LAB-REPORT-1

M. SRUJANA

March 24, 2025

1 Aim :

CLOCK USING AVR.GCC WITHOUT IC'S

2 Installation :

1. In termux: apt install avr-gcc avr-binutils avr-libc avrdude make -y

3 Apparatus :

1. Breadboard
2. 6 Seven-Segment Displays
3. Jumper wires
4. Arduino
5. Arduino USB Cable

4 Connections :

Make the connections as mentioned below:

1. These connections are for COMMON ANODE.
2. Connect the common anode pin of each 7-segment display to +5V through a 220Ω resistor.
3. Connect each segment pin (A to G) of the displays to the corresponding output pins of the Arduino.
4. The display will be multiplexed by turning on one digit at a time while updating the corresponding segments.

SEVENSEG	PINS	RESISTOR USAGE
A	2	NO
B	3	NO
C	4	NO
D	5	NO
E	6	NO
F	7	NO
G	8	NO
DOT	GND	NO

Table 1: Seven-Segment Display Pin Mapping and Resistor Usage

7-Segment Display COM'S	Common Pin	Resistor (Ohms)
H1 (Hours Tens)	Pin 9	220Ω
H2 (Hours Ones)	Pin 10	220Ω
M1 (Minutes Tens)	Pin 11	220Ω
M2 (Minutes Ones)	Pin 12	220Ω
S1 (Seconds Tens)	Pin 14 (A0)	220Ω
S2 (Seconds Ones)	Pin 15 (A1)	220Ω

Table 2: Common anode connections and resistors for each COM

5 Code :

```
#define F_CPU 16000000UL
#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/delay.h>

const uint8_t digit_map[] = {
    0b00000000, 0b11100100, 0b10010000, 0b11000000, 0b01100100,
    0b01001000, 0b00001000, 0b11100000, 0b00000000, 0b01000000
};

volatile uint8_t hours = 1, minutes = 11, seconds = 30;
uint8_t digits[6];

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;
}

void update_time() {
    seconds++;
    if (seconds >= 60) { seconds = 0; minutes++; }
    if (minutes >= 60) { minutes = 0; hours++; }
    if (hours >= 24) { hours = 0; }
    update_digits();
}

ISR(TIMER1_COMPA_vect) {
    update_time();
}

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]);
        }
    }
}
```

6 Execution :

1. To compile main.c file : avr-gcc -mmcu=atmega328p -Os -o main.elf main.c
2. To convert main.elf file to main.hex : avr-objcopy -O ihex -R .eeprom main.elf main.hex
3. Moving the Compiled File to ArduinoDroid : mv main.hex /sdcard/ArduinoDroid/precompiled