

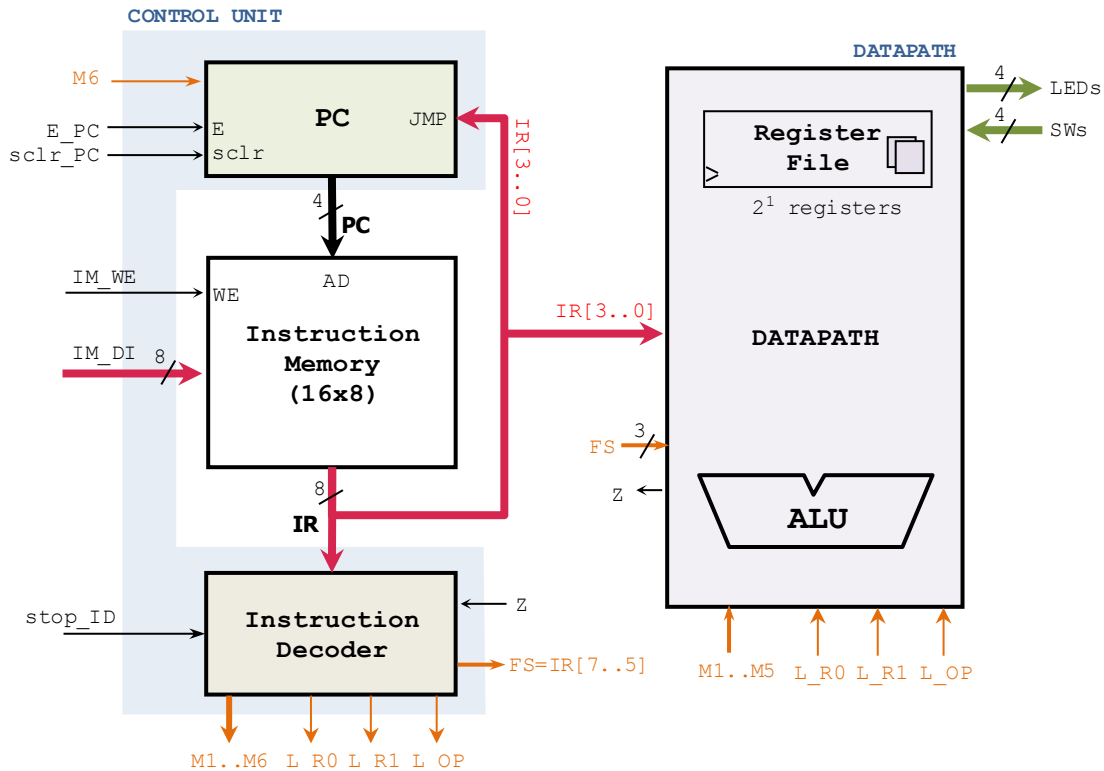
Solutions - Homework 4

(Due date: April 11th @ 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 4 to 11: 4, 5, ... 11, 4, 5, ... 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.

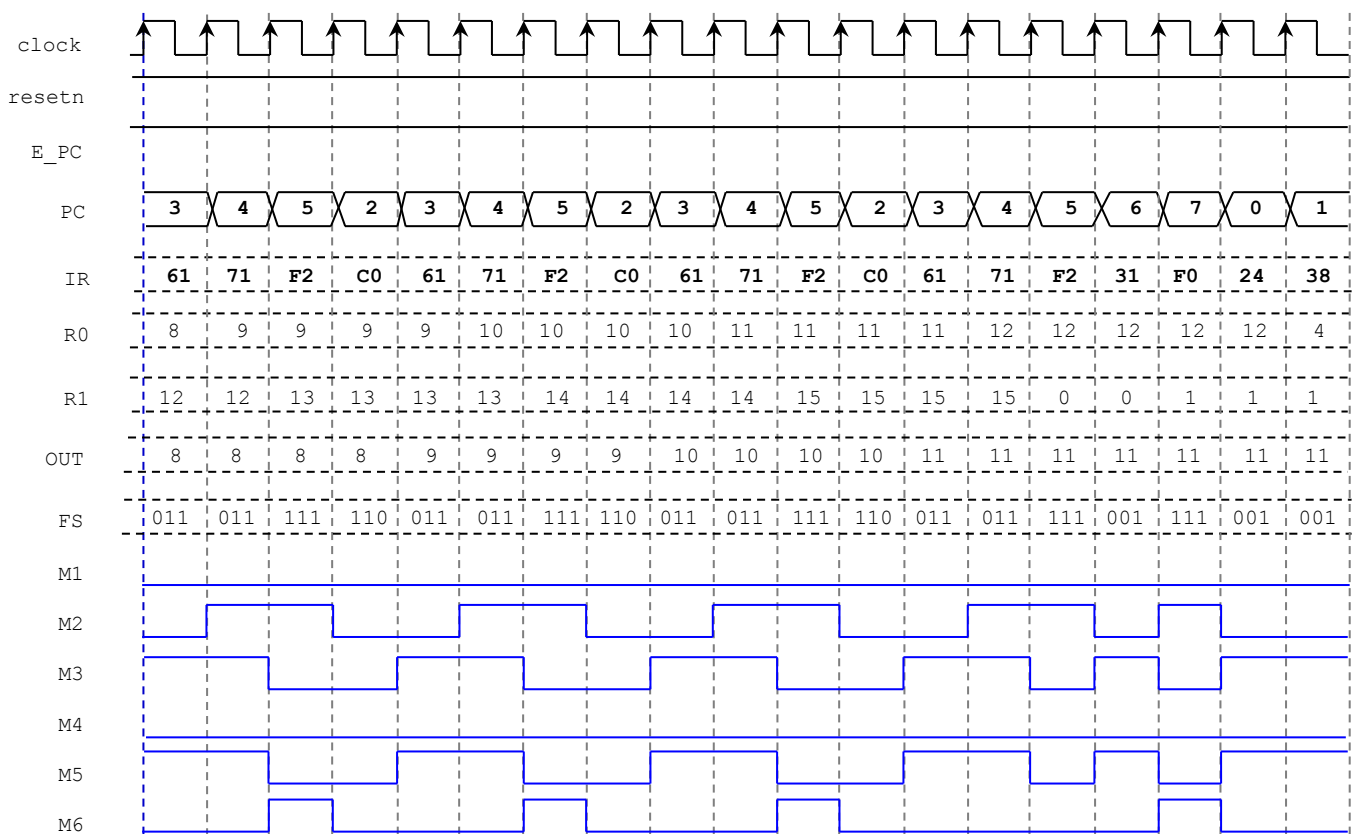
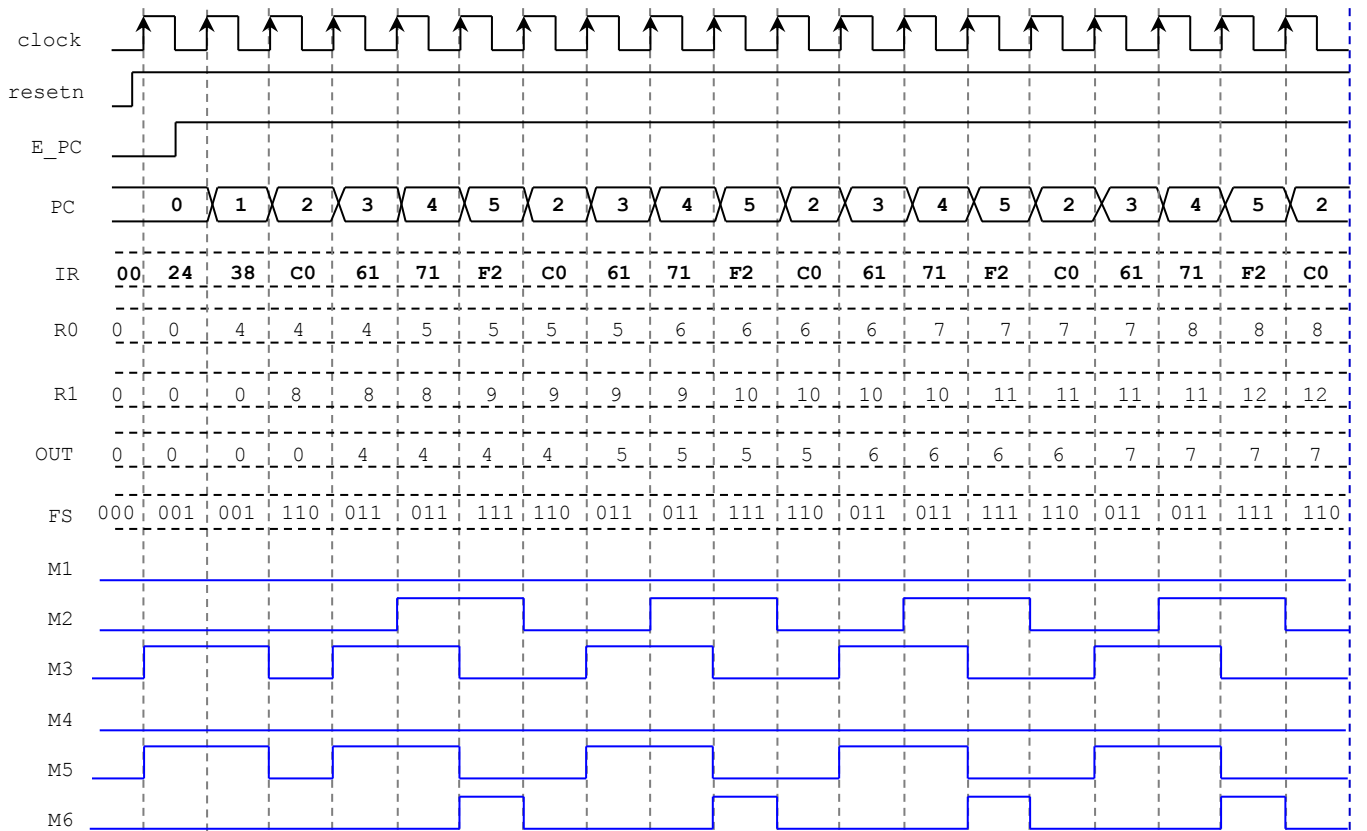
4 to 11 = 8 to 15

Address	Instruction Memory	Assembly Instruction
0000	00100011	start: loadi R0,4
0001	00111001	loadi R1,8
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, 5	1001100	000	---	NA	101	NA	1001100000---101
1	LDI R3, 4	1001100	011	---	NA	100	NA	1001100011---100
2	LDI R2, 7	1001100	010	---	NA	110	NA	1001100010---111
3	SUB R7, R3, R0	0000101	111	011	000	NA	NA	0000101111011000
4	ADI R5, R7, 7	1000010	101	111	NA	111	NA	1000010101111111
5	ST R3, R5	0100000	---	011	101	NA	NA	0100000---011101
6	XOR R0, R5, R2	0001010	000	101	010	NA	NA	0001010000101010
7	LD R6, R3	0010000	110	011	---	NA	NA	0010000110011---
8	BRN R7, -5	1100001	NA	111	NA	NA	111011	1100001111111011
9	SHL R7, R2	0001110	111	---	010	NA	NA	0001110111---010
10	JMP R2	1110000	NA	010	NA	NA	-----	1110000---010---

- ✓ 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	R2	R3	R5	R6	R7
000000	0005	0000	0000	0000	0000	0000
000001	0005	0000	0004	0000	0000	0000
000010	0005	0007	0004	0000	0000	0000
000011	0005	0007	0004	0000	0000	FFFF
000100	0005	0007	0004	0006	0000	FFFF
000101	0005	0007	0004	0006	0000	FFFF
000110	0001	0007	0004	0006	0000	FFFF
000111	0001	0007	0004	0006	0006	FFFF
001000	0001	0007	0004	0006	0006	FFFF
000011	0001	0007	0004	0006	0006	0003
000100	0001	0007	0004	000A	0006	0003
000101	0001	0007	0004	000A	0006	0003
000110	000D	0007	0004	000A	0006	0003
000111	000D	0007	0004	000A	000A	0003
001000	000D	0007	0004	000A	000A	0003
001001	000D	0007	0004	000A	000A	000E
001010	000D	0007	0004	000A	000A	000E

PROBLEM 3 (17 PTS)

- **"Simple Computer"**: 8 registers, 64-word IM, 16 bits per instruction (see Notes - Unit 6)
- ✓ Write an assembly program that stores numbers from 37 down to 25 in Data Memory (DM) on addresses 0 to 12. 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.

address	DM
000000	25
000001	24
000010	23
000011	22
000100	21
000101	20
000110	1F
000111	1E
001000	1D
001001	1C
001010	1B
001011	1A
001100	19
...	

Address	Instruction Memory	Assembly Instructions
000000	1001100 010 --- 101	start: ldi R2,5
000001	1001100 110 --- 110	ldi R6,6
000010	1000010 110 110 110	adi R6,R6,6
000011	0000000 100 110 ---	mov R4,R6
000100	0000010 110 100 110	add R6,R4,R6
000101	0000001 110 110 ---	loop: inc R6,R6
000110	0100000 --- 100 110	st R4,R6
000111	1100000 111 100 001	brz R4,-7
001000	0000110 100 100 ---	dec R4,R4
001001	1110000 --- 010 ---	jmp R2

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).