Digital Clock

Pothuri Rahul - EE24BTECH11050

1 Introduction

This project presents a digital clock using an Arduino Uno, a 7447 BCD to 7-segment decoder IC, and six common cathode 7-segment displays. The clock is designed to display time in HH:MM:SS format. It utilizes multiplexing to control all displays using a single 7447 IC. The Arduino manages time updates and ensures proper digit transitions. The clock runs in real-time and resets after 24 hours. This project demonstrates the practical implementation of BCD to 7-segment conversion and multiplexing in digital electronics.

2 Hardware things

2.1 Components used :

- Breadboard
- 744 IC (BCD to 7 segment converter)
- Six 7 segment display
- Arduino board
- Connecting cables and wires
- Mobile

2.2 Circuit design

2.2.1 7447 to Arduino

These are the following connections I made between 7447 IC and Arduino

7447	**ARDUINO**
A (pin 7)	2
B (pin 1)	3
C (pin 2)	4
D (pin 6)	5
V_{cc} (pin 16)	5 V
GND (pin 8)	GND

2.2.2 7447 to 7 segment display

Following connections are to be done between the 7447 and all six 7 - segment displays.

7447	7 segment display
a (pin 13)	a
b (pin 12)	b
c (pin 11)	c
d (pin 10)	d
e (pin 9)	e
f (pin 15)	f
g (pin 14)	g

Actually we use six 7 segment displays to get HH:MM:SS format. let us consider H_1 display is tens of hours , H_2 display is ones of hours , M_1 display is tens of minutes , M_2 display is ones of minutes , S_1 display is tens of seconds and S_2 display is ones of seconds.

We use the multiplexing technique to control six displays with a single 7447 IC.For that let us connect the com 7 segment displays to Arduino and GND pins of 7 segment displays to ground .

Display	Pin	Arduino pin
S_2	COM	6
S_1	COM	7
M_2	COM	8
M_1	COM	9
H_2	COM	10
H_1	COM	11

3 Software things

3.1 Code used

The cpp code used is the following

```
// 7447 BCD input pins
const int A = 2, B = 3, C = 4, D = 5;
// Display common pins
const int digits[] = {11, 10, 9, 8, 7, 6};

int hours = 17, minutes = 7, seconds = 5;
unsigned long prevMillis = 0;
const int interval = 1000;  // Update time every 1 second

void setup() {
    Serial.begin(9600);

    // Set up 7447 BCD pins as output
```

```
    for (int i = A; i <= D; i++) {
        pinMode(i, OUTPUT);
        digitalWrite(i, LOW);
    }

    // Set up display common pins as output
    for (int i = 0; i < 6; i++) {
        pinMode(digits[i], OUTPUT);
        digitalWrite(digits[i], HIGH);  // Start with all displays OFF
    }

    Serial.println("Clock Started!");
}

void loop() {
    unsigned long currentMillis = millis();

    // Update time every second
    if (currentMillis - prevMillis >= interval) {
        prevMillis = currentMillis;
        updateClock();
        Serial.print("Time: ");
        Serial.print(hours); Serial.print(":");
        Serial.print(minutes); Serial.print(":");
        Serial.println(seconds);
    }

    // Refresh the display continuously
    multiplexDisplay();
}

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

void multiplexDisplay() {
    for (int i = 0; i < 6; i++) {
        int digitValue = getDigit(i);
        showDigit(digitValue, i);
    }
}

void showDigit(int num, int digitIndex) {
    turnOffDisplays();
    digitalWrite(digits[digitIndex], HIGH);  // Activate display
    set7447(num);
    delay(5);  // Small delay to allow visibility
    digitalWrite(digits[digitIndex], LOW); // Turn off display
}

```

```
73  void set7447(int num) {
74      digitalWrite(A, num & 1);
75      digitalWrite(B, (num >> 1) & 1);
76      digitalWrite(C, (num >> 2) & 1);
77      digitalWrite(D, (num >> 3) & 1);
78  }
79
80  void turnOffDisplays() {
81      for (int i = 0; i < 6; i++) {
82          digitalWrite(digits[i], LOW);  // Turn off all displays
83      }
84  }
85
86  int getDigit(int index) {
87      switch (index) {
88          case 0: return hours / 10;
89          case 1: return hours % 10;
90          case 2: return minutes / 10;
91          case 3: return minutes % 10;
92          case 4: return seconds / 10;
93          case 5: return seconds % 10;
94          default: return 0;
95      }
96  }
```

Reference : Used some AI Tools like chat-gpt for getting the code.

3.2 Explanation of the code

The given Arduino code implements a digital clock (HH:MM:SS) using a 7447 BCD to 7-segment decoder with multiplexing.

3.2.1 Key Features

- **Pin Definitions:**
  - `A, B, C, D` → BCD input pins for 7447.
  - `digits[]` → Stores the 6 display common pins.
- **Time Variables:**
  - `hours, minutes, seconds` → Stores current time.
  - `prevMillis, interval` → Tracks updates every 1 second.
- `setup()`: Initializes serial communication and sets display pins OFF.
- `loop()`:
  - Updates time every second using millis().
  - Calls `multiplexDisplay()` to refresh the 7-segment display.
- `updateClock()`: Increments seconds, minutes, and hours like a real clock.

- `multiplexDisplay()`: Switches between six digits rapidly using multiplexing.
- `showDigit(num, digitIndex)`:
  - Turns off all displays.
  - Activates the required digit.
  - Sends the corresponding BCD number to the 7447 decoder.
- `set7447(num)`: Converts the digit to BCD format for the 7447 IC.
- `turnOffDisplays()`: Turns off all displays to prevent ghosting.
- `getDigit(index)`: Extracts individual digits from HH:MM:SS.

3.2.2 How it Works

- Uses multiplexing to display six digits using one 7447 IC.
- millis() ensures time updates every second without delay(), allowing smooth display refresh.
- The method is power-efficient, simple, and can scale to a full digital clock with minimal hardware.

4 Working process

- The Arduino tracks the time (seconds, minutes, hours) using the millis() function.
- The time values are converted into BCD format and sent to the 7447 decoder IC.
- The 7447 IC drives the 7-segment displays according to the BCD input.
- Multiplexing technique is used to control six displays with a single 7447 IC.
- The display updates every second, ensuring proper timekeeping.
- The clock resets to 00:00:00 after reaching 24:00:00.

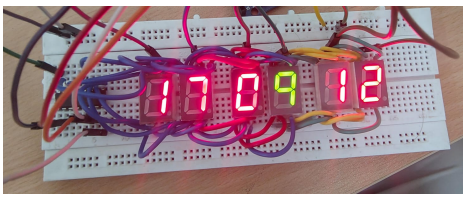

Figure 1: Clock made showing the time 17:09:12

5 Future Enhancements

- Using Real - Time Clock (RTC) would be better since we can get the real-life time even when their is no power is given to arduino.
- Adding push buttons to set buttons will be useful.
- Some gets habituated to 12 - hours formart , for them it would be useful to add another display to mention AM and PM.
- We can make an alarm using a buzzer and little changes in code.
- Instead of 7 segment display , we can also use LED display so that we can add some more features like Date, Temperature by making some changes in the code and using some other sensors to measure the temperature.

6 Conclusion

By doing this project I learned the usage of 7447 , 7 segment display and multiplexing. This project successfully implemented a fully functional digital clock using an Arduino Uno, a 7447 decoder, and six 7-segment displays. The use of multiplexing allowed efficient control of the displays with a single decoder IC .