King Saud University
College of Computer and Information Sciences
Computer Science Department

Course Code	CSC 220		
Course Title	Computer Organization		
Semester	S1 – 1443 (Fall-2021-22)		
Exam	Final Exam		
Date	04/01/2022	**Duration**	**3 Hours**
Student Name			
Student ID			
Section No.			

Course Learning Outcomes		**Relevant question**	**Full mark**	**Student's mark**
CLO 2.1	*Combinational and sequential circuits design*	**1**	**8**	
CLO 2.2	*Register, counter, and RAM design*	**2**	**6**	
CLO 2.2	*Register, counter, and RAM design*	**3**	**6**	
CLO 2.3	*Arithmetic and Logic Unit (ALU) design*	**4**	**6**	
CLO 2.4	*Datapath and CPU design*	**5**	**6**	
CLO 1.3	*Instruction set and machine programming*	**6**	**8**	
Total			**40**	

Feedback/Comments:

Question 1. (8 Marks: 2+2+2+2)

(a) Determine the decimal value represented by 1000 1011 in each of the following systems.

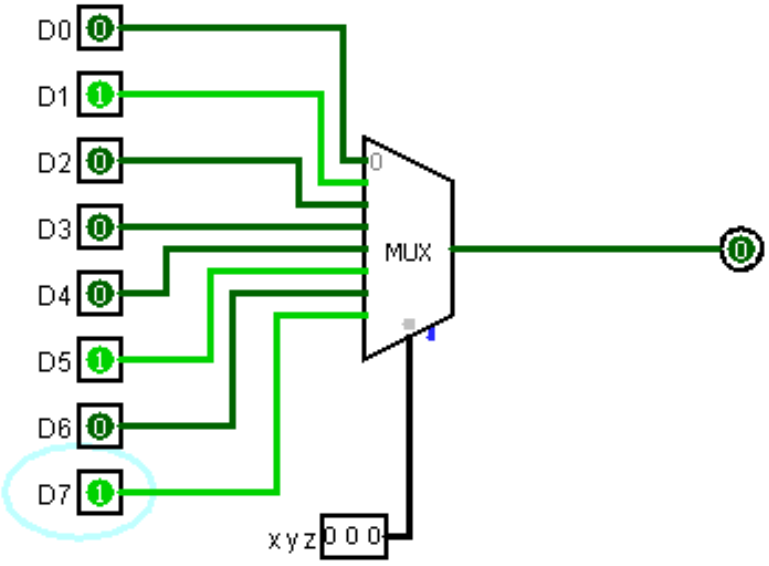
1. Unsigned notation?	139
2. Two's complements?	-117

(b) For an 8-bit adder, write the following values

i. Number of inputs?	17
ii. Number of Outputs?	9
iii. Size of truth table?	217
iv. How many function to be optimized?	9

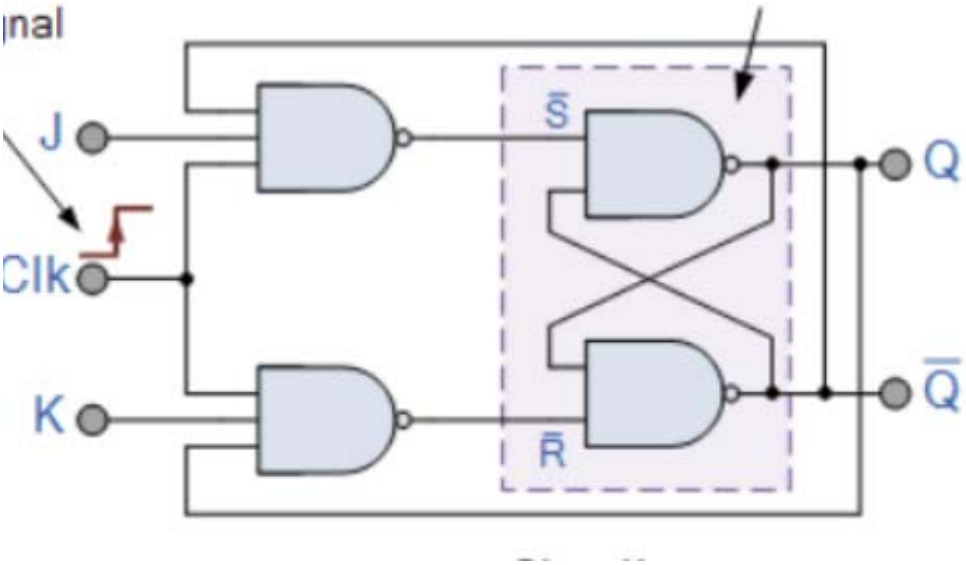
(c) Implement the following function using an 8X1 multiplexer

F(x,y,z) = xy'z + x'y'z + xyz

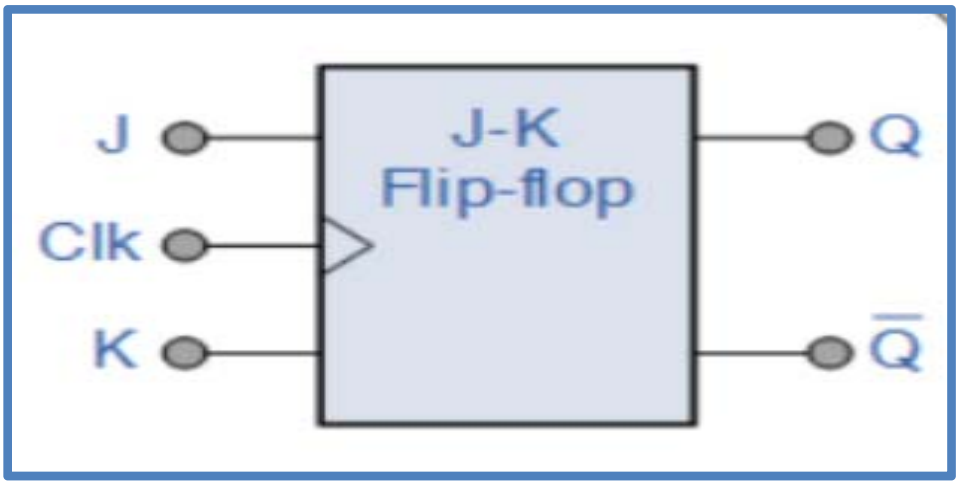


(d) Show how to:

i. implement a JK flip-flop using NAND gates

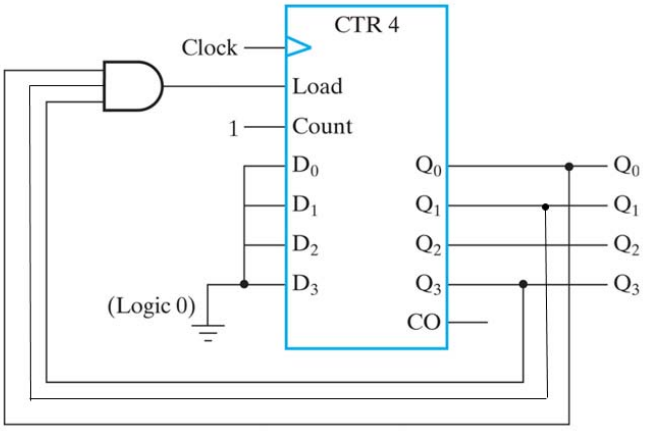


v. obtain T Flip Flop from the JK structure.



Question 2 (6 Marks: 2+1+3)

(a) Determine the cycle described by the following 4-bit synchronous counter.



The counter describes the cycle 0,1,2,3,4,5,6,7,8,9,10,11 / 0, 1 , 2 3…….

(b) Calculate the memory space in Bytes considering the following memory (RAM) diagram.

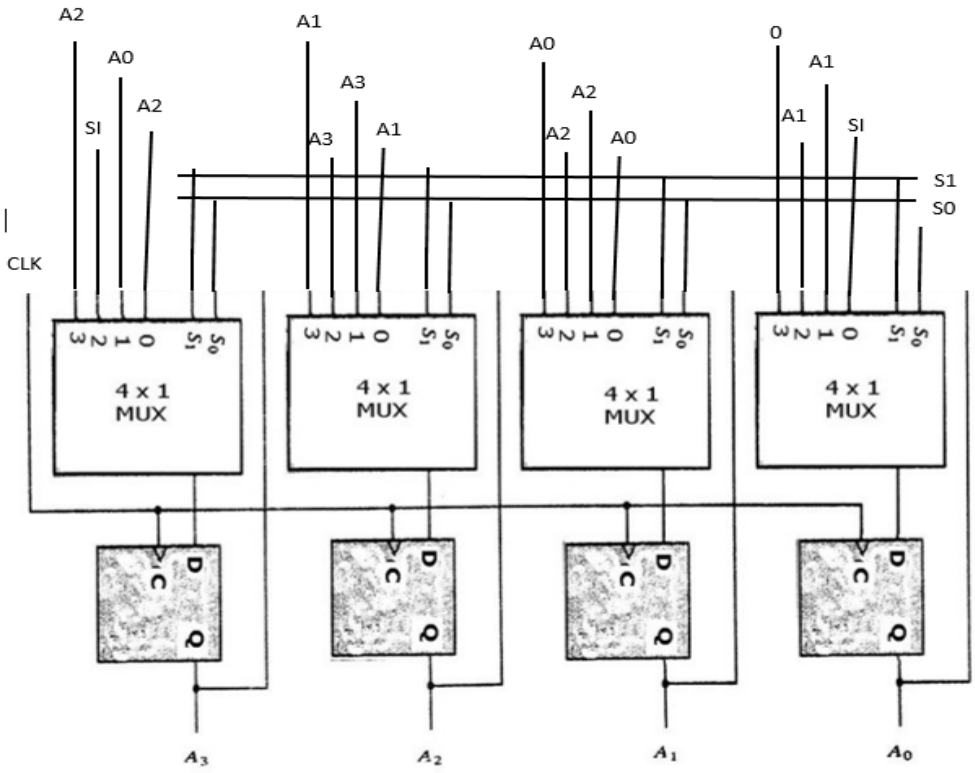
The memory has the size (2^16) Bytes

(c) Show how to design a 128 × 8 RAM Using 32 × 8 RAM chips

Question 3 (6 Marks: 3+3)

(a) Design 4-bits register using necessary flip-flops and MUXs that performs the following operations

S1	S0	Operation
0	0	Shift left
0	1	Rotate Right
1	0	Shift right
1	1	Logical Shift Left



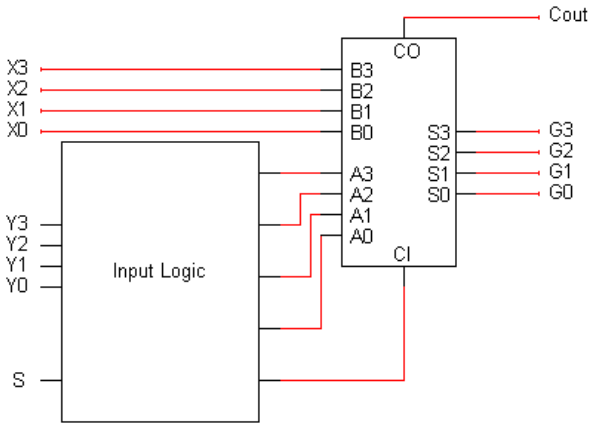
(b) Consider the following **RTL program** with the initial values of 8-bit registers R1 = 0001 0111, R2 = 1110 0111, R3 = 0000 0000 (2's complement representation). Show the contents of the registers after execution of each micro-operation sequentially.

micro-operations	R1	R2	R3
R3 ← R1 + R2	0001 0111	1110 0111	1111 1110

R1 ← R3 + 1	1111 1111	1110 0111	1111 1110
R2 ← SL R1	1111 1111	1111 1110	1111 1110

Question 4 (6 Marks: 3+3)

(a) In order to design an **input logic** for a 4-bit-arithmetic unit as shown below and considering the operation table with eight-arithmetic operations, **write the required adder's inputs** (A, B, CI) in terms of user input X and Y.



Answer of Question 6 (a)

Selection			Arithmetic operation		Required adder inputs		
S_2	**S_1**	**S_0**	**G (A + B + CI)**	**Description**	1111	X	0
0	0	0	X – 1	(decrement)	1111	X	1
0	0	1	X	(transfer)	Y'	X	0
0	1	0	X + Y'	(1C subtraction)	Y'	X	1
0	1	1	X + Y' + 1	(2C subtraction)	Y	X	0
1	0	0	X + Y	(add)	Y	X	1
1	0	1	X + Y + 1		0000	X	0
1	1	0	X	(transfer)	0000	X	1
1	1	1	X + 1	(increment)	1111	X	0

(b) Design the **input logic** using necessary logic gates for the 4-bit arithmetic unit above and considering the same set of operations. Here, the inputs for the circuit are three-bit selections S(S_2, S_1, S_0), 4-bit data Y(Y_3, Y_2, Y_1, Y_0) and the 4-bit outputs are A(A_3, A_2, A_1, A_0). Show the followings:

i. Abbreviated and full truth table,
ii. Simplified function with k-map,
iii. The diagram of the input logic (Note: *MUX-based implementation is not acceptable*).

Answer of Question 4 (b)

i.

S_2	S_1	A
0	0	1111
0	1	Y'
1	0	Y
1	1	0000

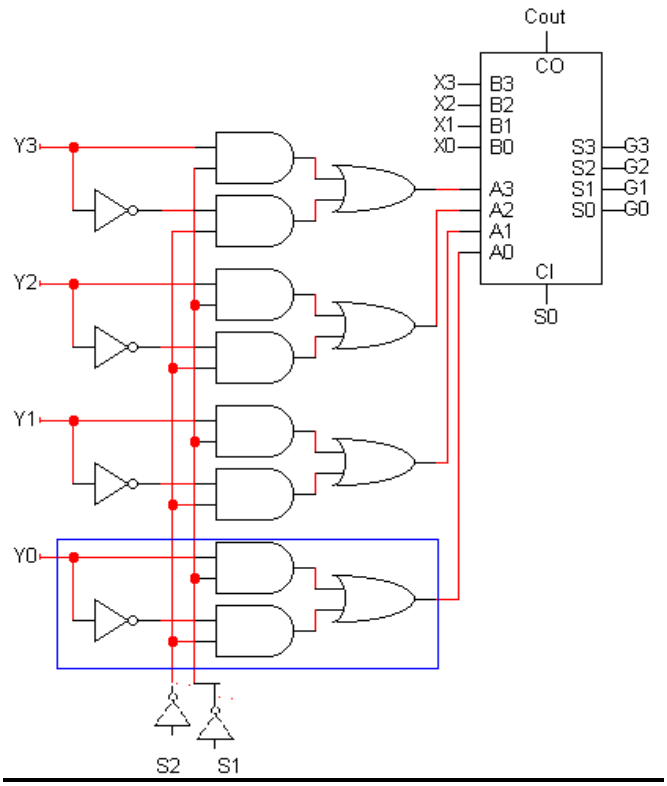
S_2	S_1	Y_i	A_i
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	0

ii.

		S_1		
	1	1	0	1
S_2	0	1	0	0
		Y_i		

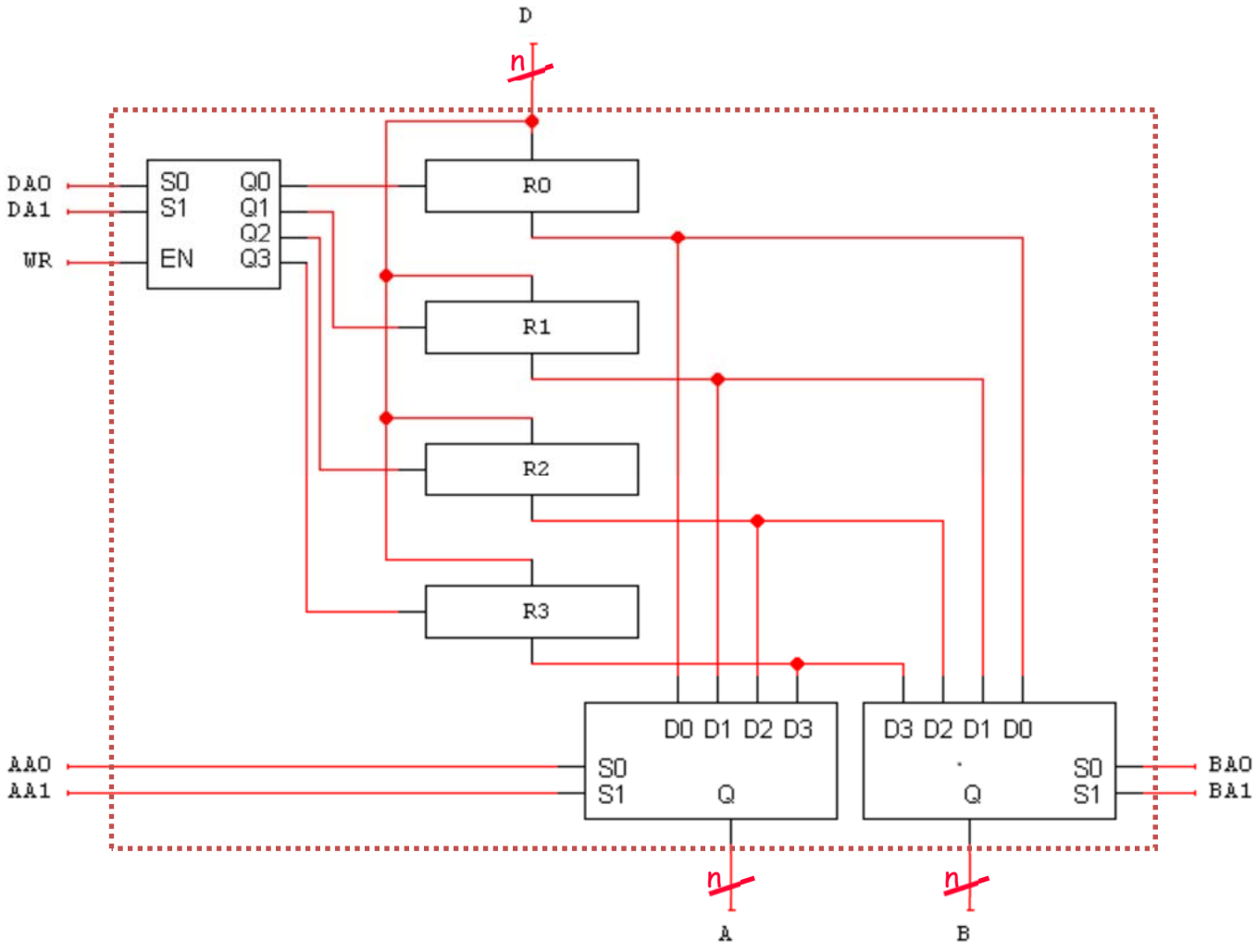
$$A_i = S'_2 Y_i + S'_1 Y'_i$$

iii.



Question 5 (6 Marks: 4+2)

(a) Show how to design a **4 x n** register file using necessary registers, MUXs, and decoders.



(b) Compute the control words for the following micro-operations for the datapath below described by the Table 8.5.

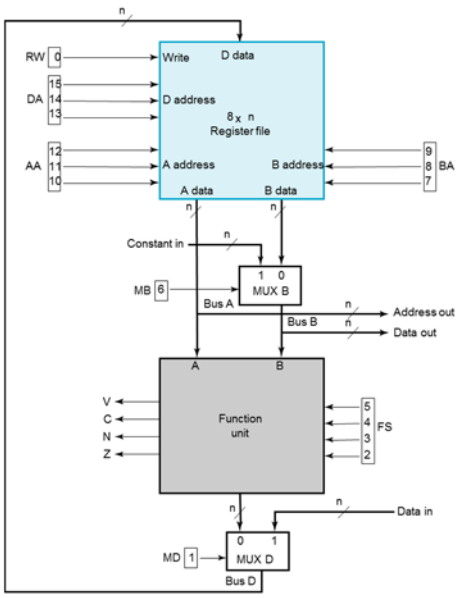


TABLE 8-5
Encoding of Control Word for the Datapath

DA, AA, BA		MB		FS		MD		RW	
Function	Code	Function	Code	Function	Code	Function	Code	Function	Code
$R0$	000	Register	0	$F = A$	0000	Function	0	No Write	0
$R1$	001	Constant	1	$F = A + 1$	0001	Data in	1	Write	1
$R2$	010			$F = A + B$	0010				
$R3$	011			$F = A + B + 1$	0011				
$R4$	100			$F = A + \overline{B}$	0100				
$R5$	101			$F = A + \overline{B} + 1$	0101				
$R6$	110			$F = A - 1$	0110				
$R7$	111			$F = A$	0111				
				$F = A \wedge B$	1000				
				$F = A \vee B$	1001				
				$F = A \oplus B$	1010				
				$F = \overline{A}$	1011				
				$F = B$	1100				
				$F = \text{sr } B$	1101				
				$F = \text{sl } B$	1110				

Micro-operations	DA (15-13)	AA (12-10)	BA (9-7)	MB (6)	FS (5-2)	MD (1)	RW (0)
R2←R3 – R1	010	011	001	0	0101	0	1
R3←M(R2)	011	010	xxx	x	xxx	1	1
R0←R0 ⊕ R1	000	000	001	0	1010	0	1
R2←sr R7	010	xxx	111	0	1101	0	1

Question 6 (8 Marks: 2 + 2 + 4)

TABLE 8-8
Instruction Specifications for the Simple Computer

Instruction	Opcode	Mnemonic	Format	Description	Status Bits
Move A	0000000	MOVA	RD, RA	R[DR] ← R[SA]*	N, Z
Increment	0000001	INC	RD, RA	R[DR] ← R[SA] + 1*	N, Z
Add	0000010	ADD	RD, RA, RB	R[DR] ← R[SA] + R[SB]*	N, Z
Subtract	0000101	SUB	RD, RA, RB	R[DR] ← R[SA] − R[SB]*	N, Z
Decrement	0000110	DEC	RD, RA	R[DR] ← R[SA] − 1*	N, Z
AND	0001000	AND	RD, RA, RB	R[DR] ← R[SA] ∧ R[SB]*	N, Z
OR	0001001	OR	RD, RA, RB	R[DR] ← R[SA] ∨ R[SB]*	N, Z
Exclusive OR	0001010	XOR	RD, RA, RB	R[DR] ← R[SA] ⊕ R[SB]*	N, Z
NOT	0001011	NOT	RD, RA	R[DR] ← $\overline{\text{R[SA]}}$*	N, Z
Move B	0001100	MOVB	RD, RB	R[DR] ← R[SB]*	
Shift Right	0001101	SHR	RD, RB	R[DR] ← sr R[SB]*	
Shift Left	0001110	SHL	RD, RB	R[DR] ← sl R[SB]*	

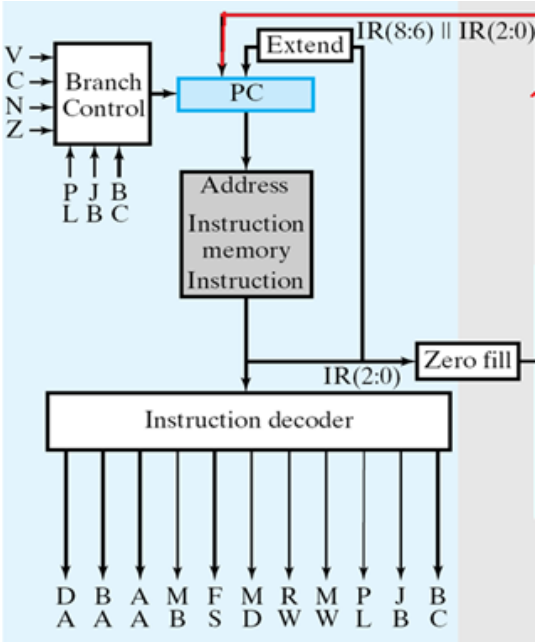
Load Immediate	1001100	LDI	RD, OP	R[DR] ← zf OP*	
Add Immediate	1000010	ADI	RD, RA, OP	R[DR] ← R[SA] + zf OP*	N, Z
Load	0010000	LD	RD, RA	R[DR] ← M[SA]*	
Store	0100000	ST	RA, RB	M[SA] ← R[SB]*	
Branch on Zero	1100000	BRZ	RA, AD	if (R[SA] = 0) PC ← PC + se AD, if (R[SA] ≠ 0) PC ← PC + 1	N, Z
Branch on Negative	1100001	BRN	RA, AD	if (R[SA] < 0) PC ← PC + se AD, if (R[SA] ≥ 0) PC ← PC + 1	N, Z
Jump	1110000	JMP	RA	PC ← R[SA]*	

* For all of these instructions, PC ← PC + 1 is also executed to prepare for the next cycle.

(a) Consider table 8-8 containing instruction specification for a simple computer. Translate the following instructions into 16-bit binary machine codes (**NB:** use 0 for don't care).

Instruction	**Binary machine code**			
	Opcode	**RD/AD(left)**	**RA**	**RB/OP/AD(Right)**
ADD R0, R5, R3	**0000010**	**000**	**101**	**011**
ST (R5), R6	**0100000**	**000**	**101**	**110**
ADI R2, R7 5	**1000010**	**010**	**111**	**101**
BRZ R2 AD (AD = 101 110)	**1100000**	**101**	**010**	**110**

(b) Show the design of a single cycle control unit involving Program Counter, Instruction Register, Instruction Decoder and a Branch Control Logic.



(c) Consider the following set of instructions written in assembly code and the content of the memory. What will be the final value of register R0, R1, R2 and the memory location 311 after executing these instructions?

Solution:

```
LDI     R0, 250
LD      R1, (R0)

INC     R0, R0
LD      R2, (R0)
ADD     R1, R1, R2

INC     R0, R0
LD      R2, (R0)
ADD     R1, R1, R2

INC     R0, R0
LD      R2, (R0)
ADD     R1, R1, R2

SHR  R1, R1
SHR R1, R1

LDI     R0, 311
ST      (R0), R1
```

address	memory
…	…
250	**90**
251	**-3**
252	25
253	9
…	…
311	…

R0	R1	R2	Memory location 311
311	30	9	30

THE END