



TRƯỜNG ĐẠI HỌC BÁCH KHOA ĐẠI HỌC QUỐC GIA TP.HCM

BỘ MÔN ĐIỆN TỬ

THIẾT KẾ HỆ THỐNG NHÚNG

Voltmeter

Bùi Thị Huyền Như
Mã Chí Nhân

2212464
2212359

Đại diện: Bùi Thị Huyền Như
Gmail: nhu.buiiud@hcmut.edu.vn
Tel: 0706.229.026

GV: Nguyễn Phan Hải Phú

Ngày 20 tháng 11 năm 2024

Mục lục

1	Đặt vấn đề	2
2	Yêu cầu hệ thống	2
3	Những vấn đề thiết kế	3
4	Đặc tả hệ thống	3
4.1	Đặc tả phần cứng	4
4.1.1	Yêu cầu	4
4.1.2	Chọn linh kiện	6
4.2	Đặc tả phần mềm	7
5	Kế hoạch làm việc nhóm	8
6	Thực hiện hệ thống	9
6.1	Phần mềm	9
6.2	Phần cứng	18
7	Hoàn thiện sản phẩm	22
7.1	Các thông số của sản phẩm:	22
7.2	Hướng dẫn sử dụng:	22
7.2.1	Một số lưu ý trước khi sử dụng:	22
7.2.2	Hướng dẫn hoạt động:	23
8	Kết luận	24
9	Tài liệu kham khảo	25

Danh sách hình vẽ

1	Sơ đồ khối	4
2	Sơ đồ khối chi tiết cho hệ thống	4
3	Nút nhấn	6
4	Terminal	6
5	Biến trở	6
6	Dao động thạch anh	6
7	Màn hình LCD	6
8	Vị điều khiển	6
9	Sơ đồ khối phần mềm	7
10	Sơ đồ chân ATmega16-PU	9
11	Các thanh ghi và bit cấu hình ADC chế độ ngõ vào đơn	13
12	Bảng tra các bit cấu hình ADC chế độ ngõ vào đơn	13
13	Các thanh ghi và bit cấu hình ADC chế độ ngõ vào vị sai	14
14	Các bit chọn độ lợi vị sai	15
15	Sơ đồ nguyên lý mô phỏng trong phần mềm Protues	19
16	Nạp chương trình bằng Progisp	19
17	Thử nghiệm trên breadboard	20
18	Sơ đồ nguyên lý trong phần mềm Altium	20
19	Mô hình 3D của hệ thống	21
20	Mạch PCB	21
21	Sản phẩm phần cứng	22

1 Đặt vấn đề

Trong các hệ thống điện và điện tử hiện nay, việc đo lường chính xác các thông số điện là rất quan trọng để đảm bảo sự hoạt động ổn định và hiệu quả của các thiết bị. Một trong những thông số cơ bản và quan trọng nhất cần đo trong hệ thống điện là điện áp. Đo điện áp giúp đánh giá trạng thái hoạt động của các mạch điện, từ đó phát hiện và khắc phục sự cố kịp thời, bảo đảm các thiết bị hoạt động trong phạm vi điện áp an toàn và hiệu quả.

Sự phát triển của công nghệ và nhu cầu sử dụng ngày càng cao đòi hỏi các thiết bị đo điện áp không chỉ có độ chính xác cao mà còn phải có khả năng đo được các mức điện áp rộng, dễ dàng sử dụng và giá thành hợp lý. Việc này đặt ra yêu cầu cao đối với các thiết bị đo, trong hầu hết những ứng dụng yêu cầu tính chính xác và độ tin cậy cao như trong các mạch điện tử, thiết bị gia dụng hay các hệ thống điện công nghiệp.

Tuy nhiên, do kiến thức còn hạn chế, nhóm chỉ có thể cải thiện về độ chính xác của thiết bị đo điện áp bằng cách áp dụng các kiến thức đã học và tìm hiểu thêm một số các kiến thức liên quan. Mục tiêu của nhóm là tạo ra một thiết bị đo điện áp có thể đo được các mức điện áp thấp trong các linh kiện điện tử, với mức giá hợp lý và độ chính xác cao. Đây cũng là cơ hội để nhóm có thể tự tay xây dựng một sản phẩm hoàn chỉnh, đồng thời hiểu rõ hơn về quy trình thiết kế và các vấn đề thực tế, không có trong lý thuyết.

Từ những yêu cầu và mục tiêu đó, nhóm tiến hành thiết kế và triển khai một hệ thống đo điện áp nhỏ gọn, dễ sử dụng và đáp ứng được yêu cầu về độ chính xác cho người dùng.

2 Yêu cầu hệ thống

- Tên: Voltmeter
- Mục tiêu:
Voltmeter đo điện áp trong khoảng từ 0 đến 5 volt. Có chế độ đo ngõ vào đơn và chế độ đo ngõ vào vi sai với gain là x1, x10, x200.
- Input và Output:
 - Input:
 - + 6 nút nhấn: reset, ngõ vào đơn, ngõ vào vi sai, gain x1, gain x10, gain x200.
 - + Biến trở: điều chỉnh độ phân giải của màn hình LCD.
 - + 5 Terminal: 1 chân điện áp ngõ vào đơn, 2 chân điện áp ngõ vào vi sai, 2 chân nguồn cung cấp cho hệ thống.
 - Output:
 - + màn hình LCD 16x2: Hiển thị các thông tin, yêu cầu và điện áp đo được.
- Trường hợp sử dụng:
 - Khi cắm điện, màn hình LCD sẽ hiển thị các yêu cầu. Chọn chế độ đo và độ lợi bằng cách nhấn nút tương ứng. Cấp điện áp ngõ vào qua cổng terminal tương ứng. Điện áp sẽ liên tục hiển thị lên màn hình. Hệ thống sẽ cảnh báo ngược chiều điện áp nếu mắc sai chiều. Nếu muốn chọn lại chế độ và gain, nhấn reset. Ngoài ra, người dùng có thể xoay biến trở để tăng giảm độ tương phản của LCD.
- Chức năng:
 - Do điện áp trong khoảng từ 0 đến 5 volt. Có chế độ đo ngõ vào đơn và chế độ đo ngõ vào vi sai với gain là x1, x10, x200, hiển thị lên LCD.
- Hiệu suất:
 - Độ chia nhỏ nhất: $Q = \frac{R}{2^B} = \frac{5}{2^{10}} \approx 0.00488V$

- Sai số lượng tử hiệu dụng $e_{rms} = \frac{Q}{\sqrt{12}} \approx 0.001409$
- Tỉ số tín hiệu trên nhiễu (SNR): $20\log_{10}\left(\frac{R}{Q}\right) \approx 60dB$
- Chi phí: 250.000 VND
Bao gồm:
 - Chi phí nghiên cứu: 500.000 VND chi phí vật liệu, 1.200.000 VND chi phí con người
 - Chi phí sản xuất: 200.000 VND.
- Công suất cung cấp: Sử dụng adapter chuyển 220VAC thành 5VDC.
- Công suất tiêu thụ: $\leq 10W$
- Kích thước/ trọng lượng: 15x15(cm).
- Cài đặt: Cầm tay.

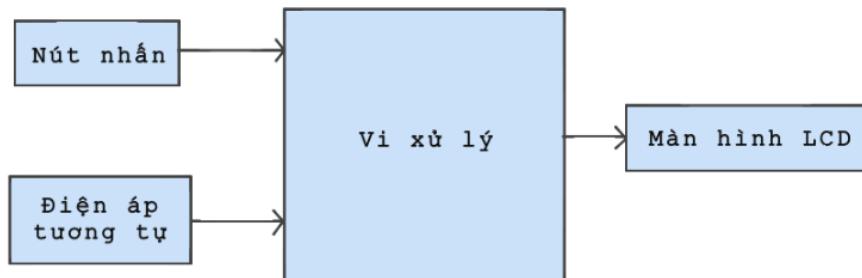
3 Những vấn đề thiết kế

- Ràng buộc:
 - Sai số $<1\%$.
 - Giá thành: $\leq 300.000đ$.
 - Công suất: $\leq 10W$.
 - Kích thước: diện tích $\leq 300cm^2$.
 - Tuổi thọ: trên 7 năm.
 - Thời gian đáp ứng: $\leq 50ms$.
- Vấn đề tính năng: Cảnh báo ngược chiều áp.
- Vấn đề thời gian thực: Soft real-time.
Vì hệ thống chỉ sử dụng để đo điện áp từ 0 đến 5V. Nên nếu trong thời gian vi điều khiển thực hiện, có xảy ra hiện tượng quá áp cũng không gây ảnh hưởng lớn đến người sử dụng.
- Đồng thời:
Đồng thời đọc giá trị điện áp ngõ vào, chuyển đổi giá trị điện áp, thực hiện tính toán để tìm giá trị áp ngõ vào và hiển thị lên LCD.
- Vấn đề phản ứng:
 - Tương tác không liên tục: cảm biến để đo áp khi cần đo.
 - Đáp ứng không theo chu kỳ: Không thể đo sự thay đổi điện áp giữa hai lần lấy mẫu và xử lý liên tiếp, với độ trễ $T_d \approx 0.5s$.

4 Đặc tả hệ thống

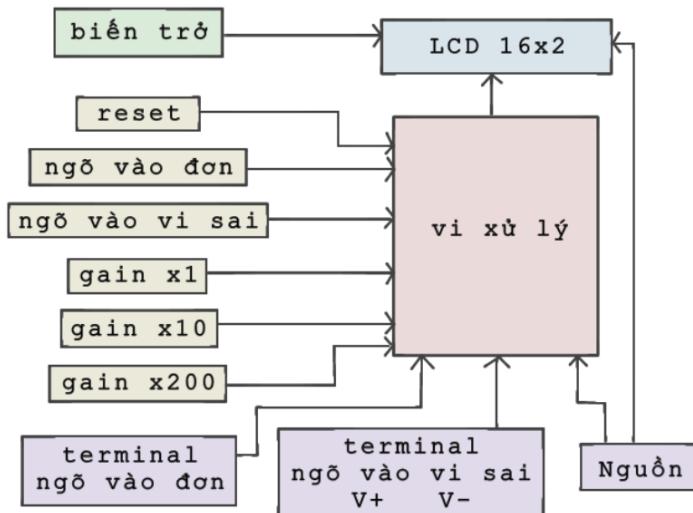
- Nguyên lý hoạt động:
Khi cảm biến, màn hình LCD sẽ hiển thị các yêu cầu. Chọn chế độ đo và độ lợi bằng cách nhấn nút tương ứng, với hai chế độ là đo ngõ vào đơn và đo ngõ vào vi sai, độ lợi là gain x1, x10 và x200. Cấp điện áp ngõ vào qua cổng terminal tương ứng. Điện áp sẽ liên tục hiển thị lên màn hình. Nếu muốn chọn lại chế độ và gain, nhấn reset. Ngoài ra, người dùng có thể xoay núm vặn để tăng giảm độ tương phản của LCD. Rút nguồn điện khi không sử dụng.
- Môi trường hoạt động:
Môi trường có nhiệt độ từ $10 - 35^\circ C$. Trong không gian thoáng, tránh nắng nóng và ẩm ướt sẽ gây hao mòn linh kiện. Hệ thống không có chức năng chống nước.

- Sơ đồ khói:



Hình 1: Sơ đồ khói

- Sơ đồ khói chi tiết:



Hình 2: Sơ đồ khói chi tiết cho hệ thống

4.1 Đặc tả phần cứng

4.1.1 Yêu cầu

- Nút nhấn:
 - + Mục đích: Cung cấp giao diện cho 6 nút nhấn.
 - + Yêu cầu: Nút nhấn phải ổn định, có tuổi thọ dài, cổng kết nối chật chẽ.

Thành phần phần cứng	Giao diện	Ghi chú
Nút reset, ngõ vào đơn, ngõ vào vi sai, gain x1, gain x10, gain x200.	Single end	Được nối trong bo mạch, nút là nút nhấn nhả, khoảng cách giữa 2 chân cắm là 6mm.

- Terminal:
 - + Mục đích: Cung cấp giao diện để Cắm điện áp ngõ vào, kết nối adaptor với mạch để tạo nguồn cấp cho vi điều khiển và LCD.

- + Yêu cầu: Các mối nối phải chắc chắn, ổn định.

Thành phần phần cứng	Giao diện	Ghi chú
Terminal 3 chân lây áp: 1 ngõ vào đơn, 2 chân điện áp ngõ vào vi sai	Dầu nối vít (Screw terminal)	Dầu nối có thể vặn dây dẫn vào, nối các đầu dây đưa ra bên ngoài bo mạch.
Terminal 2 chân kết nối với adaptor.	Dầu nối vít (Screw terminal)	Dầu nối có thể vặn dây dẫn vào, nối với 2 đầu của adaptor.

- Biến trở:
 - + Mục đích: Điều chỉnh độ tương phản cho LCD.
 - + Yêu cầu: Biến trở $10\text{ k}\Omega$.

Thành phần phần cứng	Giao diện	Ghi chú
Biến trở	$10\text{ k}, 3$ chân	

- Dao động thạch anh:
 - + Mục đích: Tạo dao động cho vi điều khiển.
 - + Yêu cầu: Chính xác, ổn định.

Thành phần phần cứng	Giao diện	Ghi chú
Dao động thạch anh	8 MHz	

- Màn hình LCD:
 - + Mục đích: Hiển thị các thông tin yêu cầu và điện áp đo được.
 - + Yêu cầu: Kích thước: $80 \times 36 \times 12.5\text{ mm}$.
- Vi điều khiển:
 - + Mục đích: Nhận các tín hiệu từ nút nhấn để xử lý, đọc điện áp ngõ vào đơn và ngõ vào vi sai, ADC và tính toán điện áp ngõ vào, điều khiển LCD hiện các yêu cầu và giá trị điện áp.
 - + Yêu cầu: Có chức năng ADC ngõ vào đơn, ADC ngõ vào vi sai với các độ lợi, thực hiện yêu cầu của hệ thống. Bộ nhớ $\geq 16\text{ Kbytes Flash ROM, SRAM} \geq 1\text{ Kbytes}$.

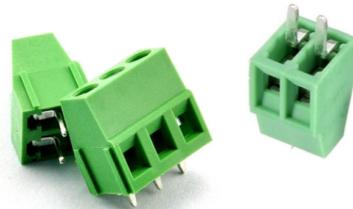
Thành phần phần cứng	Giao diện	Ghi chú
Vi điều khiển	40 chân, PU	Cần chọn loại vi điều khiển phù hợp, đầy đủ các chức năng.

4.1.2 Chọn linh kiện

- Nút nhấn:
Nút Nhấn 6X6X5MM 2 Chân Cắm.
- Terminal:
+ Domino KF128-2P-5.08MM.
+ Domino KF128-3P-5.08MM.



Hình 3: Nút nhấn



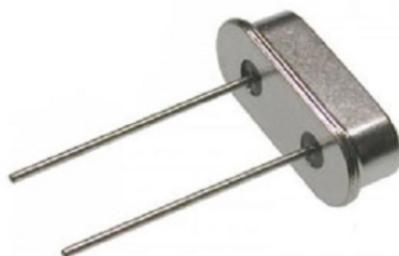
Hình 4: Terminal

- Biến trớ:
WH148-3P-103 Biến Trở Volume Đơn
10 KOhm 20% 0.125W 3 Chân.



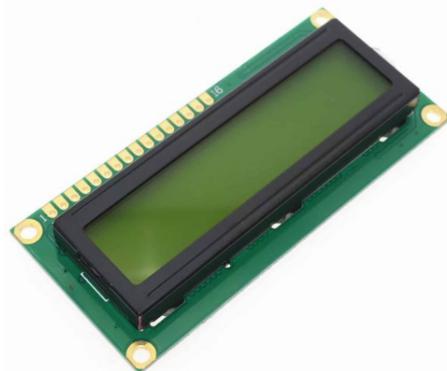
Hình 5: Biến trớ

- Dao động thạch anh:
Thạch anh 8Mhz HC49 DIP.



Hình 6: Dao động thạch anh

- Màn hình LCD: LCD1602 5V.
Datasheet: [LCD 1602](#)
- Vi điều khiển: ATMEGA16-16PU.
Datasheet: [ATmega16](#)

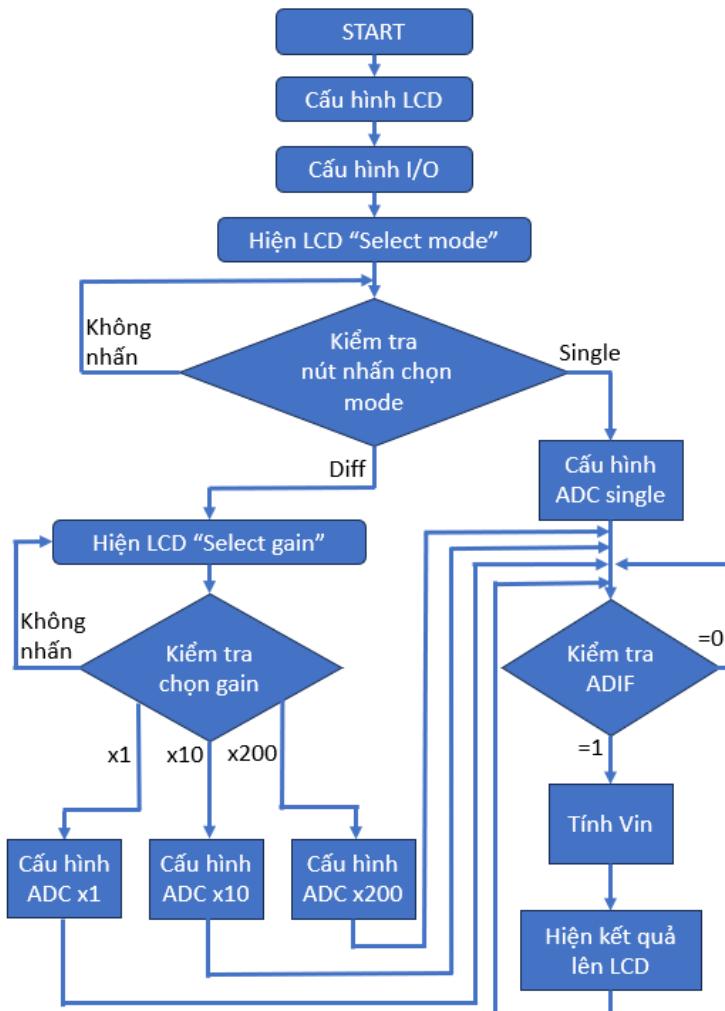


Hình 7: Màn hình LCD



Hình 8: Vi điều khiển

4.2 Đặc tả phần mềm



Hình 9: Sơ đồ khái niệm phần mềm

Khối	Chức năng	Yêu cầu
Nút nhấn chọn chế độ	Liên tục kiểm tra nút nhấn để gọi hàm thực hiện chế độ tương ứng với nút nhấn	Thực hiện chống run bằng phần mềm
Nút nhấn chọn độ lợi	Liên tục kiểm tra nút nhấn để gọi hàm cấu hình ADC với độ lợi tương ứng	Thực hiện chống run bằng phần mềm
LCD	Hiển thị các yêu cầu và giá trị điện áp ngõ vào	Giao tiếp LCD 8bits
Lấy mẫu và tính toán	Sử dụng chế độ ADC để lấy mẫu điện áp ngõ vào, thực hiện tính toán chuyển giá trị điện áp số thành giá trị điện áp tương tự	Lấy 3 chữ số sau dấu phẩy với chế độ ngõ vào đơn, vi sai x1, vi sai x10. Lấy 2 chữ số sau dấu phẩy với vi sai x200

5 Kế hoạch làm việc nhóm

- Xây dựng nhóm:

Thành Viên	Vai trò	Chữ ký
Bùi Thị Huyền Như	Thiết kế hệ thống, thiết kế phần cứng, viết báo cáo.	
Mã Chí Nhân	Thiết kế phần mềm, làm phần cứng, viết báo cáo.	
Nhiệm vụ	Trách nhiệm	
Phát triển kiến trúc của hệ thống	Bùi Thị Huyền Như	
Viết chương trình thực hiện chức năng của hệ thống	Mã Chí Nhân	
Tích hợp và kiểm tra	Cả 2	
Thiết kế PCB	Bùi Thị Huyền Như	
Mua linh kiện, in mạch, làm mạch	Cả 2	
Kiểm tra chức năng mạch in	Cả 2	
Viết báo cáo và làm slide	Cả 2	

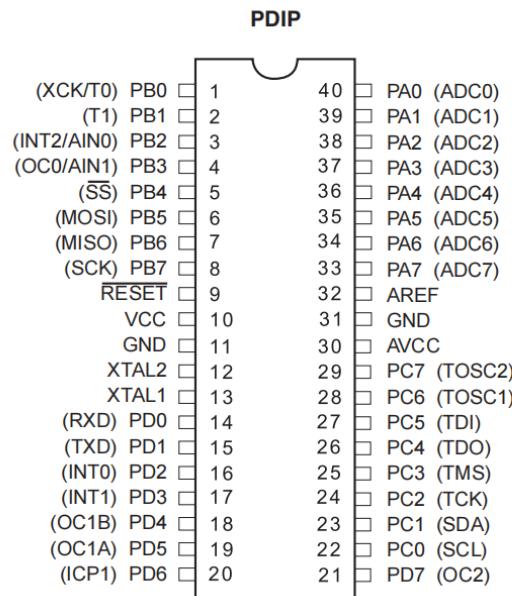
- Xây dựng kế hoạch thực hiện hệ thống:

Tên sản phẩm	Voltmeter		
Đặc điểm chính	Do điện áp ngõ vào.		
Thời gian dự kiến	3 tuần Bắt đầu: 4/11/2024 Kết thúc: 25/11/2024		
Ngân sách	Dụng cụ: 500.000 VND Vật liệu: 180.000 VND		
Giá dự kiến	300.000VND		
Thành viên	Bùi Thị Huyền Như, Mã Chí Nhân		
Thời khóa biểu	Tuần 1	Tuần 2	Tuần 3
1. Viết báo cáo. 2. Thiết kế kiến trúc hệ thống. 3. Thiết kế phần cứng. 3.1. Thiết kế giao diện. 3.2. Lắp mạch (breadboard). 4. Viết phần mềm. 4.1. Tìm thuật toán. 4.2. Viết chương trình. 5. Tích hợp và kiểm tra. 5.1. Mô phỏng hoạt động. 5.2. Kiểm tra và sửa lỗi. 6. Hoàn thiện sản phẩm. 6.1. Vẽ PCB, in mạch PCB. 6.2. Đóng gói sản phẩm.	→	→	→

6 Thực hiện hệ thống

6.1 Phần mềm

Phần mềm sử dụng ngôn ngữ C và trình biên dịch là Microchip Studio. Do nhóm sử dụng vi điều khiển ATmega16 thuộc họ AVR, nên để dễ tương tác với các thanh ghi chức năng của vi điều khiển, nhóm dùng thư viện `<avr/io.h>`. Nhóm cũng sử dụng thư viện `"util/delay.h"` có sẵn để dễ thực hiện điều khiển LCD và các thao tác cần có thời gian chờ như chống rung nút nhấn bằng phần mềm. Do cách bố trí chân của vi điều khiển, nên để điều khiển LCD chế độ 8 bit dữ liệu và 3 bit điều khiển. Để tối ưu cho phần vẽ PCB thì nhóm chọn 8 bit dữ liệu là `PORTD` và 3 bit điều khiển là `RS, RW, E` lần lượt là `PB5, PB6, PB7` vì các chân xuất dữ liệu đều nằm bên phải của vi điều khiển. Ngoài nút nhấn `reset`, hệ thống còn 5 nút nhấn bao gồm `Single, Diff, gain x1, gain x10, gain x200`. Để vẽ PCB thuận lợi, nhóm sắp xếp các nút nhấn nằm phía bên phải của vi điều khiển bao gồm `PORTA` và `PORTC`, nhưng `PORTC` thường được dùng để nạp chương trình vào vi điều khiển, thế nên 5 chân còn lại của `PORTA` được dùng làm ngõ vào cho các nút nhấn từ theo thứ tự lần lượt `PA3, PA4, PA5, PA6, PA7` được kết nối với `Single, Diff, gain x1, gain x10, gain x200`. Ba chân còn lại của `PORTA` gồm `PA0, PA1, PA2` là các chân ngõ vào `ADC` vì đây là các chân cố định được nhà sản xuất gán cho chức năng `ADC`.



Hình 10: Sơ đồ chân ATmega16-PU

Listing 1: Chỉ dẫn

```

1  /*
2   * Voltmetter.c
3   *
4   * Created: 9/13/2024 2:01:26 AM
5   * Author : Chisnhaan
6   */
7
8 #include <avr/io.h>
9 #define F_CPU 8000000UL
10 #include "util/delay.h"

```

```

11 #include <string.h>
12 #include <stdio.h>
13 #define LCD_data_DDR DDRD
14 #define LCD_data_PORT PORTD
15 #define LCD_com_DDR DDRB
16 #define LCD_com_PORT PORTB
17
18 #define RS 5
19 #define RW 6
20 #define EN 7
21
22 #define BUTTON_DDR DDRA
23 #define BUTTON_PORT PORTA
24 #define Single_mode 3
25 #define Diff_mode 4
26 #define Gain_1 5
27 #define Gain_10 6
28 #define Gain_200 7
29
30 void LCD_com(unsigned char cmnd);
31 void LCD_char(unsigned char char_data);
32 void LCD_string(char *str);
33 void Single();
34 void Diff();
35 void Diff_1();
36 void Diff_10();
37 void Diff_200();

```

Hàm main bao gồm các chức năng: cấu hình ngõ vào ngõ ra, hiện tiêu đề và yêu cầu nhấn nút lên LCD sau đó chờ nút nhấn chọn chế độ *Single* hay *Diff*. Cấu hình ngõ vào ngõ ra được thực hiện bằng cách tác động lên các thanh ghi *DDR* đã được chỉ dẫn từ trước phần đầu chương trình. Hiện tiêu đề *ESD Project* bằng cách ghi lệnh, dữ liệu vào LCD qua hai hàm *LCD_com* và *LCD_string* sẽ được mô tả ở phần sau. Nhiệm vụ chờ nút nhấn được thực hiện trong vòng lặp vô tận *while(1)*, nếu phát hiện có tín hiệu nút nhấn được nhấn thì sẽ thực hiện chống rung phím bằng phần mềm sau đó kiểm tra lại nút nhấn để xem có thực sự được nhấn hay không, nếu không thực sự nhấn sẽ quay lại kiểm tra tiếp trạng thái nút.

Listing 2: hàm main

```

1 int main(void)
2 {
3     LCD_com_DDR |= (1<<RS)|(1<<EN)|(1<<RW);
4     LCD_data_DDR = 0xFF;
5
6     BUTTON_DDR |= (0<<Single_mode)|(0<<Diff_mode)|(0<<Gain_1)|(0<<
7         Gain_10)|(0<<Gain_200);
8     BUTTON_PORT |= (1<<Single_mode)|(1<<Diff_mode)|(1<<Gain_1)|(1<<
9         Gain_10)|(1<<Gain_200);
10
11     LCD_com(0x38);
12     LCD_com(0x0E);
13     LCD_com(0x01);
14
15     LCD_com(0X83);
16     LCD_string("ESD\u2022Project");
17     LCD_com(0xC4);
18     LCD_string("Voltmeter");
19     _delay_ms(100);
20 }

```

```

19     LCD_com(0x01);
20     LCD_com(0x80);
21     LCD_string("Click the button");
22     LCD_com(0xC0);
23     LCD_string("to select mode!");
24     _delay_ms(100);

25
26     LCD_com(0x01);
27     LCD_string("Single");
28     LCD_com(0xC0);
29     LCD_string("Differential");

30
31     while (1)
32     {
33         if((PIN_A & (1<<Diff_mode)) == 0)
34         {
35             _delay_ms(10);
36             if((PIN_A & (1<<Diff_mode)) == 0)
37                 Diff();
38         }
39
40         if((PIN_A & (1<<Single_mode)) == 0)
41         {
42             _delay_ms(10);
43             if((PIN_A & (1<<Single_mode)) == 0)
44                 Single();
45         }
46     }
47 }
48

```

Các hàm để tác động với LCD bao gồm: *LCD_com*, *LCD_string* và *LCD_char*. Hàm *LCD_com* có chức năng xuất dữ liệu nhận vào ra 8 chân dữ liệu của LCD, được dùng để ghi một lệnh vào LCD. *LCD_char* cũng có chức năng tương tự như *LCD_com* khi xuất 8 bit đối số nhận vào ra LCD, tuy nhiên nhóm tạo 2 hàm để dễ kiểm soát việc ghi lệnh hay ghi dữ liệu. Hàm *LCD_string* được phát triển từ *LCD_char* để có thể ghi một chuỗi ký tự vào LCD bằng cách dùng biến con trỏ. Quy trình của việc ghi lệnh và ghi dữ liệu ra LCD được thực hiện theo các bước:

Cách gửi lệnh ra LCD:

- Xuất mức thấp chân RW để chọn chế độ ghi.
- Xuất mức thấp chân RS để chọn ghi lệnh.
- Gửi byte lệnh ra các chân data (D0 đến D7).
- Tạo một cạnh xuống ở chân E để ghi vào LCD.
- Delay một khoảng thời gian (1ms) để chờ LCD thực hiện lệnh

Cách gửi dữ liệu ra LCD:

- Xuất mức thấp chân RW để chọn chế độ ghi.
- Xuất mức cao chân RS để chọn ghi dữ liệu.
- Gửi mã ASCII lệnh ra các chân dữ liệu (D0 đến D7).
- Tạo một cạnh xuống ở chân E để ghi vào LCD.

- Delay một khoảng thời gian (1ms) để chờ LCD thực hiện lệnh

Listing 3: LCD

```

1   void LCD_com(unsigned char cmnd)
2 {
3     LCD_com_PORT &= ~(1<<RW);
4     LCD_com_PORT &= ~(1<<RS);
5     LCD_data_PORT = cmnd;
6     LCD_com_PORT |= (1<<EN);
7     _delay_ms(1);
8     LCD_com_PORT &= ~(1<<EN);
9     _delay_ms(5);
10 }
11
12 void LCD_char(unsigned char char_data)
13 {
14     LCD_com_PORT &= ~(1<<RW);
15     LCD_com_PORT |= (1<<RS);
16     LCD_data_PORT = char_data;
17     LCD_com_PORT |= (1<<EN);
18     _delay_ms(1);
19     LCD_com_PORT &= ~(1<<EN);
20     _delay_ms(5);
21 }
22
23 void LCD_string(char *str)
24 {
25     int i;
26     for(i = 0; str[i] != 0; i++)
27     {
28         LCD_char(str[i]);
29     }
30 }
```

Hàm *Single* sẽ xử lý các tác vụ khi người dùng chọn vào chế độ ngõ vào đơn. Trước tiên, sẽ thực hiện việc cấu hình ADC ngõ vào đơn điện áp tham chiếu là 5V, cho phép ADC với giá trị bộ chia là 128, chế độ tự kích và kích lần đầu tiên (hình 11, 12). Nhóm chọn ngõ vào ADC là chân *PA2* vì ở chế độ vi sai, nhà sản xuất hỗ trợ các chế độ với độ lợi khác nhau ở hai chân *PA0* và *PA1*, vì vậy, 2 chân đó được để trống cho chế độ ngõ vào vi sai. Sau khi thực hiện cấu hình ADC, chương trình liên tục kiểm tra cờ *ADIF* của *ADCSRA* để chờ ADC xử lý xong, giá trị ngõ ra được lưu trong thanh ghi *ADC*, từ đó thực hiện việc tính tinh diện áp ngõ vào *Vin* qua công thức. Cuối cùng sẽ tách từng ký tự của giá trị *Vin* có được từ phương trình (1) và lần lượt hiện lên LCD.

$$ADC = \frac{Vin \cdot 1024}{Vref} \quad (1)$$

Voltmeter

ADMUX – ADC Multiplexer Selection Register

Bit	7	6	5	4	3	2	1	0	
	REFS1	REFS0	ADLAR	MUX4	MUX3	MUX2	MUX1	MUX0	ADMUX
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

ADCSRA – ADC Control and Status Register A

Bit	7	6	5	4	3	2	1	0	
	ADEN	ADSC	ADATE	ADIF	ADIE	ADPS2	ADPS1	ADPS0	ADCSRA
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

SFIOR – Special FunctionIO Register

Bit	7	6	5	4	3	2	1	0	
	ADTS2	ADTS1	ADTS0	-	ACME	PUD	PSR2	PSR10	SFIOR
Read/Write	R/W	R/W	R/W	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Hình 11: Các thanh ghi và bit cấu hình ADC chế độ ngõ vào đơn

Table 22-3. Voltage Reference Selections for ADC

REFS1	REFS0	Voltage Reference Selection
0	0	AREF, Internal Vref turned off
0	1	AVCC with external capacitor at AREF pin
1	0	Reserved
1	1	Internal 2.56V Voltage Reference with external capacitor at AREF pin

Table 22-4. Input Channel and Gain Selections

MUX4:0	Single Ended Input	Positive Differential Input	Negative Differential Input	Gain
00000	ADC0	N/A		
00001	ADC1			
00010	ADC2			
00011	ADC3			
00100	ADC4			
00101	ADC5			
00110	ADC6			
00111	ADC7			

Table 22-6. ADC Auto Trigger Source Selections

ADTS2	ADTS1	ADTS0	Trigger Source
0	0	0	Free Running mode
0	0	1	Analog Comparator
0	1	0	External Interrupt Request 0
0	1	1	Timer/Counter0 Compare Match
1	0	0	Timer/Counter0 Overflow
1	0	1	Timer/Counter1 Compare Match B
1	1	0	Timer/Counter1 Overflow
1	1	1	Timer/Counter1 Capture Event

Hình 12: Bảng tra các bit cấu hình ADC chế độ ngõ vào đơn

Listing 4: ADC Single

```

1   void Single()
2 {
3     float Vin;
4     int V, V1, V2, V3, V4;
5     LCD_com(0x01);
6     LCD_string("Single\u25aa mode");
7     while(1)
8     {
9       ADMUX |= (1<<REFS0)|(1<<MUX1);
10      ADCSRA |= (1<<ADEN) | (1<<ADSC) | (1<<ADATE) | (1<<ADPS2) | (1<<
11        ADPS1) | (1<<ADPS0);
12      SFIOR |= 0;
13      while(!(ADCSRA & (1<<ADIF)));
14      ADCSRA |= (1<<ADIF);
15      Vin = (ADC*4.8828);
16      V = Vin;
17      V1 = (V/1000);
18      V2 = (V%1000)/100;
19      V3 = ((V%1000)%100)/10;
20      V4 = (((V%1000)%100)%10);
21      LCD_com(0xc5);
22      LCD_char(V1+0x30);
23      LCD_char(',');
24      LCD_char(V2+0x30);
25      LCD_char(V3+0x30);
26      LCD_char(V4+0x30);
27    }
}

```

Tương tự như hàm *Single*, hàm *Diff* cũng giải quyết tác vụ khi người dùng nhấn chọn vào chế độ vi sai. Tuy nhiên, trong chế độ vi sai còn có 3 tùy chọn độ lợi vi sai. Do đó, hàm *Diff()* sẽ hiện yêu cầu người dùng chọn độ lợi vi sai và liên tục kiểm tra nút nhấn của từng chế độ cũng như thực hiện chống rung phím bằng phần mềm bằng cách delay. Khi người dùng chọn một trong 3 độ lợi vi sai thông qua nút nhấn, hàm vi sai với độ lợi tương ứng sẽ được gọi. Trong các hàm *Diff_1* *Diff_10* và *Diff_200* sẽ thực hiện cấu hình ADC ngõ vào vi sai với độ lợi tương ứng (hình 14), chế độ chuyển đổi 1 lần (hình 13). Đối với *gain* = 1 chiều điện áp sẽ là *PA0(+)*, *PA1(-)* còn với độ lợi *gain* = 10 và *gain* = 200 thì chiều điện áp sẽ là *PA0(-)*, *PA1(+)*.

ADCSRA – ADC Control and Status Register A

Bit	7	6	5	4	3	2	1	0	
	ADEN	ADSC	ADATE	ADIF	ADIE	ADPS2	ADPS1	ADPS0	ADCSRA
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

SFIOR – Special FunctionIO Register

Bit	7	6	5	4	3	2	1	0	
	ADTS2	ADTS1	ADTS0	-	ACME	PUD	PSR2	PSR10	SFIOR
Read/Write	R/W	R/W	R/W	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Hình 13: Các thanh ghi và bit cấu hình ADC chế độ ngõ vào vi sai

Table 22-4. Input Channel and Gain Selections

MUX4:0	Single Ended Input	Positive Differential Input	Negative Differential Input	Gain
01000		ADC0	ADC0	10x
01001		ADC1	ADC0	10x
01010 ⁽¹⁾		ADC0	ADC0	200x
01011 ⁽¹⁾		ADC1	ADC0	200x
01100		ADC2	ADC2	10x
01101		ADC3	ADC2	10x
01110 ⁽¹⁾		ADC2	ADC2	200x
01111 ⁽¹⁾		ADC3	ADC2	200x
10000		ADC0	ADC1	1x

Hình 14: Các bit chọn độ lợi vi sai

Sau khi đã cấu hình phù hợp, chương trình liên tục kiểm tra cờ *ADIF* để đợi quá trình chuyển đổi hoàn tất, sau đó sẽ tính lại *Vin* có được từ phương trình (2). Sau khi đã có *Vin*, hàm sẽ thực hiện việc lấy từng ký tự của *Vin* và lần lượt hiện giá trị lên LCD. Nếu người dùng mắc ngược chiều điện áp, trên LCD sẽ hiện dòng chữ *Opposite* thể hiện rằng người dùng đã đang đo một điện áp ngược chiều *ADC*, nếu muốn đo đúng giá trị, phải thực hiện mắc ngược dây lại.

$$ADC = \frac{(V_+ - V_-).Gain.512}{Vref} \quad (2)$$

Listing 5: ADC Diff

```

1   void Diff()
2 {
3     LCD_com(0x01);
4     LCD_string("Diff\u2022mode");
5     _delay_ms(100);
6     LCD_com(0x01);
7     LCD_string("select\u2022Gain");
8
9   while(1)
10 {
11
12     if((PINB & (1<<Gain_200)) == 0)
13     {
14       _delay_ms(10);
15       if((PINB & (1<<Gain_200)) == 0)
16       {
17         Diff_200();
18       }
19     }
20
21     if((PINB & (1<<Gain_10)) == 0)
22     {
23       _delay_ms(10);
24       if((PINB & (1<<Gain_10)) == 0)
25       {
26         Diff_10();
27       }
28     }
29 }
```

Voltmeter

```
30         if((PINB & (1<<Gain_1)) == 0)
31     {
32         _delay_ms(10);
33         if((PINB & (1<<Gain_1)) == 0)
34     {
35             Diff_1();
36         }
37     }
38
39 }
40 }
41 }
42
43 void Diff_1()
44 {
45     long double Vin = 0;
46     int V = 0, V1, V2, V3, V4, V5;
47     LCD_com(0x01);
48     LCD_string("Diff\u20a9x1");
49     ADMUX |= (1<<REFS0)|(1<<MUX4); //vi sai G = 1 PA0(+) , PA1(-)
50     ADCSRA |= (1<<ADEN)|(1<<ADPS2) |(1<<ADPS1)|(1<<ADPS0);
51     SFIOR |= 0;
52     while(1)
53     {
54         ADCSRA |= (1<<ADSC);
55         while(!(ADCSRA & (1<<ADIF))); //check co ADIF
56         ADCSRA |= (1<<ADIF); // xoa co ADIF
57         Vin = ((ADC&(0X1FF))*9.765625); // Vin =
58                         ADC*5000/512 ;lay 3 so thap phan sau dau phay
59
60         if(Vin==0)
61     {
62         LCD_com(0xC2);
63         LCD_string("\u20a9Opposite\u20a9");
64     }
65     else
66     {
67         V = Vin;
68         V1 = (V/10000);
69         V2 = (V%10000)/1000;
70         V3 = ((V%10000)%1000)/100;
71         V4 = (((V%10000)%1000)%100)/10;
72         V5 = (((V%10000)%1000)%100)%10;
73
74         LCD_com(0Xc2);
75         LCD_string("Vi=");
76         LCD_char(V1+0x30);
77         LCD_char(V2+0x30);
78         LCD_char(',');
79         LCD_char(V3+0x30);
80         LCD_char(V4+0x30);
81         LCD_char(V5+0x30);
82
83     }
84 }
85 }
86 }
```

Voltmeter

```
87 void Diff_10()
88 {
89     long double Vin = 0;
90     long int V = 0, V1, V2, V3, V4, V5;
91     LCD_com(0x01);
92     LCD_string("Diff\u2022x10");
93     ADMUX |= (1<<REFS0)|(1<<MUX3)|(1<<MUX0);           //vi sai G = 10
94         PA0(-), PA1(+)
95     ADCSRA |= (1<<ADEN)|(1<<ADPS2) |(1<<ADPS1)|(1<<ADPS0);
96     SFIOR |= 0;
97     while(1)
98     {
99
100        ADCSRA |= (1<<ADSC);
101        while(!(ADCSRA & (1<<ADIF))); //check co ADIF
102        ADCSRA |= (1<<ADIF);           // xoa co ADIF
103        Vin = ((ADC&(0X1FF))*0.9765625);
104            // Vin = ADC*5000/512 ;lay 3 so thap phan sau dau phay
105        if(Vin==0)
106        {
107            LCD_com(0xC2);
108            LCD_string("\u2022Opposite\u2022");
109        }
110        else
111        {
112            V = Vin;
113            V1 = (V/10000);
114            V2 = (V%10000)/1000;
115            V3 = ((V%10000)%1000)/100;
116            V4 = (((V%10000)%1000)%100)/10;
117            V5 = (((V%10000)%1000)%100)%10;
118            LCD_com(0Xc2);
119            LCD_string("Vi=");
120            LCD_char(V1+0x30);
121            LCD_char(V2+0x30);
122            LCD_char(',');
123            LCD_char(V3+0x30);
124            LCD_char(V4+0x30);
125            LCD_char(V5+0x30);
126            LCD_com(0x80);
127        }
128    }
129
130 void Diff_200()
131 {
132     double Vin = 0;
133     int V = 0, V1, V2, V3, V4, V5;
134     LCD_com(0x01);
135     LCD_string("Diff\u2022x200");
136     ADMUX |= (1<<REFS0)|(1<<MUX3)|(1<<MUX1)|(1<<MUX0); //vi sai G =
137         200 PA0(-), PA1(+)
138     ADCSRA |= (1<<ADEN) |(1<<ADPS2) |(1<<ADPS1);      //mode chuyen doi
139         1 lan
140     while(1)
141     {
142         ADCSRA |= (1<<ADSC);
143         while(!(ADCSRA & (1<<ADIF))));
```

```

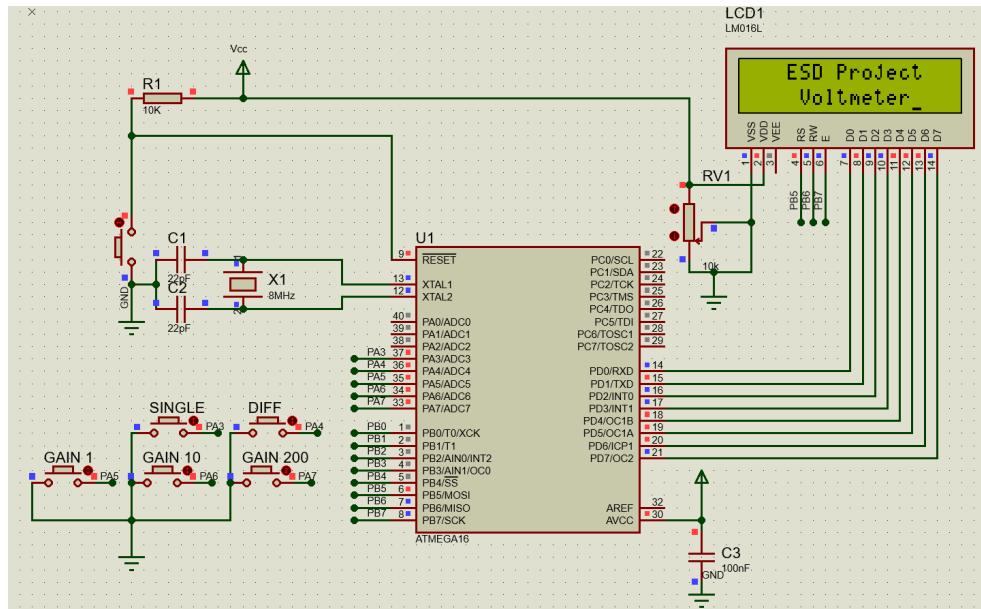
141     ADCSRA |= (1<<ADIF);
142     Vin = ((ADC&(0X1FF))*25/512); // Vin =
143     ADC*5000/512 ;lay 3 so thap phan sau dau phay
144     if(Vin==0)
145     {
146         LCD_com(0xC2);
147         LCD_string("«Opposite»");
148     }
149     else
150     {
151         V = Vin;
152         V1 = (V/10000);
153         V2 = (V%10000)/1000;
154         V3 = (((V%10000)%1000)/100)/100;
155         V4 = (((((V%10000)%1000))%100)/100)/10;
156         V5 = (((((V%10000)%1000))%100)%10;
157         LCD_com(0xc2);
158         LCD_string("Vi=");
159         LCD_char(V1+0x30);
160         LCD_char(V2+0x30);
161         LCD_char(',');
162         LCD_char(V3+0x30);
163         LCD_char(V4+0x30);
164         LCD_char(V5+0x30);
165         LCD_com(0x80);
166     }
167 }
```

Dể kiểm soát chương trình do các thành viên cùng viết, nhóm sử dụng nền tảng Github để trao đổi và lưu trữ chương trình. Github giúp quá trình của nhóm được thuận lợi hơn do có hỗ trợ lưu trữ các phiên bản code khác nhau của mỗi lần điều chỉnh. Github còn thể hiện được xung đột chương trình trước và sau ghi ghép, điều này rất có ý nghĩa với nhóm khi nhiều thành viên cùng chỉnh sửa có thể gây ra xung đột chương trình. Đường dẫn đến Github cho bài tập lớn này [tại đây](#)

6.2 Phần cứng

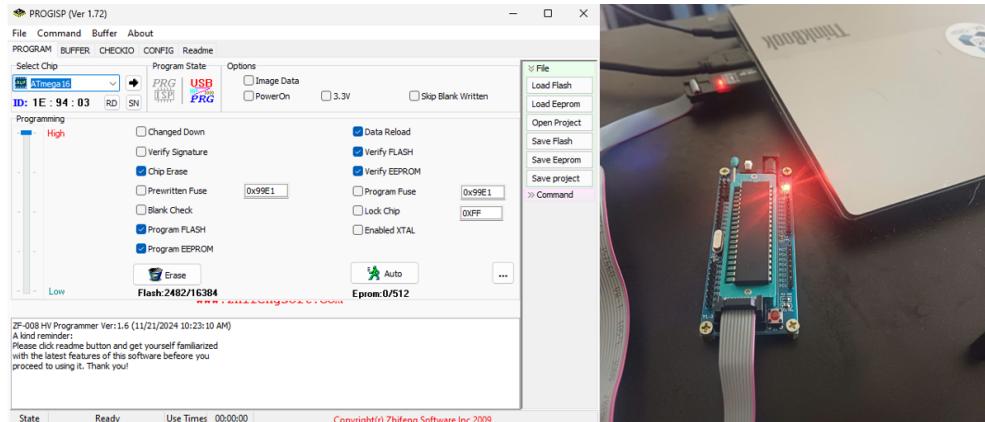
Nhóm thực hiện mô phỏng phần cứng và kiểm tra lỗi phần mềm bằng Protues với sơ đồ kết nối phần cứng như sau:

Voltmeter



Hình 15: Sơ đồ nguyên lý mô phỏng trong phần mềm Protues

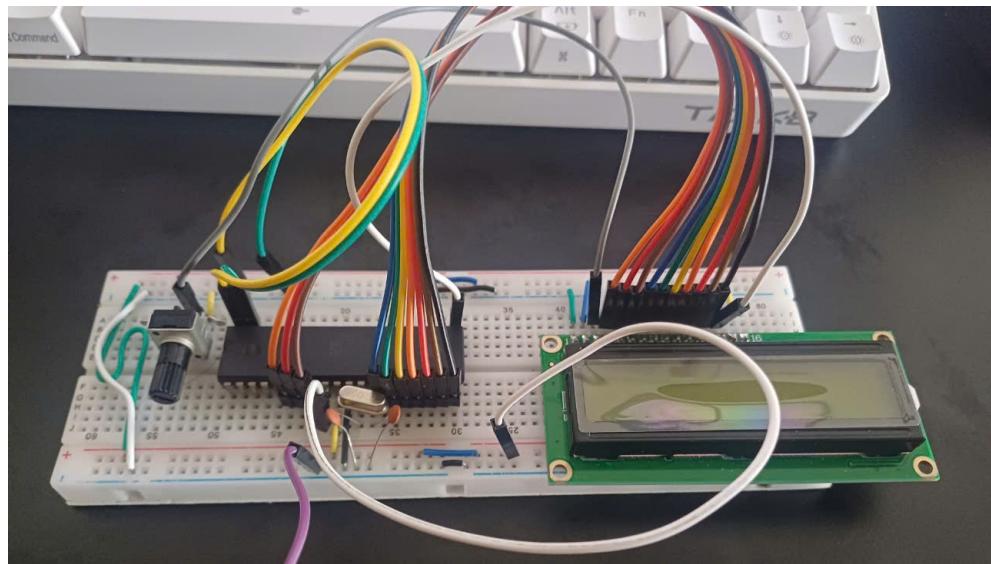
Khi mạch nguyên lý đã chạy đúng, nhóm đã thực hiện thử nghiệm mạch thực tế. Phần mềm để nạp chương trình vào vi điều khiển là *Progisp* dùng để nạp chương trình cho các dòng vi điều khiển AVR, vì vậy nó phù hợp với vi điều khiển nhóm sử dụng. Kết nối phần cứng giữa máy tính và vi điều khiển cho việc nạp chương trình thông qua USB.



Hình 16: Nạp chương trình bằng Progisp

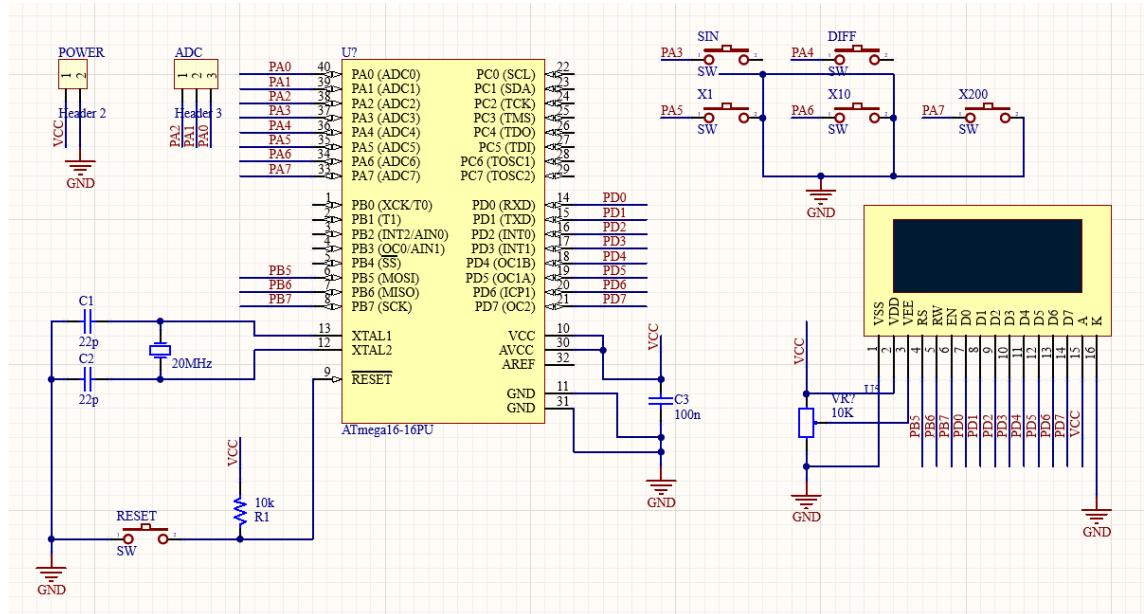
Để dễ dàng thử nghiệm mạch thực tế, nhóm sử dụng breadboard cho việc lắp mạch và kiểm tra hoạt động của hệ thống trên thực tế.

Voltmeter



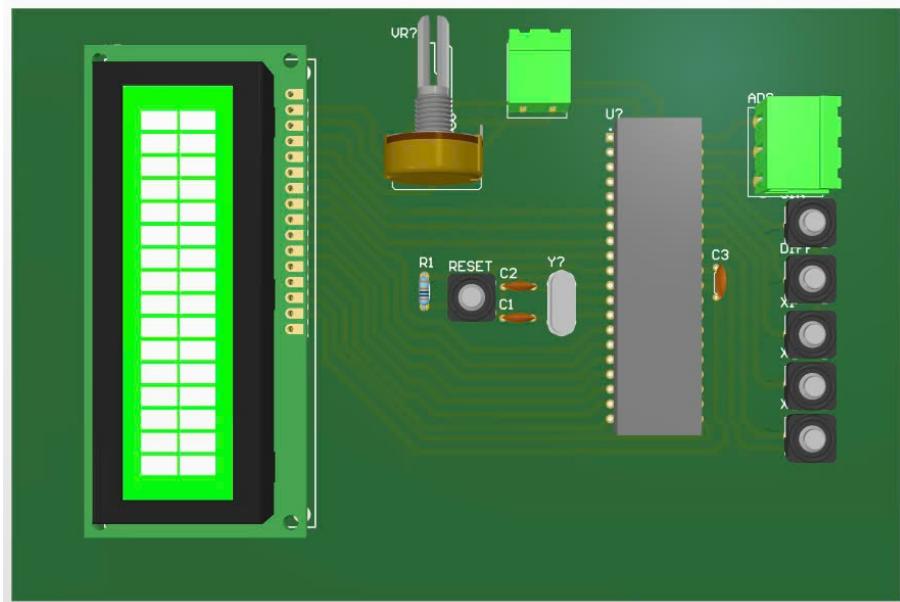
Hình 17: Thử nghiệm trên breadboard

Khi mạch của nhóm đã hoạt động theo đúng chức năng. Nhóm thực hiện vẽ PCB bằng phần mềm Altium. Sau đây là kết quả thực hiện trên Altium bao gồm: sơ đồ nguyên lý, mô hình 3D của mạch và file mạch in PCB.

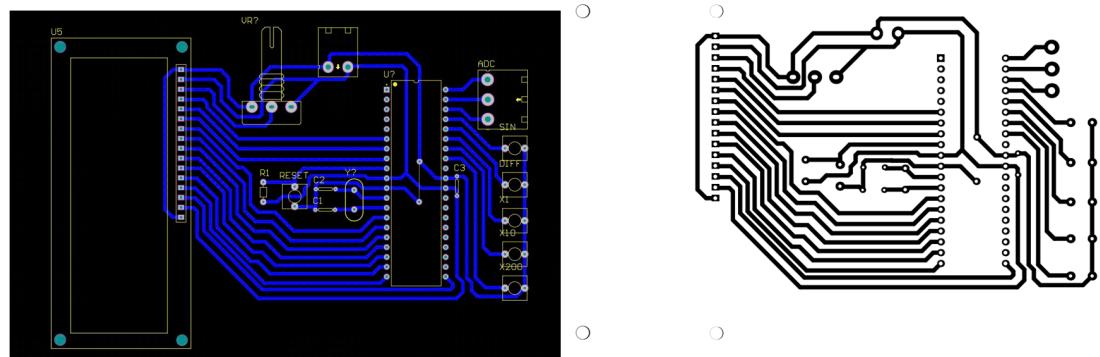


Hình 18: Sơ đồ nguyên lý trong phần mềm Altium

Voltmeter

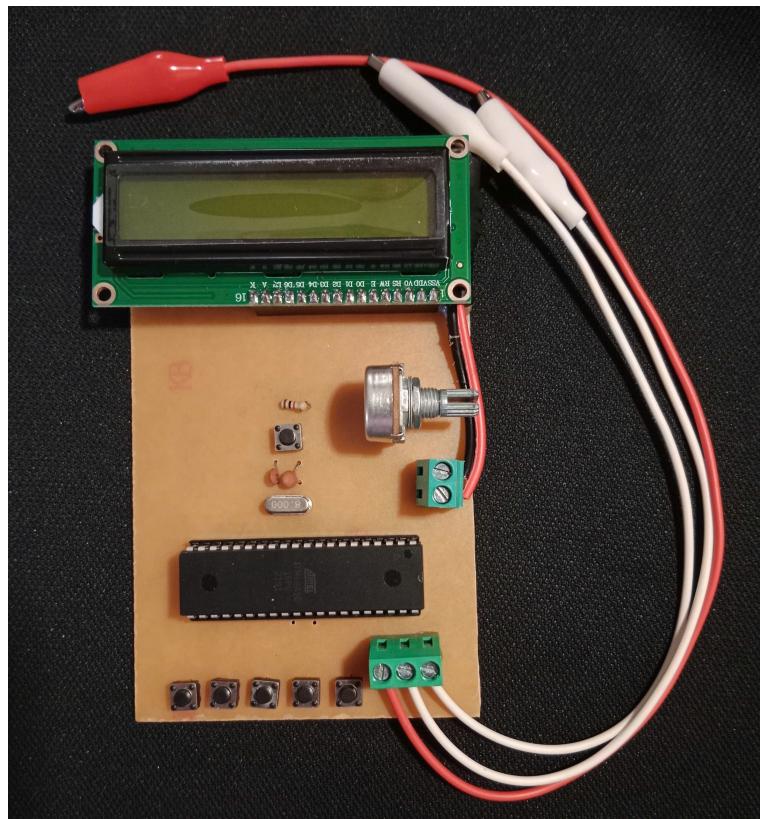


Hình 19: Mô hình 3D của hệ thống



Hình 20: Mạch PCB

Sau bước thiết kế phần cứng, nhóm thực hiện thi công mạch in bằng board mạ đồng. Sử dụng phương pháp in mực sau đó tẩy bằng dung dịch $FeCl_3$ để giữ lại phần mạch cần thiết. Thực hiện hàn các linh kiện vào mạch theo như sơ đồ nguyên lý. Hình 21 là sản phẩm hoàn thiện của nhóm.



Hình 21: Sản phẩm phần cứng

7 Hoàn thiện sản phẩm

7.1 Các thông số của sản phẩm:

- Giá: 250.000 VND
- Công suất: 5W
- Kích thước: 13x8x2 cm
- Độ chia nhỏ nhất: 0.005V
- Sai số: 0.5%
- Thời gian đáp ứng: 0.1s
- Tuổi thọ: 10 năm (dự đoán)

7.2 Hướng dẫn sử dụng:

7.2.1 Một số lưu ý trước khi sử dụng:

Thiết bị cần được giữ trong môi trường thoáng mát (nhiệt độ từ 10 – 35°C), tránh nắng nóng và môi trường ẩm ướt, sẽ gây hao mòn và hư hại cho linh kiện. Hệ thống không có chức năng chống nước.

Máy đo điện áp có tầm đo đổi với chế độ điện áp ngõ vào đơn là 5 Volt. Đổi với ngõ vào vi sai, chế độ gain x1 tầm đo là 5 volt, gain x10 là 0.5 volt, gain x200 là 0.025 volt. Người dùng

cần hiểu rõ sản phẩm và sử dụng đúng chức năng của sản phẩm. Nếu đưa điện áp ngõ vào vượt ngoài mức điện áp quy định, hệ thống không hiển thị được chính xác. Nếu điện áp cấp vào quá lớn, sản phẩm có thể bị hư hại.

Sản phẩm đi kèm với dây cáp nguồn là adaptor 220VAC - 5VDC. Nếu dây nguồn bị hư hại, cần mua lại dây nguồn tương thích, đúng yêu cầu. Nếu sử dụng sai dây nguồn hệ thống sẽ không hoạt động và có thể gây hư hại cho sản phẩm.

7.2.2 Hướng dẫn hoạt động:

Khi cần đo điện áp, người dùng cần cắm phích điện trực tiếp vào nguồn điện 220VAC - 50Hz (diện áp tiêu chuẩn cho các thiết bị gia dụng ở Việt Nam). Trên màn hình LCD sẽ hiện dòng chữ "Voltmeter" và yêu cầu chọn chế độ. Hệ thống có hai chế độ đo điện áp: đo điện áp ngõ vào đơn hoặc đo điện áp ngõ vào vi sai. Người dùng có thể chọn bằng cách nhấn vào nút tương ứng trên sản phẩm.

Khi đã chọn chế độ đo, nếu lựa chọn là ngõ vào đơn, người dùng chỉ cần cắm dây kẹp ngõ vào đơn (màu đỏ) vào những vị trí muốn đo. Giá trị điện áp sẽ liên tục hiện trên màn hình LCD.

Nếu lựa chọn là đo ngõ vào vi sai (tức là đo độ chênh lệch điện áp giữa hai điểm), màn hình sẽ hiện yêu cầu chọn độ lợi, bao gồm gain x1, gain x10 và gain x200. Người dùng cần chọn chế độ bằng cách nhấn vào nút tương ứng trên sản phẩm. Sau khi đã chọn độ lợi, người dùng cần cắm hai dây màu trắng theo đường dẫn như sau:

- gain x1: Cắm dây bên trái với điện áp (+), dây bên phải với điện áp (-).
- gain x10: Cắm dây bên phải với điện áp (+), dây bên trái với điện áp (-).
- gain x200: Cắm dây bên phải với điện áp (+), dây bên trái với điện áp (-).

Điện áp vi sai sẽ liên tục được hiện trên màn hình LCD. Lưu ý, nếu mắc dây ngược chiều điện áp (đảo chiều dây + và -), trên màn hình sẽ hiện "OPPOSITE".

Nếu muốn chọn lại chế độ đo, người dùng nhấn nút "RESET" trên sản phẩm và thực hiện lại quy trình hướng dẫn. Ngoài ra, cần rút điện khi không sử dụng.

8 Kết luận

Báo cáo này đã trình bày đầy đủ, chi tiết về quá trình thiết kế, xây dựng, kế hoạch và thực hiện hệ thống. Qua các phần, nhóm đã đi qua toàn bộ các khâu trong một quy trình thiết kế hệ thống nhúng, từ việc xác định yêu cầu hệ thống, đặc tả hệ thống, thiết kế phần cứng, phần mềm, cho đến việc thi công và hoàn thiện sản phẩm. Trong quá trình thực hiện, nhóm đã gặp phải một số thách thức nhưng đã giải quyết hiệu quả nhờ sự phối hợp chặt chẽ và áp dụng các phương pháp thiết kế hợp lý và tìm tòi, học hỏi thêm các kiến thức liên quan như cách vẽ PCB bằng Altium, cách nạp chương trình vào vi điều khiển, cách lựa chọn mua các linh kiện, in mạch, rửa mạch, hàn mạch, ... Qua đó có những bài học và kết luận được đúc kết trong suốt quá trình.

Về mặt tích cực, nhóm đã thành công thiết kế, hoàn thiện một sản phẩm đúng theo mục tiêu và yêu cầu ban đầu. Sản phẩm đáp ứng được đầy đủ các tiêu chí về ràng buộc. Đó là: Giá thành sản phẩm là 250.000 VND (≤ 300.000 VND); công suất ≤ 10 Watt, kích thước sản phẩm là 13x8x2 cm (diện tích $\leq 300cm^2$); tuổi thọ theo dự đoán là trên 7 năm (vì nhóm không có đủ thời gian kiểm nghiệm nên chỉ có thể đưa ra dự đoán tương đối chính xác); thời gian đáp ứng $<50ms$. Và quan trọng nhất là: độ chính xác của sản phẩm đáp ứng được yêu cầu, với sai số là 0.5%.

Ngoài ra, trong suốt quá trình, mỗi người trong nhóm đều học được cách làm việc nhóm sao cho hiệu quả, thích ứng được với việc làm nhóm và giải quyết các vấn đề khi làm sai yêu cầu, sản phẩm chưa hoạt động đúng,... và có thêm rất nhiều bài học kinh nghiệm khác để cuối cùng có thể đưa ra được một sản phẩm hoạt động ổn định và một bài báo cáo hoàn thiện đúng với tiến độ đưa ra.

Tuy nhiên, vẫn còn một số mặt hạn chế. Dù đáp ứng tiêu chí ban đầu nhưng nhóm vẫn chưa thể cải thiện được nhiều yếu tố của một máy đo volt. Ngoài ra, nhóm chưa tiếp xúc nhiều với các linh kiện điện tử nên tốn rất nhiều thời gian để nghiên cứu cách nối dây, chỉnh sửa để mạch hoạt động. Và vì chưa có nhiều kinh nghiệm trước đó, nhóm đã phải làm lại một số giai đoạn nhiều lần, tốn thêm chi phí nghiên cứu cho sản phẩm.

Sau cùng, hệ thống đã được xây dựng và hoạt động ổn định, đáp ứng đầy đủ các yêu cầu ban đầu. Nhóm cũng đưa ra các bước hướng dẫn sử dụng chi tiết cung cấp giúp người dùng dễ dàng làm quen và sử dụng hệ thống một cách hiệu quả. Bên cạnh đó, nhóm cũng đưa ra hướng phát triển tiếp theo của sản phẩm nên là giảm giá thành của sản phẩm và nâng tầm đo điện áp lên mức cao hơn. Với kết quả đạt được, nhóm tin rằng sản phẩm này sẽ có ứng dụng thực tế cao và phát triển, mở rộng hơn trong tương lai.

Hiện tại, sản phẩm và báo cáo đã hoàn thành nhưng có thể vẫn còn nhiều thiếu sót, nhóm rất mong nhận được sự góp ý của giảng viên.

9 Tài liệu kham khảo

Anasa Store, "Hướng dẫn sử dụng Mạch nạp ISP 89S/AVR", 21 tháng 10, 2016

Bùi Quốc Bảo, "Bài giảng Thiết Kế Hệ Thống Nhúng", Bộ môn Điện Tử, Trường Đại Học Bách Khoa - Đại Học Quốc Gia Thành Phố Hồ Chí Minh

Dặng Ngọc Hạnh, "Bài giảng Xử Lý Số Tín Hiệu", Bộ môn Viễn Thông, Trường Đại Học Bách Khoa - Đại Học Quốc Gia Thành Phố Hồ Chí Minh

Dặng Nguyên Châu, "Bài giảng Xử Lý Số Tín Hiệu", Bộ môn Viễn Thông, Trường Đại Học Bách Khoa - Đại Học Quốc Gia Thành Phố Hồ Chí Minh

Hà Hoàng Kha, "Bài giảng Xử Lý Số Tín Hiệu", Bộ môn Viễn Thông, Trường Đại Học Bách Khoa - Đại Học Quốc Gia Thành Phố Hồ Chí Minh 2024

Hoàng Trang, Lưu Phú, Nguyễn Lý Thiên Trường, Lê Thị Kim Anh, Nguyễn Trọng Luật, Bùi Quốc Bảo và Trần Hoàng Linh, "Giáo trình Vi Xử Lý", Bộ môn Điện Tử, Trường Đại Học Bách Khoa - Đại Học Quốc Gia Thành Phố Hồ Chí Minh 2024

Lập Trình Nhúng A-Z, "[Lập Trình ATTiny13] Bài 3: ADC", 20 tháng 7, 2021

Lập trình - Điện Tử, "Lập trình AVR - ATMega16 / ATMega32", 21 tháng 8, 2023

Lưu Phú, "Bài giảng Vi Xử Lý", Bộ môn Điện Tử, Trường Đại Học Bách Khoa - Đại Học Quốc Gia Thành Phố Hồ Chí Minh 2024

Microchip, "ATmega16 Datasheet"

Nguyễn Lý Thiên Trường, "Bài giảng Vi Xử Lý", Bộ môn Điện Tử, Trường Đại Học Bách Khoa TPHCM 2024

Nguyễn Phan Hải Phú, "Bài giảng Vi Xử Lý", Bộ môn Điện Tử, Trường Đại Học Bách Khoa - Đại Học Quốc Gia Thành Phố Hồ Chí Minh 2024

Nguyễn Trung Hiếu, "Bài giảng: ADC and Analog Comparator", Bộ môn Điện Tử, Trường Đại Học Bách Khoa - Đại Học Quốc Gia Thành Phố Hồ Chí Minh, 23 tháng 11, 2023

Nguyễn Trung Hiếu, "Bài giảng Thiết Kế Hệ Thống Nhúng", Bộ môn Điện Tử, Trường Đại Học Bách Khoa - Đại Học Quốc Gia Thành Phố Hồ Chí Minh

Nguyễn Thanh Tuấn, "Bài giảng Xử Lý Số Tín Hiệu", Bộ môn Viễn Thông, Trường Đại Học Bách Khoa - Đại Học Quốc Gia Thành Phố Hồ Chí Minh 2024

Nguyễn Trọng Luật, "Bài giảng Vi Xử Lý", Bộ môn Điện Tử, Trường Đại Học Bách Khoa - Đại Học Quốc Gia Thành Phố Hồ Chí Minh 2024

Lê Thị Kim Anh, "Bài giảng Vi Xử Lý", Bộ môn Điện Tử, Trường Đại Học Bách Khoa - Đại Học Quốc Gia Thành Phố Hồ Chí Minh 2024