🚦 Traffic Light Controller Using PIC18F4520 (Assembly)

This program implements a simple **traffic light controller** using **Assembly Language** for the **PIC18F4520 microcontroller**. The lights follow a standard traffic sequence:

1. **Green ON** → Wait **5 seconds**
2. **Yellow ON** → Wait **1 second**
3. **Red ON** → Wait **5 seconds**
4. **Repeat Cycle**

🔍 Line-by-Line Explanation

```
ORG 0x00  ; Reset vector (Execution starts here)
GOTO MAIN ; Jump to MAIN (start of main program)
```

- The program starts execution from address **0x00**.
- **GOTO MAIN** sends execution to the `MAIN` label where the actual code begins.

```
ORG 0x08  ; Interrupt vector (not used here)
```

- This is the **interrupt vector**. It is required by convention but not used in this code.

```
CBLOCK 0x20 ; Start of variable block (RAM address 0x20)
       COUNT1 ; First delay counter
       COUNT2 ; Second delay counter
ENDC ; End of variable declaration
```

- **CBLOCK** defines memory locations starting from **0x20**.
- **COUNT1 and COUNT2** are used as delay counters.

```
ORG 0x20  ; Start of main program memory
```

- This tells the assembler that the **actual program** starts from memory location **0x20**.

🌟 Main Program (Initialization)

```
assembly
CopyEdit
MAIN:
    CLRF LATB    ; Clear LATB (turn off all LEDs)
    CLRF PORTB   ; Clear PORTB (ensure all outputs are LOW)
    MOVLW 0xF8   ; Load WREG with 0xF8 (11111000 in binary)
    MOVWF TRISB  ; Set RB0, RB1, RB2 as outputs (other bits remain inputs)
```

- **CLRF LATB & CLRF PORTB**: Clears **PORTB** and **LATB**, turning off all LEDs.
- **MOVLW 0xF8 & MOVWF TRISB**:
  - **TRISB = 0xF8 (11111000)**: Sets **RB0, RB1, RB2 as outputs** (0 = output).
  - Other pins (RB3-RB7) remain as inputs.

🔄 Main Loop (Traffic Light Logic)

```
LOOP:
    MOVLW 0x04   ; Load 00000100 (Green ON)
    MOVWF LATB   ; Write to LATB (Turns ON Green LED at RB2)
    CALL DELAY_5S ; Wait 5 seconds

    MOVLW 0x02   ; Load 00000010 (Yellow ON)
    MOVWF LATB   ; Write to LATB (Turns ON Yellow LED at RB1)
    CALL DELAY_1S ; Wait 1 second

    MOVLW 0x01   ; Load 00000001 (Red ON)
    MOVWF LATB   ; Write to LATB (Turns ON Red LED at RB0)
    CALL DELAY_5S ; Wait 5 seconds

    GOTO LOOP   ; Repeat traffic light sequence
```

- **Green LED ON (RB2) → 5 sec delay**
- **Yellow LED ON (RB1) → 1 sec delay**
- **Red LED ON (RB0) → 5 sec delay**
- **Repeat indefinitely**

⍰ Delay Subroutines

1-Second Delay Routine

```
DELAY_1S:
    MOVLW 0xFF   ; Load WREG with 0xFF (255)
    MOVWF COUNT1 ; Store in COUNT1
DELAY_LOOP1:
    MOVLW 0xFF   ; Load WREG with 0xFF (255)
    MOVWF COUNT2 ; Store in COUNT2
DELAY_LOOP2:
    DECFSZ COUNT2,F  ; Decrement COUNT2, Skip if Zero
    GOTO DELAY_LOOP2 ; Repeat until COUNT2 = 0
    DECFSZ COUNT1,F  ; Decrement COUNT1, Skip if Zero
    GOTO DELAY_LOOP1 ; Repeat outer loop
    RETURN
```

- **Creates ~1-second delay** using nested loops.
- **COUNT1 and COUNT2** decrement **from 255 to 0**, creating a delay.

5-Second Delay Routine

```
DELAY_5S:
    CALL DELAY_1S  ; Call 1-second delay 5 times
    CALL DELAY_1S
    CALL DELAY_1S
    CALL DELAY_1S
    CALL DELAY_1S
    RETURN
```

- **Calls the 1-second delay 5 times** to create a **5-second delay**.

📌 Algorithm (Traffic Light Controller)

1. **Initialize PORTB**:
   - Clear all bits
   - Configure RB0, RB1, and RB2 as **outputs**.
2. **Turn ON Green Light (RB2) for 5 seconds**.
3. **Turn OFF Green, Turn ON Yellow Light (RB1) for 1 second**.
4. **Turn OFF Yellow, Turn ON Red Light (RB0) for 5 seconds**.
5. **Repeat the process indefinitely**

🎯 Expected LED Behavior

State	RB2 (Green)	RB1 (Yellow)	RB0 (Red)	Duration
Step 1	ON (1)	OFF (0)	OFF (0)	**5 sec**
Step 2	OFF (0)	ON (1)	OFF (0)	**1 sec**
Step 3	OFF (0)	OFF (0)	ON (1)	**5 sec**
Loop Back	**Repeat from Step 1**			

✅ Key Features

✓ Uses simple bit manipulation to control LEDs
✓ Implements precise delays using software loops
✓ Uses `COMF` for toggling and simple state transitions
✓ Optimized for PIC18F4520

◆ What You Can Modify

- **Change delay values** (Adjust `CALL DELAY_1S` in `DELAY_5S`).
- **Add more traffic light states** (e.g., blinking green before yellow).
- **Extend for pedestrian crossing** by adding a separate LED sequence.