Embedded Systems Capstone Project

AVR-Based LC Meter with Colpitts Oscillator

Mentor: Dr. Poonam Kasturi

Team Members:

Team Leader: Ram Tripathi (Roll No: 22HEL2231)
Member 1: Astha Bharadhwaj (Roll No: 22HEL2262)
Member 2: Manan Sharma (Roll No: 22HEL2250)
Member 3: Jainit Tyagi (Roll No: 22HEL2248)

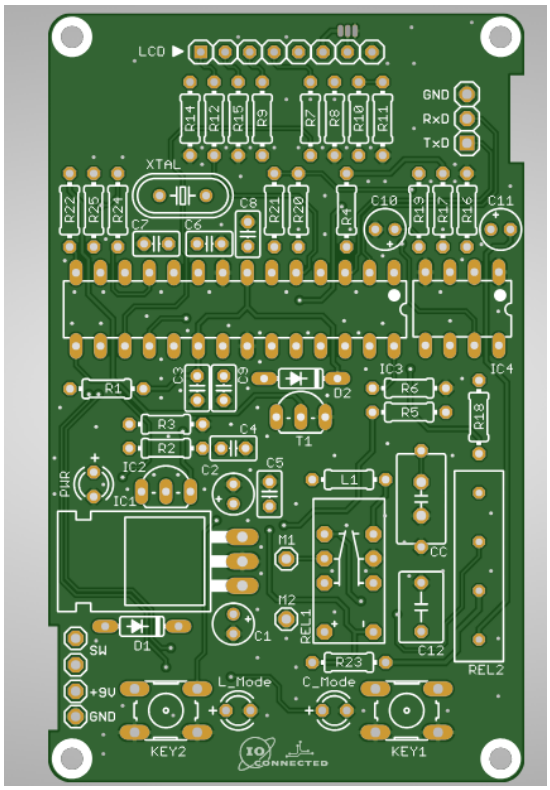

Figure 1: PCB layout in Altium Designer

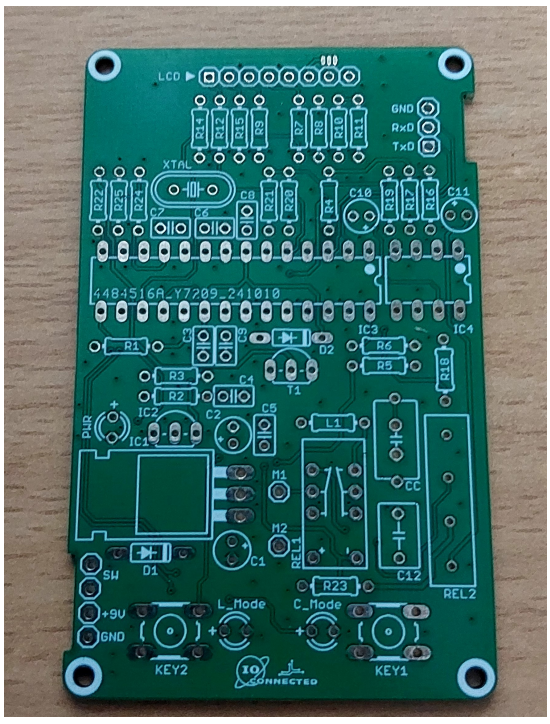

Figure 2: PCB layout after fabrication

Abstract

The AVR-Based LC Meter is a tool for measuring unknown inductance and capacitance values. At the heart of the system is a Colpitts oscillator, which provides stable oscillations to calculate these values based on frequency. This report details the design, implementation, and application of the LC meter.

Contents

1 Introduction

The LC meter is designed to measure the values of inductors and capacitors using an AVR microcontroller. By leveraging the principles of the Colpitts oscillator, the system determines unknown values based on frequency calculations.

2 Schematic Diagram

The schematic diagram has been created using Octopart library of Altium Designer.

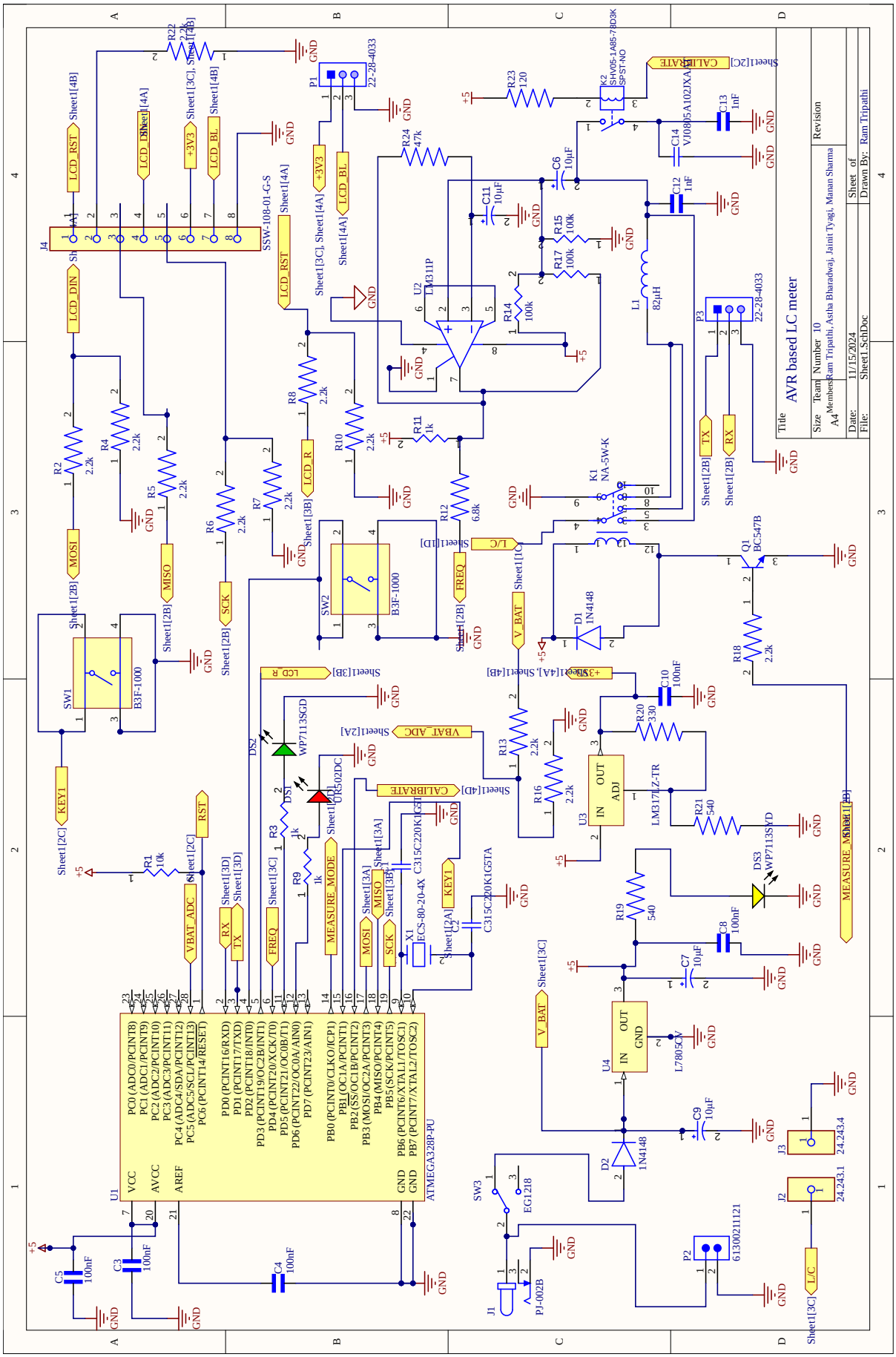

Figure 3: Schematic diagram of the AVR-Based LC Meter.

3 PCB Layout

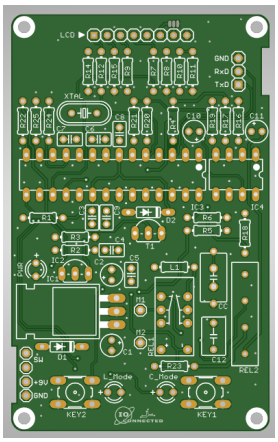

Figure 4: Front view of the PCB layout.

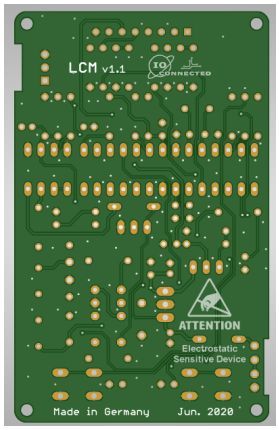

Figure 5: Back view of the PCB layout.

4 Colpitts Oscillator and Frequency Calculation

The Colpitts oscillator is a fundamental component in LC meters, generating oscillations at a frequency determined by the inductance L and equivalent capacitance C_{eq}. This frequency is given by:

$$f = \frac{1}{2\pi\sqrt{L \cdot C_{\text{eq}}}} \tag{1}$$

where:

- f is the frequency of oscillation,
- L is the inductance, and
- $C_{\text{eq}} = \frac{C_1 \cdot C_2}{C_1 + C_2}$ is the equivalent capacitance of two series capacitors C_1 and C_2.

To determine an unknown inductance or capacitance, we can rearrange this formula. For unknown inductance L:

$$L = \frac{1}{(2\pi f)^2 \cdot C_{\text{eq}}} \tag{2}$$

Similarly, for an unknown capacitance C_{eq}:

$$C_{\text{eq}} = \frac{1}{(2\pi f)^2 \cdot L} \tag{3}$$

By measuring the frequency f and knowing either L or C_{eq}, we can accurately compute the other, making the Colpitts oscillator valuable in LC meter applications.

5 Use Cases

- Measurement of inductors and capacitors in electronic circuits.
- Troubleshooting and testing passive components.
- Educational tool for understanding LC circuits and oscillators.

6 Tech Stack

- Programming Environment: Microchip Studio, AVRDUDES
- Microcontroller: AVR (e.g., ATmega328)
- Programming Language: C
- PCB Design Software: Altium Desiger
- Simulation Tools: Proteus, Keysght power analyzer In Altium

7 Code

```
/**
 * AVR LC/Frequency Meter
/

#define __AVR_ATmega328P__

#include <avr/interrupt.h>
#include <binary.h>
#include <HardwareSerial.h>
#include <pins_arduino.h>
#include <WConstants.h>
#include <wiring.h>
#include <wiring_private.h>
#include <math.h>
#include <WProgram.h>
#include <EEPROM/EEPROM.h>
#include <LiquidCrystal/LiquidCrystal.h>
#include <FreqCounter/FreqCounter.h>

enum MeterMode {
    L,
    C,
    F
};

//Frequency input is digital pin 5

const int pinRS = 3;
const int pinEn = 4;
const int pinD4 = 16;
const int pinD5 = 17;
const int pinD6 = 8;
const int pinD7 = 9;
const int pinBACKLIGHT = 10;

const int btn1 = 0;
const int btn2 = 1;
const int btn3 = 2;

const int pinBtn1 = 11;
const int pinBtn2 = 12;
const int pinBtn3 = 15;

const int pinRelay = 14;
const int pinLCMode = 2;

float F0 = 348000;
const float Cth = 1000 * 1e-12; //
    theoretical 1000pF
const float Lth = 221 * 1e-6; //
    theoretical 221uH

boolean backLightOn = false;
boolean dispFreq = false;

float l0 = Lth;
float c0 = Cth;
MeterMode currentMode;

LiquidCrystal lcd(pinRS, pinEn, pinD4,
    pinD5, pinD6, pinD7);
unsigned long frq;

const int BTN_PINS[] = {pinBtn1, pinBtn2
```

```
    , pinBtn3};
boolean btnPressed[] = {false, false,
    false};

void delayMilliseconds(int ms) {
    for (int i = 0; i < ms; i++) {
        delayMicroseconds(1000);
    }
}

boolean buttonPressed(int btnIdx) {
    if (digitalRead(BTN_PINS[btnIdx]) ==
         LOW && !btnPressed[btnIdx]) {
        delayMilliseconds(20);
        if (digitalRead(BTN_PINS[btnIdx])
             == LOW) {
            btnPressed[btnIdx] = true;
            return true;
        }
    }

    return false;
}

boolean buttonRleased(int btnIdx) {
    if (digitalRead(BTN_PINS[btnIdx]) ==
         HIGH && btnPressed[btnIdx]) {
        delayMilliseconds(20);
        if (digitalRead(BTN_PINS[btnIdx])
             == HIGH) {
            btnPressed[btnIdx] = false;
            return true;
        }
    }

    return false;
}

float calcV(float f, float VRef) {
    float v = 0;

    v = ((F0 * F0) / (f * f) - 1.0) *
        VRef;

    return v;
}

void checkLCMode() {
    lcd.clear();

    if (digitalRead(pinLCMode) == LOW) {
        currentMode = L;
        lcd.print("Mode: L");
    } else {
        currentMode = C;
        lcd.print("Mode: C");
    }

    if (dispFreq) {
        lcd.setCursor(0, 0);
        lcd.print("Frequency: ");
    }
}

void setup() {
    for (int i = 0; i < 3; i++) {
        pinMode(BTN_PINS[i], INPUT);
        digitalWrite(BTN_PINS[i], HIGH);
    }

    pinMode(pinLCMode, INPUT);
    digitalWrite(pinLCMode, HIGH);

    pinMode(pinBACKLIGHT, OUTPUT);
    digitalWrite(pinBACKLIGHT, LOW);

    pinMode(pinRelay, OUTPUT);
    digitalWrite(pinRelay, LOW);
    lcd.begin(16, 2);
    checkLCMode();
}

void displayFreq(long fin) {
    lcd.setCursor(0, 1);

    float f = 0;
    if (fin < 1000) {
        f = 1.0 * ((float) fin);
        lcd.print(f, 0);
        lcd.print(" Hz");
    } else if (fin >= 1000) {
        f = ((float) fin) / 1000.0;
        lcd.print(f, 3);
        lcd.print(" KHz");
    }
}

void displayV(long fin) {
    float f = (float) fin;
    float v = 0;
    lcd.setCursor(0, 1);

    switch (currentMode) {
        case C:
            v = calcV(f, c0);

            if (v < 1e-9) {
                v = v * 1e12; // pico
                lcd.print(v);
                lcd.print(" ");
                lcd.print("pF");
            } else if (v >= 1e-9 && v < 1
                e-6) {
                v = v * 1e9; // n
                lcd.print(v);
                lcd.print(" ");
```

```
            lcd.print("nF");
        } else {
            lcd.print("---");
        }
        break;
    case L:
        v = calcV(f, l0);

        if (v < 1e-6) {
            v = v * 1e9; //nH
            lcd.print(v);
            lcd.print(" ");
            lcd.print("nH");
        } else if (v >= 1e-6 && v < 1
            e-3) {
            v = v * 1e6; //uH
            lcd.print(v);
            lcd.print(" ");
            lcd.print("uH");
        } else if (v >= 1e-3 && v <
            1) {
            v = v * 1e3;
            lcd.print(v);
            lcd.print(" ");
            lcd.print("mH");
        } else if (v >= 1 && v < 100)
             {
            lcd.print(v);
            lcd.print(" ");
            lcd.print("H");
        } else {
            lcd.print("---");
        }
        break;
    case F:
        break;
    }

}

void loop() {
    FreqCounter::f_comp = 106;
    FreqCounter::start(1000);
    while (FreqCounter::f_ready == 0)
        frq = FreqCounter::f_freq;

    checkLCMode();

    if (buttonPressed(btn1)) {
        backLightOn = !backLightOn;
        if (backLightOn) {
            digitalWrite(pinBACKLIGHT,
                HIGH);
        } else {
            digitalWrite(pinBACKLIGHT,
                LOW);
        }
    } else if (buttonPressed(btn2)) {
        switch (currentMode) {
            case C:
                c0 = calcV(frq, Cth) +
                    Cth;
                F0 = frq;
                break;
            case L:
                l0 = calcV(frq, Lth) +
                    Lth;
                F0 = frq;
                break;
            default:
                break;
        }

        lcd.print(" Cal");
    } else if (buttonPressed(btn3)) {
        dispFreq = !dispFreq;

        if (dispFreq)
            digitalWrite(pinRelay, HIGH);
        else
            digitalWrite(pinRelay, LOW);
    }

    if (dispFreq)
        displayFreq(frq);
    else
        displayV(frq);

    for (int i = 0; i < 3; i++) {
        if (buttonRleased(i)) {
            //the button is released, no
                action is necessary.
        }
    }
}
```

8 Conclusion

This report highlights the design and application of the AVR-Based LC Meter, demonstrating its capability to measure unknown inductance and capacitance values using the Colpitts oscillator.