TRAFFIC LIGHT CONTROLLER

BY
SUBATHRA S

TRAFFIC LIGHT CONTROLLER

OBJECTIVE

To write an assembly language program in 8085 to interface traffic light controller with 8085 Microprocessor trainer kit and simulating the sequence of traffic light states.

APPARATUS REQUIRED

- 8085 Microprocessor trainer kit.
- Traffic light controller.
- Power Supply.
- Flat Ribbon Cable.

DESCRIPTION

Combination of Red, Amber and Green LEDs are provided to indicate Halt, Wait and Go states for vehicles. Combination of Red and Green LEDs are provided for pedestrian crossing. 36 LEDs are arranged in the form of an intersection. At the left corner of each road, a group of 5 LEDs (Red, Amber and Green) are arranged in the form of a T-section to control the traffic of that road. Each road is named as North N, South S, East E and West W.

L_1,L_{10}, L_{19} and L_{28} (Red) are for stop signal for the vehicles on the road N,S,W and E respectively.

L_2,L_{11},L_{20} and L_{29} (Amber) indicate wait state for the vehicles on the road N,S,E and W respectively.

L_3,L_4 and L_5 (Green) are for left, straight and right turn for the vehicles on the road S.

Similarly L_{12} - L_{13} - L_{14} , L_{23} - L_{22} - L_{21} and L_{32} - L_{31} - L_{30} simulates same function for the roads E, N & W respectively. A total of 16 LEDs (2 Red & 2 Green at each road) are provided for pedestrian crossing. L_7 - L_9, L_{16} - L_{18}, L_{25} - L_{27} & L_{34} - L_{36} (Green) when on allows pedestrians to cross and L_6 - L_8, L_{15} - L_{17}, L_{24} - L_{26} & L_{33} - L_{35} (Red) when on alarms the pedestrians to wait.

To minimize the hardware pedestrians indicator LEDs (both Green and Red) are connected to some port lines (PC_4 to PC_7) with Red inverted. Red LED's L_{10} and L_{28} are connected to port lines PC_2 to PC_3 while L_1 and L_{19} are connected to lines PC_0 and PC_1 after inversion. All other LEDs (Amber and Green) are connected to Port A and port B.

INSTALLATION PROCEDURE

SDA_85M to NIFC_11 interface connection details:

1.Connect p3 on 85M to the connector C1 on the interface using a 26 core FRC.

Care should be taken such that, pin1 of P3 on the kit coincides with pin1 of the cable [Observe the notch on the cable connector]

2.Power connection:

Connect +5v,GND to the interface. Color codes of power connection on the interface

+5v - Orange, Blue, White
GND - Black.

3.Enter the Program.

4. Execute the program,

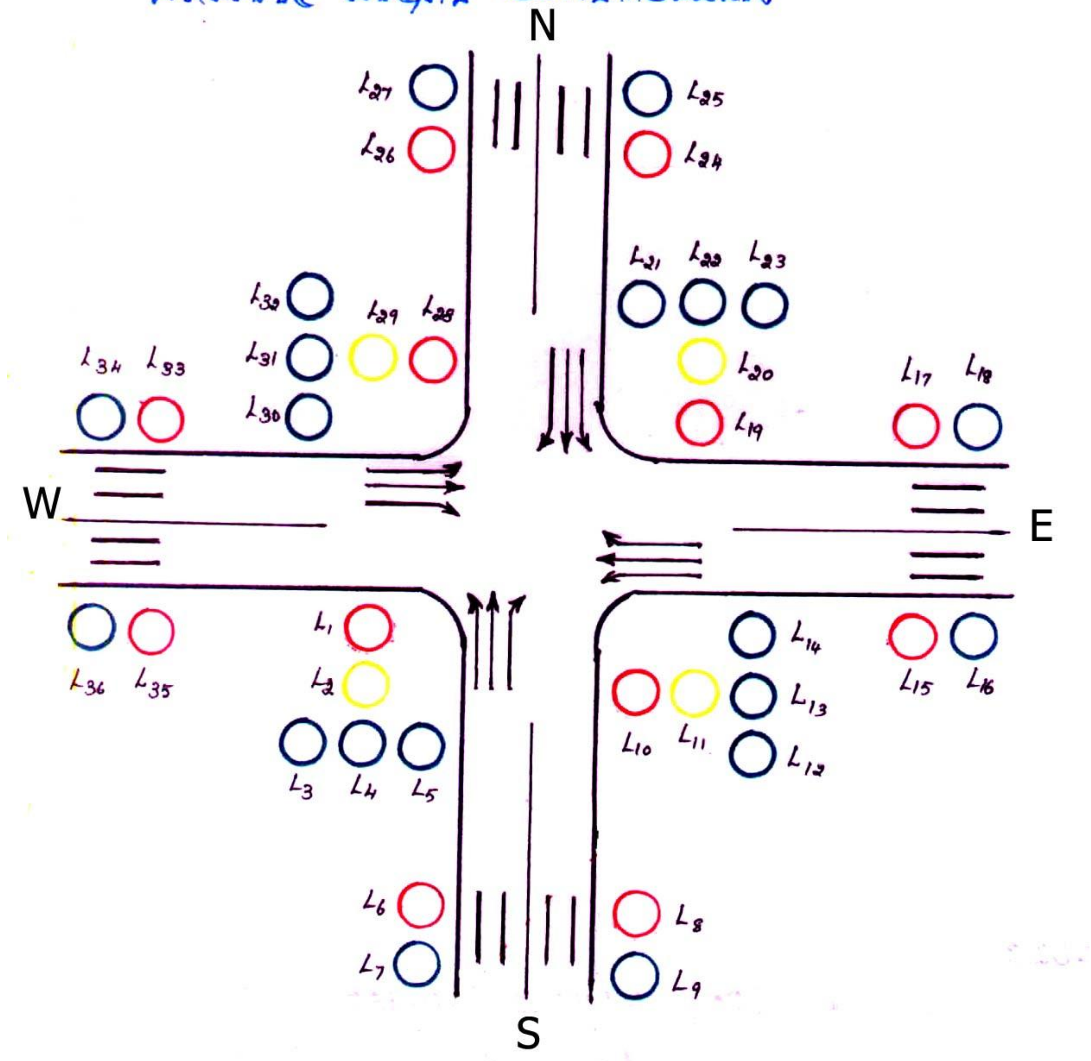
Go <Starting address> <EXE>

The LEDs on the interface glow according to certain sequence.

ASSEMBLY LANGUAGE PROGRAM

ADDRESS	LABEL	MNEMONICS	OPCODE/OPERAND
C000		**MVI A,80$_H$**	3E 80
C002		**OUT CWR**	D3 DB
C004	**REPEAT**	**MVI E,03$_H$**	06 03
C006		**LXI H,C100$_H$**	21 00 C1
C009	**NEXTSTAT**	**MOV A,M**	7E
C00A		**OUT PORTA**	D3 D8
C00C		**INX H**	23
C00D		**MOV A,M**	7E
C00E		**OUT PORTB**	D3 D9
C010		**INX H**	23
C011		**MOV A,M**	7E
C012		**OUT PORTC**	D3 DA
C014		**CALL DELAY**	CD 1F C0
C017		**INX H**	23
C018		**DCR E**	05
C019		**JNZ NEXTSTAT**	C2 09 C0
C01C		**JMP REPEAT**	C3 04 C0
C01F	**DELAY**	**LXI D,3000$_H$**	11 00 30
C022	**L2**	**MVI C,FF$_H$**	0E FF
C024	**L1**	**DCR C**	0D
C025		**JNZ L1**	C2 24 C0
C028		**DCX D**	1B
C029		**MOV A,D**	7A
C02A		**ORA E**	B3
C02B		**JNZ L2**	C2 22 C0
C02E		**RET**	C9

TRAFFIC LIGHT CONTROLLER



G	G	G	Y	G	G	G	Y	
L_{14}	L_{13}	L_{12}	L_{11}	L_5	L_4	L_3	L_2	PORT A

G	G	G	Y	G	G	G	Y	
L_{32}	L_{31}	L_{30}	L_{29}	L_{23}	L_{22}	L_{21}	L_{20}	PORT B

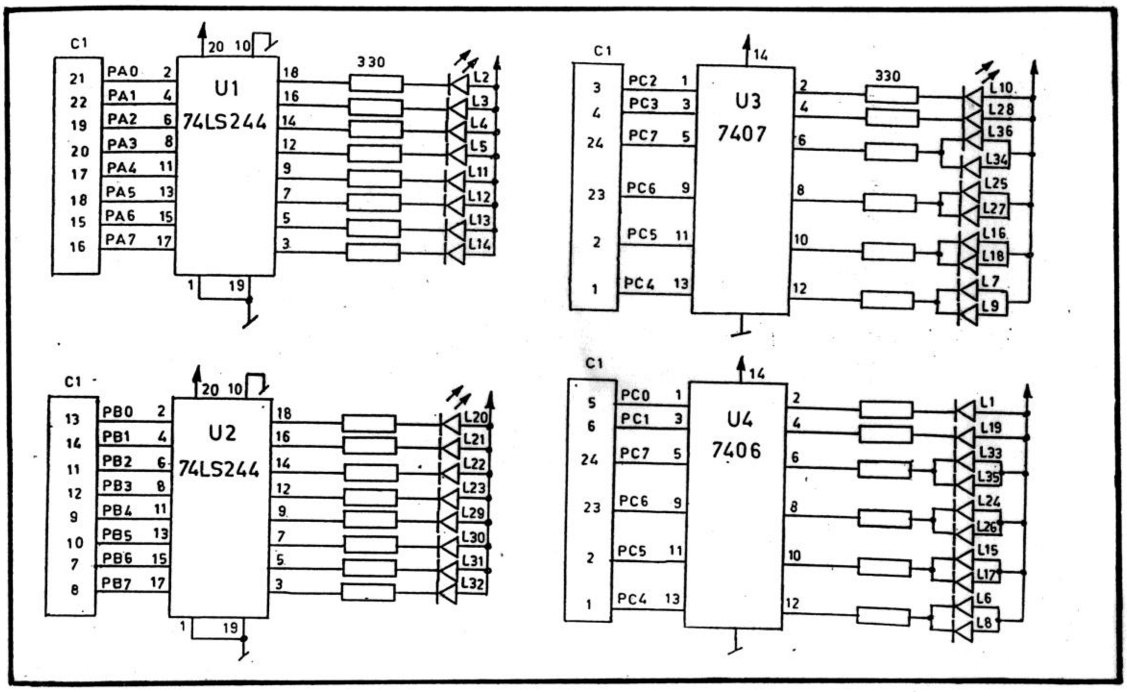
R	G	R	G	R	G	R	G	R	R	R	R	
▨ L_{33} L_{35}	L_{34} L_{36}	▨ L_{24} L_{26}	L_{25} L_{27}	▨ L_{15} L_{17}	L_{16} L_{18}	▨ L_6 L_8	L_7 L_9	L_{28}	L_{10}	▨ L_{19}	▨ L_1	PORT C

▨ POSITIVE LOGIC ('1' TO GLOW)

□ NEGATIVE LOGIC ('0' TO GLOW)

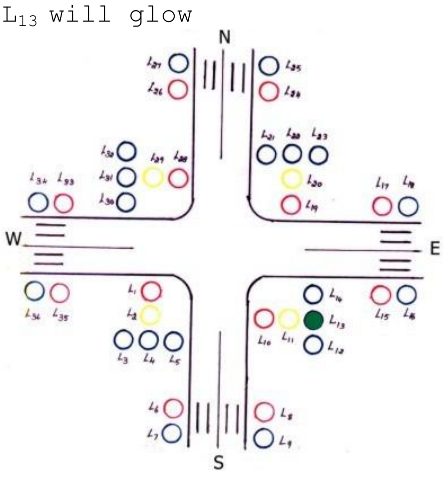
Circuit Diagram

Traffic Light Stimulator



PROGRAM TRACE

<table>
<tr><th>LABEL</th><th>MNEMONICS</th><th>DESCRIPTION</th></tr>
<tr><td></td><td>MVI A,80_H</td><td>Initializing the ports of the PPI 8255 as O/P ports by writing the control word as 80_H.
<table>
<tr><td>DATA BITS</td><td>D₇</td><td>D₆</td><td>D₅</td><td>D₄</td><td>D₃</td><td>D₂</td><td>D₁</td><td>D₀</td></tr>
<tr><td></td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>COMMENT</td><td>I/O mode</td><td colspan="2">Mode0</td><td>PortA O/P</td><td>PortC Upper O/P</td><td>Mode0</td><td>PortB O/P</td><td>PortC Lower O/P</td></tr>
</table>
80_H is moved to accumulator.
<b>REGISTERS</b>
<table>
<tr><td><b>A</b></td><td><b>80</b></td><td>XX</td><td>F</td></tr>
<tr><td>B</td><td>XX</td><td>XX</td><td>C</td></tr>
<tr><td>D</td><td>XX</td><td>XX</td><td>E</td></tr>
<tr><td>H</td><td>XX</td><td>XX</td><td>L</td></tr>
</table></td></tr>
<tr><td></td><td>OUT CWR</td><td>Control word specify the I/O function for each ports of 8255.</td></tr>
<tr><td>REPEAT</td><td>MVI E,03_H</td><td>Initialize E register with number of sequence.
<b>REGISTERS</b>
<table>
<tr><td>A</td><td>80</td><td>XX</td><td>F</td></tr>
<tr><td>B</td><td>XX</td><td>XX</td><td>C</td></tr>
<tr><td>D</td><td>XX</td><td><b>03</b></td><td><b>E</b></td></tr>
<tr><td>H</td><td>XX</td><td>XX</td><td>L</td></tr>
</table></td></tr>
</table>

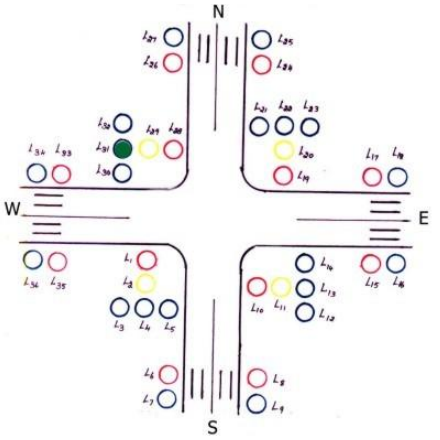
<table>
<tr>
<td></td>
<td>LXI H,C100$_H$</td>
<td>
Initialize the memory pointer at C100$_H$ .i.e. loads the 16-bit data in the register pair designated.
<b>REGISTERS</b>
<table>
<tr><td>A</td><td>80</td><td>XX</td><td>F</td></tr>
<tr><td>B</td><td>XX</td><td>XX</td><td>C</td></tr>
<tr><td>D</td><td>XX</td><td>03</td><td>E</td></tr>
<tr><td><b>H</b></td><td><b>C1</b></td><td><b>00</b></td><td><b>L</b></td></tr>
</table>
C100$_H$ is the memory pointer to the first data of the sequence.
<b>MEMORY</b>
<table>
<tr><td><b>C100</b></td><td><b>BF</b></td><td>HL memory pointer</td></tr>
<tr><td>C101</td><td>BF</td><td></td></tr>
<tr><td>C102</td><td>AF</td><td></td></tr>
<tr><td>C103</td><td>EE</td><td></td></tr>
<tr><td>C104</td><td>EE</td><td></td></tr>
</table>
</td>
</tr>
<tr>
<td>NEXTSTAT</td>
<td>MOV A,M</td>
<td>
Memory pointer content BF$_H$ is moved to accumulator.
<b>REGISTERS</b>
<table>
<tr><td><b>A</b></td><td><b>BF</b></td><td>XX</td><td>F</td></tr>
<tr><td>B</td><td>XX</td><td>XX</td><td>C</td></tr>
<tr><td>D</td><td>XX</td><td>03</td><td>E</td></tr>
<tr><td>H</td><td>C1</td><td>00</td><td>L</td></tr>
</table>
<table>
<tr><td>LED no</td><td>L$_{14}$</td><td>L$_{13}$</td><td>L$_{12}$</td><td>L$_{11}$</td><td>L$_5$</td><td>L$_4$</td><td>L$_3$</td><td>L$_2$</td></tr>
<tr><td><b>PORTA</b> bits BF$_H$</td><td>1</td><td><b>0</b></td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td></tr>
<tr><td>LED status</td><td>Will not glow</td><td><b>GLOW</b> Since Negative logic</td><td colspan="6">Will not glow</td></tr>
</table>
when the portA bit is ‘1’ then LED is in OFF state,
when the portA bit is ‘0’ then LED is in ON state.
</td>
</tr>
<tr>
<td></td>
<td>OUT PORTA</td>
<td>
L$_{13}$ will glow


</td>
</tr>
<tr>
<td></td>
<td>INX H</td>
<td>Increment the HL register pair by 1.The instruction views the contents of the HL registers as a 16-bit number. No flags are affected.</td>
</tr>
</table>

<table>
<tr><td></td><td></td><td>
<b>REGISTERS</b>
<table>
<tr><td>A</td><td>BF</td><td>XX</td><td>F</td></tr>
<tr><td>B</td><td>XX</td><td>XX</td><td>C</td></tr>
<tr><td>D</td><td>XX</td><td>03</td><td>E</td></tr>
<tr><td><b>H</b></td><td><b>C1</b></td><td><b>01</b></td><td><b>L</b></td></tr>
</table>
$C101_H$ is the memory pointer to input data sequence.

<b>MEMORY</b>
<table>
<tr><td>C100</td><td>BF</td><td></td></tr>
<tr><td><b>C101</b></td><td><b>BF</b></td><td>← HL memory pointer</td></tr>
<tr><td>C102</td><td>AF</td><td></td></tr>
<tr><td>C103</td><td>EE</td><td></td></tr>
<tr><td>C104</td><td>EE</td><td></td></tr>
</table>
</td></tr>
<tr><td></td><td>MOV A,M</td><td>
Memory pointer content BF_H is moved to accumulator.

<b>REGISTERS</b>
<table>
<tr><td><b>A</b></td><td><b>BF</b></td><td>XX</td><td>F</td></tr>
<tr><td>B</td><td>XX</td><td>XX</td><td>C</td></tr>
<tr><td>D</td><td>XX</td><td>03</td><td>E</td></tr>
<tr><td>H</td><td>C1</td><td>01</td><td>L</td></tr>
</table>
<table>
<tr><td>LED no</td><td>L_{32}</td><td>L_{31}</td><td>L_{30}</td><td>L_{29}</td><td>L_{23}</td><td>L_{22}</td><td>L_{21}</td><td>L_{20}</td></tr>
<tr><td><b>PORTB</b> bits BF_H</td><td>1</td><td><b>0</b></td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td></tr>
<tr><td>LED status</td><td>Will not glow</td><td><b>GLOW</b> Since Negative Logic</td><td colspan="6">Will not glow</td></tr>
</table>
when the portB bit is ‘1’ then LED is in OFF state,
when the portB bit is ‘0’ then LED is in ON state.
</td></tr>
<tr><td></td><td>OUT PORTB</td><td>
L_{31} will glow



</td></tr>
<tr><td></td><td>INX H</td><td>
Increment the HL register pair by 1.

<b>REGISTERS</b>
<table>
<tr><td>A</td><td>BF</td><td>XX</td><td>F</td></tr>
<tr><td>B</td><td>XX</td><td>XX</td><td>C</td></tr>
<tr><td>D</td><td>XX</td><td>03</td><td>E</td></tr>
<tr><td><b>H</b></td><td><b>C1</b></td><td><b>02</b></td><td><b>L</b></td></tr>
</table>
$C102_H$ is the memory pointer to input data sequence.
</td></tr>
</table>

<table>
<tr><td></td><td></td><td>

MEMORY

<table>
<tr><td>C100</td><td>BF</td><td></td></tr>
<tr><td>C101</td><td>BF</td><td></td></tr>
<tr><td><b>C102</b></td><td><b>AF</b></td><td>← HL memory pointer</td></tr>
<tr><td>C103</td><td>EE</td><td></td></tr>
<tr><td>C104</td><td>EE</td><td></td></tr>
</table>

</td></tr>
<tr><td></td><td>MOV A,M</td><td>

Memory pointer content AF_H is moved to accumulator.

REGISTERS

<table>
<tr><td><b>A</b></td><td><b>AF</b></td><td>XX</td><td>F</td></tr>
<tr><td>B</td><td>XX</td><td>XX</td><td>C</td></tr>
<tr><td>D</td><td>XX</td><td>03</td><td>E</td></tr>
<tr><td>H</td><td>C1</td><td>02</td><td>L</td></tr>
</table>

<table>
<tr><td>LED no</td><td>L_{33}
L_{35}</td><td>L_{34}
L_{36}</td><td>L_{24}
L_{26}</td><td>L_{25}
L_{27}</td><td>L_{15}
L_{17}</td><td>L_{16}
L_{18}</td><td>L_6
L_8</td><td>L_7
L_9</td><td>L_{28}</td><td>L_{10}</td><td>L_{19}</td><td>L_1</td></tr>
<tr><td>LED glow</td><td>1</td><td>0</td><td>1</td><td>0</td><td>1</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td></tr>
<tr><td><b>PORTC</b> bits AF_H</td><td colspan="2">1</td><td colspan="2">0</td><td colspan="2">1</td><td colspan="2">0</td><td>1</td><td>1</td><td>1</td><td>1</td></tr>
<tr><td>LED status</td><td colspan="2">L_{33},L_{35}
<b>GLOW</b></td><td colspan="2">L_{25},L_{27}
<b>GLOW</b></td><td colspan="2">L_{15},L_{17}
<b>GLOW</b></td><td colspan="2">L_7,L_9
<b>GLOW</b></td><td colspan="2">Will not glow</td><td colspan="2"><b>GLOW</b></td></tr>
</table>

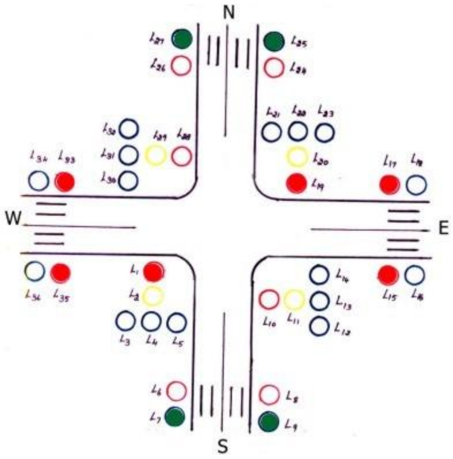
</td></tr>
<tr><td></td><td>OUT PORTC</td><td>

L_1 L_{19} => glow; since positive logic. Only when ‘1’ is present in this place the LED will glow.

L_{10} L_{28} => does not glow; since negative logic. This will not glow because only when ‘0’ is present in this, the LED will glow. Here ‘11’ is present hence it will not glow.

L_7,L_9 L_{25},L_{27} => will glow; since negative logic. Here we have zero.

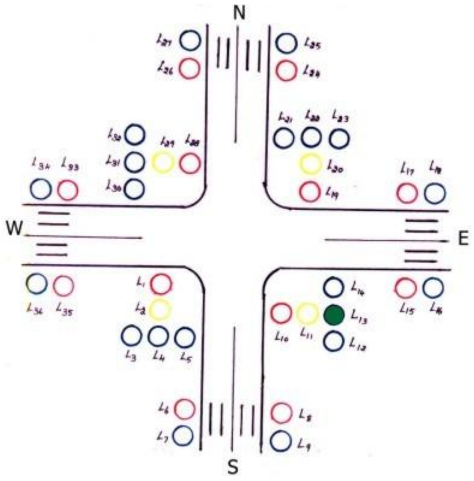
L_{33},L_{35} L_{15},L_{17} => will glow; since positive logic. Here we have one.

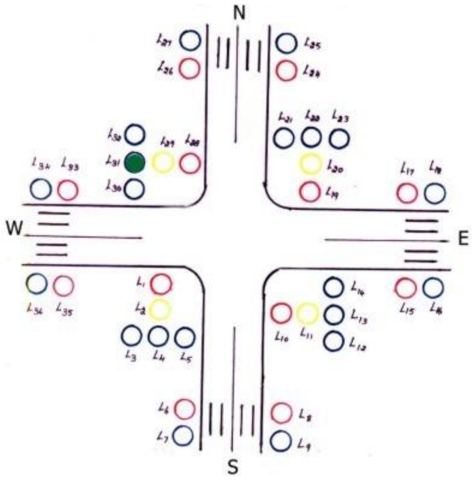


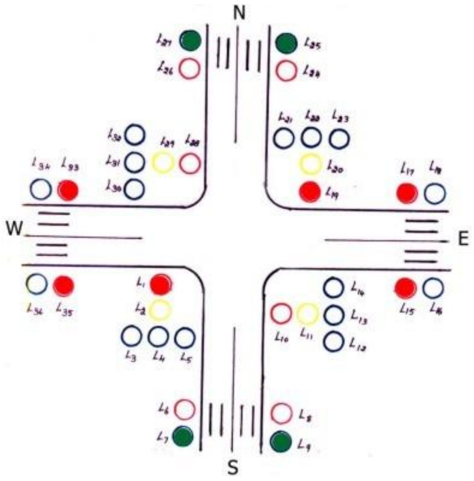
</td></tr>
<tr><td></td><td>CALL</td><td>In order to make these LEDs glowing visible to the</td></tr>
</table>

	DELAY	programmer/user, delay was provided.
	INX H	Increment the HL register pair by 1. **REGISTERS** A AF XX F B XX XX C D XX 03 E **H C1 03 L** $C103_H$ is the memory pointer to input data sequence. MEMORY C100 BF C101 BF C102 AF **C103 EE** ← HL memory pointer C104 EE
	DCR E	E register was decremented by 1 indicating remaining number of sequence. **REGISTERS** A AF XX F B XX XX C D XX **02 E** H C1 03 L
	JNZ NEXTSTAT	Now the next sequence is being looped.
	JMP REPEAT	Once again the three sequence are executed.

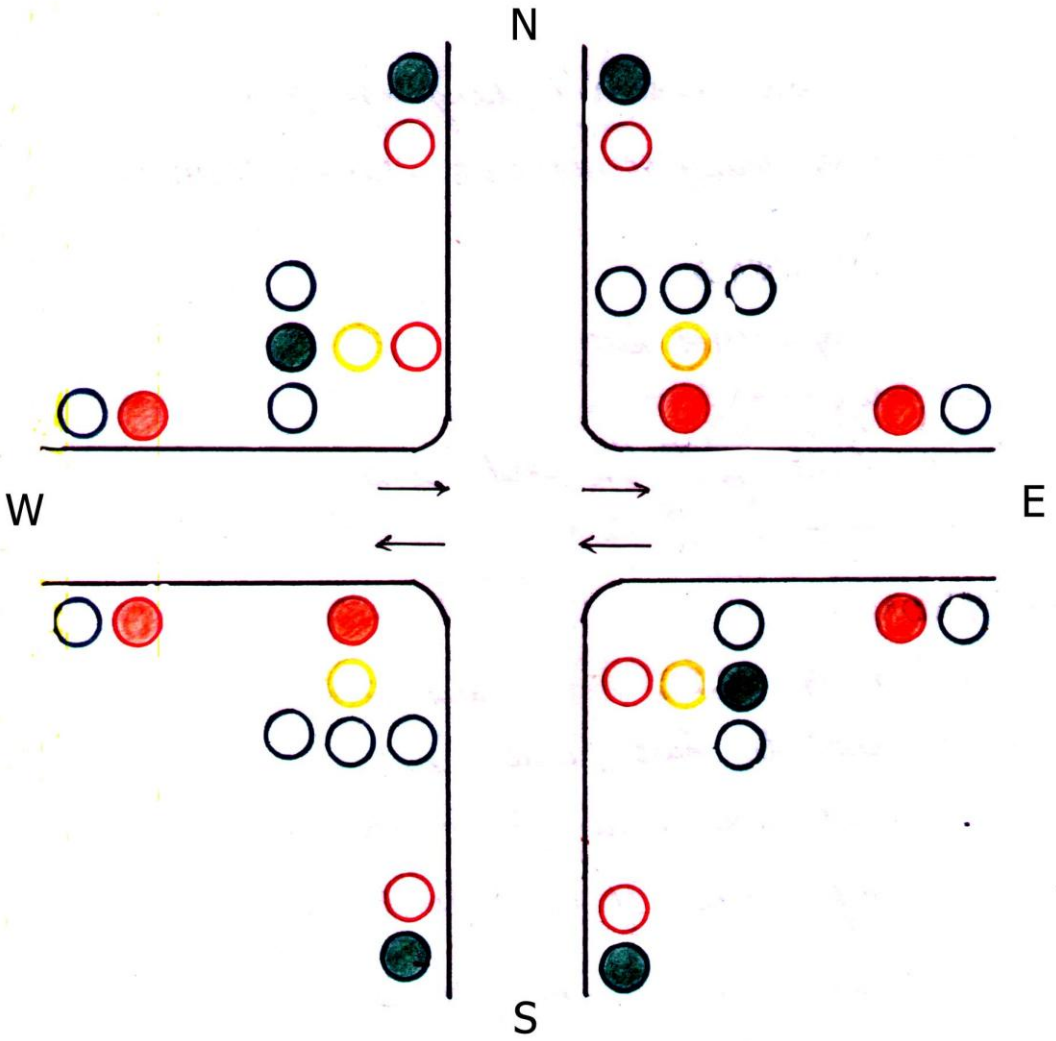
W → E
E → W


N
W
E
S
L27
L26
L25
L24
L32
L31
L30
L29
L28
L21
L22
L23
L20
L19
L34
L33
L17
L18
L1
L2
L3
L4
L5
L34
L35
L14
L13
L12
L10
L11
L15
L16
L6
L7
L8
L9


N
W
E
S
L27
L26
L25
L24
L32
L31
L30
L29
L28
L21
L22
L23
L20
L19
L34
L33
L17
L18
L1
L2
L3
L4
L5
L34
L35
L14
L13
L12
L10
L11
L15
L16
L6
L7
L8
L9


N
W
E
S
L27
L26
L25
L24
L32
L31
L30
L29
L28
L21
L22
L23
L20
L19
L34
L33
L17
L18
L1
L2
L3
L4
L5
L34
L35
L14
L13
L12
L10
L11
L15
L16
L6
L7
L8
L9

W --> E
E --> W



PORT A - BF_H

PORT B - BF_H

PORT C - AF_H

Now the next sequence is being traced.

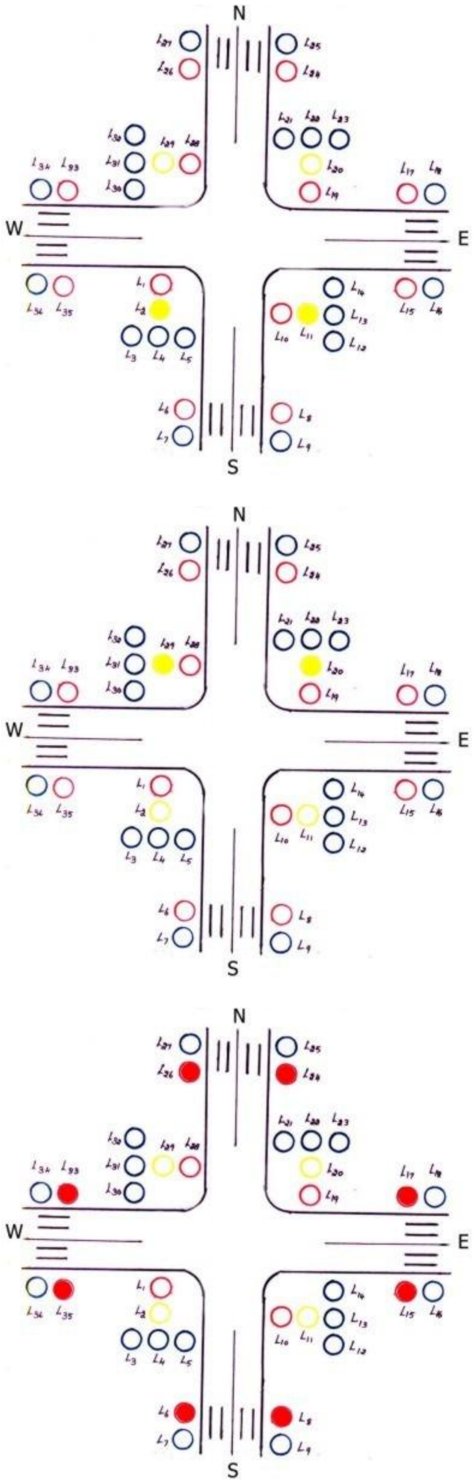
When E=02
PORTS CONFIGURATION & DISPLAY.

LED no	L_{14}	L_{13}	L_{12}	L_{11}	L_5	L_4	L_3	L_2
PORTA bits EE_H	1	1	1	0	1	1	1	0
LED status	Will not glow			**GLOW** Since Negative logic	Will not glow			**GLOW** Since Negative logic

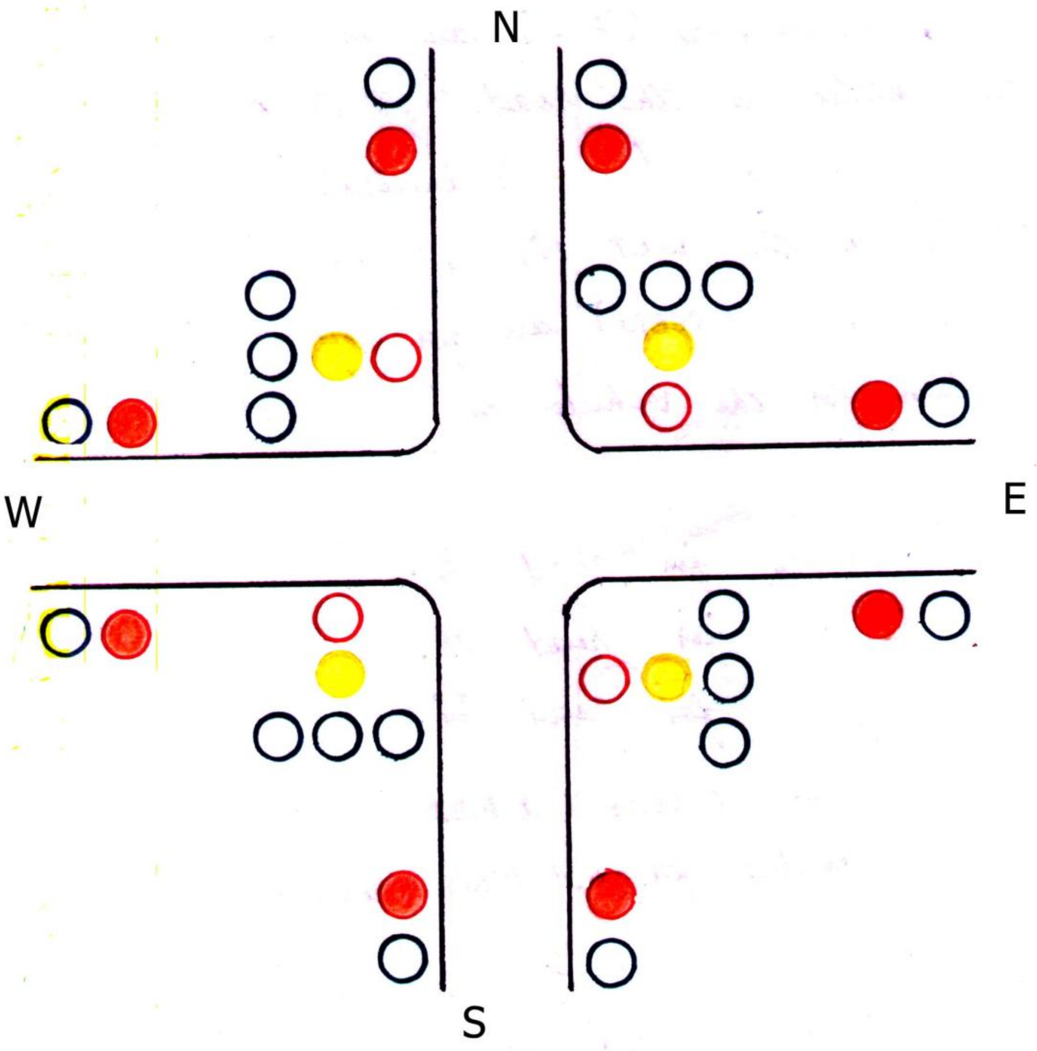
LED no	L_{32}	L_{31}	L_{30}	L_{29}	L_{23}	L_{22}	L_{21}	L_{20}
PORTB bits EE_H	1	1	1	0	1	1	1	0
LED status	Will not glow			**GLOW** Since Negative logic	Will not glow			**GLOW** Since Negative logic

LED no	L_{33} L_{35}	L_{34} L_{36}	L_{24} L_{26}	L_{25} L_{27}	L_{15} L_{17}	L_{16} L_{18}	L_6 L_8	L_7 L_9	L_{28}	L_{10}	L_{19}	L_1
LED glow	1	0	1	0	1	0	1	0	0	0	1	1
PORTC bits AC_H	1		1		1		1		1	1	0	0
LED status	L_{33},L_{35} **GLOW**		L_{24},L_{26} **GLOW**		L_{15},L_{17} **GLOW**		L_6 L_8 **GLOW**		Will not glow			

WAITING SEQUENCE



WAITING SEQUENCE



PORT A – EE_H

PORT B – EE_H

PORT C – FC_H

Now the next sequence is being traced.

When E=01
PORTS CONFIGURATION & DISPLAY.

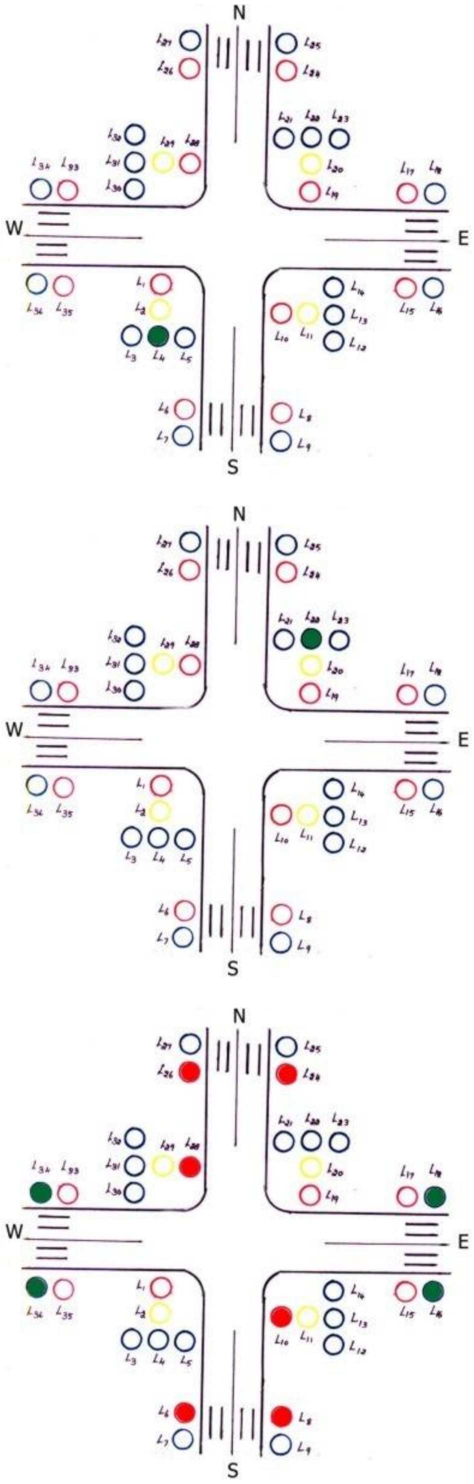
LED no	L_{14}	L_{13}	L_{12}	L_{11}	L_5	L_4	L_3	L_2
PORTA bits FB$_H$	1	1	1	1	1	0	1	1
LED status	Will not glow					**GLOW** Since Negative logic	Will not glow	

LED no	L_{32}	L_{31}	L_{30}	L_{29}	L_{23}	L_{22}	L_{21}	L_{20}
PORTB bits FB$_H$	1	1	1	1	1	0	1	1
LED status	Will not glow					**GLOW** Since Negative logic	Will not glow	

LED no	L_{33} L_{35}	L_{34} L_{36}	L_{24} L_{26}	L_{25} L_{27}	L_{15} L_{17}	L_{16} L_{18}	L_6 L_8	L_7 L_9	L_{28}	L_{10}	L_{19}	L_1
LED glow	1	0	1	0	1	0	1	0	0	0	1	1
PORTC bits 50$_H$	0		1		0		1		0	0	0	0
LED status	L_{34},L_{36} **GLOW**		L_{24},L_{26} **GLOW**		L_{16},L_{18} **GLOW**		L_6 L_8 **GLOW**		**GLOW**		Will not glow	

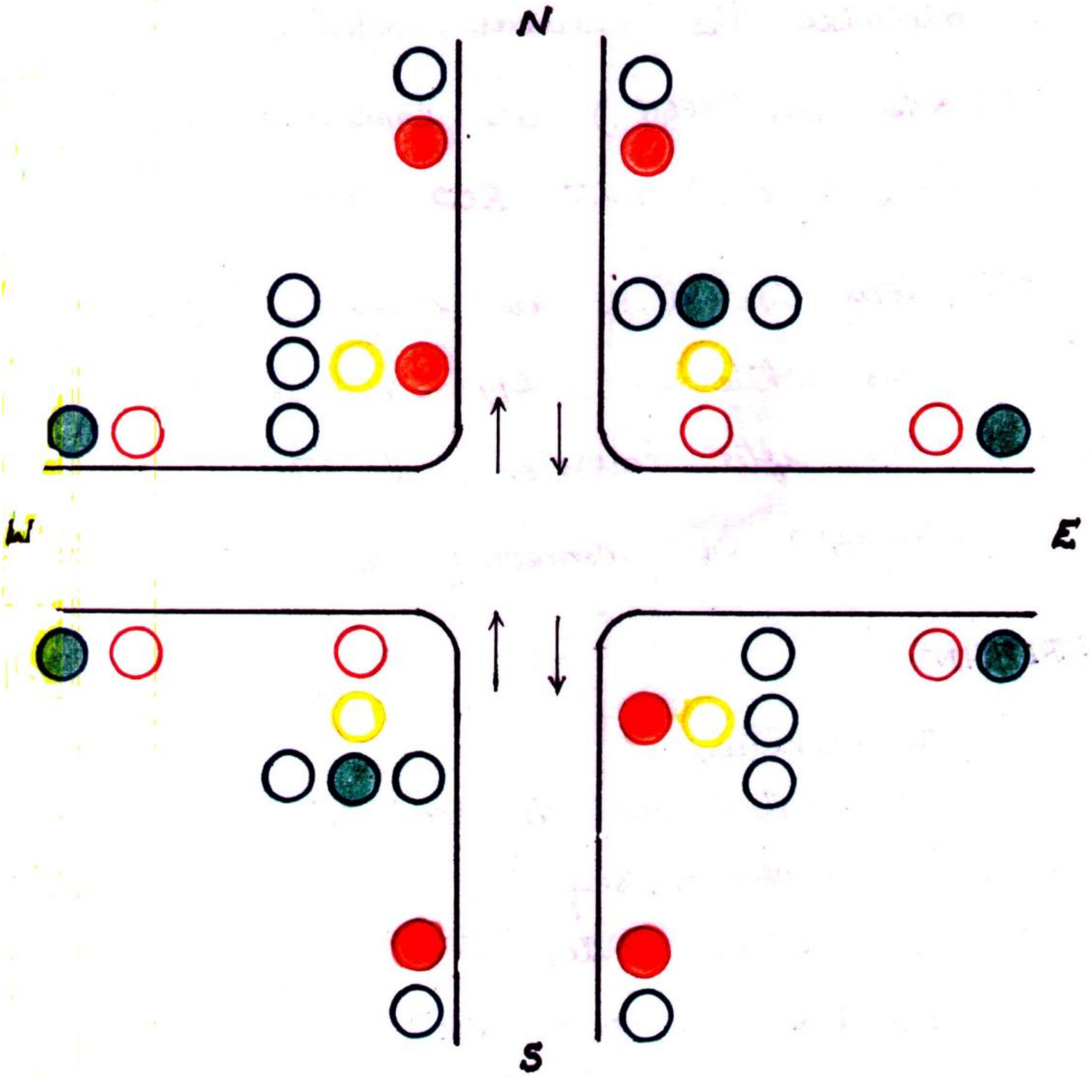
Thus the LEDs glow, when E=00 the sequence is terminated and next cycle starts.

N→S
S →N


N
S
E
W
L1
L2
L3
L4
L5
L6
L7
L8
L9
L10
L11
L12
L13
L14
L15
L16
L17
L18
L19
L20
L21
L22
L23
L24
L25
L26
L27
L28
L29
L30
L31
L32
L33
L34
L35

$N \rightarrow S$

$S \rightarrow N$



PORT A - FB_H

PORT B - FB_H

PORT C - 50_H

DELAY SUBPROGRAM

DELAY	LXI D,3000$_H$	Initialize the memory pointer at C100$_H$ .i.e. loads the 16-bit data in the register pair designated. **REGISTERS** A XX XX F B XX XX C **D 30 00 E** H XX XX L C100$_H$ is the memory pointer to the first data of the sequence. **MEMORY** **3000 XX** ← HL memory pointer 3001 XX 3002 XX 3003 XX 3004 XX
L2	MVI C,FF$_H$	Move FF$_H$ immediately in to C register. **REGISTERS** A XX XX F B XX **FF C** D 30 00 E H XX XX L
L1	DCR C	Move FF$_H$ immediately in to C register. **REGISTERS** A XX XX F B XX **FE C** D 30 00 E H XX XX L
	JNZ L1	Loop until C = 00
	DCX D	Decrement the DE register pair by 1. **REGISTERS** A XX XX F B XX XX C **D 2F FF E** H XX XX L
	MOV A,D	D register content 2F$_H$ is moved to accumulator. **REGISTERS** **A 2F** XX F B XX XX C D 2F FF E H XX XX L
	ORA E	OR the accumulator content with E register content FF => 1111 1111 2F => 0010 1111 ----------- 1111 1111 => FF

		REGISTERS **A** **FF** XX F B XX XX C D 2F FF E H XX XX L
	JNZ L2	Only when DE=0000,this loop will end.
	RET	Return to main program

EXECUTION

ADDRESS	DATA
C100	BF $_H$
C101	BF $_H$
C102	AF $_H$
C103	EE $_H$
C104	EE $_H$
C105	FC $_H$
C106	FB $_H$
C107	FB $_H$
C108	50 $_H$

VERIFICATION

The LEDs on the interface glow according to the given sequence.

REFERENCE

1. Ramesh S.Gaonkar, Microprocessor Architecture, Programming, and Applications, Fourth Edition, Penram International Publishing (India), 2000.

2. S.Subathra, “Advanced Microprocessor Laboratory”, Record work, Adhiparashakthi Engineering College, Melmaruvathur, October 2002

3. S.Subathra, “Programming in 8085 Microprocessor and its applications – An Innovative Analysis”, Technical Report, Adhiparashakthi Engineering College, Melmaruvathur, March 2003

4. Micro-85 EB, Technical Reference, Version – 2.0, CAT #M85 EB-001 VI Microsystems Pvt. Ltd., Chennai.