FORMAN CHRISTIAN COLLEGE

(A CHARTERED UNIVERSITY)

CSCS 306 - A

FALL 24

4-Digit Stopwatch using Arduino

Assignment 1 Report

Hafsah Shahbaz – 251684784

Daim Bin Khalid - 251686775

Syeda Manal Ammad - 251606966

14th October 6, 2024

Table of Contents

Introduction

This project involves the design and implementation of a 4-digit stopwatch using Arduino. The stopwatch counts minutes and seconds on a four-digit, multiplexed 7-segment display. Users can input minutes and seconds using a keypad, which are then displayed on the stopwatch. Once started, the stopwatch counts down and resets to zero when the time runs out. When the countdown reaches zero, the display shows **FCCU** and blinks it four times before resetting. The stopwatch can also be started to count-up if started at 0. The counter can be paused and reset as well.

This project highlights skills such as interfacing Arduino with external hardware components like 7-segment displays, push buttons, and keypads. The challenge of the project is to utilize limited Arduino pins ideally while using a shift register to counter this problem. The code logic also emphasizes real-time counting using millis(), ensuring accurate timekeeping.

Components Used

1. Arduino Uno R3

2. 1x 7-Segment Anode Display (4 Digits)

3. 1x Shift Register (74HC595)

4. Keypad (4x4)

5. 4x 330 Ohm Resistors

6. Jumper Wires and Breadboard

Circuit Diagram (TinkerCAD)

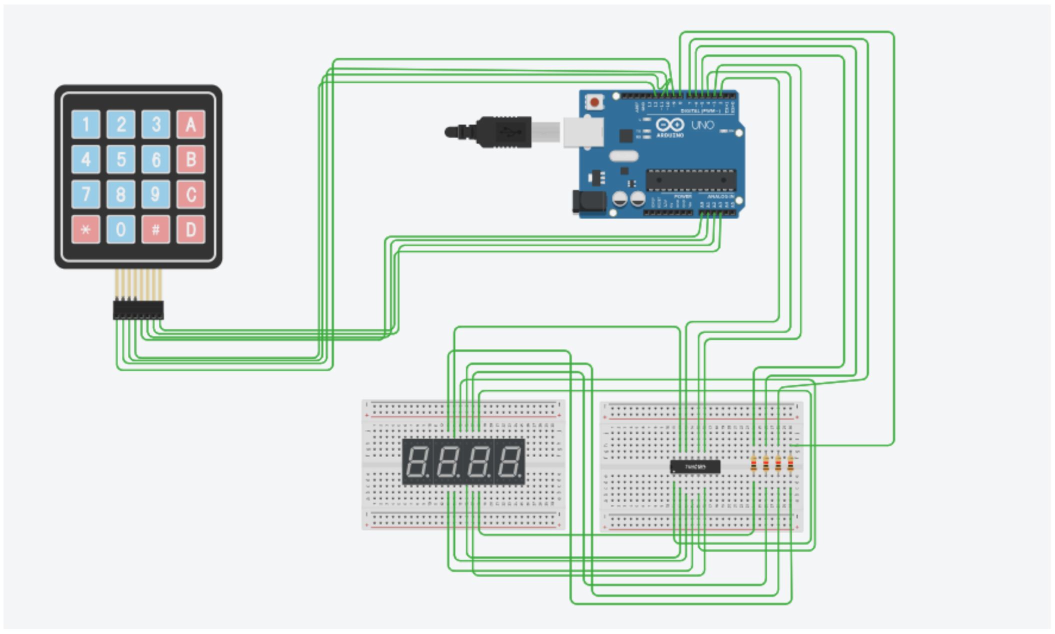
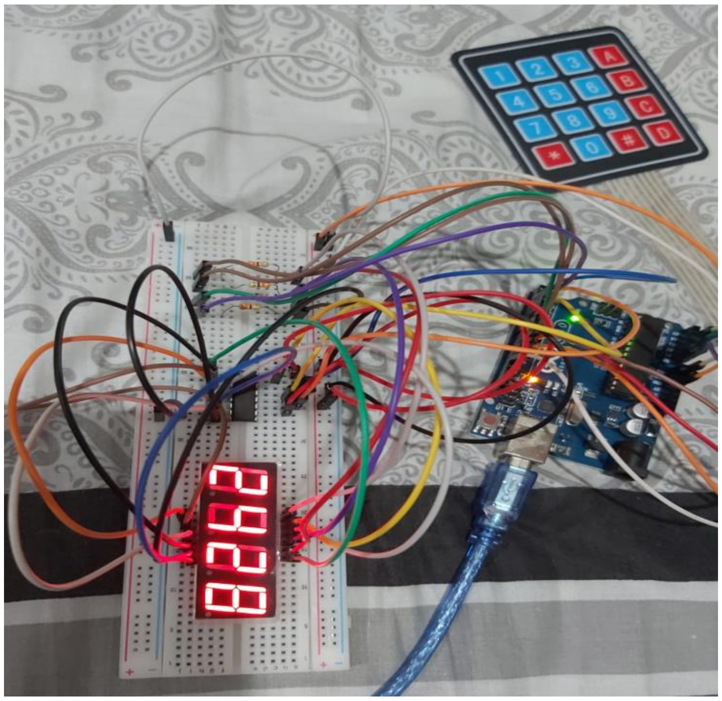


Image of the Working Hardware



Functionality and Code Breakdown

1. Stopwatch Setup and Pin Configuration

Summary: The setup initializes pins and configures the 4-digit 7-segment display for use with a shift register.

```
const int dataPin = 2;     // DS (Data Pin)

const int latchPin = 4;    // ST_CP (Latch Pin)

const int clockPin = 3;    // SH_CP (Clock Pin)
```

dataPin, latchPin, clockPin: Define the pins used for controlling the 74HC595 shift register that drives the 7-segment display.

```
pinMode(DIGIT_1_PIN, OUTPUT);

pinMode(DIGIT_2_PIN, OUTPUT);

pinMode(DIGIT_3_PIN, OUTPUT);

pinMode(DIGIT_4_PIN, OUTPUT);
```

DIGIT_X_PIN: Defines the pins connected to each of the 4 digits of the 7-segment display.

2. Handling Keypad Inputs

Summary: The code captures keypad inputs to set minutes/seconds, start/stop the stopwatch, and reset the timer.

```
switch (key) {

  case 'A': // Edit seconds

  case 'B': // Edit minutes

  case 'C': // Start/Stop stopwatch

  case 'D': // Reset stopwatch
```

Keypad Input Handling: Detects which key has been pressed ('A' to edit seconds, 'B' to edit minutes, 'C' to start/stop the stopwatch, and 'D' to reset).

3. Time Modification Logic

Summary: When the user presses a number key after selecting "Edit Minutes" or "Edit Seconds," the value is updated.

```
if (editingSeconds && key >= '0' && key <= '9') {
  int value = key - '0';  // Convert char to int
  seconds = (seconds * 10 + value) % 60; // Limit to 0-59
}
```

Editing Seconds/Minutes: Allows the user to set the time by converting the character input from the keypad into an integer and updating the time.

4. Countdown and Count-up Timer Logic

Summary: The stopwatch can operate in countdown mode (default) or count-up mode (if time is 00:00).

```
if (countingUp) {
  seconds++;
  if (seconds > 59) { seconds = 0; minutes++; }
} else {
  seconds--;
  if (seconds < 0) { seconds = 59; minutes--; }
}
```

Count-Up Logic: If countingUp is true, the seconds increment and the stopwatch behaves as a count-up timer.

Countdown Logic: If countingUp is false, the seconds decrement, and the stopwatch counts down.

5. Displaying Time on the 7-Segment Display

Summary: Time (minutes and seconds) is continuously updated and displayed using multiplexing on the 4 digits of the display.

```
void displayTime() {
  digitalWrite(DIGIT_1_PIN, HIGH); // Show ones of seconds
  showDigit(digitCode[seconds % 10]);
  digitalWrite(DIGIT_1_PIN, LOW);
  digitalWrite(DIGIT_2_PIN, HIGH); // Show tens of seconds
  showDigit(digitCode[seconds / 10]);
  digitalWrite(DIGIT_2_PIN, LOW);

  digitalWrite(DIGIT_3_PIN, HIGH); // Show ones of minutes
  showDigit(digitCode[minutes % 10]);
  digitalWrite(DIGIT_3_PIN, LOW);

  digitalWrite(DIGIT_4_PIN, HIGH); // Show tens of minutes
  showDigit(digitCode[minutes / 10]);
  digitalWrite(DIGIT_4_PIN, LOW);
}
```

Multiplexing Digits: Each digit is displayed one by one using multiplexing, where only one digit is active at a time.

showDigit(): Sends the appropriate digit pattern to the shift register based on the current time.

6. Displaying "FCCU"

Summary: When the countdown reaches 00:00, the display blinks "FCCU" four times.

```
void displayFCCU() {
  for (int blinkCount = 0; blinkCount < 4; blinkCount++) {  //
Blink 4 times
    // Display "FCCU" for 500ms
    for (int i = 0; i < 15; i++) {  // Refresh display for 500ms
(5ms * 100 = 500ms)
      // Display F
      showDigit(FCCU[0]);
      digitalWrite(DIGIT_4_PIN, HIGH);
      delay(5);
      digitalWrite(DIGIT_4_PIN, LOW);
```

```
        // Display C (first C)
        showDigit(FCCU[1]);
        digitalWrite(DIGIT_3_PIN, HIGH);
        delay(5);
        digitalWrite(DIGIT_3_PIN, LOW);

        // Display C (second C)
        showDigit(FCCU[1]);
        digitalWrite(DIGIT_2_PIN, HIGH);
        delay(5);
        digitalWrite(DIGIT_2_PIN, LOW);

        // Display U
        showDigit(FCCU[2]);
        digitalWrite(DIGIT_1_PIN, HIGH);
        delay(5);
        digitalWrite(DIGIT_1_PIN, LOW);
      }

      // Turn off the display (clear) for 500ms
      clearDisplay();
      delay(500);
    }
  }
```

FCCU Blinking: The letters 'F', 'C', 'C', and 'U' are displayed using the predefined segment codes in the chars[] array. The display blinks 4 times before the stopwatch is reset.

7. Reset Functionality

Summary: Resets the stopwatch to 00:00 and stops it from running or counting up.

```
void resetStopwatch() {

  minutes = 0;

  seconds = 0;

  running = false;

  countingUp = false;

  displayTime();

}
```

Reset Stopwatch: Resets the minutes, seconds, running, and countingUp variables, stopping the timer and displaying "00:00".

8. Clear Display

Summary: Turns off all segments of the display for a short period to achieve the blink effect.

```
void clearDisplay() {

  shiftOut(dataPin, clockPin, MSBFIRST, 0xFF); // Clear digit
(turn off segments)

}
```

9. showDigit() Function

Summary: Sends the appropriate binary pattern to the shift register to display a specific digit.

```
void showDigit(byte digitValue) {

  digitalWrite(latchPin, LOW);

  shiftOut(dataPin, clockPin, MSBFIRST, digitValue); // Send
digit pattern

  digitalWrite(latchPin, HIGH);

}
```

showDigit(): Uses shiftOut to transmit the digit pattern (e.g., for numbers 0-9 or letters like 'F', 'C', 'U') to the display.

References

1. 74HC595N Shift Register Datasheet: https://datasheet.octopart.com/74HC595N-Philips-datasheet-7085704.pdf

2. Keypad Library Documentation: Keypad | Arduino Documentation

3. 4 Digit 7-Segment Display Tutorial: https://youtu.be/3m4jhmafg8E?si=kEl6X-NRuPXs0exU