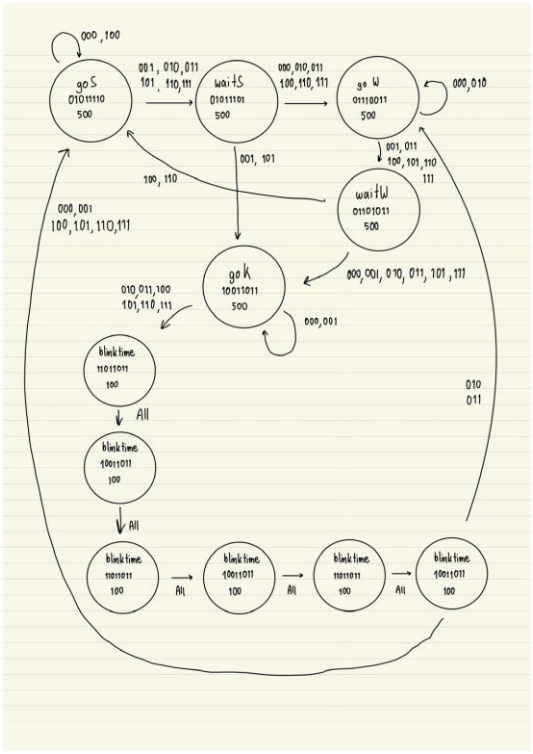
State transition graph

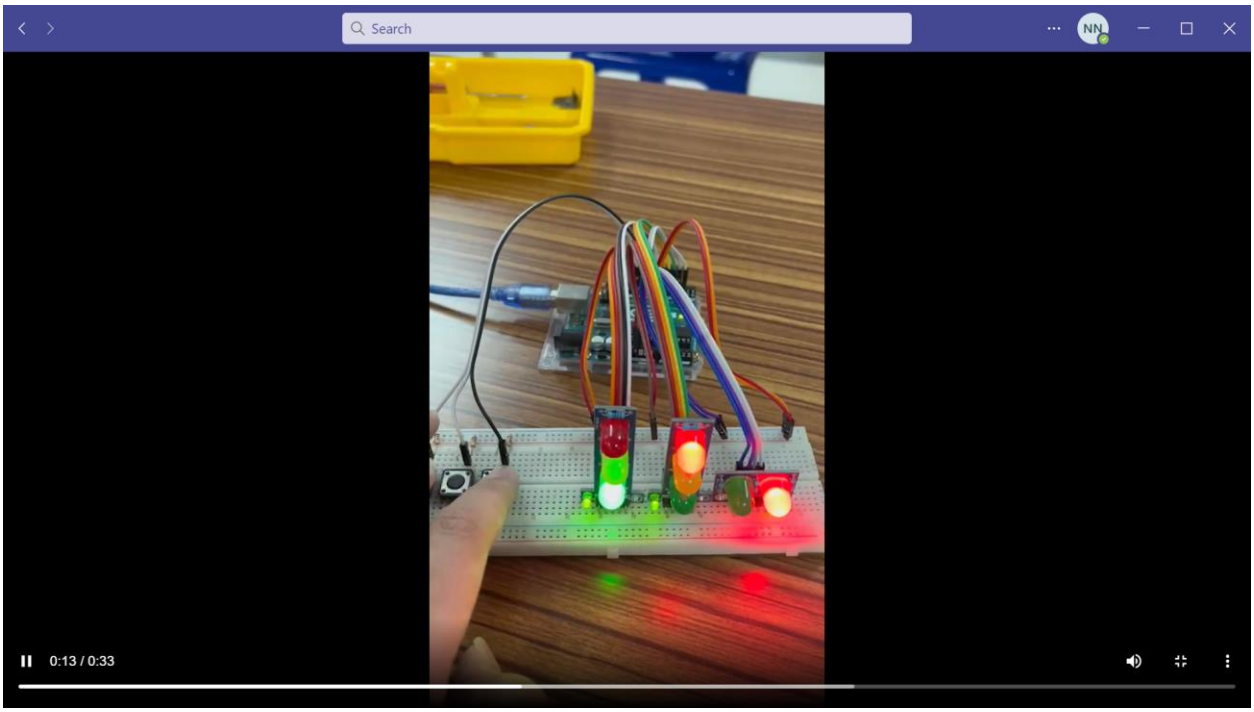


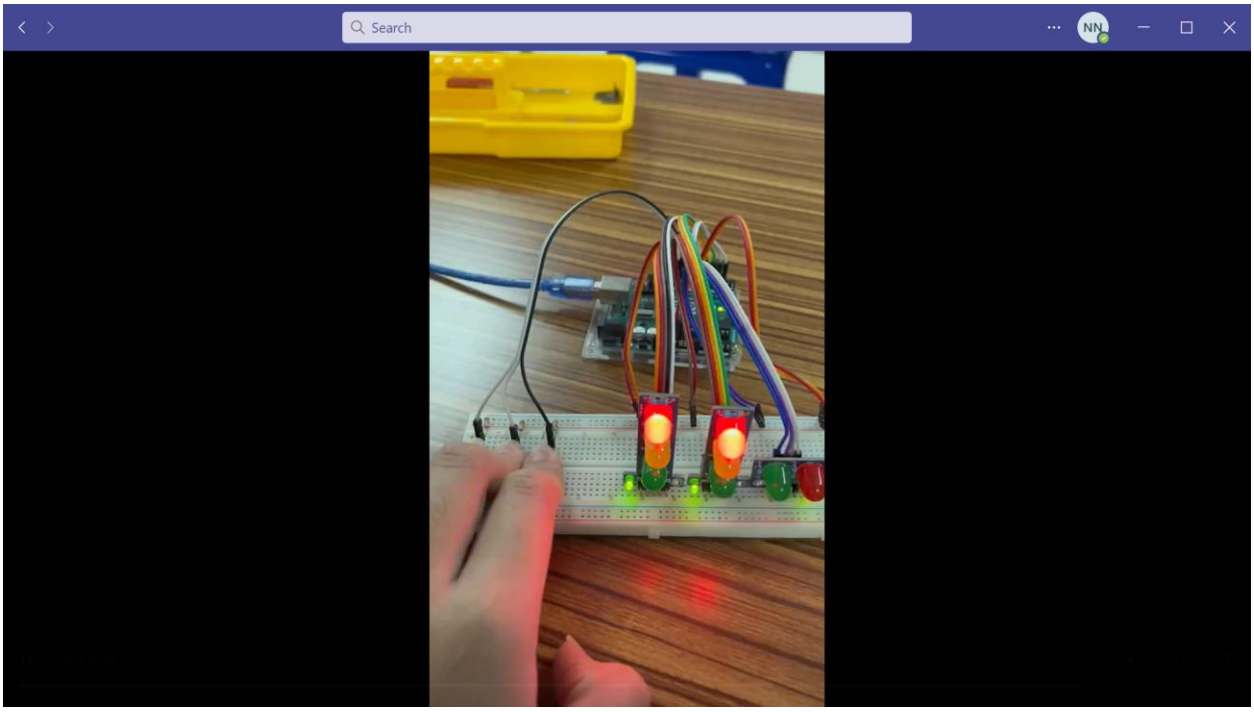
State transition table

State Transition Table

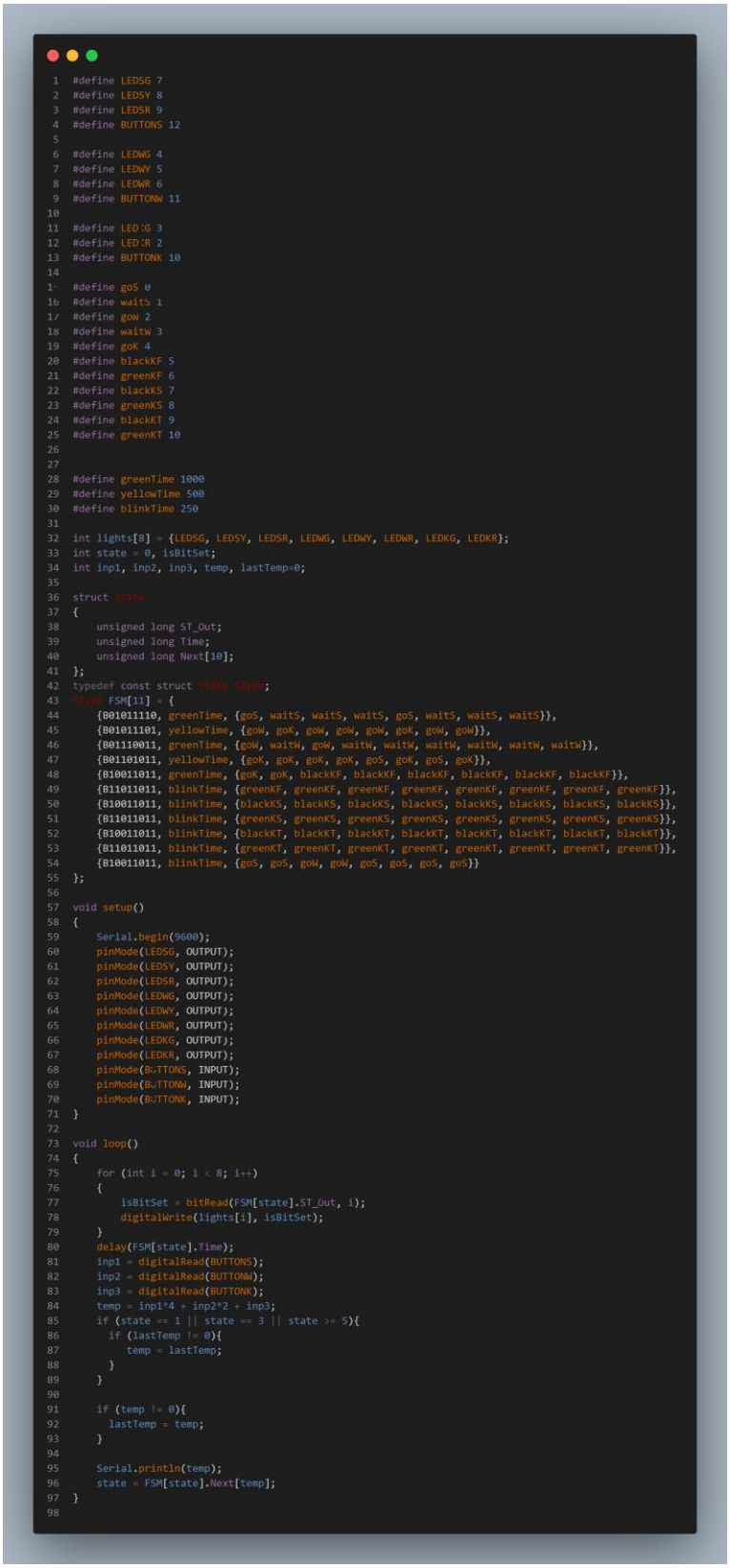
Num	Name	Lights	Time	In=0	In=1	In=2	In=3	In=4	In=5	In=6	In=7
0	goS	01011110	500	goS	waitS	waitS	waitS	goS	waitS	waitS	waitS
1	waitS	01011101	500	goW	goK	goW	goW	goW	goK	goW	goW
2	goW	01110011	500	goW	waitW	goW	waitW	waitW	waitW	waitW	waitW
3	waitW	01101011	500	goK	goK	goK	goK	goS	goK	goS	goK
4	goK	10011011	500	goK	goK	blackKF	blackKF	blackKF	blackKF	blackKF	blackKF
5	blinktime	11011011	100	greenKF	greenKF	greenKF	greenKF	greenKF	greenKF	greenKF	greenKF
6	blinktime	10011011	100	blackKS	blackKS	blackKS	blackKS	blackKS	blackKS	blackKS	blackKS
7	blinktime	11011011	100	greenKS	greenKS	greenKS	greenKS	greenKS	greenKS	greenKS	greenKS
8	blinktime	10011011	100	blackKT	blackKT	blackKT	blackKT	blackKT	blackKT	blackKT	blackKT
9	blinktime	11011011	100	greenKT	greenKT	greenKT	greenKT	greenKT	greenKT	greenKT	greenKT
10	blinktime	10011011	100	goS	goS	goW	goW	goS	goS	goS	goS

รูป





Source code



```
#define LEDSG 7
#define LEDSY 8
#define LEDSR 9
#define BUTTONS 12

#define LEDWG 4
#define LEDWY 5
#define LEDWR 6
#define BUTTONW 11

#define LEDKG 3
#define LEDKR 2
#define BUTTONK 10

#define goS 0
#define waitS 1
#define goW 2
#define waitW 3
#define goK 4
#define blackKF 5
#define greenKF 6
#define blackKS 7
#define greenKS 8
#define blackKT 9
#define greenKT 10


#define greenTime 1000
#define yellowTime 500
#define blinkTime 250

int lights[8] = {LEDSG, LEDSY, LEDSR, LEDWG, LEDWY, LEDWR, LEDKG, LEDKR};
int state = 0, isBitSet;
int inp1, inp2, inp3, temp, lastTemp=0;

struct State
{
    unsigned long ST_Out;
    unsigned long Time;
    unsigned long Next[10];
};
typedef const struct State State;
State FSM[11] = {
    {B01011110, greenTime, {goS, waitS, waitS, waitS, goS, waitS, waitS, waitS}},
    {B01011101, yellowTime, {goW, goK, goW, goW, goW, goK, goW, goW}},
    {B01110011, greenTime, {goW, waitW, goW, waitW, waitW, waitW, waitW, waitW, waitW}},
    {B01101011, yellowTime, {goK, goK, goK, goK, goS, goK, goS, goK}},
    {B10011011, greenTime, {goK, goK, blackKF, blackKF, blackKF, blackKF, blackKF, blackKF}},
    {B11011011, blinkTime, {greenKF, greenKF, greenKF, greenKF, greenKF, greenKF, greenKF, greenKF}},
    {B10011011, blinkTime, {blackKS, blackKS, blackKS, blackKS, blackKS, blackKS, blackKS, blackKS}},
    {B11011011, blinkTime, {greenKS, greenKS, greenKS, greenKS, greenKS, greenKS, greenKS, greenKS}},
    {B10011011, blinkTime, {blackKT, blackKT, blackKT, blackKT, blackKT, blackKT, blackKT, blackKT}},
    {B11011011, blinkTime, {greenKT, greenKT, greenKT, greenKT, greenKT, greenKT, greenKT, greenKT}},
    {B10011011, blinkTime, {goS, goS, goW, goW, goS, goS, goS, goS}}
};

void setup()
{
    Serial.begin(9600);
    pinMode(LEDSG, OUTPUT);
    pinMode(LEDSY, OUTPUT);
    pinMode(LEDSR, OUTPUT);
    pinMode(LEDWG, OUTPUT);
    pinMode(LEDWY, OUTPUT);
    pinMode(LEDWR, OUTPUT);
    pinMode(LEDKG, OUTPUT);
    pinMode(LEDKR, OUTPUT);
    pinMode(BUTTONS, INPUT);
    pinMode(BUTTONW, INPUT);
    pinMode(BUTTONK, INPUT);
}

void loop()
{
    for (int i = 0; i < 8; i++)
    {
        isBitSet = bitRead(FSM[state].ST_Out, i);
        digitalWrite(lights[i], isBitSet);
    }
    delay(FSM[state].Time);
    inp1 = digitalRead(BUTTONS);
    inp2 = digitalRead(BUTTONW);
    inp3 = digitalRead(BUTTONK);
    temp = inp1*4 + inp2*2 + inp3;
    if (state == 1 || state == 3 || state >= 5){
      if (lastTemp != 0){
        temp = lastTemp;
      }
    }

    if (temp != 0){
      lastTemp = temp;
    }

    Serial.println(temp);
    state = FSM[state].Next[temp];
}

```

คำอธิบาย

เราจะปุ่ม3ปุ่มเป็นเลขฐานสองแล้วแปลงเป็นฐาน10จะได้ค่าinputออกมา8ค่าและให้ทำงานตา State transition graph มีการป้องกันการปล่อยมือด้วยการเก็บตัวแปรไว้ในกรณีที่ปล่อยมือขณะไฟเหลือง