COMPUTER ORGANIZATION

(Machine Instructions, Addressing Models, ALU & Data Path)

SOLUTIONS

1. Which of following may not definitely occur in an instruction cycle?

(i) Fetch cycle
(ii) Execution cycle
(iii) Indirect cycle

(a) (i) only
(b) (i) and (ii)
(c) (iii) only
(d) (ii) and (iii)

Solution: Option (c)

Explanation:

For direct addressing mode, indirect cycle will not happen.

2. Most relevant addressing mode to write position independent code is

(a) Direct
(b) Indirect
(c) Relative
(d) Indexed Mode

Solution: Option (c)

Explanation:

Relative mode always finds based on PC value.

3. Which set of instruction transfers the memory word specified by the effective address to AC or Load to AC?

(a) $\begin{array}{c} \mathrm{DR} \leftarrow \mathrm{M[AR]} \\ \mathrm{AC} \leftarrow \mathrm{AC} + \mathrm{DR}, \mathrm{E} \leftarrow \mathrm{C_{OUT}}, \mathrm{SC} \leftarrow 0 \end{array}$

(b) $\begin{array}{c} \mathrm{DR} \leftarrow \mathrm{M[AR]} \\ \mathrm{AC} \leftarrow \mathrm{DR}, \mathrm{SC} \leftarrow 0 \end{array}$

(c) $\mathrm{M[AR]} \leftarrow \mathrm{AC}, \mathrm{SC} \leftarrow 0$

(d) $\begin{array}{c} \mathrm{DR} \leftarrow \mathrm{M[AR]} \\ \mathrm{AC} \leftarrow \mathrm{AC} \wedge \mathrm{DR}, \mathrm{SC} \leftarrow 0 \end{array}$

Solution: Option (b)

Explanation:

Memory word should be loaded into DR first, and then into AC.

4. A certain machine uses expanding opcode. It has 16-bit instructions and 6-bit addresses. It supports one address and two address instructions only. If there are ‘n’ two address instructions, the maximum number of one address instruction is

(a) $2^{16} – n$
(b) $2^{10} – n$
(c) $(2^4 – n) \times 2^6$
(d) 2^{10}

Solution: Option (c)

5. Booth’s algorithm is used in floating point

(a) Addition
(b) Subtraction
(c) Multiplication
(d) Division

Solution: Option (c)

6. Consider the following format of 32 bit floating point number:

Sign: 1 bit
Exponent: 8 bits
Mantissa: 23 bits

The mantissa is normalized and has an implied “1” on the left of the point. Normalized form of mantissa is 1.MMMMM………

The exponent is formatted using excess-127 notation, with an implied base of 2

What will be the decimal value of the following 32 floating point number stored in above mentioned format?

1 10000010 11110110000000000000000

(a) – 15.6875
(b) – 19.8976
(c) 14.1123
(d) None of these

Solution: Option (a)

Explanation:

Mantissa: $1.1111011_2 = 1.9609375_{10}$

Exponent: $10000010_2 = 130_{10}$ (because is excess – 127) = 3

Sign 1: negative number, - 1.9609375 * 2^3 = – 15.6875

7. For a carry look ahead adder, the general formula for g_i and p_i are

$g_i = a_i \cdot b_i$
$P_i = a_i + b_i$ where g_i is the 'generate' part and p_i is the 'propagate' part

Assume you are adding two 4 – bit numbers $a_3a_2a_1a_0 + b_3b_2b_1b_0$

and c_0 is the carry-in to the least significant bit (LSB) (normally, a ripple-carry adder has no carry-in to the LSB, but pretend you have a full adder adding the LSB).

Write a formula for c_2 for a carry look ahead adder.

(a) $(a_1b_1) + \left((a_0b_0)(a_1 + b_1) + (a_1 + b_1)(a_0 + b_0)c_0\right)$
(b) $(a_1b_1) + (a_1b_1)(a_0 + b_0) + (a_1 + b_1)(a_0 + b_0)c_0$
(c) $(a_1b_1) + (a_0b_0)(a_1 + b_1)c_0 + (a_1 + b_1)(a_0 + b_0)c_1$
(d) $(a_1b_1) + (a_0b_0)(a_1 + b_1) + (a_1 + b_1)(a_0 + b_0)$

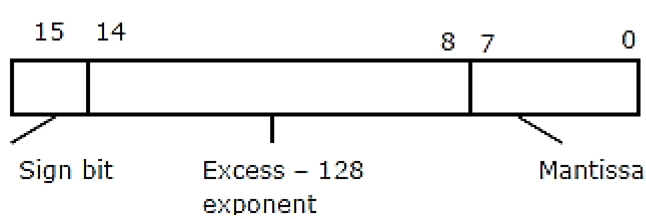
Solution: Option (a)

Explanation:

The general carry formula is: $c_{i+1} = g_i + p_ic_i$
$\therefore c_2 = g_1 + p_1c_1, c_1 = g_0 + p_0c_0$

Putting the formula of c_1 in c_2 we get, $c_2 = g_1 + p_1(g_0 + p_0c_0)$
$= (a_1b_1) + (a_0b_0)(a_1 + b_1) + (a_1 + b_1)(a_0 + b_0)c_0$

8. Consider the following floating –point format:



Mantissa is in fraction in sign magnitude form.

What will be the value of mantissa in hexadecimal for number 56.75?

(a) D3
(b) E3
(c) F3
(d) A3

Solution: Option (b)

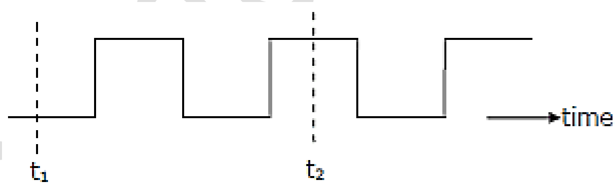
Explanation:

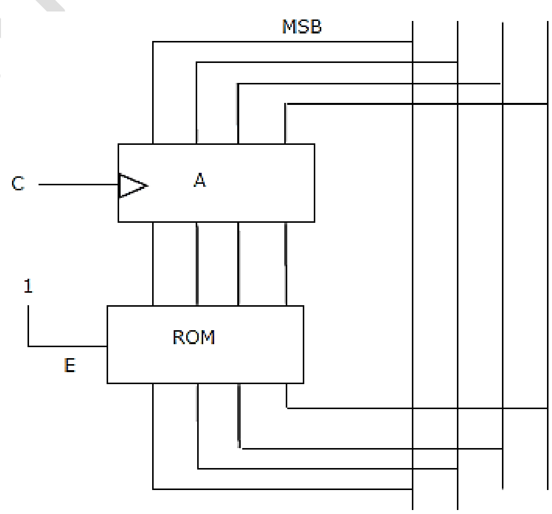
Number 56.75 is represented as, $111000.11 \times 20 = .11100011 \times 2^6$

Sign bit = 0, Exponent = 6 + 128, Mantissa = .11100011 = E3

9. In the figure, A is a parallel – in, parallel – out 4 bit register which loads at the rising edge of the clock C. The input lines are connected to a 4 bit bus W. Its output acts as the input to a 16 × 4 ROM whose output is floating when the enable input E is a 0. A partial table of the contents of the ROM is as follows:

Address	0	2	4	6	8	10	11	14
Data	0011	1111	0100	1010	1011	1000	0010	1000





The clock to the register is shown and the data on the W bus at time t_1 is 0110. The data on the bus at time t_2 is

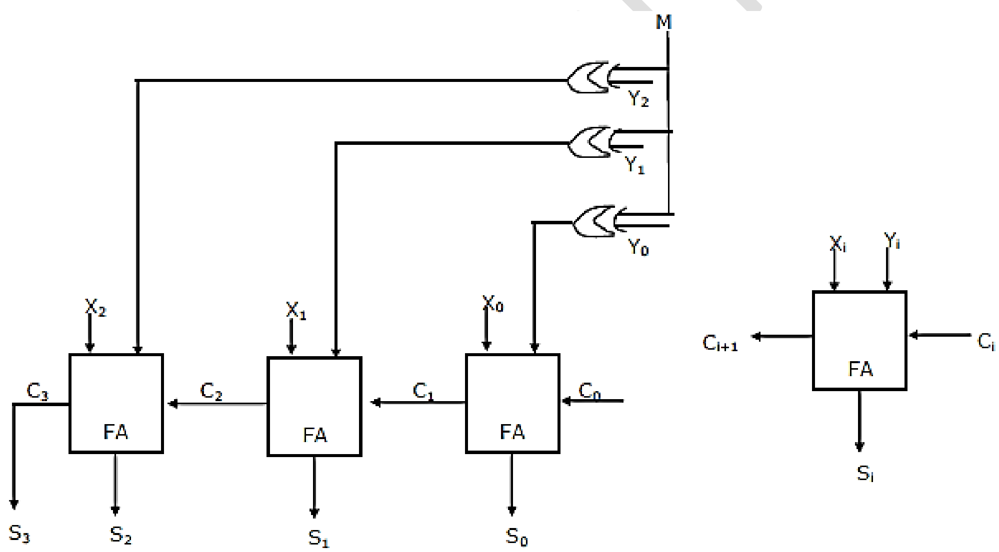
Solution: From 1030 To 970

Explanation:

At the first rising edge of clock after t = t2, O/P of shift register is 0110 = 6 in decimal. At the address 6 data 1010 is stored which is applied to the I/P of shift register and at the next rising edge, the O/P of register is 1010 = 10. At the address 10 of ROM, it contains 1000.

10. In the adder circuit shown below, $X = X_2X_1X_0$, $Y = Y_2Y_1Y_0$ are the inputs, and $S = S_3S_2S_1S_0$ is the output. M and C are control input lines, and FA refers to a full adder.

In this problem + represents binary addition, and – represents binary subtraction in either one's or two's complement form.



The logic expressions describing a full adder are

(a) $S_i = X_i \oplus Y_i \oplus C_i, C_{i+1} = C_i \cup X_iY_i$
(b) $S_i = X_i \oplus Y_i \oplus C_i, C_{i+1} = (X_i \oplus Y_i)C_i \cup X_iY_i$
(c) $S_i = X_i \oplus Y_i \oplus C_i, C_{i+1} = C_i \cap X_iY_i$
(d) $S_i = X_i \oplus Y_i \oplus C_i, C_{i+1} = C_i(X_i \cup Y_i)$

Solution: Option (b)

Explanation:

The truth table for a full adder is given below:

X_i	Y_i	C_i	S_i	C_{i+1}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

From this table it can be shown that
$S_i = X_i \oplus Y_i \oplus C_i$
and
$C_{i+1} = (X_i \oplus Y_i)C_i \cup X_iY_i$