AUTOMATED TRAFFIC SIGNAL CONTROLLER

VIKRAM BANERJEE
MRINAL KANTI MANDAL
DR ANIRUDHA GHOSAL

This automated traffic signal controller can be made by suitably programming a GAL device. (For GAL programming you may refer to the construction project published on page 52 in EFY's September issue.) Its main features are:

1. The controller assumes equal traffic density on all the roads.

2. In most automated traffic signals the free left-turn condition is provided throughout the entire signal period, which poses difficulties to the pedestrians in crossing the road, especially when the traffic density is high. This controller allows the pedestrians to safely cross the road during certain periods.

3. The controller uses digital logic, which can be easily implemented by using logic gates.

4. The controller is a generalised one and can be used for different roads with slight modification.

5. The control can also be exercised manually when desired.

The time period for which green, yellow, and red traffic signals remain 'on' (and then repeat) for the straight moving traffic is divided into eight units of 8 seconds (or multiples thereof) each. Fig. 1 shows the flow of traffic in all permissible directions during the eight time units of 8 seconds each. For the left- and right-turning traffic and pedestrians crossing from north to south, south to north, east to west, and west to east, only green and red signals are used.

Table I shows the simultaneous states of the signals for all the traffic. Each row represents the status of a signal for 8 seconds. As can be observed from the table, the ratio of green, yellow, and red signals is 16:8:40 (= 2:1:5) for the straight moving traffic. For the turning traffic the ratio of green and red signals is 8:56 (= 1:7), while for pedestrians crossing the road the ratio of green and red signals is 16:48 (= 2:6).

In Table II (as well as Table I) X, Y, and Z are used as binary variables to

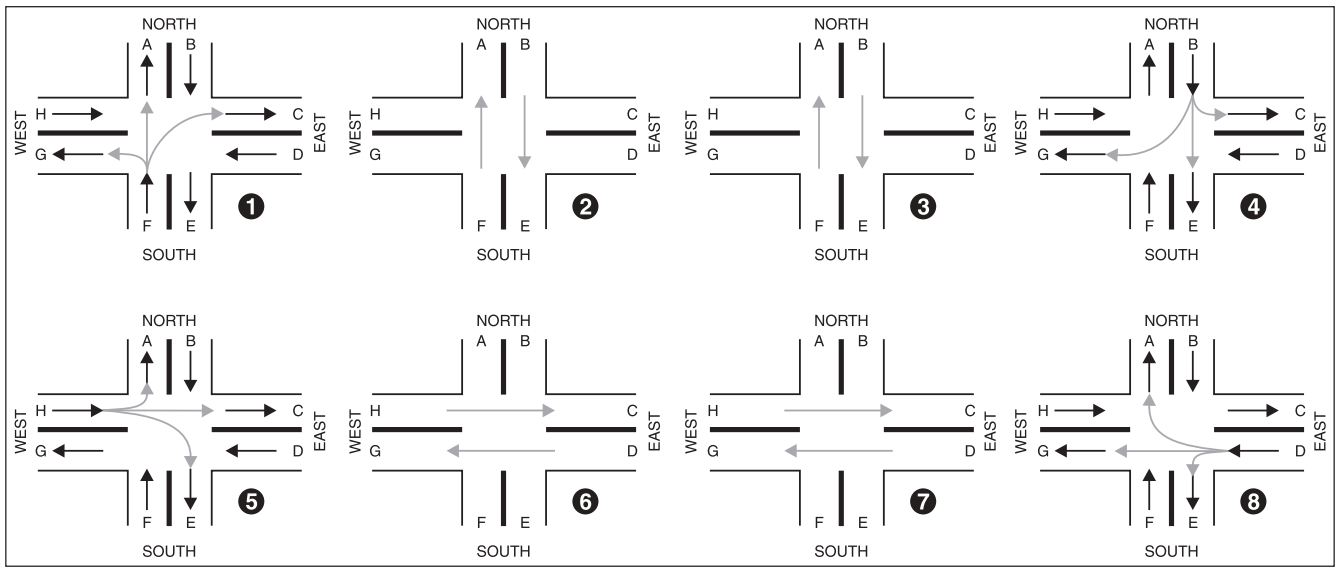


Fig. 1: Flow of traffic in all possible directions

TABLE I
Simultaneous States of Signals for All the Traffic

X	Y	Z	B-C/B-G Lt/Rt	B-E St	D-E/D-A Lt/Rt	D-G St	F-G/F-C Lt/Rt	F-A St	H-A/H-E Lt/Rt	HC St	WALK (N-S)/(S-N)	WALK (E-W)/(W-E)
0	0	0	R	R	R	R	G	G	R	R	R	R
0	0	1	R	G	R	R	R	G	R	R	G	R
0	1	0	R	G	R	R	R	Y	R	R	G	R
0	1	1	G	Y	R	R	R	R	R	R	R	R
1	0	0	R	R	R	R	R	R	G	G	R	R
1	0	1	R	R	R	G	R	R	R	G	R	G
1	1	0	R	R	R	G	R	R	R	Y	R	G
1	1	1	R	R	G	Y	R	R	R	R	R	R

TABLE II
Boolean Functions for All the Signal Conditions

Signal	Reference	Boolean functions
Green	B-C(Lt)/B-G (Rt)	X'YZ
Green	B-E (St)	XYZ' + X'Y'Z
Red	B-E (St)	X + Y'Y'Z'
Yellow	B-E (St)	X'YZ
Green	D-E (Lt)/D-A (Rt)	XYZ
Green	D-G (St)	XYZ' + XY'Z
Red	D-G (St)	X' + XY'Z'
Yellow	D-G (St)	XYZ
Green	F-G(Lt)/F-C (Rt)	X'Y'Z'
Green	F-A (St)	X'Y'
Red	F-A (St)	X + X'YZ
Yellow	F-A (St)	X'YZ'
Green	H-A (Lt)/H-E (Rt)	XY'Z'
Green	H-C (St)	XY'
Red	H-C (St)	X' + XYZ
Yellow	H-C (St)	XYZ'
Green	Walk (N-S/S-N)	X'YZ' + X'Y'Z
Green	Walk (E-W/W-E)	XYZ' + XY'Z

***Note**. X', Y', and Z' denote complements of variables X, Y, and Z, respectively.*

depict the eight states of 8 seconds each. Letters A through H indicate the left and right halves of the roads in four directions as shown in Fig. 1. Two letters with a dash in between indicate the direction of permissible movement from a road. Straight direction is indicated by St, while left and right turns are indicated by Lt and Rt, respectively.

The Boolean functions for all the signal conditions are shown in Table II. The left- and the right-turn signals for the traffic have the same state, i.e. both are red or green for the same duration, so their Boolean functions are identical and they should be connected to the same control output.

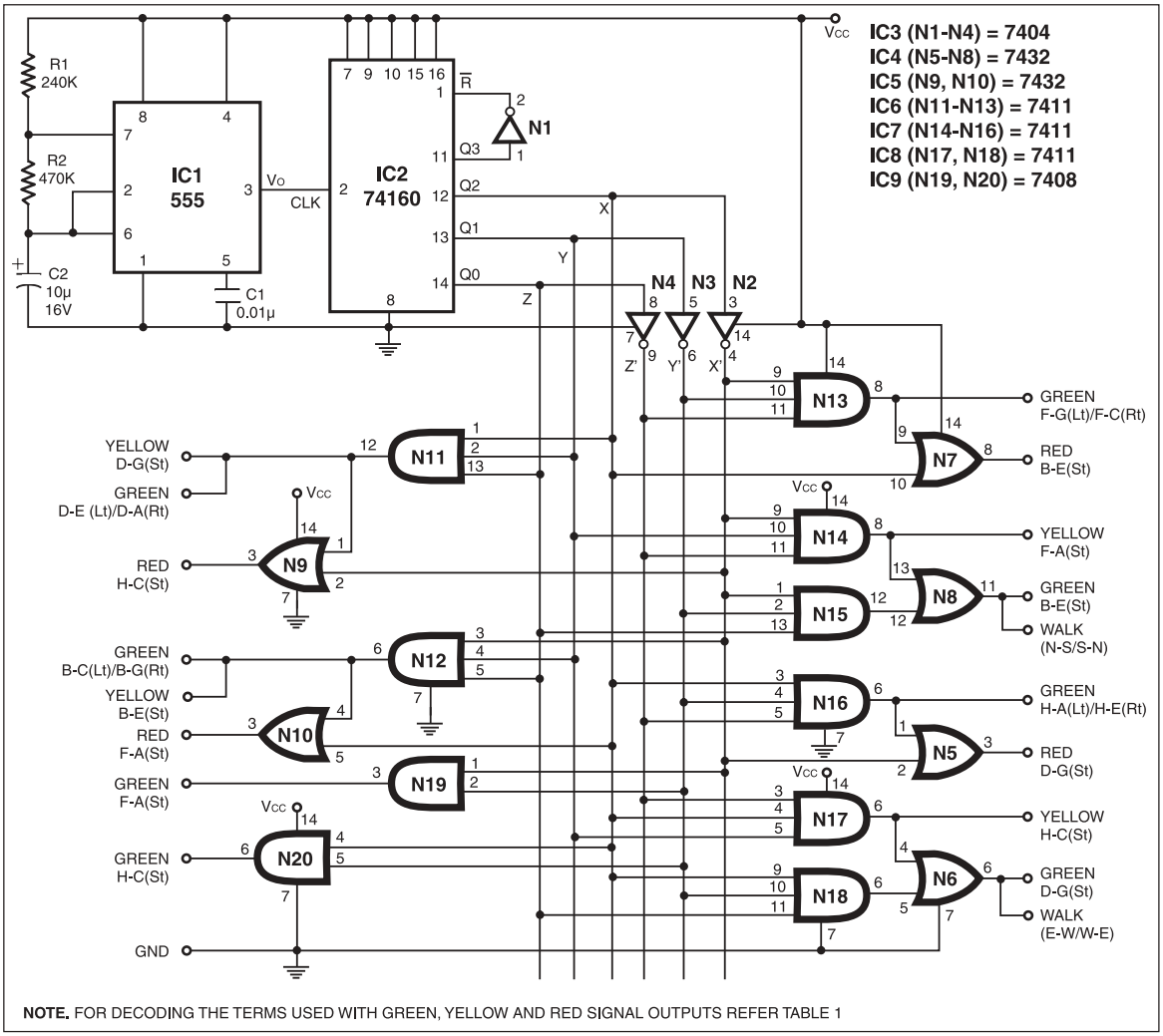
The circuit diagram for realising these Boolean functions is shown in Fig. 2. Timer 555 (IC1) is wired as an astable multivibrator to generate clock signal for the 4-bit counter 74160 (IC2). The time duration of IC1 can be adjusted by varying the value of resistor R1, resistor R2, or capacitor C2 of the clock circuit. The 'on' time duration T is given by the following relationship:

T = 0.695C2(R1 + R2)

IC2 is wired as a 3-bit binary counter by connecting its Q3 output to reset pin 1 via inverter N1. Binary outputs Q2, Q1, and Q0 form variables X, Y, and Z, respectively. These outputs, along with their complimentary outputs X', Y', and Z', respectively, are used as inputs to the rest of the logic circuit to realise various outputs satisfying Table I.

You can simulate various traffic lights using green, yellow, and red LEDs and feed the outputs of the circuit to respective LEDs via current-limiting resistors of 470 ohms each to check the working of the circuit. Here, for turning traffic and pedestrians crossing the road, only green signal is made available. It means that for the remaining period these signals have to be treated as 'red'.

In practice, the outputs of Fig. 2 should be connected to solidstate relays to operate high-power bulbs. Further, if a particular signal condition (such as turning signal) is not applicable to a given road, the output of that signal condition should be



NOTE. FOR DECODING THE TERMS USED WITH GREEN, YELLOW AND RED SIGNAL OUTPUTS REFER TABLE 1

Fig. 2: The circuit diagram for traffic light signalling

Table III
Execution Results of Software Program

SIG-B	SIG-D	SIF-F	SIG-H	WALK(N-S)	WALK(E-W)
G G R Y	G G R Y	G G R Y	G G R Y	G R	G R
0 0 1 0	0 1 0 0	1 1 0 0	0 0 1 0	0 1	0 1
0 1 0 0	0 1 0 0	0 1 0 0	0 0 1 0	1 0	0 1
0 1 0 0	0 1 0 0	0 0 0 1	0 0 1 0	1 0	0 1
1 0 0 1	0 1 0 0	0 0 1 0	0 0 1 0	0 1	0 1
0 0 1 0	0 1 0 0	0 0 1 0	1 1 0 0	0 1	0 1
0 0 1 0	1 0 0 0	0 0 1 0	0 1 0 0	0 1	1 0
0 0 1 0	1 0 0 0	0 0 1 0	0 0 0 1	0 1	1 0
0 0 1 0	0 0 1 1	0 0 1 0	0 0 1 0	0 1	0 1

Note. *The first column under G (green) in each group of four signals indicates the turn signal, while the next three columns under GRY indicate signal for the straight traffic.*

connected to green signal of the next state (refer Table I).

The traffic signals can also be controlled manually, if desired. Any signal state can be established by entering the binary value corresponding to that particular state into the parallel input pins of the 3-bit counter. Similarly, the signal can be reset at any time by providing logic 0 at the reset pin (pin 1) of the counter using an external switch.

A software program to verify the functioning of the circuit using a PC is given below. (Source code and executable file will be provided in the next month's EFY-CD.) When executing the program, keep pressing Enter key to get the next row of results. The test results on execution of the program is shown in Table III.

This circuit costs around Rs 125.

TRAFFIC.C

```
#include < stdio.h >
#include < conio.h >
#define TRUE 1
#define False 0

int not(int x);
int or2(int x,int y);
int or3(int x,int y,int z);
int and2(int x,int y);
int and3(int x,int y,int z);
int main(void)
{
int a,b,c;
int seq,green_bl,green_bs,red_bs,yellow_bs;
int green_dl,green_ds,red_ds,yellow_ds;
int green_fl,green_fs,red_fs,yellow_fs;
int green_hl,green_hs,red_hs,yellow_hs;
int walk_ns,stop_ns;
int walk_ew,stop_ew;

clrscr();
printf(" SIG-B SIG-D SIF-F SIG-H
WALK(N-S) WALK(E-W)\n");
printf("G G R Y G G R Y G G R Y G G R Y
G R G R\n");

for(seq = 0;seq < 8;seq + + )
{
c = (seq&1);b = (seq&2) > > 1;a = (seq&4) > > 2;
green_bl = and3(not(a),b,c);
green_bs = or2(and3(not(a),b,not(c)),and3(not(a),not(b),c));
red_bs = or2(a,and3(not(a),not(b),not(c)));
yellow_bs = and3(not(a),b,c);
green_dl = and3(a,b,c);
green_ds = or2(and3(a,b,not(c)),and3(a,not(b),c));
red_ds = or2(not(a),and3(a,not(b),not(c)));
yellow_ds = and3(a,b,c);
green_fl = and3(not(a),not(b),not(c));
green_fs = and2(not(a),not(b));
red_fs = or2(a,and3(not(a),b,c));
yellow_fs = and3(not(a),b,not(c));
green_hl = and3(a,not(b),not(c));
green_hs = and2(a,not(b));
red_hs = or2(not(a),and3(a,b,c));
yellow_hs = and3(a,b,not(c));
walk_ns = green_bs;
stop_ns = or3(and3(not(a),not(b),not(c)),and3(not(a),b,c),a);
walk_ew = green_ds;
stop_ew = or3(not(a),and3(a,b,c),and3(a,not(b),not(c)));
printf("%d %d %d %d %d %d %d %d
%d %d %d %d %d %d %d %d %d %d
%d %d\n",
    green_bl,green_bs,red_bs,yellow_bs,
    green_dl,green_ds,red_ds,yellow_ds,
    green_fl,green_fs,red_fs,yellow_fs,
    green_hl,green_hs,red_hs,yellow_hs,
    walk_ns,stop_ns,
    walk_ew,stop_ew);
    getch();
    }
    return;
    }
    int and2(int x,int y)
    {
    return(x && y);
    }
    int and3(int x,int y,int z)
    {
    return(x && y && z);
    }
    int or2(int x,int y)
    {
    return(x || y);
    }
    int or3(int x,int y,int z)
    {
    return(x || y || z);
    }
    int not(int x)
    {
    return(!x);
    }
```