University of Dhaka

Department of Computer Science and Engineering

CSE 3116 -Microcontroller Lab
Batch 28 / 3RD Year 1ST Semester

Blinky Program and Road Traffic Management

Submitted To:
Dr. Mosaddek Tushar,Professor,CSEDU
Jargis Ahmed,Lecturer,CSEDU

Group Number: B_9(Even)

Submitted By:
Farzana Tasnim (14)
Amina Islam (36)
Rezaunnabi Ruhan (58)

1 Blinky Program

1.1 Design

GPIO Configuration

- **Pin Used**: GPIOA pin 5 (**PA5**) for LED control.
- **Configuration** (in `GPIO_Init`):
  - Enable GPIOA clock (`RCC->AHB1ENR |= 0x1U`).
  - Set PA5 as output (`GPIOA->MODER |= 0x1U << 10`, sets `MODER[11:10] = 01`).
  - Configure PA5 for very high-speed operation (`GPIOA->OSPEEDR |= 0x3U << 10`).

Control Function:

- `GPIO_ON(5)`: Sets PA5 high (`GPIOA->BSRR = 0x1U << 5`) to turn the LED on.
- `GPIO_OFF(5)`: Sets PA5 low (`GPIOA->BSRR = 0x1U << (5 + 16)`) to turn the LED off.

Main Loop Operation

- **Functionality**: Continuously toggles the LED on PA5.
- **Implementation** (in `main`):
  - Initialize clock (`initClock()`) and GPIO (`GPIO_Init()`).
  - Enter infinite loop:
    - Turn LED on (`GPIO_ON(5)`).
    - Delay using a `for` loop (`for(volatile int i = 0; i < 1000000; i++)`).
    - Turn LED off (`GPIO_OFF(5)`).
    - Delay again with the same loop.

1.2 Required Hardwares

1. LED(1)
2. Register(1)
3. Breadboard
4. STM32F446RE
5. Wires

1.3 Result

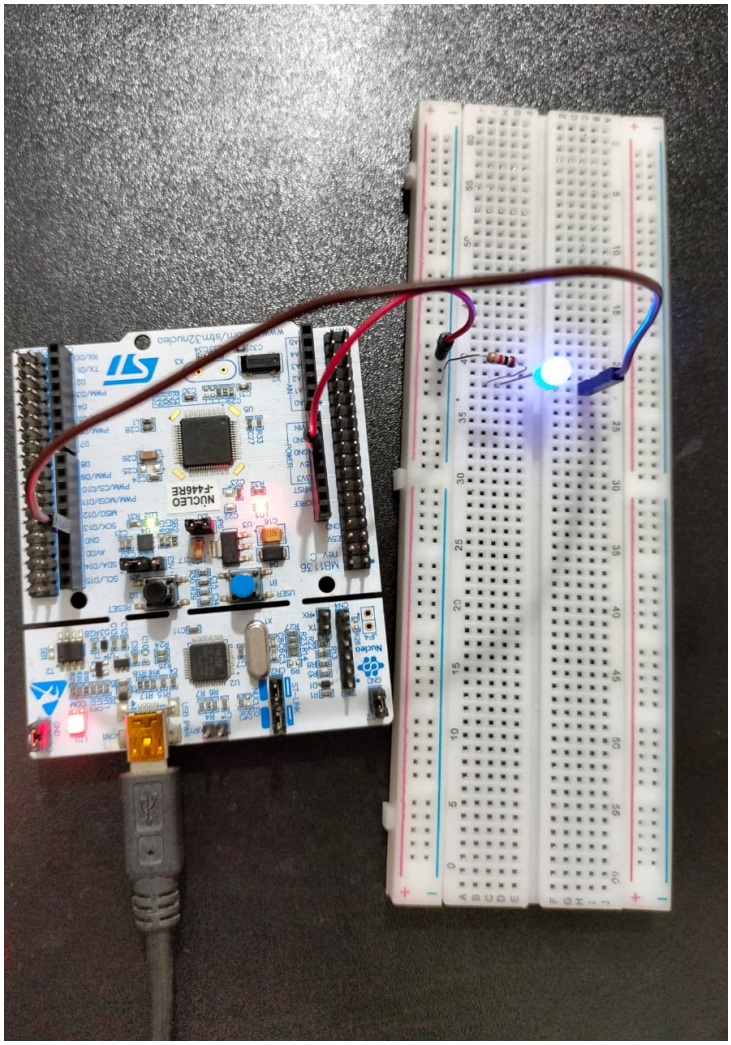

Figure 1: Blinking Led Circuit

2 Traffic Management

2.1 Design

Hardware Configuration

- **GPIO Setup**:
  - *Port B (PB0–PB5)*: Configured as outputs for traffic light signals:
    - **PB0: NS Red, PB1: NS Yellow, PB2: NS Green**
    - **PB3: EW Red, PB4: EW Yellow, PB5: EW Green**
  - *Port A (PA6–PA7)*: Configured as outputs for load indicator LEDs:
    - **PA6: NS Load Indicator, PA7: EW Load Indicator**
  - All pins are set to **high-speed** operation.

- **Clock Configuration**:
  - Configures AHB , APB1 and APB2 .

Software Design

- **Traffic States**: Defined as an enumeration with three states:
  - `NORTH_SOUTH_GREEN`: NS green, EW red
  - `YELLOW`: NS yellow, EW yellow
  - `EAST_WEST_GREEN`: EW green, NS red
- **Traffic Light Control**:
  - The `set_traffic_lights` function sets GPIO pins based on the current state, ensuring only the appropriate lights are active.
  - States transition in a fixed sequence: NS Green → NS Yellow → EW Green → EW Yellow, looping indefinitely.
- **Traffic Load Simulation**:
  - The `simulate_random_load` function generates random binary values (0 or 1) for NS and EW traffic loads.
  - Load indicators (PA6, PA7) reflect the simulated load (high/low).
  - The green light duration dynamically adjusts based on traffic load—set to 20 seconds under high load conditions and 10 seconds under low load—whereas the yellow light duration is fixed at 5 seconds regardless of load.
- **Main Loop**:
  - Initializes clock, system init, and traffic init.
  - Enters an infinite loop in `traffic_control`, cycling through traffic states with adaptive green timings based on simulated load.

Timing and Operation

The system operates as a state machine with adaptive timing:

- **NS Green**: 10s (low load) or 20s (high load), followed by 5s yellow.
- **EW Green**: 10s (low load) or 20s (high load), followed by 5s yellow.

2.2 Required Hardwares

1. LEDs(2 red, 2 green, 2 yelllow, 2 white)
2. 1k ohm Register(8)
3. Breadboard
4. STM32F446RE
5. Wires

2.3 Result

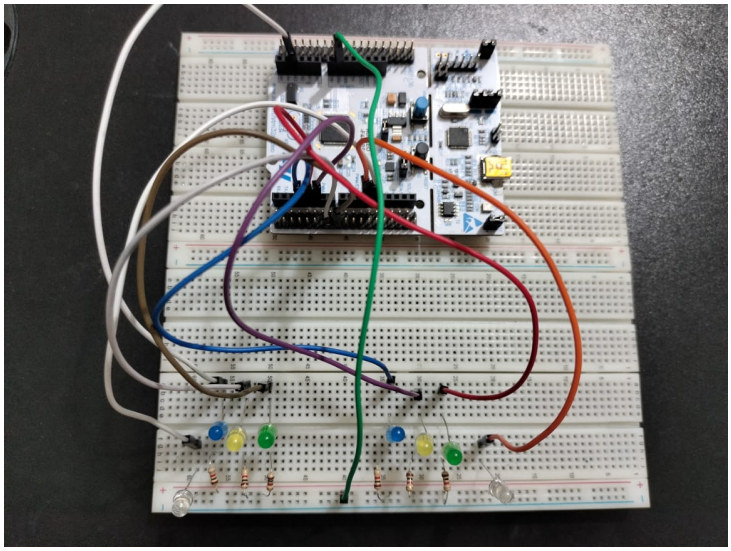

Figure 2: Left:North-South ,Right:East-West ,Traffic lights(Red, Green, Blue LEDs), Loads-White LEDs

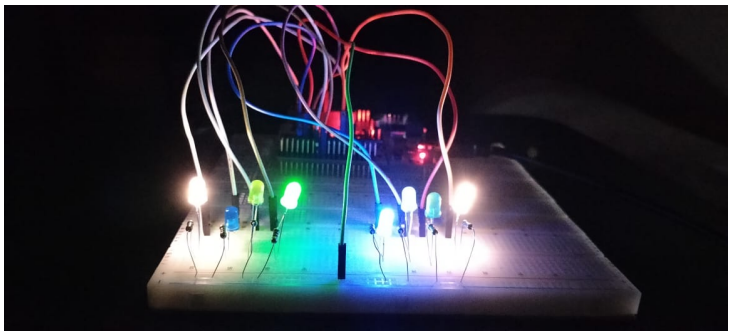

Figure 3: NS-Green, EW-Red , Load on both side.

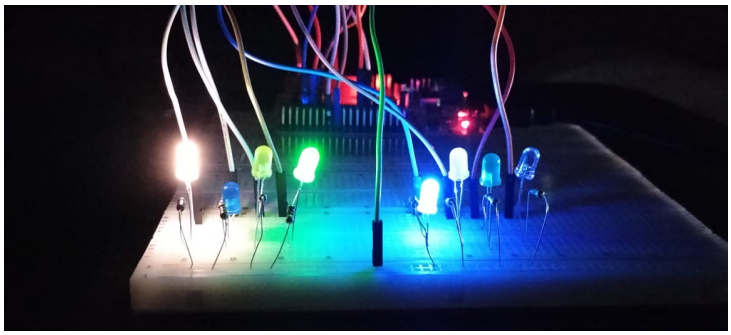

Figure 4: NS-Green, EW-Red , Load on NS side.

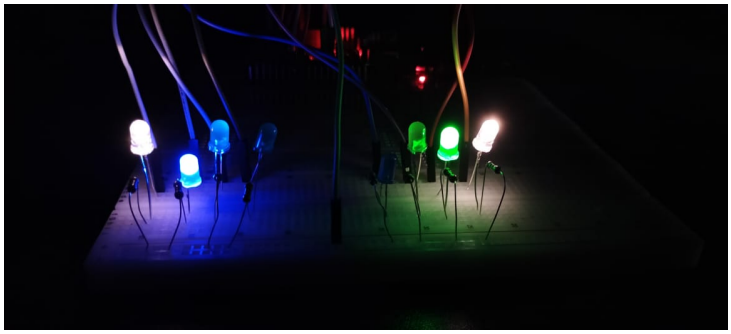

Figure 5: NS-Red, EW-Green , Load on Both side.

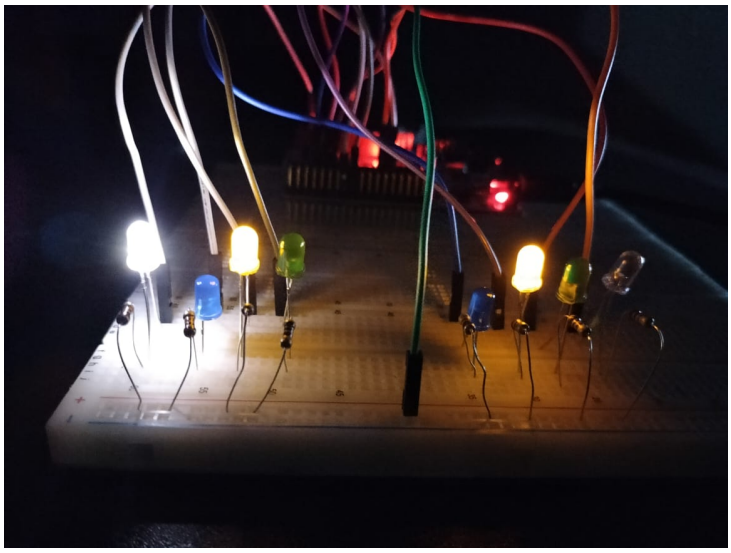

Figure 6: Transition from Green to Red and vice versa.