CECS 346 Spring 2025 Project 1

Traffic Light Controller

By

Eric Santana & Bao Luong & Noah

Luu & Thaisinge Kour

Date: 3/8/2025

The traffic light controller is an embedded system using a TM4C123 microcontroller and a Moore finite state machine (FSM) to manage traffic at a intersection. It takes inputs from traffic sensors and a pedestrian button to control traffic and pedestrian lights.

Introduction

The traffic light controller is a real-time embedded system implemented on a TM4C123 microcontroller using a Moore finite state machine (FSM). It manages traffic at a intersection with two roads (South and West) and a pedestrian crossing. Inputs include car sensors for each road and a pedestrian button, while outputs control traffic lights and pedestrian signals.

The system follows predefined timing sequences using the SysTick timer, ensuring fair traffic flow by cycling through green, yellow, and red lights. The pedestrian signal transitions through walk, hurry up (flashing), and don't walk states. The FSM ensures safe and efficient traffic management while handling multiple requests in a round-robin fashion.

Operation

The Traffic Light Controller operates using a Moore FSM on a TM4C123 microcontroller, managing an intersection with southbound, westbound, and pedestrian traffic. The system detects vehicles via sensors (SW1 for southbound, SW2 for westbound) and allows pedestrian crossings via a push button. Traffic lights (Port B) and pedestrian signals (Port E3) change based on priority, with pedestrians having the highest priority. Upon power-up, the system starts with a green light for southbound traffic. When a vehicle is detected in the westbound direction, the system transitions through yellow before granting westbound green. If a pedestrian presses the button, the system ensures a safe crossing by activating the white LED, followed by flashing warnings before returning to normal traffic flow. The SysTick timer manages state delays, ensuring smooth real-time transitions. Users can observe automatic changes, manually trigger sensors, and reset the system if needed. Proper wiring and configuration are essential for correct operation.

Theory

The Traffic Light Controller project is based on the Moore Finite State Machine (FSM) theory, where the system transitions between predefined states based on sensor inputs and time delays. Using the TM4C123 microcontroller, the system manages an intersection by controlling traffic lights through GPIO registers and synchronizing state changes using the SysTick timer for real-time operation. The FSM ensures smooth transitions between southbound, westbound, and pedestrian signals, prioritizing pedestrian safety by overriding vehicle flow when the pedestrian button is pressed. The sensor-based decision-making system simulates real-world traffic management by detecting vehicles through input switches and adjusting signals accordingly. This implementation demonstrates key embedded systems concepts, including register-level programming, digital logic, real-time synchronization, and priority-based control, ensuring an efficient and safe traffic light operation.

Hardware Design

Inputs (Sensors - Port E)

- Traffic Sensors & Pedestrian Button (Active HIGH)
  - PE2 - Southbound Traffic Sensor (1 = Vehicle Detected)
  - PE1 - Westbound Traffic Sensor (1 = Vehicle Detected)
  - PE0 - Pedestrian Button (1 = Button Pressed)

Outputs (Traffic & Pedestrian Lights)

Traffic Lights (Port B)

Traffic Light	**Port B Pin**	**Meaning (1 = ON)**
West Red	PB5	West Traffic **STOP**
West Yellow	PB4	West Traffic **CAUTION**
West Green	PB3	West Traffic **GO**
South Red	PB2	South Traffic **STOP**
South Yellow	PB1	South Traffic **CAUTION**

South Green	PB0	South Traffic **GO**

Pedestrian Signals (Port F)

Pedestrian Signal	Port F Pin	Meaning (1 = ON)
Walk	PF3	Pedestrians **Cross**
Don't Walk	PF1	Pedestrians **Stop**

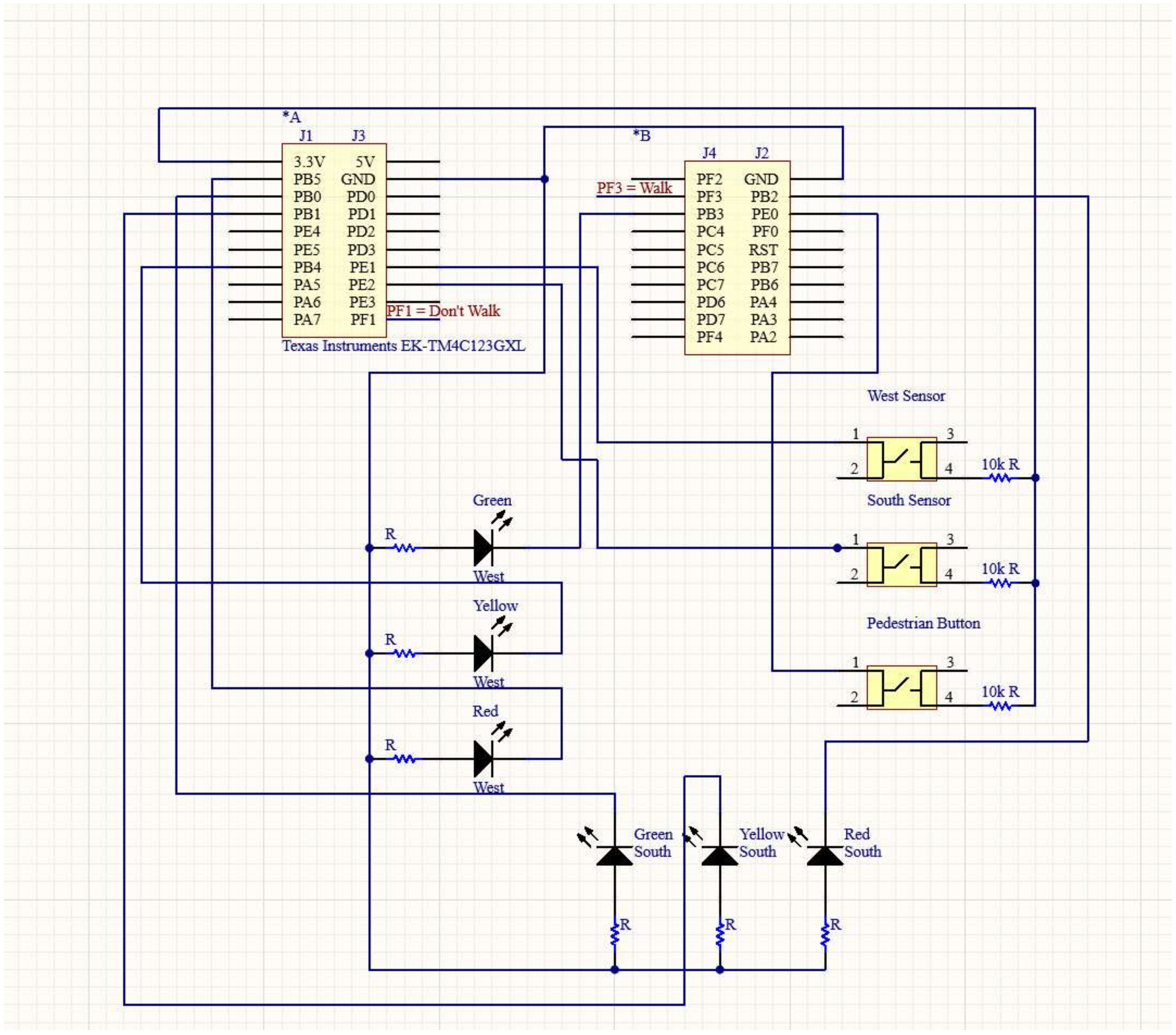
Microcontroller Pin Configuration

- **Port E (Inputs)**
  - Configured as digital inputs with internal pull-down resistors.
  - Sensors and pedestrian button activate when HIGH (1).
- **Port B (Traffic Lights)**
  - Configured as digital outputs.
  - HIGH (1) turns the corresponding light ON.
- **Port F (Pedestrian Signals)**
  - Configured as digital outputs.
  - HIGH (1) activates the corresponding pedestrian signal.

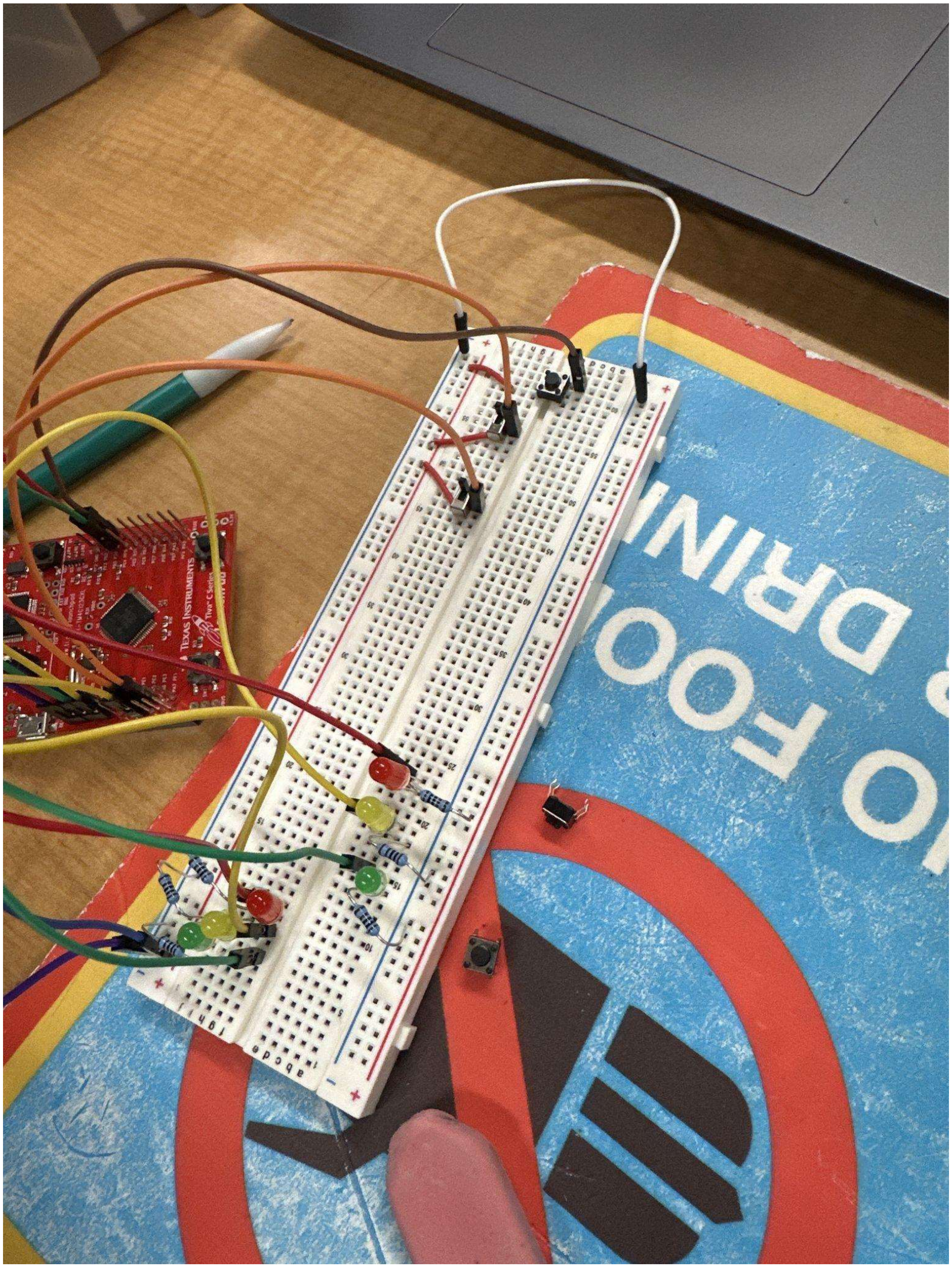
4. System Operation

- **Normal Traffic Flow:**
  - Only one direction (South or West) gets a green light at a time.
  - Traffic transitions based on sensor input (if no car is detected, it remains green).
  - Yellow light turns on before switching to red.
- **Pedestrian Crossing:**
  - When the pedestrian button (PE0) is pressed, the system switches to pedestrian mode.
  - The Walk signal (PF3) turns ON.
  - The Don't Walk signal (PF1) flashes before turning solid.

Schematic


*A
J1
J3
3.3V
5V
PB5
GND
PB0
PD0
PB1
PD1
PE4
PD2
PE5
PD3
PB4
PE1
PA5
PE2
PA6
PE3
PA7
PF1
PF1 = Don't Walk
Texas Instruments EK-TM4C123GXL
*B
J4
J2
PF3 = Walk
PF2
GND
PF3
PB2
PB3
PE0
PC4
PF0
PC5
RST
PC6
PB7
PC7
PB6
PD6
PA4
PD7
PA3
PF4
PA2
West Sensor
South Sensor
Pedestrian Button
10k R
Green West
Yellow West
Red West
Green South
Yellow South
Red South
R

Breadboard



Software Design

This program implements a Moore Finite State Machine to control a traffic light system with a pedestrian crossing.

Inputs (Sensors - Port E)

- PE2 (Southbound traffic sensor)
- PE1 (Westbound traffic sensor)
- PE0 (Pedestrian crossing button)

Outputs

- **Traffic Lights (Port B)**
  - **West Traffic Light**
    - Red: PB5
    - Yellow: PB4
    - Green: PB3
  - **South Traffic Light**
    - Red: PB2
    - Yellow: PB1
    - Green: PB0
- **Pedestrian Signals (Port F)**
  - Walk: PF3
  - Don't Walk: PF1

FSM Design

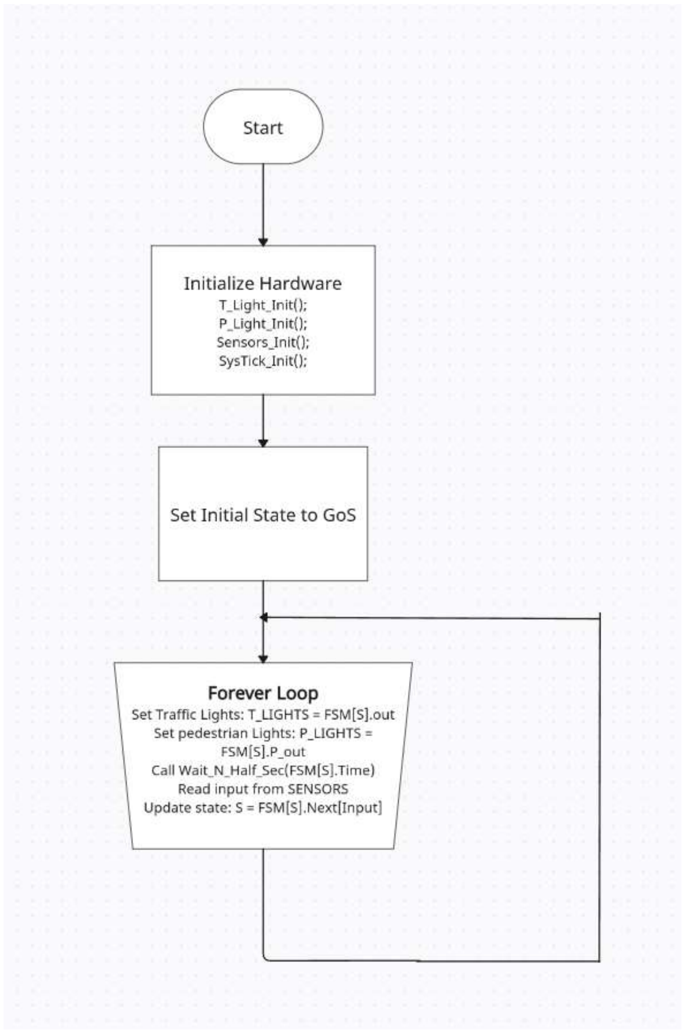
- The system follows a finite state machine with 9 states:
    - **GoS** (South traffic moves, others wait)
    - **WaitS** (South traffic yellow light, prepares to stop)
    - **GoW** (West traffic moves, others wait)
    - **WaitW** (West traffic yellow light, prepares to stop)
    - **GoP** (Pedestrian crosses)
    - **WaitPOn1** (Pedestrian hurry signal flashes)
    - **WaitPOff1** (Pedestrian hurry signal off)
    - **WaitPOn2** (Pedestrian hurry signal flashes again)
    - **WaitPOff2** (Pedestrian hurry signal off, returns to normal)
- Transitions are determined by input sensors (traffic or pedestrian detected).

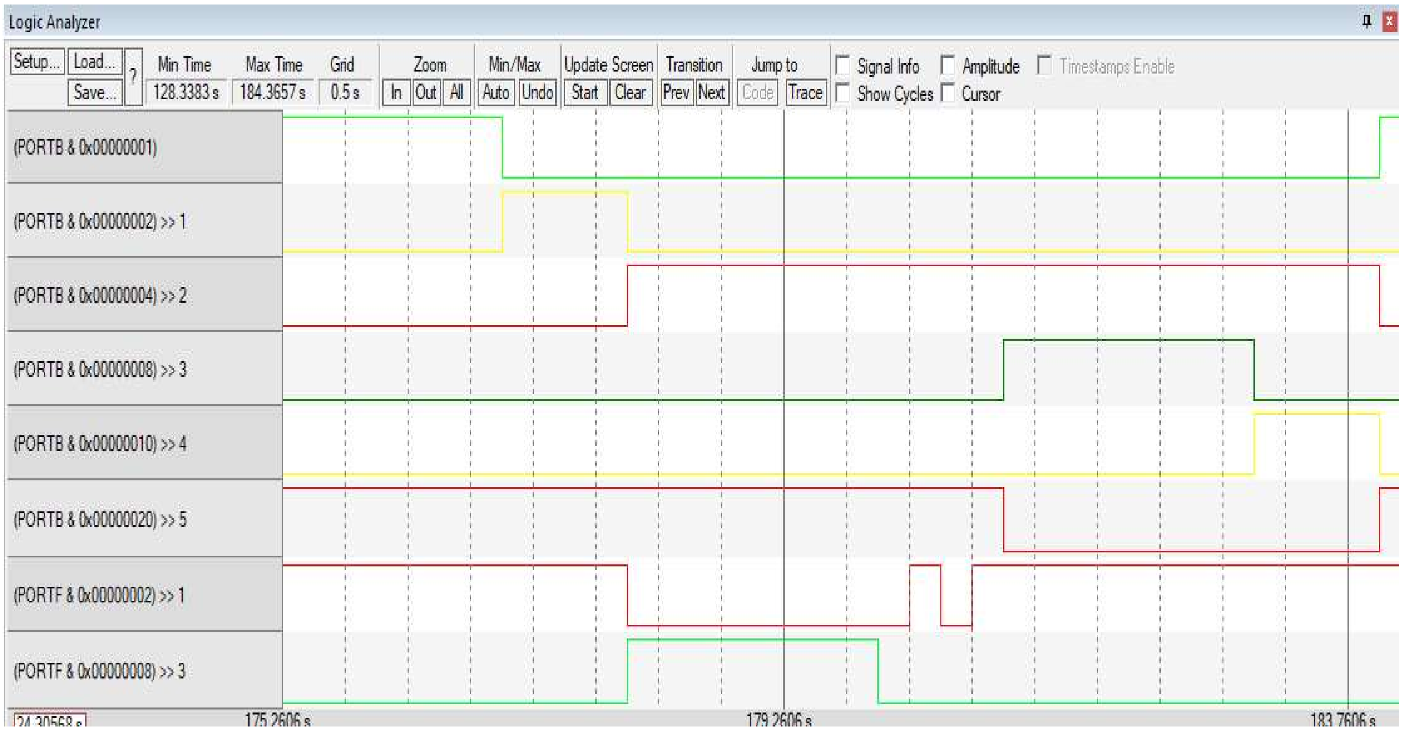
System Initialization

- **T_Light_Init()** - Configures Port B for traffic lights
- **P_Light_Init()** - Configures Port F for pedestrian lights
- **Sensors_Init()** - Configures Port E as input for traffic & pedestrian sensors
- **SysTick Timer** - Used to implement state delays

FSM (main loop)

1. Start in the GoS (South traffic green).
2. Continuously:
    a. Set traffic lights (T_LIGHTS) and pedestrian signals (P_LIGHTS).
    b. Wait for the duration defined for that state.
    c. Read sensor input to determine the next state.
    d. Transition to the appropriate state based on the input.

e. 
Start
Initialize Hardware
T_Light_Init();
P_Light_Init();
Sensors_Init();
SysTick_Init();
Set Initial State to GoS
Forever Loop
Set Traffic Lights: T_LIGHTS = FSM[S].out
Set pedestrian Lights: P_LIGHTS = FSM[S].P_out
Call Wait_N_Half_Sec(FSM[S].Time)
Read input from SENSORS
Update state: S = FSM[S].Next[Input]

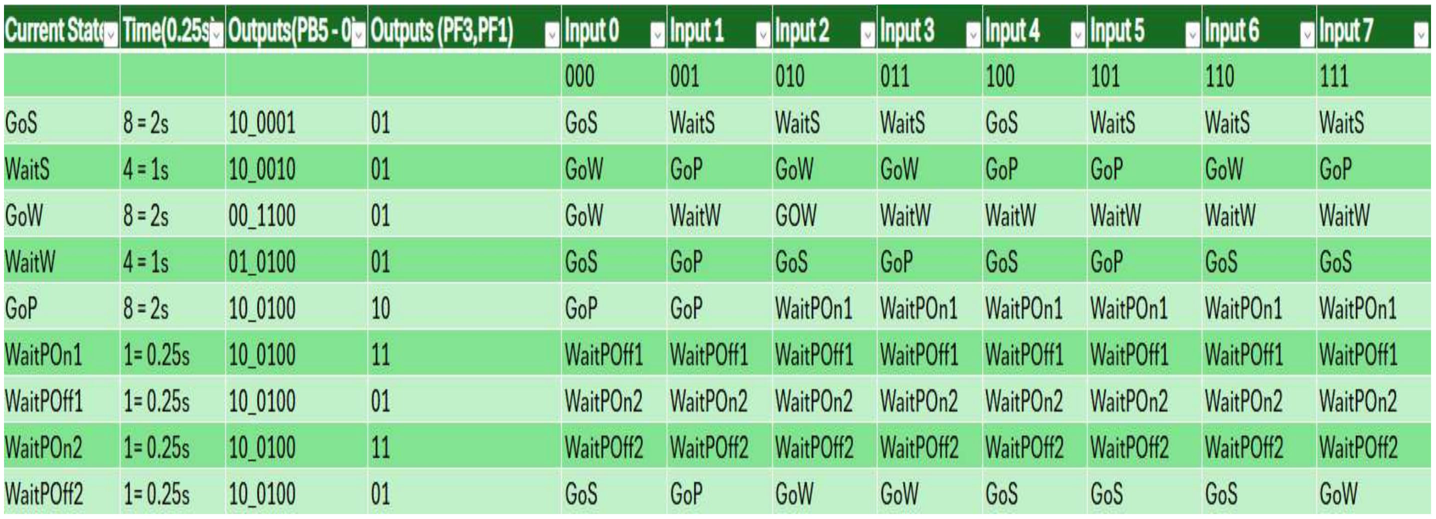


Min Time 128.3383 s
Max Time 184.3657 s
Grid 0.5 s

Software List

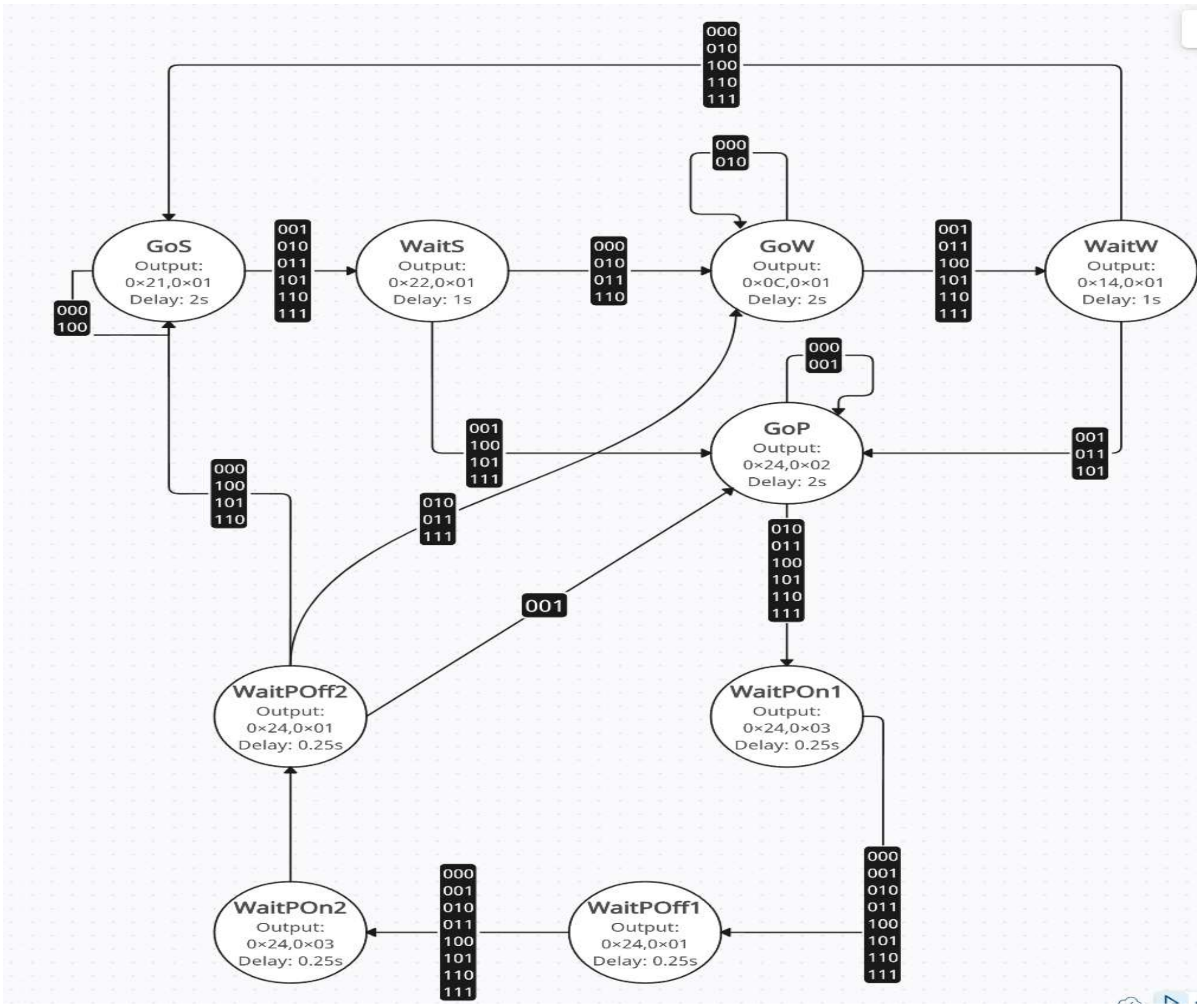
CECS_346_Project1_Traffic_Light_Controller.c
Lab4SysTick.c
Lab4SysTick.h
Stdint.h

State Table



Current Stat	Time(0.25s	Outputs(PB5 - 0	Outputs (PF3,PF1)	Input 0	Input 1	Input 2	Input 3	Input 4	Input 5	Input 6	Input 7
				000	001	010	011	100	101	110	111
GoS	8 = 2s	10_0001	01	GoS	WaitS	WaitS	WaitS	GoS	WaitS	WaitS	WaitS
WaitS	4 = 1s	10_0010	01	GoW	GoP	GoW	GoW	GoP	GoP	GoW	GoP
GoW	8 = 2s	00_1100	01	GoW	WaitW	GOW	WaitW	WaitW	WaitW	WaitW	WaitW
WaitW	4 = 1s	01_0100	01	GoS	GoP	GoS	GoP	GoS	GoP	GoS	GoS
GoP	8 = 2s	10_0100	10	GoP	GoP	WaitPOn1	WaitPOn1	WaitPOn1	WaitPOn1	WaitPOn1	WaitPOn1
WaitPOn1	1= 0.25s	10_0100	11	WaitPOff1	WaitPOff1	WaitPOff1	WaitPOff1	WaitPOff1	WaitPOff1	WaitPOff1	WaitPOff1
WaitPOff1	1= 0.25s	10_0100	01	WaitPOn2	WaitPOn2	WaitPOn2	WaitPOn2	WaitPOn2	WaitPOn2	WaitPOn2	WaitPOn2
WaitPOn2	1= 0.25s	10_0100	11	WaitPOff2	WaitPOff2	WaitPOff2	WaitPOff2	WaitPOff2	WaitPOff2	WaitPOff2	WaitPOff2
WaitPOff2	1= 0.25s	10_0100	01	GoS	GoP	GoW	GoW	GoS	GoS	GoS	GoW

State Diagram



Conclusion

The Traffic Light Controller project was a successful implementation of a Moore Finite State Machine (FSM) on the TM4C123 microcontroller, effectively managing traffic flow and pedestrian safety using GPIO inputs, sensor-based decision-making, and the SysTick timer for real-time synchronization. We successfully integrated LED signals, input switches, and state transitions, ensuring proper priority handling between vehicles and pedestrians. Challenges included GPIO configuration errors, input debouncing issues, and FSM logic refinements, which required careful debugging and optimization. Despite these hurdles, we gained valuable experience in embedded systems programming, digital logic design, and real-time control, reinforcing the importance of modular coding, precise timing, and efficient state management in embedded applications.