Interfacing LEDs and Switches

Ashish Kr. Arya

CS22B1012

What is Interfacing? Connecting the Microcontroller to the Real World

"Interfacing in embedded systems refers to the **methods and circuits used to connect a microcontroller (like our 8051) to other electronic devices or components.**"

It allows the microcontroller to **sense (input) and control (output)** its environment."

Interfacing LEDs (Output Devices):

How it Works with 8051: The 8051's port pins are configured as **digital outputs**.

The microcontroller sends a **HIGH (e.g., +5V) or LOW (e.g., 0V) signal** to an LED.

This signal controls whether current flows through the LED, making it light up or turn off.

Interfacing Switches (Input Devices):

How it Works with 8051: The 8051's port pins are configured as **digital inputs**.

The microcontroller reads the voltage level (HIGH or LOW) on these pins.

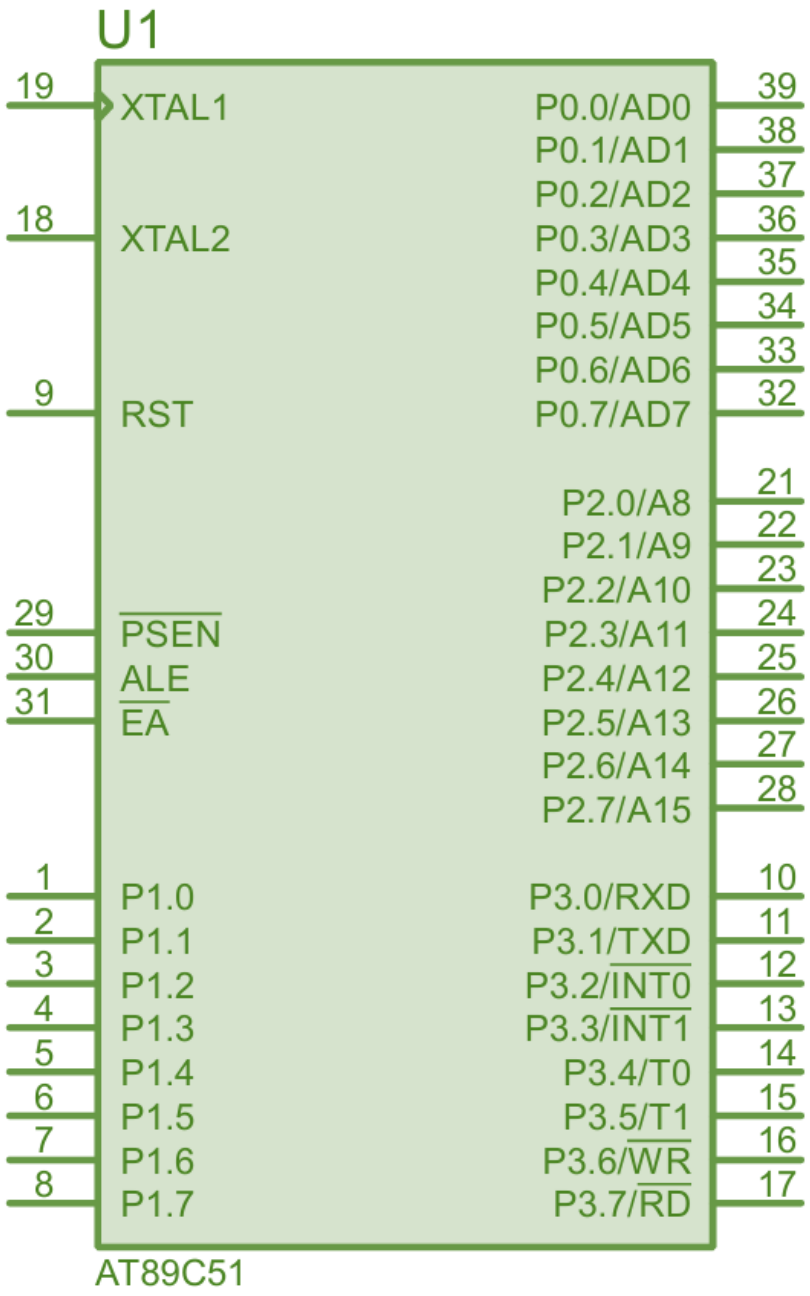
Key Considerations: Defining a Clear State Floating Input Problem: If a switch is simply connected to an input pin without further components, the pin's state can be undefined ("floating") when the switch is open.

A Mini-Project Demonstration

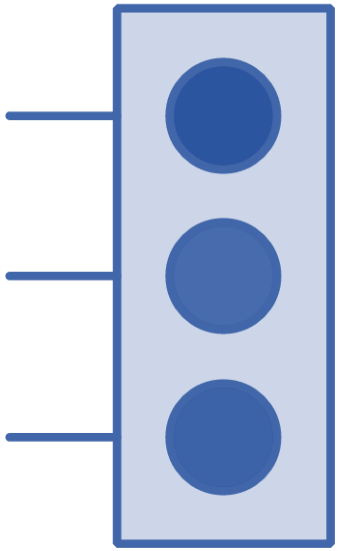
Project Overview: To design and simulate an 8051 microcontroller-based system that interfaces with LEDs (as traffic lights) and switches (for pedestrian interrupt and manual overrides).

Introduction to Key Components:

8051 Microcontroller (AT89C51)



Introduction to Key Components:

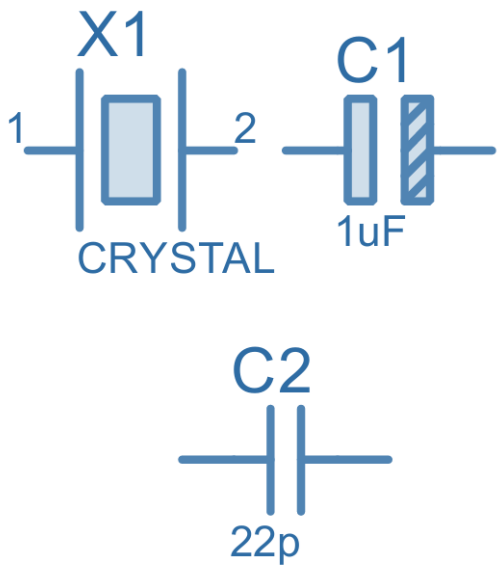


Traffic Light Component(Inbuilt LEDs along with resistors)

Push Button Switches

Resistors

Key Components(Contd.)



Crystal & Capacitors

Power Supply (BAT1)

GND

Circuit Design and Simulation Setup

8051 Core:

Crystal oscillator, reset circuit (R1, C4), EA pin connection to VCC

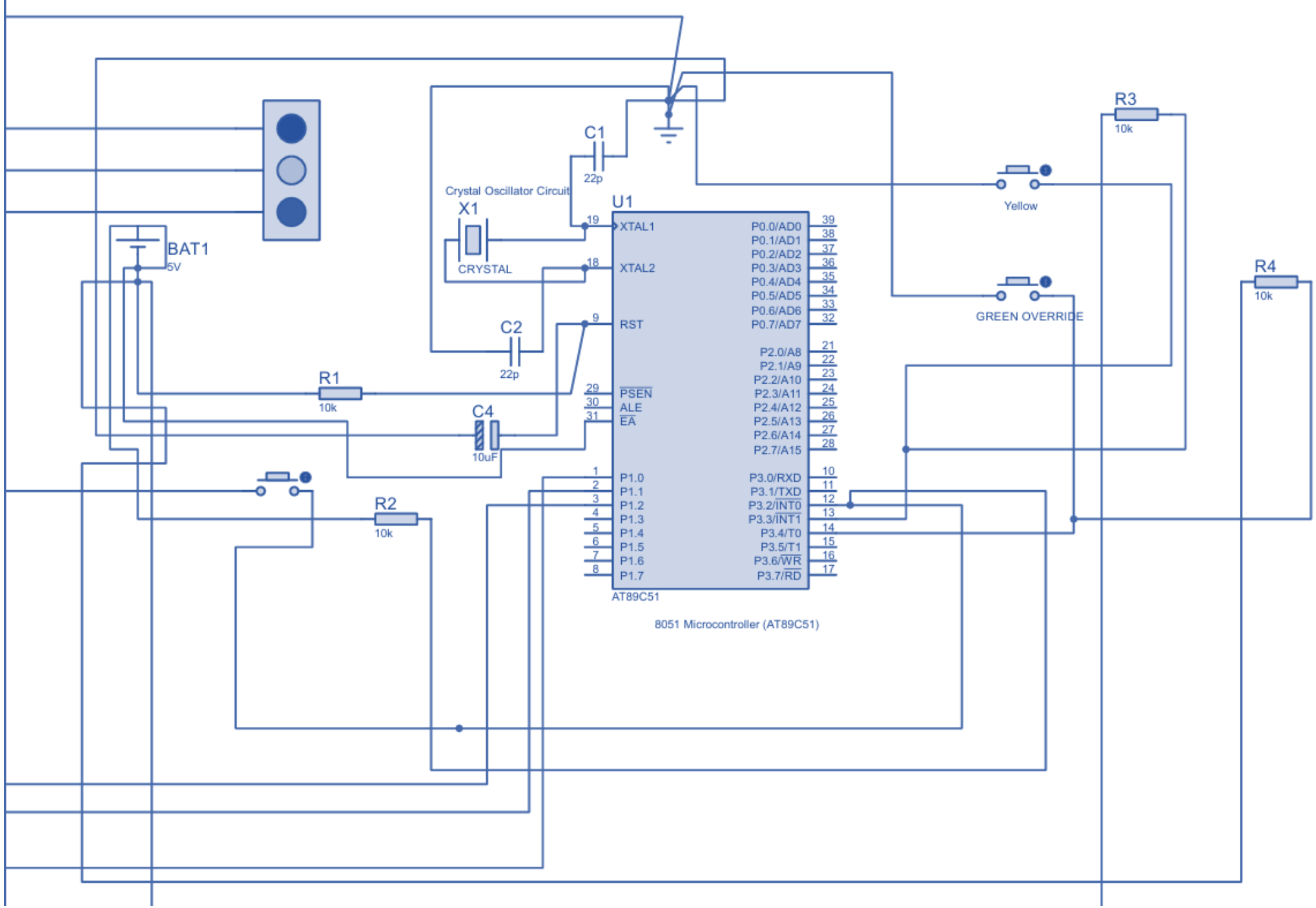
LED Interface:

P1.0, P1.1, P1.2 to the traffic light module.

Switch Interface (PULL-UP Configuration):

- Pedestrian Button (P3.2/INT0) with R2.
- Manual Yellow Switch (P3.3) with R3.
- Manual Green Switch (P3.4) with R4.

The Circuit Design



Programming the 8051: Control Logic in C

- Overall Structure: Header (reg51.h), sbit definitions for pins, global variables (flags, state), delay function, ISR, helper functions, main() loop.
- Pin Definitions: sbit declarations for LEDs and switches.

Interrupt Handling (Pedestrian Button):

- Configuration: IT0 = 1 (falling edge), EX0 = 1 (enable INT0), EA = 1 (global enable).
- Pedestrian_ISR(): Sets pedestrian_request_active flag. Mentions debounce.

Manual Override Logic (Polling in main()):

- Checking MANUAL_YELLOW_SWITCH (P3.3) and MANUAL_GREEN_SWITCH (P3.4).
- How they override the automatic sequence when pressed (active LOW).

Automatic Traffic Sequence (State Machine in main()):

- States: STATE_RED, STATE_YELLOW, STATE_GREEN.
- Software timer (current_state_timer_ticks with delay_ms(100)).
- Transition logic between states.

From Code to Hex File: The Compilation Process

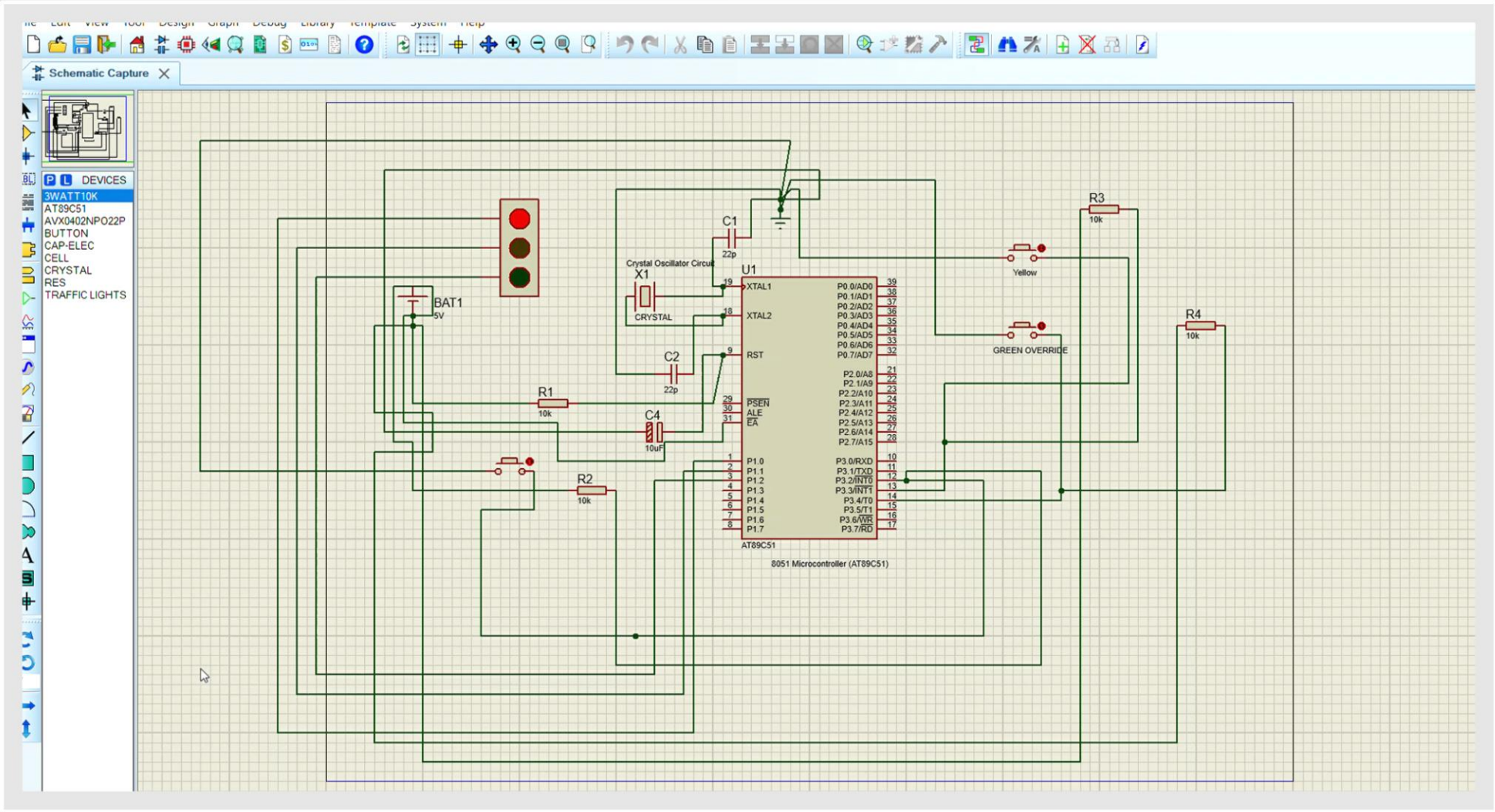
Tool Used: Keil µVision IDE, a standard development environment for 8051 microcontrollers.

```
creating hex file from ".\Objects\del"...
".\Objects\del" - 0 Error(s), 0 Warning(s).
Build Time Elapsed:  00:00:01
```

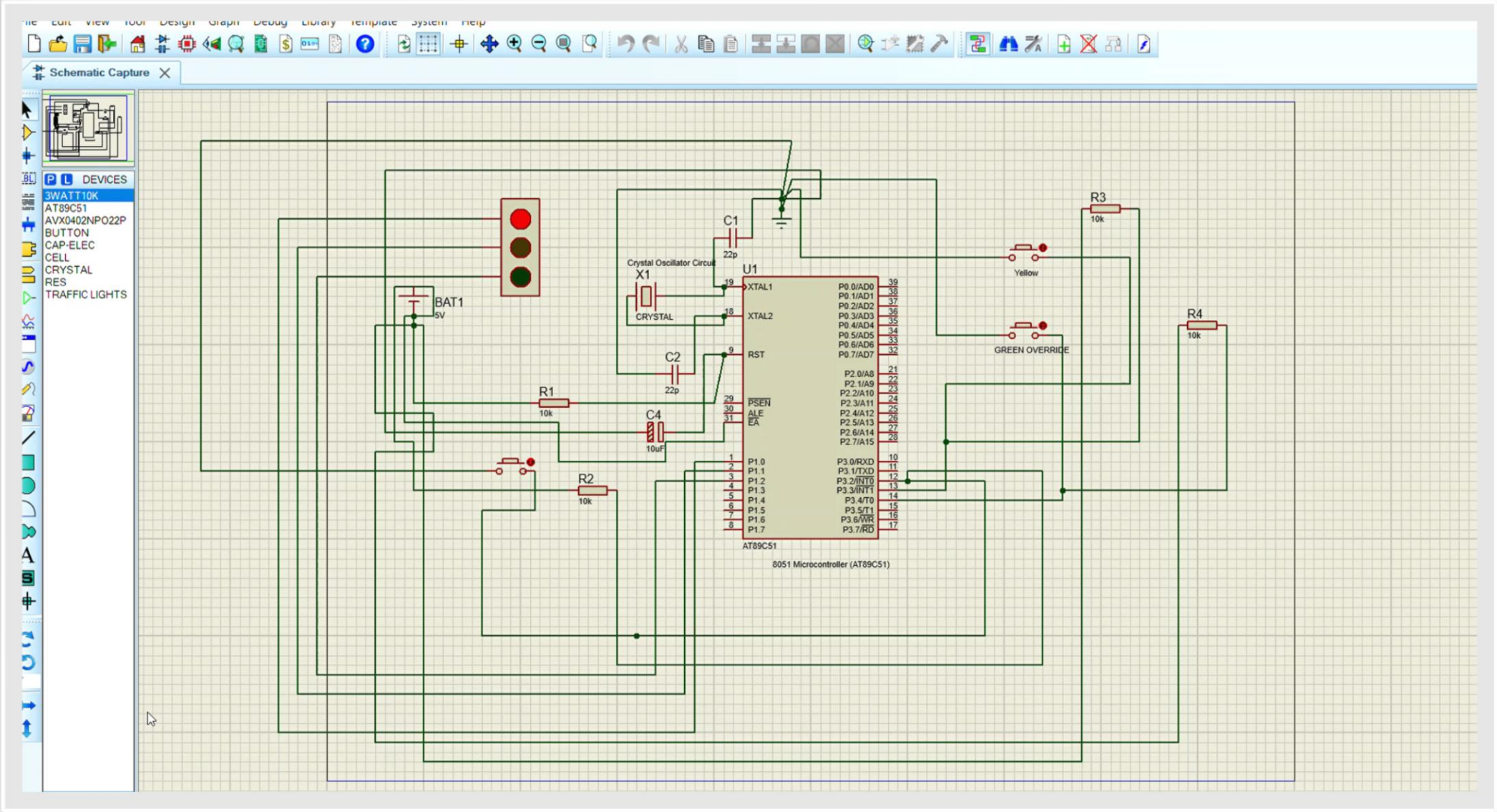
This file contains the program's instructions in a hexadecimal format that can be directly loaded into the microcontroller's memory in the Proteus simulation

Simulation and Demonstration (Proteus)

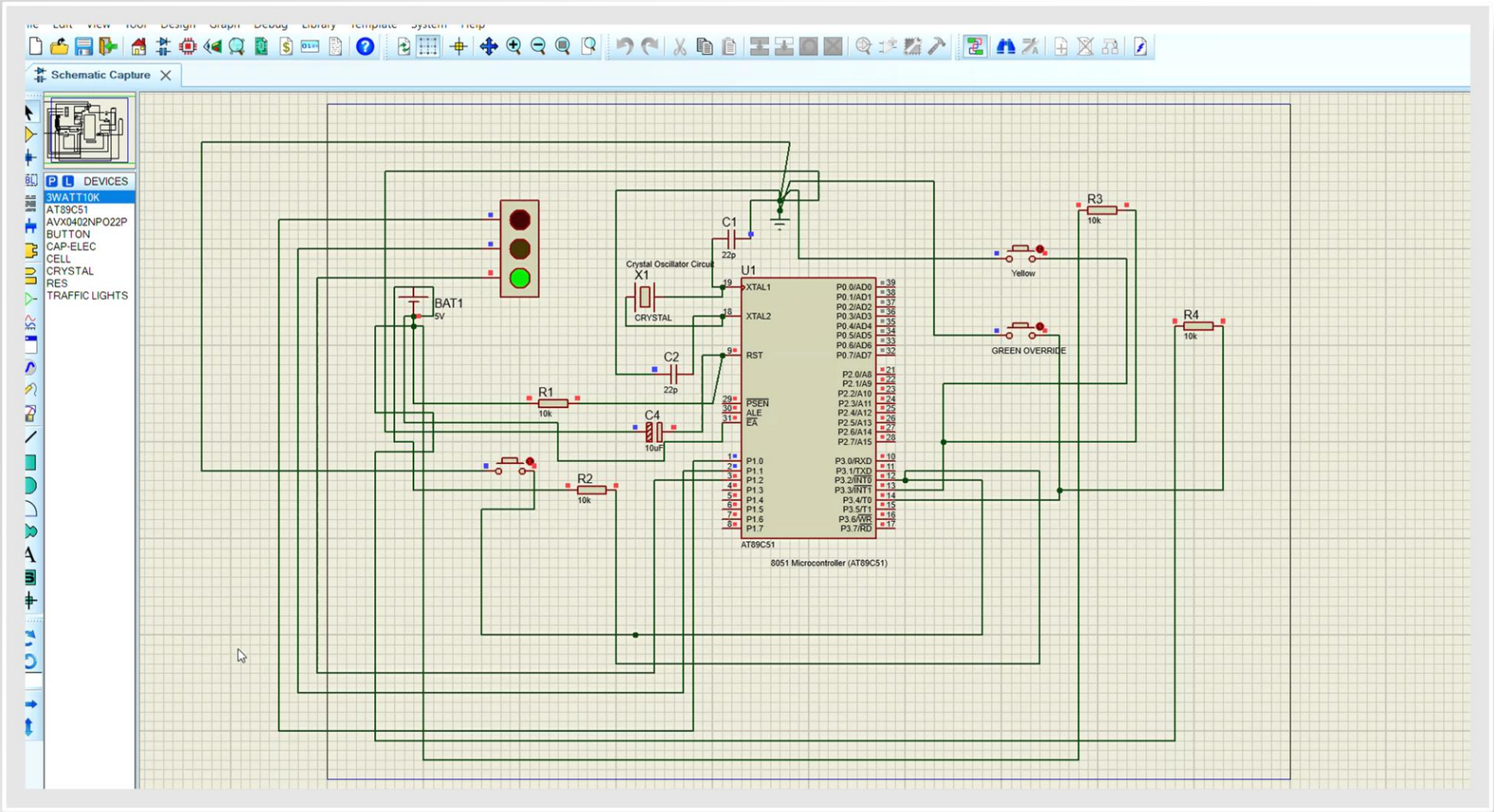
Automatic Traffic Sequence



Manual Override



Pedestrian Control(Interrupt)



Thank You