

# 24-Hour Clock built in Arduino Uno using AVR GCC

EE24BTECH11002 - Agamjot Singh

March 24, 2025

## Features

- The code is written in AVR GCC, which is not as low level as assembly but still gives decent amount of low level control.
- Use of one decoder for each 7-segment display was avoided (only one decoder was used).

## Components and Circuit Schematic

Quantity	Component
6	Seven Segment Display
1	Arduino Uno
-	Wires
1	4-Bit Decoder (7447)

Table 1: Materials Required

- Power pins of seven segment are connected to Digital Pins of the Arduino
- Data pins of all the seven segment displays are connected to the output data pins of the decoder

The schematic for connections is as shown below,

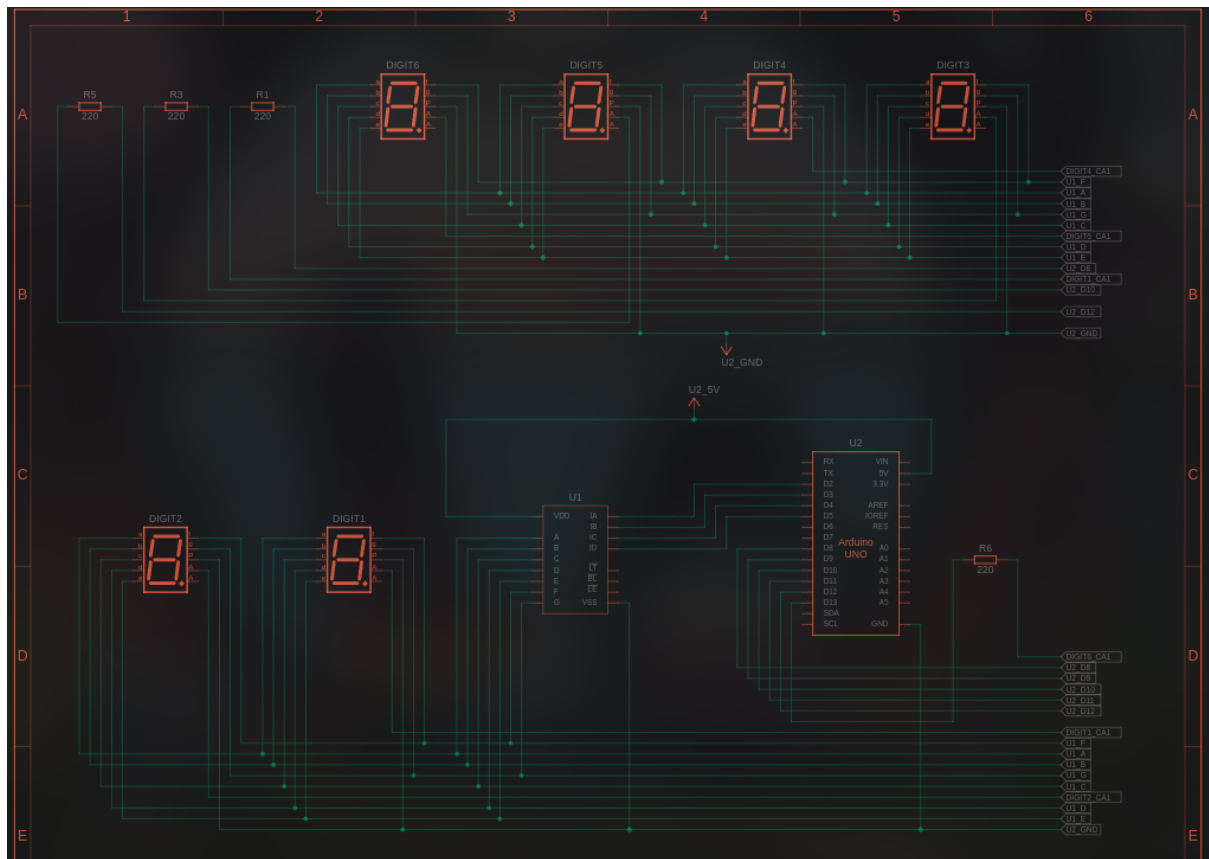


Figure 1: Schematic of Circuit.

## Code implementation and Software Multiplexing

Software Multiplexing was employed to intelligently achieve the working of 6 Seven segment displays using only one decoder.

- Power pins of seven segment are connected to Digital Pins of the Arduino
- Data pins of all the seven segment displays are connected to the output data pins of the decoder
- We exploit the fact that the human eye frame rate limitations and cycle the powers of the seven segment displays at a very high rate (about 1 ms between two digital writes) and we write to the bcd at the same time.

This multiplexing achieves the same effect as all the seven segment displays displaying the time.

## Timer Initialization and Interrupt Handling

The clock relies on precise timing through Timer1 and interrupt service routines:

```
void init_timer() {
    // Set CTC mode (Clear Timer on Compare Match)
```

```

TCCR1B |= (1 << WGM12);

// Set prescaler to 1024
TCCR1B |= (1 << CS12) | (1 << CS10);

// Set compare match value for 1s intervals (16MHz/1024)
OCR1A = 15624;

// Enable Timer1 compare match interrupt
TIMSK1 |= (1 << OCIE1A);

// Enable global interrupts
sei();
}

// Interrupt Service Routine triggered every second
ISR(TIMER1_COMPA_vect) {
    flag = 1; // Set flag to update clock values
}

```

The timer operates in CTC mode with a 1024 prescaler, generating an interrupt every second. When the interrupt occurs, the ISR sets a flag that triggers the time update in the main loop, ensuring accurate timekeeping without blocking program execution.

This uses the Arduino's internal clock to achieve very minimal time losses.

## Conclusion

This project has demonstrated my attempt to implement a fully functional timer using 7 segment displays, a decoder, and an AtMega328p microcontroller (arduino uno) implemented using AVR GCC.