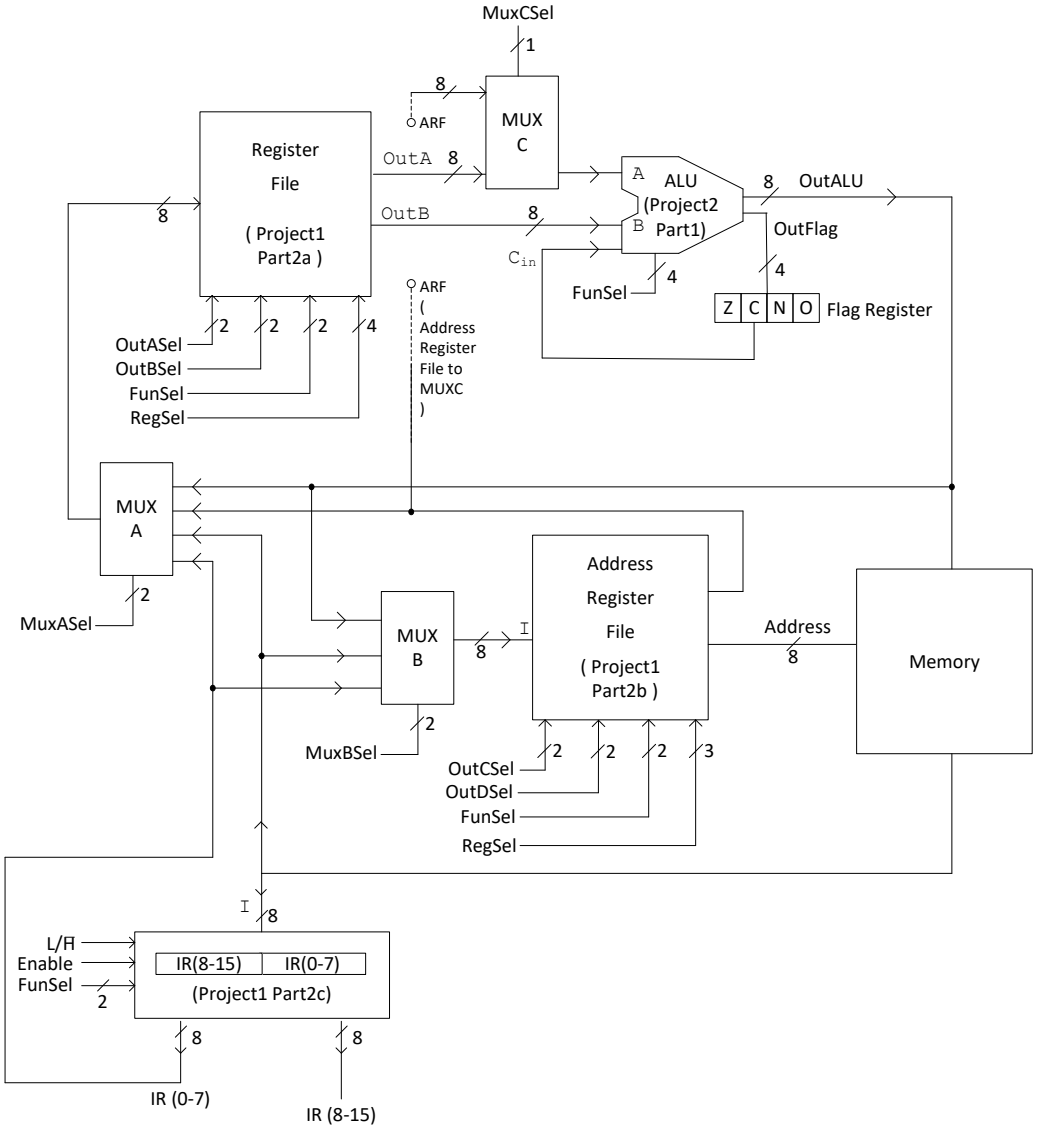
BLG222E Computer Organization

Take Home Final Exam

Due Date: 14.07.2020, 08:00

Design a **software-based (microprogrammed) control unit** for the following architecture. Use the structure shown in Figure 1 which is a slightly modified version of **Project2.**

You have to **design and decide** the size of the control memory and microinstruction format in order to control the simple computer. In addition, you might need to change the mapping algorithm as well.



MuxASel	MuxAOut
00	IROut (0-7)
01	Memory Output
10	Address Register OutC
11	OutALU

MuxBSel	MuxBOut
00	ϕ
01	IROut (0-7)
10	Memory Output
11	OutALU

MuxCSel	MuxCOut
0	MuxAOut
1	OutA

Figure 1: Slightly Modified Version of Project 2 (i.e., here MUXC takes input from Address Register File instead of MUXA)

INSTRUCTION FORMAT

There are two types of instructions as described below.

(1) Instructions with address reference has the format shown in Figure 2:

- The OPCODE is a 5-bit field (See Table 1 for the definition).
- The REGSEL is a 2-bit field (See left side of Table 2 for the definition).
- The ADDRESSING MODE is a 1-bit field (See Table 3 for the definition).
- The ADDRESS is 8 bits

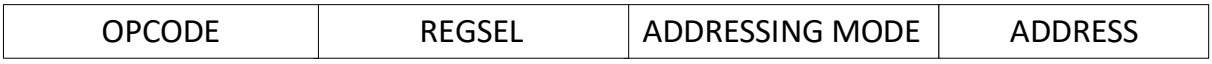


Figure 2: Instructions with an address reference

(2) Instructions without address reference has the format shown in Figure 3:

- The OPCODE is a 5-bit field (See Table 1 for the definition).
- DESTREG is a 3-bit field which specifies the destination register (See right side of Table 2 for the definition).
- SRCREG1 is a 3-bit field which specifies the first source register (See right side of Table 2 for the definition).
- SRCREG2 is a 3-bit field which specifies the second source register (See right side of Table 2 for the definition).
- The least significant two bits are unused and have the value 00.

At most one register from the address register file (AR/SP/PC) can be used on the **right hand side** of the instructions without address reference. In addition, if one of AR/SP/PC is used on the right hand side, it must be the A input of ALU (i.e., must be supplied by means of MUXC).

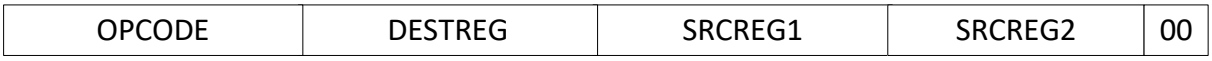


Figure 3: Instructions without an address reference

Table 1: OPCODE field and SYMBols for opertions and their descriptions

OPCODE (HEX)	SYMB	ADDRESSING MODE	DESCRIPTION
0x00	LD	IM, D	Rx ← Value (Value is described in Table 3)
0x01	ST	D	Value ← Rx
0x02	MOV	N/A	DESTREG ← SRCREG1
0x03	PSH	N/A	M[SP] ← Rx, SP ← SP - 1
0x04	PUL	N/A	SP ← SP + 1, Rx ← M[SP]
0x05	ADD	N/A	DESTREG ← SRCREG1 + SRCREG2
0x06	SUB	N/A	DESTREG ← SRCREG2 - SRCREG1
0x07	DEC	N/A	DESTREG ← SRCREG1 - 1
0x08	INC	N/A	DESTREG ← SRCREG1 + 1
0x09	AND	N/A	DESTREG ← SRCREG1 AND SRCREG2
0x0A	OR	N/A	DESTREG ← SRCREG1 OR SRCREG2
0x0B	NOT	N/A	DESTREG ← NOT SRCREG1
0x0C	LSL	N/A	DESTREG ← LSL SRCREG1
0xOD	LSR	N/A	DESTREG ← LSR SRCREG1
0x0E	BRA	IM	PC ← Value
0x0F	BEQ	IM	IF Z=1 THEN PC ← Value
0x10	BNE	IM	IF Z=0 THEN PC ← Value
0x11	CALL	IM	M[SP] ← PC, SP ← SP – 1, PC ← Value
0x12	RET	N/A	SP ← SP + 1, PC ← M[SP]

Table 2:REGSEL (Left) and DESTREG/SRCREG1/SRCREG2 (Right) select the register of interest for a particular instruction

REGSEL	REGISTER
00	R0
01	R1
10	R2
11	R3

DESTREG/SRCREG1/SRCREG2	REGISTER
000	R0
001	R1
010	R2
011	R3
100	PC
101	PC
110	AR
111	SP

Table 3: Addressing modes

ADDRESSING MODE	MODE	SYMB	Value
0	Immediate	IM	ADDRESS Field
1	Direct	D	M[AR]

EXAMPLE

The code given below adds data that are stored at M[A0]+M[A1]+M[A2]+M[A3]+M[A4] and stores the total at M[A5]. It is written as a loop that iterates 5 times.

You have to determine the binary code, write it into memory, and execute all these instructions.

```
        ORG  0x20          # Write the program starting from the address 0x20
        LD R0 IM 0x05      #  R0 is used for iteration number
        LD R1 IM 0x00      #  R1 is used to store total
        LD R2 IM 0xA0
        MOV AR R2          # AR is used to track data adrress: starts from 0xA0
LABEL:  LD R2 D            # R2 <- M[AR] (AR = 0xA0 to 0xA4)
        INC AR AR          # AR <- AR + 1 (Next Data)
        ADD R1 R1 R2       # R1 <- R1 + R2 (Total = Total + M[AR])
        DEC R0 R0          # R0 <- R0 – 1 (Decrement Iteration Counter)
        BNE IM LABEL       # Go back to LABEL if Z=0 (Itertaion Counter > 0)
        ST  R1 D           # M[AR] <- R1 (Store Total at 0xA5)
```

Submission:

Implement your design in logisim software, upload a single compressed (zip) file to Ninova before the deadline. **All the students from each group should submit the project file. Therefore, be sure that all of you uploaded the same final file to Ninova.** This compressed file should contain your design files (.circ) and a report that includes:

- the number&names of the students in the group
- information about your control unit design and microinstruction format

Group work is expected for this project. All the 4 student members of the group **must** design together. Make sure to connect pins (under Wiring group of logisim) to the inputs and control inputs of your design, so that different inputs and functions can be tested. Similarly connect your inputs and outputs to a “Hex Digit Display" in logisim (under Input/output group of logisim) so that the test outputs can be observed and use proper labelling to improve the clarity of your circuits.