# Traffic Lights with Pedestrian Crossing

Akruti Somkuwar -21BLC1024
Sri Priya -21BEC1169
Mogal Shawar Tasneem-21BEC1706

# TABLE OF CONTENTS

# OBJECTIVE

Our objective of this project is to simulate the traffic light system with pedestrian crossing.
In day mode, the sequence of traffic lights and pedestrian crossing signals, including green, yellow, and red lights for both traffic and pedestrians. In night mode, it is a constant blinking yellow light for traffic.

# CODE

# C CODE

```
#include <stdio.h>
#include <ctype.h>
//#include <time.h>
#include <unistd.h>

#define YELLOW_DURATION 2  // Duration of yellow light (seconds)
#define GREEN_FOR_TRAFFIC 15  // Duration of green light for traffic (seconds)
#define GREEN_FOR_PEDESTRIANS 10  // Duration of green light for pedestrians (seconds)
#define RED_DURATION 1  // Duration of red light (seconds)

int Traffic_light() {
  char ch;
  int isNight = 0;

  while (1) {
    printf("Enter 'n' for night mode, 'd' for day mode (or any other key to quit): ");
```

# C CODE

```
while ((getchar()) != '\n');

  ch = tolower(ch);
  if (ch == 'n') {
    isNight = 1;
    printf("Pedestrian Light: OFF\n");

    while (1) {
      printf("Traffic Light: YELLOW\n");
      sleep(YELLOW_DURATION);
    }

    isNight = 0; // Reset night mode flag
  }

else if (ch == 'd') {
```

# C CODE

```
printf("Entering day mode...\n");
    while (1) {
      // Traffic light sequence
      printf("Pedestrian Waiting: Red\n");
      printf("Traffic Light: Green\n");
      sleep(GREEN_FOR_TRAFFIC);

      printf("Traffic Light: Yellow\n");
      sleep(YELLOW_DURATION);

      printf("Traffic Light: Red\n");
      sleep(RED_DURATION);

      // Pedestrian crossing sequence
      printf("Pedestrian Crossing: Green\n");
      printf("Traffic Light: Red\n");
      sleep(GREEN_FOR_PEDESTRIANS);
```

# C Code

```
 printf("Traffic Light: Yellow\n");
        sleep(YELLOW_DURATION);
        printf("Traffic Light: Green\n");
        sleep(RED_DURATION); // Short green for traffic after pedestrian crossing
      }
    } else {
      printf("Exiting...\n");
      break;
    }
  }
  return 0;
}
int main() {
  Traffic_light();
  return 0;
}
```

# C code OUTPUT

main.c

Save Run

```c
#include <stdio.h>
#include <ctype.h>
//#include <time.h>
#include <unistd.h>

#define YELLOW_DURATION 2  // Duration of yellow light (seconds)
#define GREEN_FOR_TRAFFIC 15  // Duration of green light for traffic (seconds)
#define GREEN_FOR_PEDESTRIANS 10  // Duration of green light for pedestrians (seconds)
#define RED_DURATION 1  // Duration of red light (seconds)

int Traffic_light() {
  char ch;
  int isNight = 0;

  while (1) {
    printf("Enter 'n' for night mode, 'd' for day mode (or any other key to quit): ");
    ch = getchar();

    // Clear the newline character
    while ((getchar()) != '\n');

    ch = tolower(ch);

    if (ch == 'n') {
```

Output

```
Traffic Light: Green
Traffic Light: Yellow
Traffic Light: Red
Pedestrian Crossing: Green
Traffic Light: Red
Traffic Light: Yellow
Traffic Light: Green
Pedestrian Waiting: Red
Traffic Light: Green
Traffic Light: Yellow
Traffic Light: Red
Pedestrian Crossing: Green
Traffic Light: Red
Traffic Light: Yellow
Traffic Light: Green
Pedestrian Waiting: Red
Traffic Light: Green
Traffic Light: Yellow
Traffic Light: Red
Pedestrian Crossing: Green
Traffic Light: Red
Traffic Light: Yellow
Traffic Light: Green
Pedestrian Waiting: Red
Traffic Light: Green
SSTraffic Light: Yellow
```

# DESCRIPTION OF C CODE

1. Header Inclusions: The code includes necessary header files such as <stdio.h>, <ctype.h>, and <unistd.h>.
2. Constants: It defines several constants to represent durations of different phases of the traffic light system, such as YELLOW_DURATION, GREEN_FOR_TRAFFIC, GREEN_FOR_PEDESTRIANS, and RED_DURATION.
3. Function Definitions:
    - Traffic_light(): This function contains the main logic of the traffic light simulation. It prompts the user to choose between day and night mode and controls the behavior of the traffic lights accordingly.
    - main(): The main function of the program. It calls the Traffic_light() function to start the simulation.

# DESCRIPTION OF C CODE

4. Traffic_light() Function:
   - It begins by prompting the user to select between day and night mode.
   - If the user chooses night mode (`'n'`), it sets a flag `isNight` to 1 and continuously prints "Traffic Light: YELLOW" without changing, simulating a constant yellow light.
   - If the user chooses day mode (`'d'`), it prints the traffic light sequence for day mode, which includes green light for traffic, red light for traffic, green light for pedestrians, and red light for pedestrians.
   - The simulation continues indefinitely until the user chooses to exit by entering any other key.

5. Main Function:
   - It simply calls the `Traffic_light()` function to start the simulation.

This code provides a basic framework for simulating a traffic light system in C and can be further expanded or modified to include additional features or improve simulation accuracy.

# EMBEDDED C CODE

```
#include<reg51.h>

void T0M1delay(void);

sbit switch1 = P0^3;

int main(void) {
    int x;

   switch1 = 1; // make switch as input

    if (switch1 == 0) {
        P2 = 0x00; // pedestrian off
        P3 = 0x00;

        while (1) {
            P0 = 0x02; // traffic yellow
            P1 = 0x02;
```

# EMBEDDED C CODE

```
for (x = 0; x < 42; x++) { // 3 sec delay
            T0M1delay();
          }

          P0 = 0x00; // traffic green
          P1 = 0x00;

          for (x = 0; x < 42; x++) { // 3 sec delay
            T0M1delay();
          }
       }
    }
    else {
       while (1) {
          P0 = 0x01; // traffic red
          P1 = 0x01;
          P2 = 0X04; // pedestrian green
          P3 = 0X04;
```

# EMBEDDED C CODE

```
for (x = 0; x < 42; x++) { // 3 sec delay
            T0M1delay();
          }

          P0 = 0x02; // traffic yellow
          P1 = 0x02;
          P2 = 0X01; // pedestrian red
          P3 = 0X01;

          for (x = 0; x < 14; x++) { // 1 sec delay
             T0M1delay();
          }
        P0=0x04;//traffic go
        P1=0x04;
        P2=0X01;//pedestrian stop
        P3=0X01;
```

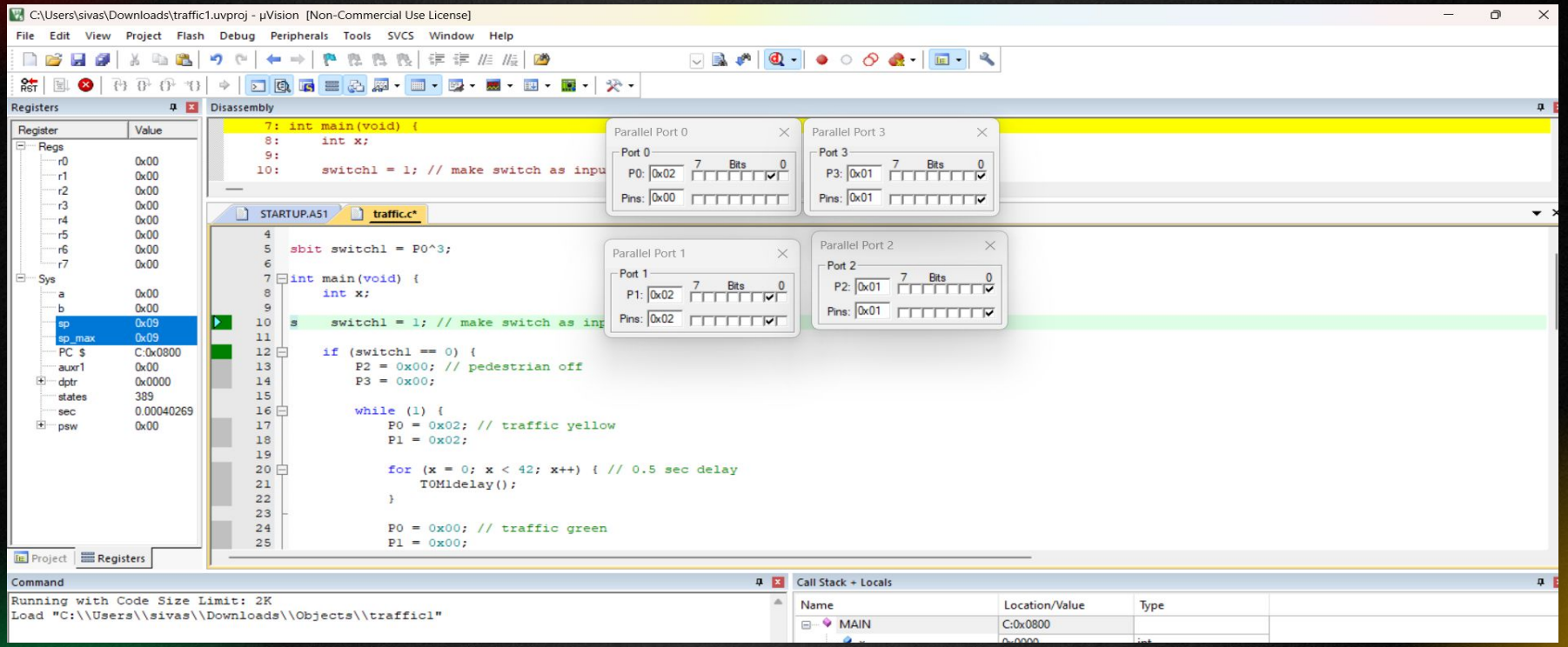
# EMBEDDED C CODE

```
for(x=0;x<42;x++){ //3 sec delay
T0M1delay();
}

P0=0x02;//traffic yellow
P1=0x02;
P2=0X01;//pedestrian stop
P3=0X01;
for(x=0;x<14;x++){ //1 sec delay
        T0M1delay();
      }
   }
}
```

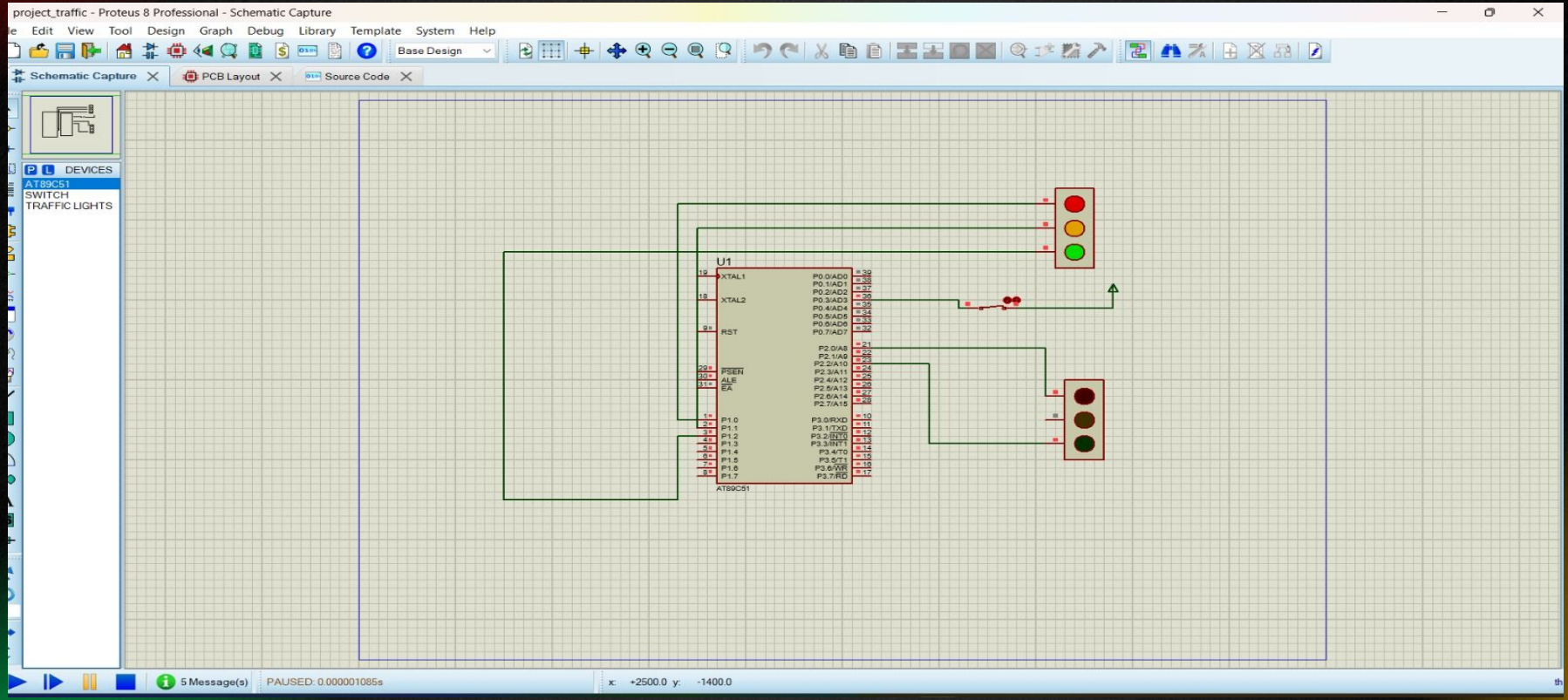
# EMBEDDED C CODE

```
void T0M1delay(void) {//defining the timer
        TMOD = 0x01; // timer 0, mode 1
        TH0 = 0x00;
        TL0 = 0x00;
        TR0 = 1; // start timer

    while (TF0==0); // wait for timer overflow flag to set
    TF0 = 0; // clear timer overflow flag
    TR0 = 0; // stop timer
}
```

# KEIL OUTPUT


C:\Users\sivas\Downloads\traffic1.uvproj - µVision [Non-Commercial Use License]
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Registers
Register Value
Regs
r0 0x00
r1 0x00
r2 0x00
r3 0x00
r4 0x00
r5 0x00
r6 0x00
r7 0x00
Sys
a 0x00
b 0x00
sp 0x09
sp_max 0x09
PC $ C:0x0800
auxr1 0x00
dptr 0x0000
states 389
sec 0.00040269
psw 0x00
Project Registers
Disassembly
7: int main(void) {
8: int x;
9:
10: switch1 = 1; // make switch as inpu
Parallel Port 0
Port 0
P0: 0x02 7 Bits 0
Pins: 0x00
Parallel Port 3
Port 3
P3: 0x01 7 Bits 0
Pins: 0x01
STARTUP.A51
traffic.c*
4
5 sbit switch1 = P0^3;
6
7 int main(void) {
8 int x;
9
10 switch1 = 1; // make switch as inp
11
12 if (switch1 == 0) {
13 P2 = 0x00; // pedestrian off
14 P3 = 0x00;
15
16 while (1) {
17 P0 = 0x02; // traffic yellow
18 P1 = 0x02;
19
20 for (x = 0; x < 42; x++) { // 0.5 sec delay
21 T0M1delay();
22 }
23
24 P0 = 0x00; // traffic green
25 P1 = 0x00;
Parallel Port 1
Port 1
P1: 0x02 7 Bits 0
Pins: 0x02
Parallel Port 2
Port 2
P2: 0x01 7 Bits 0
Pins: 0x01
Command
Running with Code Size Limit: 2K
Load "C:\\Users\\sivas\\Downloads\\Objects\\traffic1"
Call Stack + Locals
Name Location/Value Type
MAIN C:0x0800

# PROTEUS OUTPUT



# DESCRIPTION OF THE CODE

- It starts by defining the T0M1delay() function, which uses Timer 0 in mode 1 to create a delay of approximately 0.5 seconds.
- The code reads the state of switch1, likely connected to a physical switch, to determine if it should operate in pedestrian mode (switch pressed) or regular mode (switch not pressed).
- In pedestrian mode (switch pressed), it controls the traffic lights and pedestrian signals accordingly:
  - Traffic light is red, and pedestrian light is green for approximately 3 seconds.
  - Then traffic light turns yellow for approximately 1 second, while the pedestrian light remains red.
- In regular mode (switch not pressed), it simulates the traffic light sequence:
  - Traffic light starts with yellow for approximately 0.5 seconds.
  - Then it turns green for approximately 0.5 seconds.
  - The cycle repeats indefinitely.

This code assumes specific hardware connections and timings. The T0M1delay() function generates accurate delays for controlling the traffic light sequence. The switch1 input determines whether the system operates in pedestrian or regular mode.

# CONCLUSION

Microcontroller Traffic Light Control (8051 Assembly code):

- This assembly code is designed to control a traffic light system using an 8051 microcontroller.
- It includes logic to handle both regular and pedestrian modes based on the state of a physical switch (switch1).
- In pedestrian mode, it allows pedestrians to cross the road safely by controlling the traffic and pedestrian lights in a synchronized manner.
- In regular mode, it simulates the typical sequence of traffic lights, transitioning between green, yellow, and red lights for traffic.
- The code demonstrates the use of timer interrupts for generating accurate delays, essential for coordinating the timing of the traffic light sequence.
- It showcases low-level hardware control and efficient use of the microcontroller's resources to achieve the desired functionality.