Project 1: ATmega32 LED Blinker with push button

Introduction

This project demonstrates the design and implementation of a system using the ATmega32 microcontroller. The system uses a push button as input and an LED as an output.

Objective

The main objective is to make the LED blink with a 500ms on/off interval as long as the button is pressed. We are testing with two time controllers:

1. ATmega32's internal timers
2. External 8MHz crystal oscillator

Components used:

1. Breadboard
2. Atmel ICE programmer
3. ATmega32 microcontroller
4. 9V Battery
5. Voltage regulator
6. 0.1uf Capacitor
7. Red LED light
8. Push button
9. 10kΩ Resistor
10. 100Ω Resistor
11. 8MHz External clock crystal
12. Jumper wires
13. USB wire
14. Laptop

Methodology

Micro controller: ATmega32
Input: Push button to Port B, Pin 1
Output: LED connected to Port B, Pin 0
Clock Source: Tested with the internal 1MHz then, connected 8MHz crystal
Power Supply: 9V battery

The power is converted to 5V through capacitor and voltage setup. This now provides 5V power through the breadboard.

Process

1. We started with the initial setup of getting power supplied to the whole breadboard based on the slides provided.

2. Next we moved on to check if the power supply has been provided with a connected LED and check if it lit up.
3. After that, we started setting up other hardware components. At first, we connected the microcontroller.
4. Then connected the LED and push button to the designated ports mentioned above.
5. Next, we connected the ISP to the microcontroller and used the jumper wires to connect the VCC and GND to power rail and ground rail respectively.
6. Before writing the actual code for code, we tested with code provided to check any connectivity issues and if the MICROCHIP studio was able to read the programmer.
7. Once we figured out the complete setup, we moved on to the code.

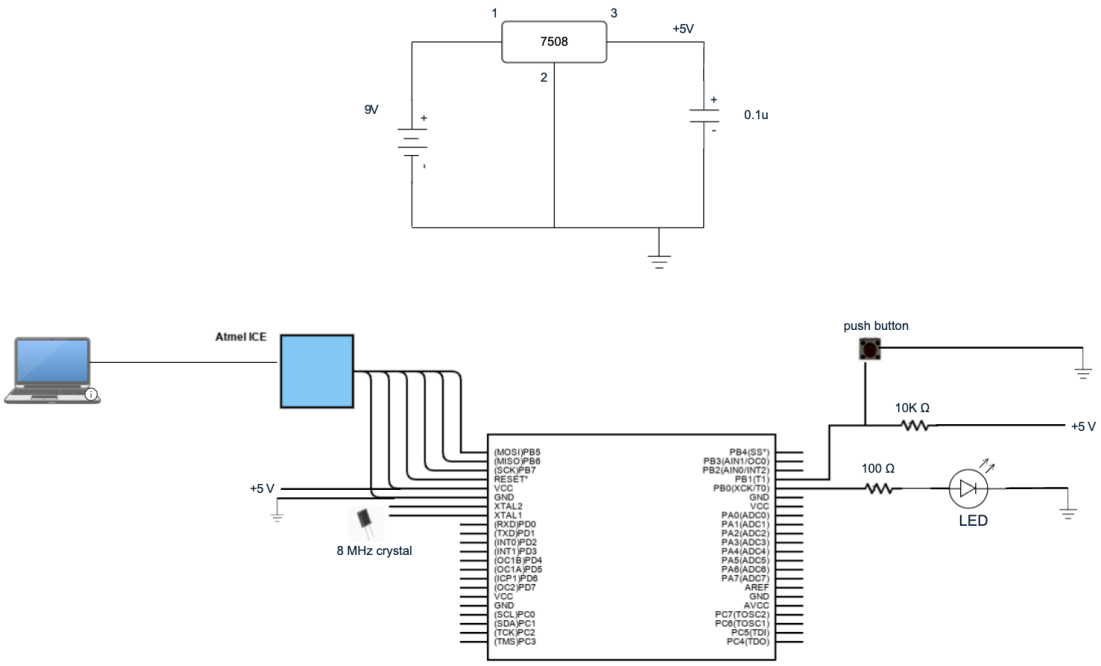
Code [all the files have been attached]

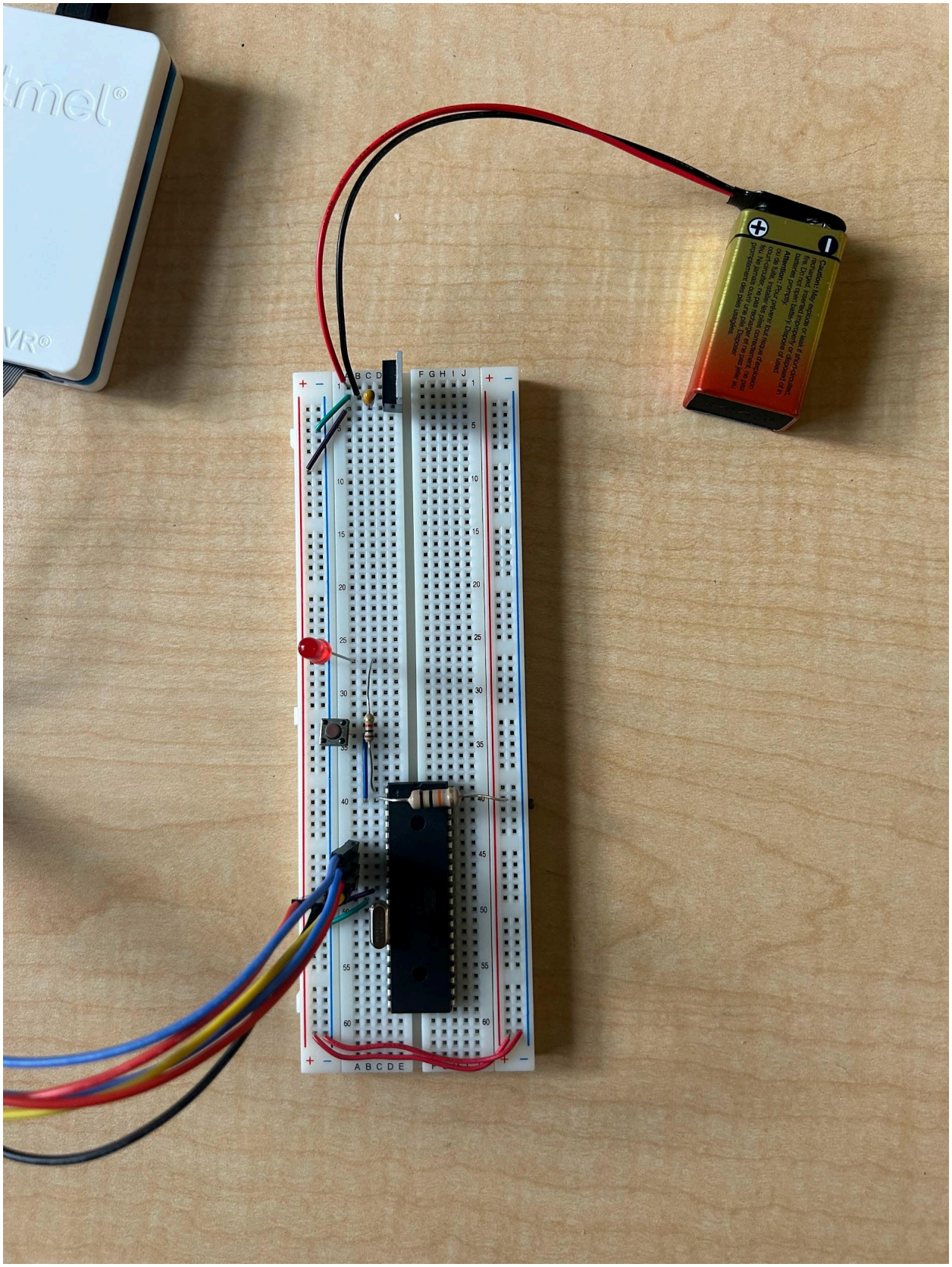
The Main function in main.c file

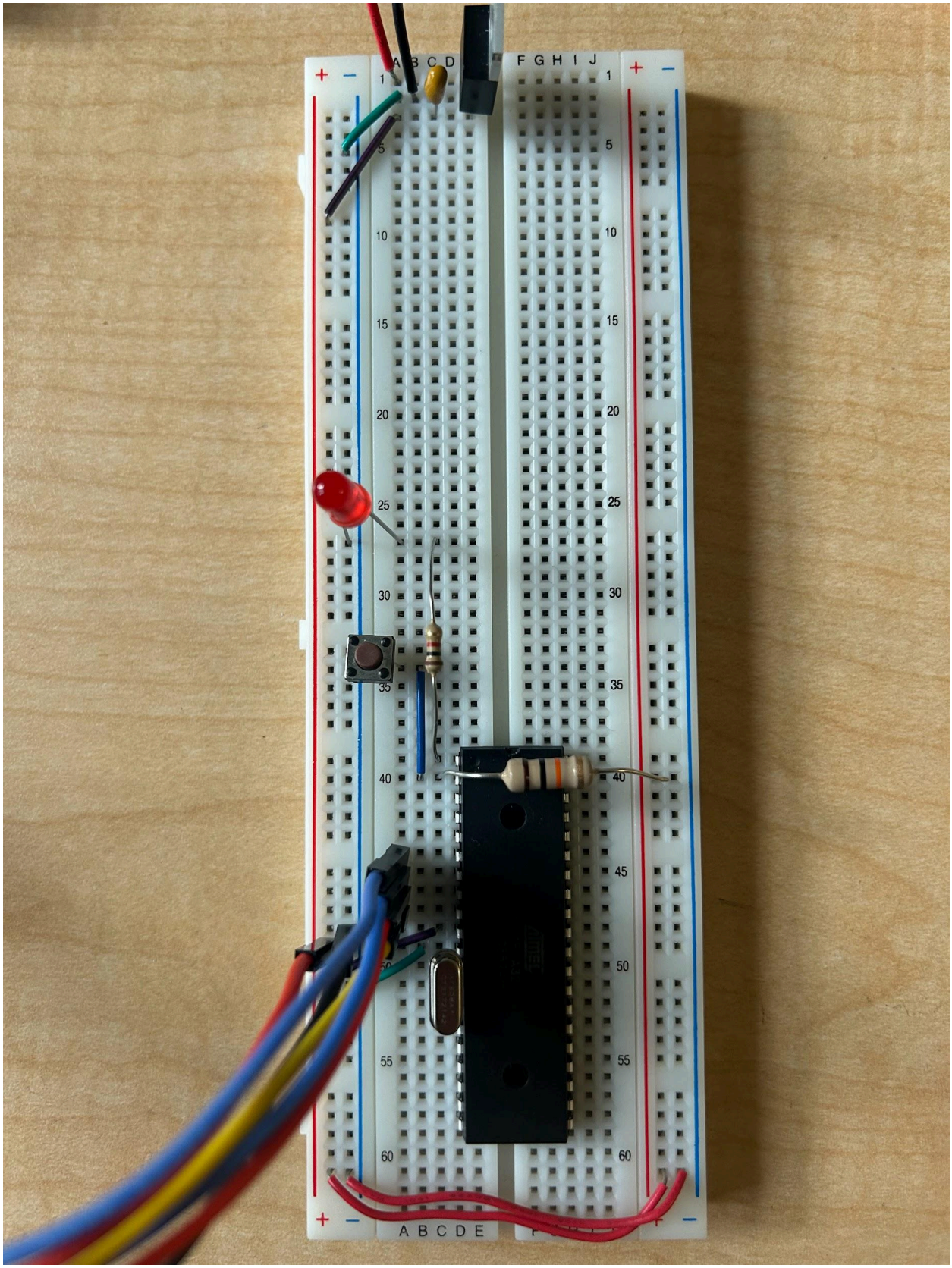
1. PB0 set as output
2. PB1 set as an input with pull-up
3. Main loop:
    a. We continuously check if the button is pressed
    b. If pressed, the LED is turned on for 500ms, then off for 500ms, using the avr_wait() function provided to us in the avr.c file
    c. If not pressed, the LED remains off.
4. For configurations, we set the external clock under the fuse. So, when we do the demo and remove the external clock, the LED doesn't turn on.

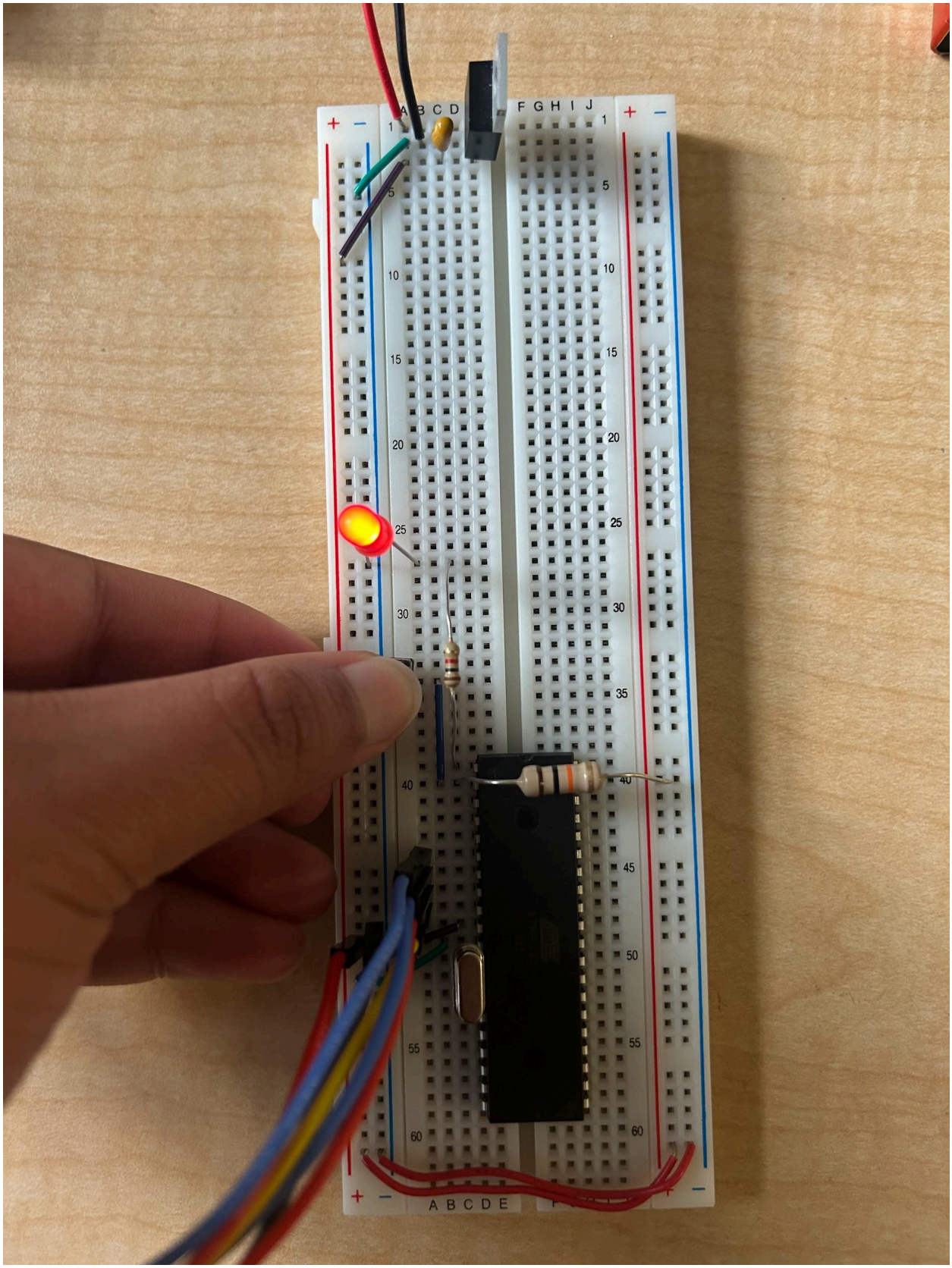
Diagram & Picture

Bread broad




A B C D E
F G H I J




A B C D
F G H I J
1
5
10
15
20
25
30
35
40
45
50
55
60
A B C D E