Hardware Project - Digital Clock

P. Shiny Diavajna
EE24BTCH11058

March 24, 2025

1 Introduction

This report details the design and implementation of a digital clock using the ATmega328p microcontroller (in Arduino) and six seven-segment displays. Timer1 interrupts are utilized for updating time, and multiplexing is used to control multiple seven-segment displays efficiently. The initial time can be manually set in the code.

2 Components Used

- Arduino UNO
- Six seven-segment displays
- Resistors
- Jumper wires
- Breadboard
- Power supply

3 Circuit Design

- Place the six seven-segment displays on the breadboard. The first two are used to display the two digits of hours, the next two for minutes, and the next two for seconds.
- Connect the 5V and GND pins from the Arduino to the power rails of the breadboard.
- Connect the COM of each display to the 5V rail through resistors.
- Connect the dot of each display to GND.

- Make connections from the Arduino pins to the seven-segment displays as per the table below.

Arduino Pin	Seven- Segment Display
2	a
3	b
4	c
5	d
6	e
7	f
8	g
9	COM of 1st Hours Display
10	COM of 2nd Hours Display
11	COM of 1st Minutes Display
12	COM of 2nd Minutes Display
A0	COM of 1st Seconds Display
A1	COM of 2nd Seconds Display

Table 1: Connections between Arduino pins and seven-segment displays

4 Code

```
// setting cpu frequency to 16 mega hz (for atmega328p)
// atmega328p microcontroller is in the arduino
// frequency is needed for timing calculations
#define F_CPU 16000000UL

// including required libraries
// they provide access to hardware registers, interrupts, delays
#include <avr/io.h> // standard input-output functions for avr
#include <avr/interrupt.h> // interrupt handling
#include <util/delay.h> // delay functions

// array to store bit patterns for each digit
// low (0) turns on the led segment
const uint8_t digit_map[] = {
    0b00000000,  // 0
    0b11100100,  // 1
    0b10010000,  // 2
    0b11000000,  // 3
    0b01100100,  // 4
    0b01001000,  // 5
    0b00001000,  // 6
```

```
    0b11100000,  // 7
    0b00000000,  // 8
    0b01000000   // 9
};

// defining time variables
volatile uint8_t hours = 5, minutes = 11, seconds = 10;

// array to store display digits
uint8_t digits[6];
// function to calculate digit values for display
void update_digits() {
    digits[0] = hours / 10;
    digits[1] = hours % 10;
    digits[2] = minutes / 10;
    digits[3] = minutes % 10;
    digits[4] = seconds / 10;
    digits[5] = seconds % 10;
}

// function to update time at every second
void update_time() {
    seconds++;
    if (seconds >= 60) { seconds = 0; minutes++; }
    if (minutes >= 60) { minutes = 0; hours++; }
    if (hours >= 24) { hours = 0; }
    update_digits();
}

// interrupt service routine (ISR) for Timer1 compare match A
ISR(TIMER1_COMPA_vect) {
    update_time();
}

// function to display a single digit (multiplexing)
void display_digit(uint8_t display, uint8_t digit) {
    PORTB &= ~(0b00011110);
    PORTC &= ~(0b00000011);
    PORTD = digit_map[digit];
    if (digit == 0 || digit == 1 || digit == 7) {
        PORTB |= (1 << PB0);
    } else {
        PORTB &= ~(1 << PB0);
    }
    if (display < 4) {
        PORTB |= (1 << (display + 1));
```

```
    } else {
        PORTC |= (1 << (display - 4));
    }
    _delay_ms(2);
}

int main(void) {
    DDRD |= 0b11111100;
    DDRB |= (1 << PB0);
    DDRB |= (1 << PB1) | (1 << PB2) | (1 << PB3) | (1 << PB4);
    DDRC |= (1 << PC0) | (1 << PC1);
    update_digits();
    TCCR1B |= (1 << WGM12) | (1 << CS12) | (1 << CS10);
    OCR1A = 15625;
    TIMSK1 |= (1 << OCIE1A);
    sei();
    while (1) {
        for (uint8_t i = 0; i < 6; i++) {
            display_digit(i, digits[i]);
        }
    }
}
```

5 Results

The clock successfully displays the time in the HH:MM:SS format, with precise timing and a clear display, which can be attributed to the usage of multiplexing and Timer1 interrupts.

6 Conclusion

This project demonstrates the use of AVR-GCC and microcontroller timers for implementing a digital clock. The current implementation requires manual time setting during initialization.

Acknowlwdgement : Code sourced from M.Srujana EE24BTECH11042