Clock

EE24BTECH11005: Credits EE24BTECH11001, EE24BTECH11002

March 24, 2025

Introduction

This project showcases an innovative clock design implemented on the AtMega328p microcontroller using the avr-gcc framework. The primary focus was on achieving maximum time accuracy. To optimize performance, time incrementation was implemented using Karnaugh maps (K-maps) instead of conventional increment commands. The entire codebase was meticulously crafted in AVR Assembly language, allowing precise control over instruction execution times and enabling fine-tuned timing adjustments.

Overview

Features

The main features of the calculator are,

- The entire code was written in AVR Assembly so it is very $bare-bones$ so more focus was placed on making the clock accurate than adding more and more functionality.
- Instead of using **inc** command, k-maps were used.
- The use of one decoder for each 7-segment display was avoided (only one decoder was used).

Hardware Overview

The hardware consists of:

Quantity	**Component**
6	7 Segment Display
1	Arduino Uno
-	Wires
1	Decoder (7447)

Table 1: Materials Required

- Decrease requirement of multiple decoders, by intelligentally connecting the circuit as shown in the schematic Figure **??**.

The schematic for connections is as shown below,

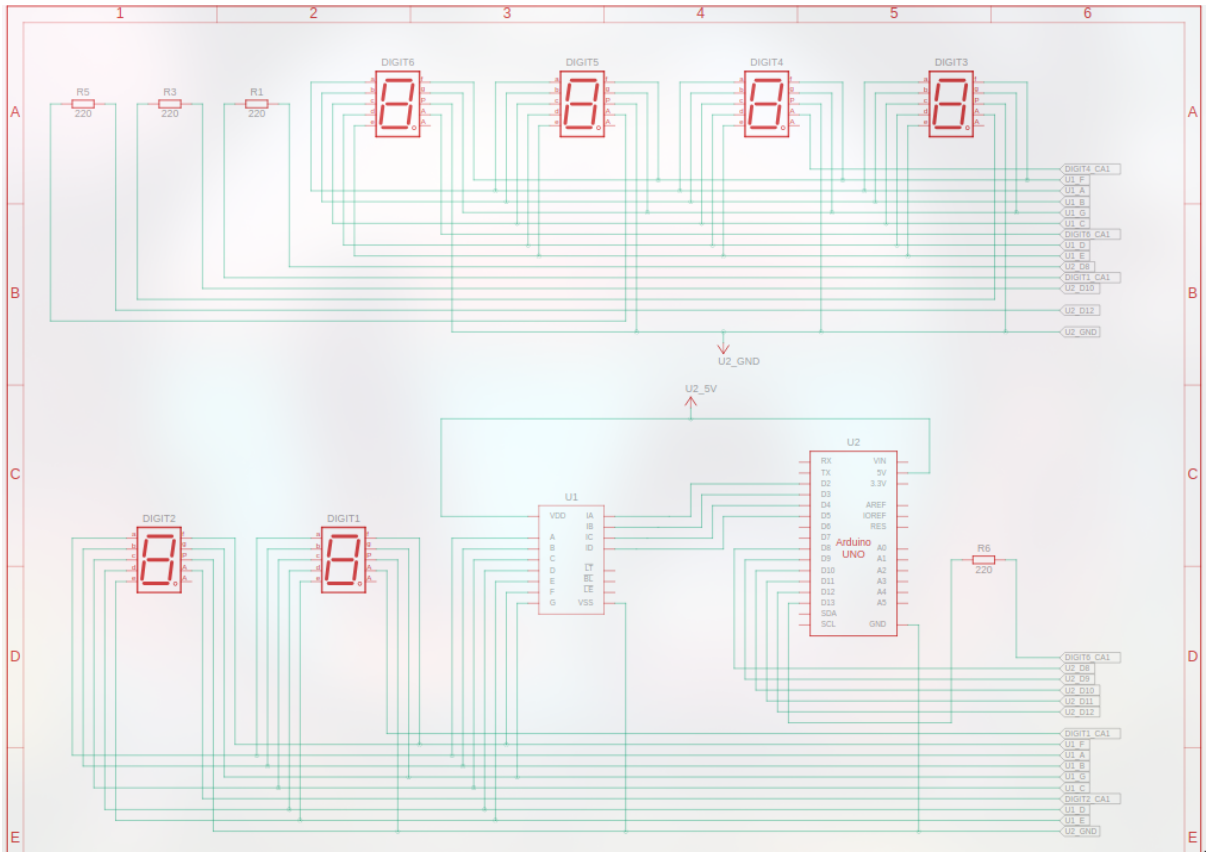

Figure 1: Schematic of Circuit.

Software Overview

The software is implemented using AVR Assembly. It includes:

- Using k-maps for implementing
- Timer was implemented through repeated looping
- Registers were intelligently used so that code was implemented using only the 16 provided registers (r15, r30)

Now let us explore some of the features mentioned above in greater depth.

In Depth

Karnaugh Maps

K-map 1

This is for one's digit of seconds, minutes,

WXYZ	ABCD
0000	0001
0001	0010
0010	0011
0011	0100
0100	0101
0101	0110
0110	0111
0111	1000
1000	1001
1001	1010

These are the K-maps for A, B, C, D

$$A = \overline{W}$$
$$B = (\overline{W}\&X\&\overline{Z})||(W\&\overline{X}\&\overline{Z})$$
$$C = (\overline{W}\&Y\&\overline{Z})||(\overline{X}\&Y\&\overline{Z})||(W\&X\&\overline{Y}\&\overline{Z})$$
$$D = (W\&X\&Y\&\overline{Z})||(\overline{W}\&\overline{X}\&\overline{Y}\&Z)$$

K-map 2

This is for ten's digit of minutes, seconds,

WXYZ	ABCD
0000	0001
0001	0010
0010	0011
0011	0100
0100	0101
0101	0000
0011	0100
0100	0101
0101	0000

These are the K-maps for A, B, C, D

$$A = 0$$
$$B = (\overline{X}\&Y\&Z)||(X\&\overline{Y}\&\overline{Z})$$
$$C = (\overline{X}\&\overline{Y}\&Z)||(\overline{X}\&Y\&\overline{Z})$$
$$D = (\overline{X}\&Y\&\overline{Z})||(\overline{X}\&\overline{Y}\&\overline{Z})$$

K-map 3

This is for one's digits of hours,

WXYZ	ABCD
0000	0001
0001	0010
0010	0000

These are the K-maps for A, B, C, D

$$A = 0$$
$$B = 0$$
$$C = (\overline{Y}\&Z)$$
$$D = (\overline{Y}\&\overline{Z})$$

K-map 4

This is for ten's digits of hours,

WXYZ	ABCD
0000	0001
0001	0000

These are the K-maps for A, B, C, D

$$A = 0$$
$$B = 0$$
$$C = 0$$
$$D = \overline{Z}$$

Timer

Using multiple loops, a timer of one second was created. Within that one second, time is displayed (using the logic mentioned above), and at the end of one second, one's digit is incremented. The sub routine that does this has a piece of code at the end to increment ten's digit of seconds if incremented value of one's digit is 0. Similar the function to increment ten's digit has a line of code to increment one's digit of minutes if value after incrementing is 0. This is how time is incremented

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 Assembly and Karnaugh Maps.