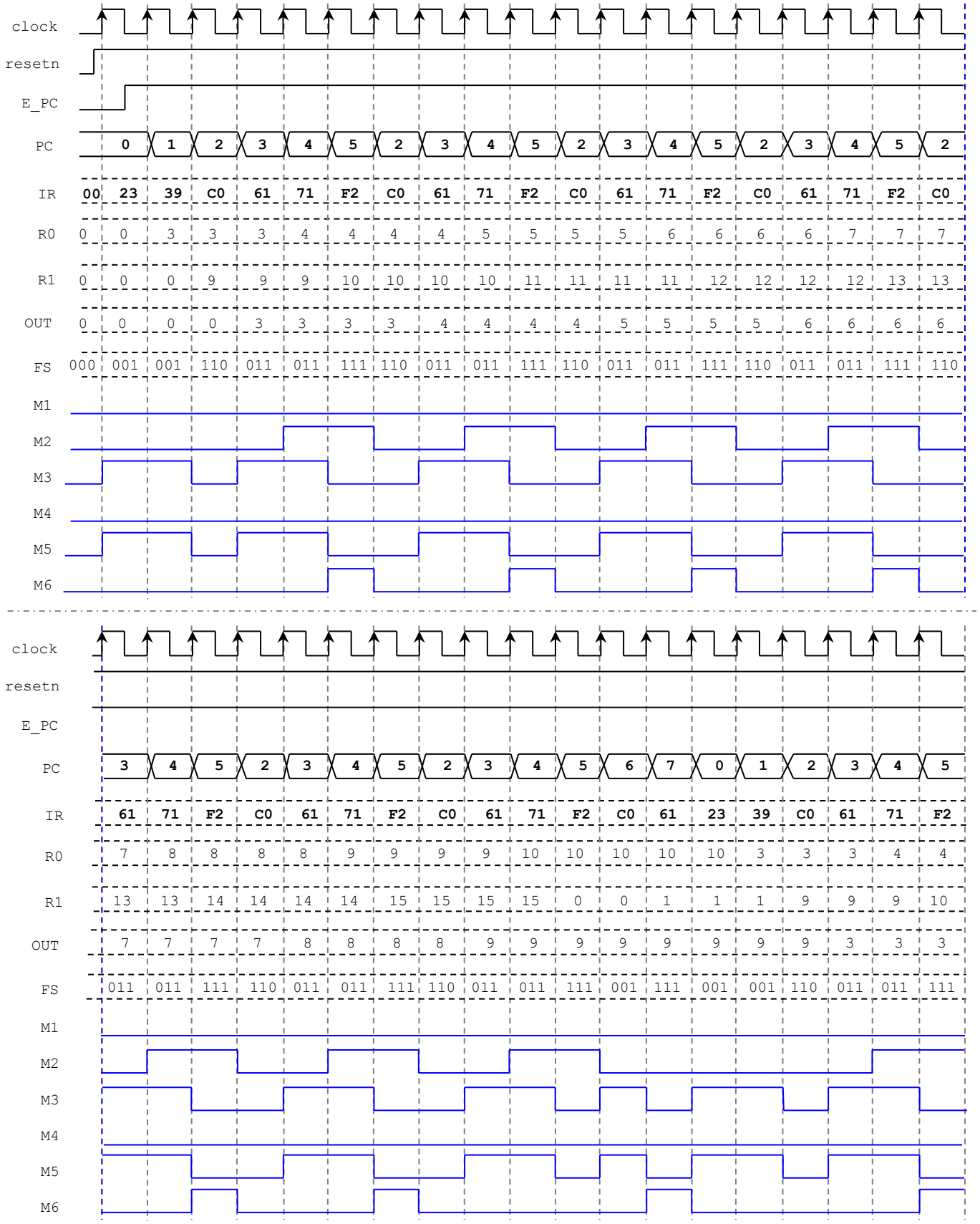
✓ Write an assembly program for a counter from 3 to 9: 3, 4, … 9, 3, 4, … The count must be shown on the output register (**OUT**). Use labels to specify any address that an instruction may jump to. You can only have up to 16 instructions.

✓ Provide the contents of the Instruction Memory. If some instruction bits are unused, you can assign 0's.

3 to 9 ≡ 9 to 15

Address	Instruction Memory	Assembly Instruction
0000	00100011	start: loadi R0,3
0001	00111001	loadi R1,9
0010	11000000	loop: out R0
0011	01100001	addi R0,1
0100	01110001	addi R1,1
0101	11110010	jnz R1,loop
0110	00110001	loadi R1,1
0111	11110000	jnz R1, start
1000		
1001		
1010		
1011		
1100		
1101		
1110		
1111		

✓ Complete the timing diagram for the execution of the previous program. Use unsigned decimal values for R0, R1, OUT, and PC. Specify IR in hexadecimal. (18 pts)
IM: Array of registers. In this case, when reading, output data appears as soon as address is ready.
Assumptions: the program is already in IM. Also: `sclr_PC=0, IM_WE=0, stop_ID=0`.



PROBLEM 2 (18 PTS)

- "**Simple Computer**": 8 registers, 64-word Instruction Memory (IM), 16 bits per instruction (see Notes – Unit 6).
  - ✓ For each of the following instructions, indicate the bits for the instruction fields (if they do not apply, write 'NA'), and the instruction memory contents (if some bits are unused, you can assign 0's). (8 pts)

Address	Instruction	OPCODE	DR	SA	SB	OP	AD	Instruction Memory contents
0	LDI R0,3	1001100	000	---	NA	011	NA	1001100000---011
1	LDI R3,2	1001100	011	---	NA	010	NA	1001100011---010
2	LDI R4,6	1001100	100	---	NA	110	NA	1001100100---110
3	SUB R7,R0,R4	0000101	111	000	100	NA	NA	0000101111000100
4	ADI R5,R4,7	1000010	101	100	NA	111	NA	1000010101100111
5	ST R5, R7	0100000	---	101	111	NA	NA	0100000---101111
6	XOR R4,R7,R5	0001010	100	111	101	NA	NA	0001010100111101
7	LD R6, R5	0010000	110	101	---	NA	NA	0010000110101---
8	BRN R7,-5	1100001	NA	111	NA	NA	111011	1100001111111011
9	ST R7,R4	0100000	---	111	100	NA	NA	0100000---111101
10	JMP R0	1110000	NA	000	NA	NA	------	1110000---000---

  - ✓ Emulate the execution of the program by completing the state of the registers (use hexadecimal values) after the instruction pointed by PC is executed. Then, complete the PC (use binary values) of the next instruction. Complete it until the JMP instruction is reached (execute this instruction as well). (10 pts)

PC	R0	R3	R4	R5	R6	R7
000000	0003	0000	0000	0000	0000	0000
000001	0003	0002	0000	0000	0000	0000
000010	0003	0002	0006	0000	0000	0000
000011	0003	0002	0006	0000	0000	FFFD
000100	0003	0002	0006	000D	0000	FFFD
000101	0003	0002	0006	000D	0000	FFFD
000110	0003	0002	FFF0	000D	0000	FFFD
000111	0003	0002	FFF0	000D	FFFD	FFFD
001000	0003	0002	FFF0	000D	FFFD	FFFD
000011	0003	0002	FFF0	000D	FFFD	0013
000100	0003	0002	FFF0	FFF7	FFFD	0013
000101	0003	0002	FFF0	FFF7	FFFD	0013
000110	0003	0002	FFE4	FFF7	FFFD	0013
000111	0003	0002	FFE4	FFF7	0013	0013
001000	0003	0002	FFE4	FFF7	0013	0013
001001	0003	0002	FFE4	FFF7	0013	0013
001010	0003	0002	FFE4	FFF7	0013	0013

PROBLEM 3 (17 PTS)

- "**Simple Computer**": 8 registers, 64-word IM, 16 bits per instruction (see Notes - Unit 6)
  - ✓ Write an assembly program for a counter from 3 to 11: 3, 4, 5, 6, 7, …, 11, 3, 4, … The count must appear on **R0**. Use labels to specify any address that an instruction may jump to.
    - Note that you can only have up to 64 instructions.
    - The ALU treats data as signed numbers. Thus, data on the registers (R0 – R7) and Data Memory (16 bits) is signed.
  - ✓ Provide the contents of the Instruction Memory (IM) and its addresses.

`* 3 to 11 = 8 downto 0`

Address	Instruction Memory	Assembly Instruction
000000	1001100011---100	start: ldi R3, 4
000001	1001100000---011	ldi R0, 3
000010	1001100001---111	ldi R1, 7
000011	1000010001001001	adi R1,R1, 1
000100	1000010000000001	loop: adi R0,R0, 1
000101	0000110001001---	dec R1,R1
000110	1100000111001011	brz R1, -5
000111	1110000---011---	jmp R3

PROBLEM 4 (20 PTS)

- "**PicoBlaze MicroProcessor**": 16 registers, 1024-word Instruction Memory, 18 bits per instruction (see Notes – Unit 6).
  - ✓ Emulate the execution of this program by completing the state of the registers (hexadecimal values) and the status flags (C, Z) after the instruction pointed by PC is executed. Then, complete the PC (hexadecimal values) of the next instruction and update SP (binary values). Complete it until the `jump` instruction (`003`) is reached (execute this instruction as well).

```
Address | Assembly Program
   000  | main: load s3,04
   001  |       load s4,05
   002  |       call umult
   003  |       jump main
   ...  |       ...
        |
   067  | umult: load s5,00
   068  |        load s7,08
   069  | loop:  sr0 s4
   06A  |        jump nc, sh_p
   06B  |        add s5, s3
   06C  |  sh_p: sra s5
   06D  |        sra s6
   06E  |        sub s7,01
   06F  |        jump nz, loop
   070  |        return
   ...  |        ...
```

PC	SP	s3	s4	s5	s6	s7	C	Z
000	11111	04	00	00	00	00	0	0
001	11111	04	05	00	00	00	0	0
002	11111	04	05	00	00	00	0	0
067	11110	04	05	00	00	00	0	0
068	11110	04	05	00	00	08	0	0
069	11110	04	02	00	00	08	1	0
06A	11110	04	02	00	00	08	1	0
06B	11110	04	02	04	00	08	0	0
06C	11110	04	02	02	00	08	0	0
06D	11110	04	02	02	00	08	0	0
06E	11110	04	02	02	00	07	0	0
06F	11110	04	02	02	00	07	0	0
069	11110	04	01	02	00	07	0	0
06A	11110	04	01	02	00	07	0	0
06C	11110	04	01	01	00	07	0	0
06D	11110	04	01	01	00	07	0	0
06E	11110	04	01	01	00	06	0	0
06F	11110	04	01	01	00	06	0	0

069	11110	04	00	01	00	06	1	1
06A	11110	04	00	01	00	06	1	1
06B	11110	04	00	05	00	06	0	0
06C	11110	04	00	02	00	06	1	0
06D	11110	04	00	02	80	06	0	0
06E	11110	04	00	02	80	05	0	0
06F	11110	04	00	02	80	05	0	0
069	11110	04	00	02	80	05	0	1
06A	11110	04	00	02	80	05	0	1
06C	11110	04	00	01	80	05	0	0
06D	11110	04	00	01	40	05	0	0
06E	11110	04	00	01	40	04	0	0
06F	11110	04	00	01	40	04	0	0
069	11110	04	00	01	40	04	0	1
06A	11110	04	00	01	40	04	0	1
06C	11110	04	00	00	40	04	1	1
06D	11110	04	00	00	A0	04	0	0
06E	11110	04	00	00	A0	03	0	0
06F	11110	04	00	00	A0	03	0	0
069	11110	04	00	00	A0	03	0	1
06A	11110	04	00	00	A0	03	0	1
06C	11110	04	00	00	A0	03	0	1
06D	11110	04	00	00	50	03	0	0
06E	11110	04	00	00	50	02	0	0
06F	11110	04	00	00	50	02	0	0
069	11110	04	00	00	50	02	0	1
06A	11110	04	00	00	50	02	0	1
06C	11110	04	00	00	50	02	0	1
06D	11110	04	00	00	28	02	0	0
06E	11110	04	00	00	28	01	0	0
06F	11110	04	00	00	28	01	0	0
069	11110	04	00	00	28	01	0	1
06A	11110	04	00	00	28	01	0	1
06C	11110	04	00	00	28	01	0	1
06D	11110	04	00	00	14	01	0	0
06E	11110	04	00	00	14	00	0	1
06F	11110	04	00	00	14	00	0	1
070	11110	04	00	00	14	00	0	1
003	11111	04	00	00	14	00	0	1

PROBLEM 5 (15 PTS)

- Attach a printout of your Project Status Report (no more than 3 pages, single-spaced, 2 columns). This report should contain the current status of the project, including more details about the design and its components. You **MUST** use the provided template (`Final Project – Report Template.docx`).