Introduction to Computer Science-101
Quiz_2

1. What is the difference between logical and arithmetic shifts? (10%)
   The logical shift operation is applied to a pattern that does not represent a signed number. The arithmetic shift operation assumes that the bit pattern is a signed number in two's complement format.

2. Show the result of the following operations. (10%)
   a. $(99)_{16}$ OR $(99)_{16}$ $(99)_{16}$
   b. $(99)_{16}$ OR $(00)_{16}$ $(99)_{16}$
   c. $(99)_{16}$ AND $(FF)_{16}$ $(99)_{16}$
   d. $(FF)_{16}$ AND $(FF)_{16}$ $(FF)_{16}$

3. Using an 8-bit allocation, first convert each of the following numbers to two's complement, do the operation, and then convert the result to decimal. (10%)
   a. 19 – 23

$00010011 - 00010111 = 000010011 + (-00010111) = 00010011 + 11101001 =$

						1	1		Carry	Decimal
	0	0	0	1	0	0	1	1		19
+	1	1	1	0	1	0	0	1		−23
	1	1	1	1	1	1	0	0		−4

   b. -19 + 23

$(-00010011) + 00010111 = 11101101 + 00010111$

1	1	1	1	1	1	1	1		Carry	Decimal
	1	1	1	0	1	1	0	1		−19
+	0	0	0	1	0	1	1	1		23
	0	0	0	0	0	1	0	0		4

4. Show the result of the following operations assuming that the numbers are stored in 16-bit two's complement representation. Show the result in hexadecimal notation. (10%)
   a. $(012A)_{16} + (0E27)_{16}$

										1		1	1	1			Carry	Hexadecimal
	0	0	0	0	0	0	0	1	0	0	1	0	1	0	1	0		012A
+	0	0	0	0	1	1	1	0	0	0	1	0	0	1	1	1		0E27
	0	0	0	0	1	1	1	1	0	1	0	1	0	0	0	1		0F51

b. $(712A)_{16} + (9E00)_{16}$ = $(0F2A)_{16}$

					Carry	Hexadecimal
	1 1 1 1					
	0 1 1 1	0 0 0 1	0 0 1 0	1 0 1 0		712A
+	1 0 0 1	1 1 1 0	0 0 0 0	0 0 0 0		9E00
	0 0 0 0	1 1 1 1	0 0 1 0	1 0 1 0		10F2A

5. Using an 8-bit allocation, first convert each of the following numbers to sign-and-magnitude representation, do the operation, and then convert the result to decimal. (10%)

a. 19+23

$19 + 23 \rightarrow A = 19 = (00010011)_2$ and $B = 23 = (00010111)_2$.

Operation is addition; sign of B is not changed. $S = A_S \text{ XOR } B_S = 0$, $R_M = A_M + B_M$ and $R_S = A_S$

		No overflow									Carry
					1		1	1	1		
A_S	0			0	0	1	0	0	1	1	A_M
B_S	0		+	0	0	1	0	1	1	1	B_M
R_S	0			0	1	0	1	0	1	0	R_M

The result is $(00101010)_2 = 42$ as expected.

b. 19-23

$19 - 23 \rightarrow A = 19 = (00010011)_2$ and $B = 23 = (00010111)_2$. Operation is subtraction, sign of B is changed. $B_S = \overline{B_S}$, $S = A_S \text{ XOR } B_S = 1$, $R_M = A_M + (\overline{B_M} + 1)$. Since there is no overflow $R_M = \overline{(R_M + 1)}$ and $R_S = B_S$

		No overflow									Carry
								1	1		
A_S	0			0	0	1	0	0	1	1	A_M
B_S	1		+	1	1	0	1	0	0	1	$(\overline{B_M}+1)$
				1	1	1	1	1	0	0	R_M
R_S	1			0	0	0	0	1	0	0	$R_M = \overline{(R_M+1)}$

The result is $(10000100)_2 = -4$ as expected.

6. Compare and contrast CISC architecture with RISC architecture. (10%)

CISC (Complex Instruction Set Computer) has a large set of instructions to execute commands at the machine level. This makes the circuitry of the CPU and the control unit very complicated. RISC (Reduced Instruction Set Computer) uses a small set of instructions. Complex operations are accomplished using a set of simple commands.

7. How many bytes of memory are needed to store a full screen of data if the screen is made of 24 lines with 80 characters in each line? The system uses ASCII code, with each ASCII character stored as a byte. (10%)

We need 24 × 80 = 1920 bytes.

8. An imaginary computer has sixteen data register (R0 to R15), 1024 words in memory, and 16 different instructions (add, subtract, and so on). If a typical instruction uses the following format: instruction M R2 .If the computer uses the

same size of word for data and instructions, what is the size of each data register? (10%)

Since the size of the instruction is 18 bits, we must have 18-bit data registers.

9. What is the size of the program counter in the computer in question 8? (10%)
   The program counter must be large enough to hold the address of a word in memory. Therefore, it must be 10 bit.

10. Using the instruction set of the simple computer in the following table, write the code for a program that performs the following calculation:

$$B \leftarrow A - 2$$

A and 2 are integers in two's complement format. The user types the value of A and the value of B is displayed on the monitor. The keyboard is assumed to be memory location $(FE)_{16}$, and the monitor is assumed to be $(FF)_{16}$.(10%)

Instruction	*Code*	*Operands*			*Action*
	d_1	d_2	d_3	d_4	
HALT	0				Stops the execution of the program
LOAD	1	R_D	M_S		$R_D \leftarrow M_S$
STORE	2	M_D		R_S	$M_D \leftarrow R_S$
ADDI	3	R_D	R_{S1}	R_{S2}	$R_D \leftarrow R_{S1} + R_{S2}$
ADDF	4	R_D	R_{S1}	R_{S2}	$R_D \leftarrow R_{S1} + R_{S2}$
MOVE	5	R_D	R_S		$R_D \leftarrow R_S$
NOT	6	R_D	R_S		$R_D \leftarrow \overline{R_S}$
AND	7	R_D	R_{S1}	R_{S2}	$R_D \leftarrow R_{S1}$ AND R_{S2}
OR	8	R_D	R_{S1}	R_{S2}	$R_D \leftarrow R_{S1}$ OR R_{S2}
XOR	9	R_D	R_{S1}	R_{S2}	$R_D \leftarrow R_{S1}$ XOR R_{S2}
INC	A	R			$R \leftarrow R + 1$
DEC	B	R			$R \leftarrow R - 1$
ROTATE	C	R	n	0 or 1	Rot_n R
JUMP	D	R	n		IF $R_0 \neq R$ then PC = *n*, otherwise continue

Key: R_S, R_{S1}, R_{S2}: Hexadecimal address of source registers
R_D: Hexadecimal address of destination register
M_S: Hexadecimal address of source memory location
M_D: Hexadecimal address of destination memory location
n: hexadecimal number
d_1, d_2, d_3, d_4: First, second, third, and fourth hexadecimal digits

Step	Code(hexadecimal)	Description
1	1FFE	// $R_F \leftarrow M_{FE}$, Input A from keyboard to R_F
2	240F	// $M_{40} \leftarrow R_F$, Store A in M_{40}
3	1040	// $M_{40} \leftarrow R_0$, Load A from M_{40} to R_0
4	B000	// $R_0 \leftarrow R_0 - 1$, Decrement A
5	B000	// $R_0 \leftarrow R_0 - 1$, Decrement A
6	2410	// $M_{41} \leftarrow R_0$, Store The result in M_{41}
7	1F41	// $R_F \leftarrow M_{41}$, Load the result to R_F
8	2FFF	// $M_{FF} \leftarrow R_F$, Send the result to the monitor
9	0000	// Halt