# American International University – Bangladesh

## Faculty of Engineering

## Department of Electrical and Electronic Engineering

| | | | |
|---|---|---|---|
| **Course Name:** | Microprocessor and Embedded Systems with Lab | **Course Code:** | EEE 4103 |
| **Semester:** | Summer 2022-2023 | **Section:** | L |
| **Faculty Name:** | **Prof. Dr. Engr. Muhibul Haque Bhuyan** | | |

| | |
|---|---|
| **Assignment No:** | 3 **(individual submission consisting of 30 marks)** |

| | | | |
|---|---|---|---|
| **Student Name:** | **IFTEKHAR, KH. TANVEER** | | |
| **Student ID:** | **18-39133-3** | **Program Name:** | **BSc in CSE** |

| | | | |
|---|---|---|---|
| **Submission Link (MS forms):** | | | |
| **Submission Date:** | **09/12/2022** | **Due Date:** | **10/12/2022** |

**Special Instruction: Question may be copied from here through copy-paste. Online submission via TEAMS is also allowed. Hardcopy should be submitted to-**

**Prof. Dr. Engr. Muhibul Haque Bhuyan**
Professor
Department of Electrical and Electronic Engineering
(Room # 514, Block D, North Wing)
American International University Bangladesh (AIUB)
408/1 Kuratoli, Dhaka 1229, Bangladesh
Email: muhibulhb@aiub.edu

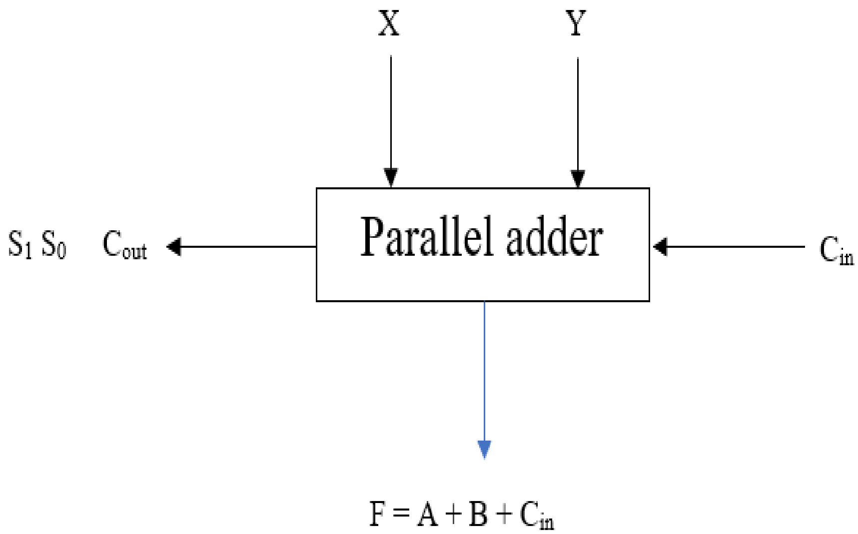
## Questions:

1. **Show the set of operations obtained by a parallel adder for a different set of inputs at the function select pins ($S_1$ $S_0$) and carry in ($C_{in}$) at the input pin.**

ID: 18-39133-3

(1) Parallel adder for different set of Inputs at the select pins ($s_1, s_0$) carry in ($c_{in}$)

| Input | | | | Output: carry | sum | | Decimal sum |
|---|---|---|---|---|---|---|---|
| $A_1$ | $A_0$ | $B_1$ | $B_0$ | $C_0$ | $S_1$ | $S_0$ | |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 |
| 0 | 0 | 1 | 0 | 0 | 1 | 0 | 2 |
| 0 | 0 | 1 | 1 | 0 | 1 | 1 | 3 |
| 0 | 1 | 0 | 0 | 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 1 | 0 | 1 | 0 | 2 |
| 0 | 1 | 1 | 0 | 0 | 1 | 1 | 3 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 4 |
| 1 | 0 | 0 | 0 | 0 | 1 | 0 | 2 |
| 1 | 0 | 0 | 1 | 0 | 1 | 1 | 3 |
| 1 | 0 | 1 | 0 | 1 | 0 | 0 | 4 |
| 1 | 0 | 1 | 1 | 1 | 0 | 1 | 5 |
| 1 | 1 | 0 | 0 | 0 | 1 | 1 | 3 |
| 1 | 1 | 0 | 1 | 1 | 0 | 0 | 4 |
| 1 | 1 | 1 | 0 | 1 | 0 | 1 | 5 |
| 1 | 1 | 1 | 1 | 1 | $\cancel{0}$1 | 0 | 6 |




Arithmetic Operation:

| | | | |
|---|---|---|---|
| F = A | X = A | Y = 0 | $C_{in}$ = 0 |
| F = A + 1 | X = A | Y = 0 | $C_{in}$ = 1 |
| F = A + B | X = A | Y = B | $C_{in}$ = 0 |
| F = A + B + 1 | X = A | Y = B | $C_{in}$ = 1 |
| F = A - B | X = A | Y = B’ | $C_{in}$ = 1 |
| F = A - B - 1 | X = A | Y = B’ | $C_{in}$ = 0 |
| F = A – 1 | X = A | Y = all 1’s | $C_{in}$ = 0 |

Logical Operation:

| | | | |
|---|---|---|---|
| NOT | F = A’ | X = A | Y = 1 |
| XOR | F = $A \oplus B$ | X = A | Y = B |
| AND | F = A . B | X = A + B’ | Y = B’ |
| OR | F = A + B | X = A + B | Y = 0 |

2. **Design an adder/subtractor circuit with one selection variable ‘S’ and two inputs ‘A’ and ‘B’: when S = 0 the circuit performs A+B. When S = 1 the circuit performs A-B by taking the 2’s complement of B.**

**ANS:**

ID: 18-39133-3

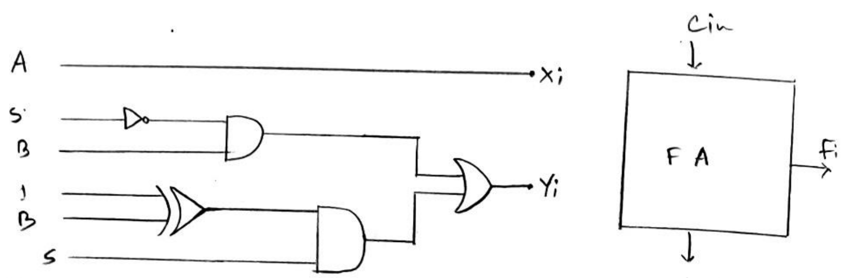
(2) Given, $S=0 \rightarrow A+B$

$S=1 \rightarrow A-B = A+(-B)$

| S | A | B | $x_i$ | $Y_i$ |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 1 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 1 | 1 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 | 0 |
| 1 | 1 | 0 | 1 | 1 |
| 1 | 1 | 1 | 1 | 0 |

$x_i = \bar{S}A + SA = A$

$Y_i = \bar{S}B + \bar{B}S$

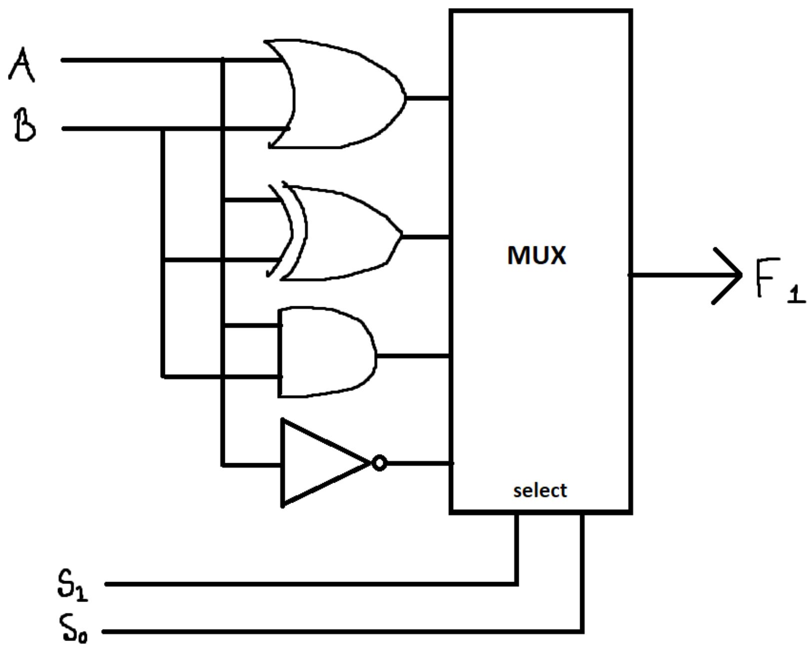



if $S = 0/1$ — $x_i = A_i$

$S=0$ — $Y_i = B_i$ ; $S=1$ — $Y_i = 2$'s complement of $B_i$

3. **Draw the combined logic and arithmetic circuit of ALU where the output can be controlled by changing the value of the mode select pin. List the logic operations that can be performed by a logic circuit based on the value of the input and select pins. Discuss the process to obtain an AND operation from an Equivalence operation with logic expression and diagram.**

**ANS:**



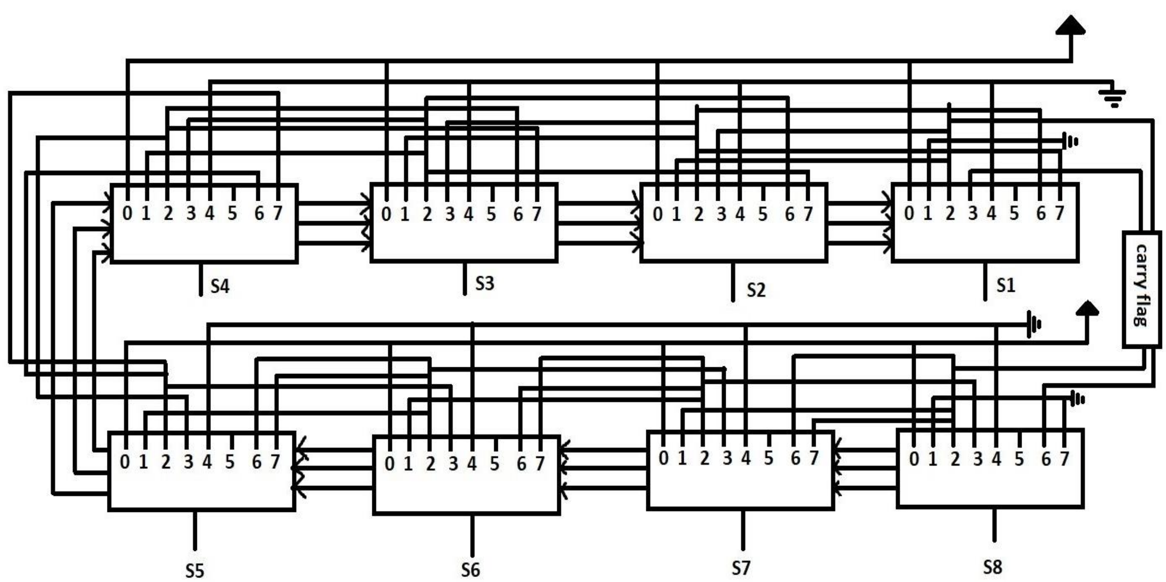

| S1 S2 | Output | Operation |
|---|---|---|
| 0 0 | F = A + B | OR |
| 0 1 | F = A ⊕ B | XOR |
| 1 0 | F = A . B | AND |
| 1 1 | F = A’ | NOT |

4. **Design an 8-bit shifter circuit for the listed shift functions provided in Table 1. Explain its operation for various cases of select inputs.**

**Table 1: Functions of control variables**

| **Binary Code** | **Functions of selection variables** | | | | | |
|---|---|---|---|---|---|---|
| | **A** | **B** | **D** | **F with $C_{in}$ = 0** | **F with $C_{in}$ = 1** | **H** |
| 0 0 0 | Input Data | Input Data | None | A-1 | A | 1’s to the output Bus |
| 0 0 1 | R1 | R1 | R1 | A+B | A+B+1 | Shift Left with $I_L$=0 |
| 0 1 0 | R2 | R2 | R2 | A-B-1 | A-B | No Shift |
| 0 1 1 | R3 | R3 | R3 | A | A+1 | Circulate Left with Carry |
| 1 0 0 | R4 | R4 | R4 | $\bar{A}$ | X | 0’s to the output Bus |
| 1 0 1 | R5 | R5 | R5 | AX**OR** B | X | - |
| 1 1 0 | R6 | R6 | R6 | A **AND** B | X | Circulate-Right with Carry |
| 1 1 1 | R7 | R7 | R7 | A **OR** B | X | Shift Right with $I_R$=0 |

**ANS:**



**5. Develop the shifts in binary and hexadecimal formats using the information provided in Table 1 for the following micro-operations:**

| | | | |
|---|---|---|---|
| **i.** | **R7←R3+R4** | **ii.** | **R4←3(R4 – 0)/3** |
| **iii.** | **R3←SHL R3** | **iv.** | **Output←R5** |
| **v.** | **R5←R1** | **vi.** | **R2←0** |
| **vii.** | **R6←Input** | **viii.** | **R6←R4-R2** |
| **ix.** | **R2←SHR R5** | **x.** | **R3←CRC R7** |

**One example is shown as follows:**

| Micro-operation | A | B | D | F | $C_{in}$ | H | In Hex |
|---|---|---|---|---|---|---|---|
| R5← CRC (R3+R4) | 011 | 100 | 101 | 001 | 0 | 110 | 7296h |

**The necessary bits for the control word are presented in Table 2.**

**Table 2: 16-bit control word sequence**

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A | | | B | | | D | | | F | | | $C_{in}$ | H | | |

**ANS:**

| Micro-operation | A | B | D | F | $C_{in}$ | H | In Hex |
|---|---|---|---|---|---|---|---|
| R7←R3+R4 | 011 | 100 | 111 | 001 | 0 | 001 | 7391h |
| R4←3(R4 – 0)/3 | 100 | 000 | 100 | 011 | 0 | 010 | 8232h |
| R3←SHL R3 | 011 | 000 | 011 | 000 | 0 | 001 | 6181h |
| Output←R5 | 101 | 000 | 000 | 011 | 0 | 001 | A031h |
| R5←R1 | 001 | 000 | 101 | 000 | 0 | 101 | 2285h |
| R2←0 | 000 | 000 | 010 | 000 | 0 | 100 | 0104h |
| R6←Input | 000 | 000 | 110 | 111 | 1 | 010 | 037Ah |
| R6←R4-R2 | 100 | 010 | 110 | 010 | 1 | 001 | 8B29h |
| R2←SHR R5 | 101 | 000 | 010 | 011 | 1 | 001 | A139h |
| R3←CRC R7 | 111 | 000 | 011 | 011 | 1 | 001 | E1B9h |

# American International University – Bangladesh
## Faculty of Engineering
## Department of Electrical and Electronic Engineering


6. **Design a 4-bit ALU for operations listed in Table 1.**

**ANS:**

ID: 18-39133-3

(6)

| ROM address | Micro instruction |
|---|---|
| 0 | $x = 1$ ; if ($q_s = 1$) then go to 1<br>if ($q_a = 1$) then go to 1<br>if ($q_a, q_b = 0$) then go to 0 |
| 1 | $B_3 \leftarrow \bar{R_5}$ |
| 2 | if ($S = 1$) then go to 4 |
| 3 | $A \leftarrow A + B$ $E \leftarrow C_{out}$ go to 0 |
| 4 | $A \leftarrow A + \bar{B} + 1$ , $E \leftarrow C_{out}$ |
| 5 | if ($E = 1$) then go to 0 $L \leftarrow 0$ |
| 6 | $A \leftarrow \bar{A}$ |
| 7 | $A \leftarrow A + 1$ , $A_s \leftarrow \bar{A}_s$ go to 0 |

| ROM Address | $x$ 1 | $S_2$ 2 | $S_1$ 3 | $S_0$ 4 | cin 5 | L 6 | y 7 | z 8 | w 9 | Address 10 | 11 | 12 | Select 13 | 14 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| 3 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 4 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 5 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 |
| 6 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 7 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 |

ID: 18-39133-3

* Encoding of ALU operations:

| OPR select | Operation | Symbol |
|---|---|---|
| 00000 | Transfer A | TSFA |
| 00001 | Increment A | INCA |
| 00010 | ADD A+B | ADD |
| 00101 | Substract A-B | SUB |
| 00110 | Decrement A | DEC A |
| 01000 | And A and B | AND |
| 01010 | OR A and B | OR |
| 01100 | XOR A and B | XOR |
| 01110 | Complement A | COM A |
| 10000 | Shift Right A | SHR A |
| 11000 | shift left A | SHL A |

7. **Develop a flow chart to perform addition and subtraction operations between two signed numbers A and B, where the conditions are A>B and A<B. Write down the microinstruction for each step of the flow chart.**

**ANS:**

ID: 18-39133-3

- We know, two signs of $A_s$ and $B_s$ are compared with XOR Gate. if the output of the gate is zero, then signs are identical. if 1 then signs will be different.

- For ADD operations, identical signs diclare the magnitudes to be added.

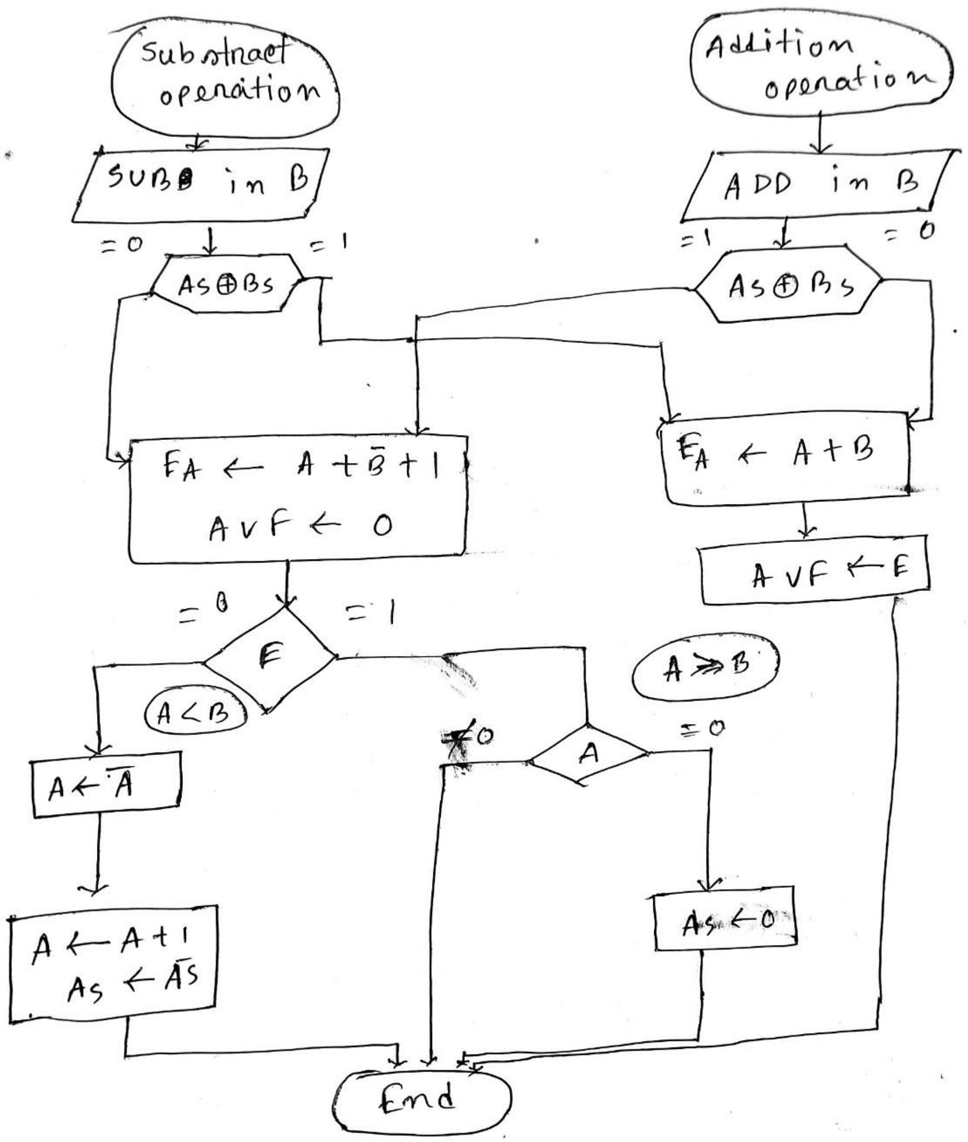
- For substract operation, different signs declare the magnitudes to be added.

- Micro-operation ~~EAA~~ A+B, The carry in E after the addition constitute an overflow if it equal to 1. The value of E transfered into add overflow flipflop AVF.

- If the number is 0, the sign A, must be made to avoid negative 0, $A_0$ in E indicates that $A < B$, $A_1$ in E indicates that $A \geq B$.

ID : 18-39133-3

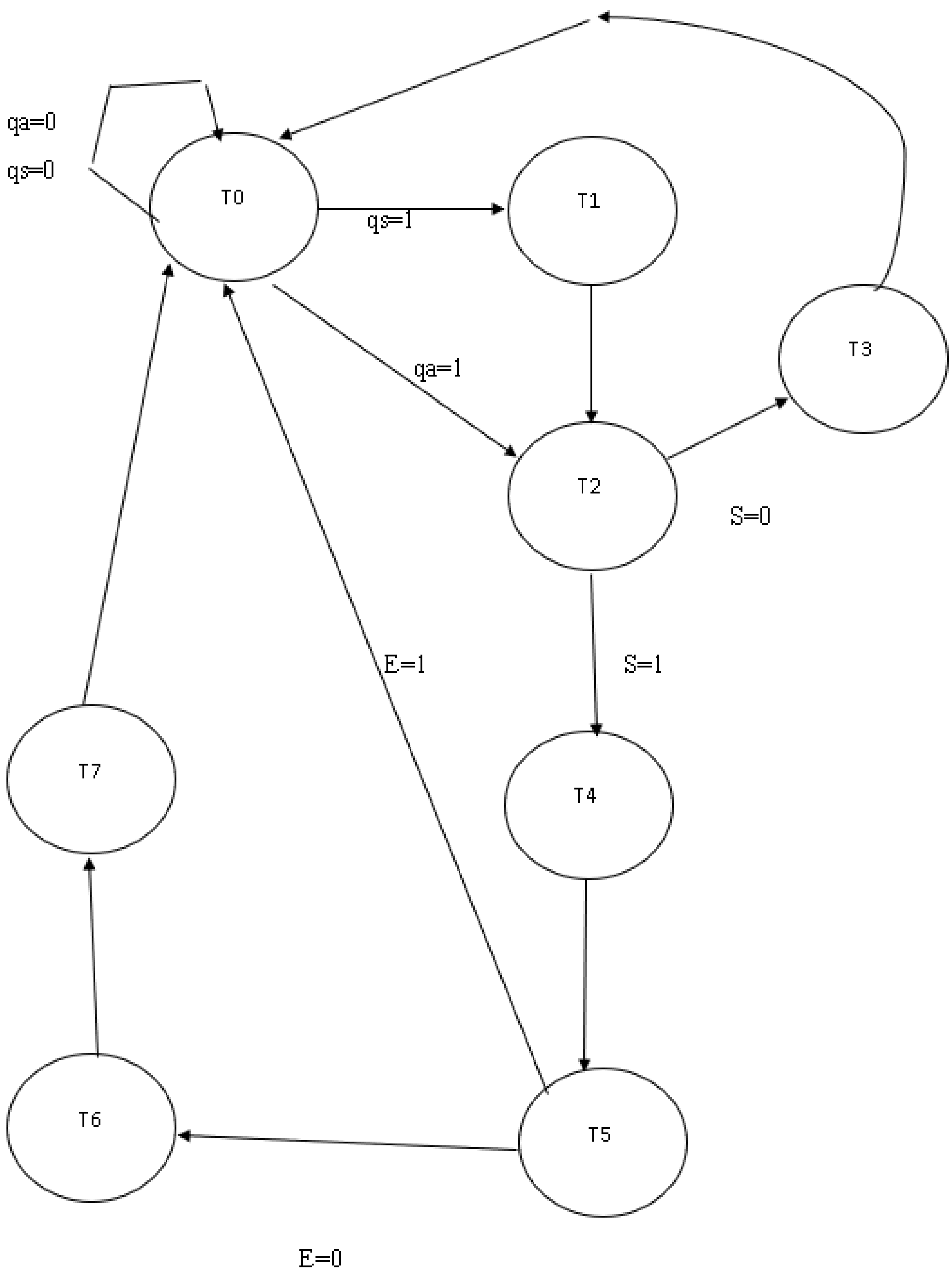
□ Flowchart



| ROM ADDRESS | MICROINSTRUCTIONS |
|---|---|
| 0 | x=1, if(qs=1) then (go to 1) if(qa=1) then (go to 2) if(qs ^ qa=0) then (go to 0) |
| 1 | Bs$\leftarrow$Bs’ |
| 2 | If (S=1) then (go to 4) |
| 3 | A$\leftarrow$A+B, E$\leftarrow$Cout ,go to 0 |
| 4 | A$\leftarrow$A+B’+1,E$\leftarrow$Cout |
| 5 | If(E=1) then (go to 0) , E$\leftarrow$0 |
| 6 | A$\leftarrow$A’ |
| 7 | A$\leftarrow$A+1,As$\leftarrow$As’, go to 0 |

8. **Draw the control state diagram to represent the sequences of the developed flow chart of question no. 7.**

**ANS:**



**9. Develop the control memory outputs for the sequence in Table 3 using the information listed in Table 1. To complete the memory outputs, use the microinstructions that you have developed in question no. 7.**

**Table 3: Control memory bit sequence**

| | | | ROM outputs | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ROM Address | | | Control Word | | | | | | | | | Address | | | Mux Select | |
| | | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 |
| 0 | 0 | 0 | | | | | | | | | | | | | | |
| 0 | 0 | 1 | | | | | | | | | | | | | | |
| 0 | 1 | 0 | | | | | | | | | | | | | | |
| 0 | 1 | 1 | | | | | | | | | | | | | | |
| 1 | 0 | 0 | | | | | | | | | | | | | | |
| 1 | 0 | 1 | | | | | | | | | | | | | | |
| 1 | 1 | 0 | | | | | | | | | | | | | | |
| 1 | 1 | 1 | | | | | | | | | | | | | | |

**ANS:**

| | | | ROM outputs | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ROM Address | | | Control Word | | | | | | | | | Address | | | Mux Select | |
| | | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 |

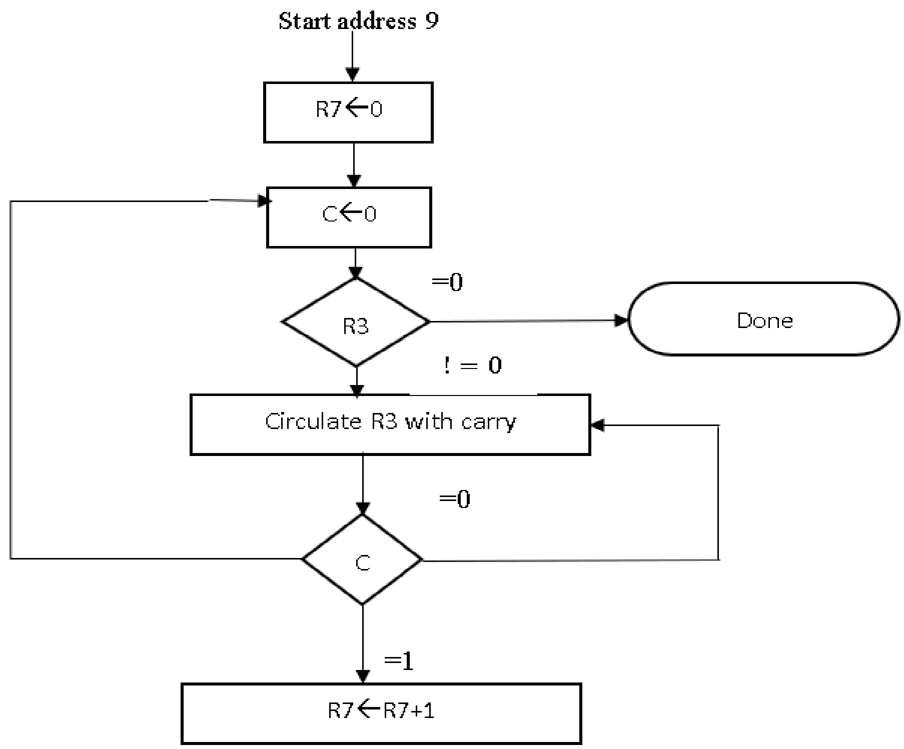
**10. Show the flow chart and develop the micro-instruction table for counting the number of 1's in register, R3, and storing the count in register R7. Assume that the starting address is 9. The operation should follow the control word format of Tables 1 and 2. If register, R3 contains the data 10011011 then what would be the contents of the register, R7 after this micro-operation is completed?**

**Control memory location contents' format:**

**Table 4: Control memory bit sequence**

| | | | | | | ROM outputs | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ROM Address | | | | | | A | | | B | | | D | | | F | | | $C_{in}$ | H | | | Mux Select | | | | Address | | | | | |
| | | | | | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 |
| 0 | 0 | 0 | 0 | 0 | 0 | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | 0 | 0 | 0 | 0 | 1 | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | 0 | 0 | 0 | 1 | 0 | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | 0 | 0 | 0 | 1 | 1 | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | 0 | 0 | 1 | 0 | 0 | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | 0 | 0 | 1 | 0 | 1 | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | 0 | 0 | 1 | 1 | 0 | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | 0 | 0 | 1 | 1 | 1 | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | 0 | 1 | 0 | 0 | 0 | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | 0 | 1 | 0 | 0 | 1 | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | 0 | 1 | 0 | 1 | 0 | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | 0 | 1 | 0 | 1 | 1 | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | 0 | 1 | 1 | 0 | 0 | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | 0 | 1 | 1 | 0 | 1 | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | 0 | 1 | 1 | 1 | 0 | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | 0 | 1 | 1 | 1 | 1 | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | 1 | 0 | 0 | 0 | 0 | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | 1 | 0 | 0 | 0 | 1 | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | 1 | 0 | 0 | 1 | 0 | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | 1 | 0 | 0 | 1 | 1 | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | 1 | 0 | 1 | 0 | 0 | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | 1 | 0 | 1 | 0 | 1 | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | 1 | 0 | 1 | 1 | 0 | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | 1 | 0 | 1 | 1 | 1 | | | | | | | | | | | | | | | | | | | | | | | | | | |

**ANS:**




| Rom address | Microinstructions |
|---|---|
| 9 | R7←0 |
| 10 | R3←R3,C←0 |
| 11 | IF (Z=1) then (go to external address) |
| 12 | R3←crc R3 |
| 13 | If (C=0) then (go to 12) |
| 14 | R7←R7+1,go to 10 |

If register R3 contains the data 10011011 then the contents of the register R7 is 00000101.

**11. Find the baud rate for the three operating modes when $f_{OSC}$ = 8 MHz and UBRRn = 24. Calculate the baud error and comment whether there will be any communication error or not.**

**ANS:**

fOSC = 8 MHz

UBRRn = 24

For asynchronous normal mode:

Baud Rate = fOSC /16(UBRRn +1)

= 8×106 /16(24+1)

=20000 bps

Baud Error Rate = (19200-20000/19200) * 100%

= -4.17% > ± 2%

So, in this mode there will be communication error for the given information.

For asynchronous double speed mode:

Baud Rate = fOSC /8(UBRRn +1)

= 8×106 /8(24+1)

=40000 bps

Baud Error Rate = (38400-40000/38400) * 100%

= -4.17% > ± 2%

So, in this mode there will be communication error for the given information.

For synchronous master mode:

Baud Rate = fOSC /2(UBRRn +1)

= 8×106 /2(24+1)

=160000 bps

Baud Error Rate = (128000-160000/128000) * 100%

= -25% > ± 2%

So, in this mode there will be communication error for the given information.

12. **Calculate the PWM frequency for the output when using fast PWM mode and phase correct PWM when $f_{OCO}$ is 8 MHz and the pre-scale factors are 1, 8, 64, 256, or 1024. Comment on the results afterward.**

**ANS:**

Given,

fOCO =8 MHz

The pre-scale factors are: 1, 8, 64, 256, or 1024.

The PWM frequency for the Fast PWM mode(Assuming Pre-scale factor =1024)

=f_clk_IO/N*256

= 8*106/ 1024*256 [fOCO = f_clk_IO]

=30.52 Hz

The PWM frequency for the Phase Correct PWM mode(Assuming Pre-scale factor =1024)

=f_clk_IO/N*510

= 8*106/ 1024*510 [fOCO = f_clk_IO]

=15.32 Hz

13. **What values should be set in the TCCR0A and TCCR0B registers to operate them in the inverting and non-inverting modes while up or down counting? Which flags are set or reset when counting is completed in the fast and phase correct PWM modes? When the Output Compare (OC0x) is cleared?**

**ANS:**

Step1 **|** Setting up the LEDs

Open up Arduino IDE and open a new sketch by going to File>New. Let's get the LEDs working first. Go ahead and delete the default

So that you get a clear screen.This `int` is for the brightness of the LED. This number can range anywhere between 0 and 255 with 0 being completely off and 255 being completely on. For example

Now let's test each LED to make sure there working. In between the `voidsetup()` and the color functions write a `voidloop()` function. `voidloop()` executes the code inside it then loops to the beginning of the function and does it all over again. `voidloop()` is were all the "Moving pieces" of your code goes. In the `voidloop()` function set all of the LEDs to completely on.

```
void loop(){
    Red(255);
    Green(255);
    Blue(255);
}
```

Now we need to define our pins the red pin is connected to pin 3. The green pin is connected to pin 4. And the blue pin is connected to pin 5. So lets make 3 `contsint` with values as `3`, `4`, and `5`.

By now your script should look like this:

```
void Blue(int VAL){
 analogWrite(redPin, 0);
 analogWrite(greenPin, 0);
 analogWrite(bluePin, VAL);
}
const int redPin = 3;
const int greenPin = 4;
const int bluePin = 5;
```

conclusion:

All three LEDs should now light up. If not check *fig1* and make sure your wiring is correct. Then check for errors in your code. If all else fails see the Questions or Comments at the bottom of the tutorial.

If all LEDs light up. Congratulations! You have completed step one of the tutorial. Move on the Step 2.

Step2

Now that we have the LED lights working we need to turn our attention to the HC-SR04 sonar sensor. But first, if you haven't already, let's turn off the LEDs by removing the color functions form the `voidloop()` like this.

```
void loop(){
//Nothing to see here :-]
}
```

First at the top of our sketch we need to set our pins for echo and trigger.

```
const int echoPin = 9;
const int trigPin = 8;
```

We also need to make another `int` to keep track of our distance. Do not set this as `const` because this value will change. To stay organized put this in-between the pin `int` and the `void setup()`

```
int distance;
```

NOTE: Do not set `int distance` to equal anything yet. We will do that later.

Next we need to set `echoPin` as `INPUT` and `trigPin` as `OUTPUT` inside of `voidsetup()`

```
void setup(){
 pinMode(redPin, OUTPUT);    /*
 pinMode(greenPin, OUTPUT);     This code should already be here
 pinMode(bluePin, OUTPUT);    */
 pinMode(echoPin, INPUT);     // Add this line
 pinMode(trigPin, OUTPUT);    // Add this line
}
```

Now we need to code the sonic burst. To save space I'm going to call it a ping from now on. So first we set `trigPin` to `LOW` then to `HIGH` for 10 µs then back to `LOW`. To stay organized lets do that in a `void`. How do you set a pin to `LOW` or `HIGH`? I'm glad you asked. First we need to understand what `LOW` and `HIGH` even mean. `LOW` and `HIGH` are constant values. When a pin is set as `OUTPUT` in the `pinMode()`, `LOW` means that a pin is at 0V. (on 5V boards). `HIGH` means that a pin is at 5V. To set a pin as `LOW` or `HIGH` we use the `digitalWrite();` function `digitalWrite();` takes in two values. First it takes in a `int` value as the pin key. Just like `analogWrite();` this value can either be in the form of physical number (like `3` for example) or as a `int` name (like `redPin` for example). The second value it takes in is a state value (normally `LOW` or `HIGH`). Now that we know how to do it, let's code the ping in a `void` function that we'll call `sendPing();`

```
void sendPing(){
 digitalWrite(trigPin, LOW);
 delayMicroseconds(5);
 digitalWrite(trigPin, HIGH);
 delayMicroseconds(10);
 digitalWrite(trigPin, LOW);
}
```

Now in the `void loop()` we need to call the `sendPing()` function then we need to figure out how long `echoPin` reads `HIGH`. This is easy with the function `pulseIn();` which takes a pin key (either as a number or `int` name) and a state (either `HIGH` or `LOW`) then it reads how long said pin key was `HIGH` or `LOW`, then returns the value as a `int` in µs. So let's make a new `int` and call it `duration`. `duration` will be our Time value in the equation above. Now we can set `distance` to equal to `duration` x `0.034/2`. Code it like this.

Step3

Now it is finally time to take all of our information we have gathered and make it do something! This is actually the simple part of project. All we have to do is write a string of `if()` statements at the bottom of `void loop()`.

First we need to decide at what distances which light we want to show.

Here are my choices;

- Red @ distances btw 33 - 50 cm
- Green @ distances btw 16 - 32 cm
- Blue @ distances btw 1 - 15 cm

Now we write our `if()` statement at the bottom of `void loop()`

```
void loop(){
sendPing();                                /*
int duration = pulseIn(echoPin, HIGH);
distance = 0.034 * duration / 2;           This code should already be
here
distance = clamp(distance, 0, 50);
printToScreen(distance);                     */
if(distance >= 0 && distance <= 15){
   Blue(255);
   Green(0);
   Red(0);
   }
else if(distance >= 16 && distance < 33){       Add these lines of code
   Green(255);
   Blue(0);
   Red(0);
   }
   else if (distance >= 33 && distance <= 50){
     Red(255);
     Green(0);
     Blue(0);
   }                                             */
}
```

NOTE: You do not HAVE TO add `Green(0);`,`Blue(0);`,`etc.` for the `if()` statement to work. I have it in there so that --- in the case that I want multiple light to light up within a said range --- I can do so by editing the null values.

**14. What is the travel time as per the following program codes of Arduino if the object is at 56 cm from the sonar sensor? Show calculations. How many LEDs are turned ON? If we want that the system will display a message for this distance that how many LEDs are turned ON/OFF (for example, "Two LEDs are turned ON and One LED is turned OFF") then what changes should we include here?**

**Program code:**

```
int distanceThreshold;
int dcm;

long readUltrasonicDistance(int triggerPin, int echoPin)
{
pinMode(triggerPin, OUTPUT); // Clear the trigger
digitalWrite(triggerPin, LOW);
delayMicroseconds(2);

// Sets the trigger pin to HIGH state for 10 microseconds
digitalWrite(triggerPin, HIGH);
delayMicroseconds(10);
digitalWrite(triggerPin, LOW);
pinMode(echoPin, INPUT);

// Reads the echo pin, and returns the sound wave travel time in microseconds
return pulseIn(echoPin, HIGH);
}

void setup()
{
serial.begin(9600);
pinMode(2, OUTPUT);
pinMode(3, OUTPUT);
pinMode(4, OUTPUT);
}

void loop()
{
// set threshold distance to activate LEDs
distanceThreshold = 80;
// measure the ping time and find the distance in cm
dcm = 0.01723*readUltrasonicDistance(7, 6);
```

```
serial.print(dcm);
serial.println(" cm");

if (dcm > distanceThreshold) {
        digitalWrite(2, LOW);
        digitalWrite(3, LOW);
        digitalWrite(4, LOW);
        }

if (dcm < distanceThreshold && dcm > distanceThreshold-30) {
        digitalWrite(2, HIGH);
        digitalWrite(3, LOW);
        digitalWrite(4, LOW);
        }

 if (dcm < distanceThreshold-30 && dcm > distanceThreshold-60) {
        digitalWrite(2, HIGH);
        digitalWrite(3, HIGH);
        digitalWrite(4, LOW);
        }

if (dcm < distanceThreshold-60 && dcm > distanceThreshold-80) {
        digitalWrite(2, HIGH);
        digitalWrite(3, HIGH);
        digitalWrite(4, HIGH);
        }
        delay(100); // Wait for 100 millisecond(s)
        }
```

**ANS:**

The speed of sound 34300cm/s.

Travel time= 56/34300= 0.0016 s

1 LED is turned on.

If we want the system to display message, we need to include some conditions in the void loop() function. The modified function-

```
void loop()

{

// set threshold distance to activate LEDs

distanceThreshold = 80;

// measure the ping time and find the distance in cm

dcm = 0.01723*readUltrasonicDistance(7, 6);
```

```
serial.print(dcm);
serial.println(" cm");
if (dcm > distanceThreshold) {
digitalWrite(2, LOW);
        digitalWrite(3, LOW);
digitalWrite(4, LOW);
Serial.println("3 LEDs are turned off");
}
if (dcm < distanceThreshold && dcm > distanceThreshold-30) {
        digitalWrite(2, HIGH);
        digitalWrite(3, LOW);
digitalWrite(4, LOW);
Serial.println("1 LEDs is turned on and 2 LEDs are turned off");
}
if (dcm < distanceThreshold-30 && dcm > distanceThreshold-60) {
        digitalWrite(2, HIGH);
        digitalWrite(3, HIGH);
digitalWrite(4, LOW);
Serial.println("2 LEDs is turned on and 1 LED are turned off");
}
if (dcm < distanceThreshold-60 && dcm > distanceThreshold-80) {
        digitalWrite(2, HIGH);
        digitalWrite(3, HIGH);
digitalWrite(4, HIGH);
Serial.println("3 LEDs are turned on");
}
delay(100); // Wait for 100 millisecond(s)
}
```

15. **What are the contents of the various registers and various buses and flags after the next clock of Fig. 1 as per Table 4 of question 10 when the ROM addresses are 001011, 001100, 001101, and 001110?**

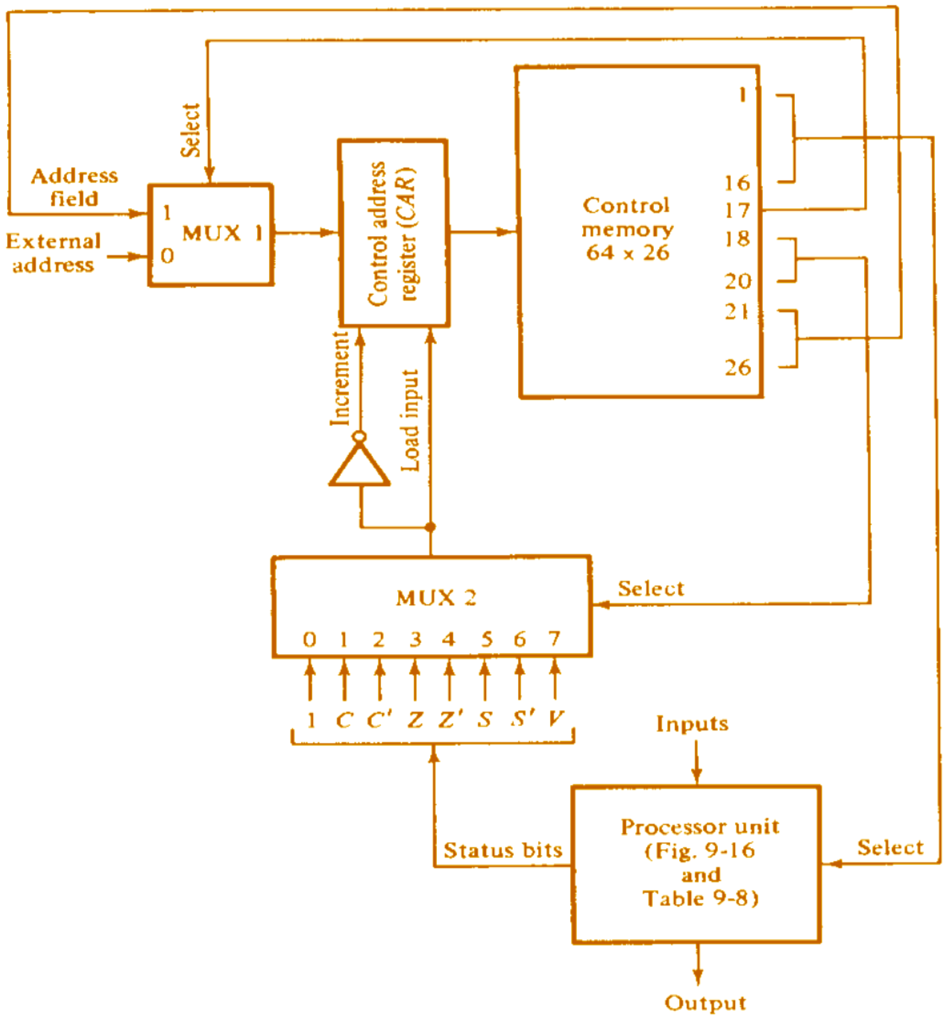



Fig. 1 Micro-program control for processor unit

**ANS:**

The register is given serial input with serial data 101101, i.e. the first digit from the left is replaced with the first digit from the right of serial data.

The contents of the register after 1st pulse 1110.

Similarly, after the second shift, the contents of the register are 0111.

after the 3rd clock pulse, we get 1011.

after the 4th clock pulse, we get 1101.

after the 5th clock pulse, we get 0110

after the 6th clock pulse, we get 1011.