REPORT

THIẾT KẾ VÀ THỰC HIỆN HỆ THỐNG ĐẾM SỐ BƯỚC CHÂN

Design and Implementation of a Pedometer

	Họ và tên (Full name)	**Mã SV (ID)**	**Đóng góp (Contribution)**
Thành viên 1 (Member 1)	Nguyễn Trường Quyền	20020708	– Mua linh kiện – Ghép nối, đóng gói hệ thống – Xây dựng thuật toán đếm bước chân từ giá trị cảm biến – Đọc giá trị cảm biến MPU6050 và xử lý – Lập trình hiển thị LCD – Làm báo cáo mục **3.2, 4**
Thành viên 2 (Member 2)	Hoàng Văn Quyến	20020710	– Mua linh kiện – Ghép nối, đóng gói hệ thống – Tạo repo github – Lập trình timer – Xử lý ngắt cho nút bấm, hiển thị LED – Làm phần còn lại của báo cáo
Tên/Địa chỉ Repo Github	Tên: BTL_NhapMonHeThongNhung Địa chỉ: https://github.com/QuyenLe1501/BTL_NhapMonHeThongNhung.git		

Mục lục

1. Giới thiệu 3

2. Yêu cầu đối với thiết kế 3

2.1. Yêu cầu đối với thiết kế 3

2.2. Đặc tả kỹ thuật 3

2.2.1. Chức năng 3

2.2.2. Thiết kế phần cứng 4

2.2.3. Thiết kế phần mềm 4

3. Thực hiện hệ thống 4

3.1. Kiến trúc phần cứng 4

3.1.1. Khối xử lý trung tâm 4

3.1.2. Cảm biến đầu vào 5

3.1.3. Các LED và SWITCH 6

3.1.4. LCD 7

3.2. Thuật toán sử dụng 7

3.2.1. Ý tưởng thuật toán 7

3.2.2. Thể hiện ý tưởng 8

3.3. Lập trình phần mềm (chi tiết tại repo github) 9

3.3.1. Khởi tạo các LED 9

3.3.2. Khởi tạo các Switch 9

3.3.3. Khởi tạo LCD 10

3.3.4. Khởi tạo Timer 10

3.3.5. Khởi tạo, đọc giá trị MPU6050 11

3.3.6. Chương trình điều khiển (Hàm main()) 12

4. Kiểm chứng 13

5. Kết luận 14

6. Lời cảm ơn 14

Danh sách hình 15

Danh sách Bảng 16

1. Giới thiệu

Dự án này nhằm mục đích đếm số bước chân dựa trên dữ liệu từ cảm biến gia tốc MPU6050. STM32f103c8t6 được sử dụng làm vi xử lý chính để xử lý dữ liệu từ cảm biến và hiển thị kết quả lên LCD 16x2. Cảm biến gia tốc MPU6050 được kết nối với STM32f103c8t6 thông qua các chân giao tiếp I2C để thu thập dữ liệu gia tốc. Kết quả đếm số bước chân được hiển thị trên màn hình LCD16x2, cho phép người dùng dễ dàng quan sát và theo dõi số bước đã đi. Có hệ thống nút bấm và đèn led để chuyển đổi cũng như hiện thị trạng thái hoạt động của hệ thống.

2. Yêu cầu đối với thiết kế

2.1. Yêu cầu đối với thiết kế

Yêu cầu thiết kế:

- *Chức năng:* Đếm số bước chân khi người sử dụng thiết bị di chuyển và hiển thị kết quả trên màn LCD.
- *Các đầu vào:*
    - Hệ thống sử dụng cảm biến gia tốc MPU6050 để đo số bước chân.
    - SW1 để chuyển đổi bộ đếm bước chân hoạt động giữa trạng thái hoạt động và trạng thái dừng. Khi bộ đếm dừng hoạt động, nếu bấm SW1 bộ đếm chuyển sang trạng thái hoạt động, và ngược lại. Khi chuyển từ trạng thái dừng sang trạng thái hoạt động, bộ đếm tiếp tục đếm từ 0.
    - SW2 để xóa (reset) kết quả đếm của bộ đếm về 0.
- *Các đầu ra:*
    - Hệ thống có 2 lối ra trạng thái: LED xanh nhấp nháy với tần số 1Hz khi bộ đếm hoạt động; tắt khi hệ thống dừng hoạt động. LED đỏ sáng khi hệ thống dừng hoạt động; tắt khi hệ thống hoạt động.
    - LCD: Hiện thị số bước chân
- Bộ đếm sử dụng một timer để xác định khoảng thời gian nhấp nháy LED xanh.

2.2. Đặc tả kỹ thuật

2.2.1. Chức năng

- Hệ thống đếm bước chân dựa trên dữ liệu cảm biến gia tốc MPU6050.
- Kết quả được hiển thị trên màn hình LCD16x2.
- Nút SW1 chuyển đổi trạng thái hoạt động và trạng thái dừng của bộ đếm.
- Nút SW2 sẽ đặt kết quả đếm về 0.

- Led xanh nháy với tần số 1Hz khi bộ đếm hoạt động, tắt khi bộ đếm dừng hoạt động.
- Led đỏ sáng khi hệ thống dừng hoạt động và ngược lại.

2.2.2. Thiết kế phần cứng

- Vi điều khiển STM32f103c8t6.
- Cảm biến gia tốc MPU6050.
- LCD16x2.
- 2 bút bấm, 1 led xanh, 1 led đỏ.

2.2.3. Thiết kế phần mềm

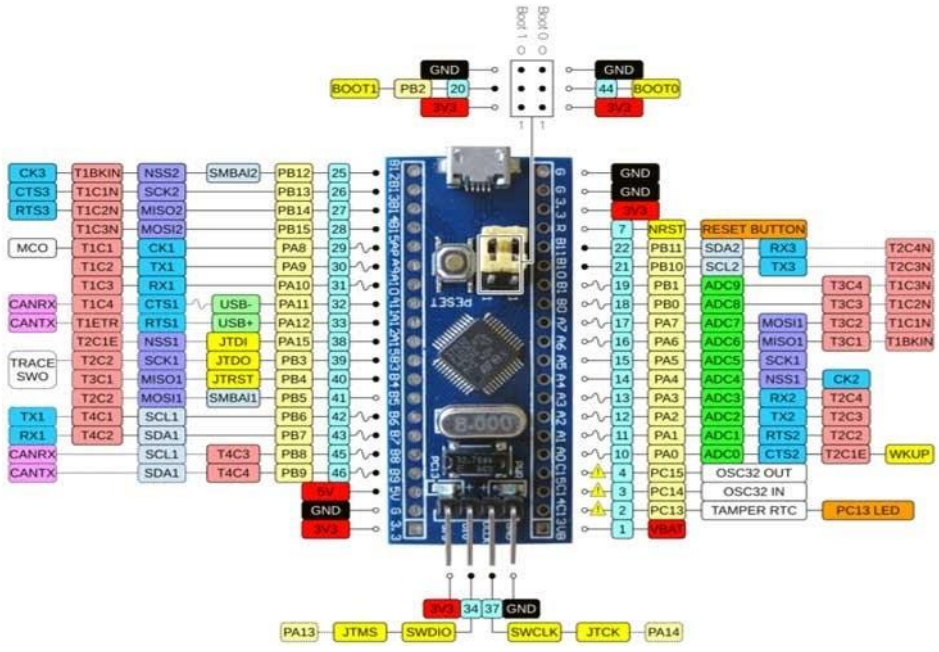
- Ngôn ngữ lập trình: C.
- Môi trường phát triển: Keil uVersion 5.
- Quản lý code: Github.

3. Thực hiện hệ thống

3.1. Kiến trúc phần cứng

3.1.1. Khối xử lý trung tâm

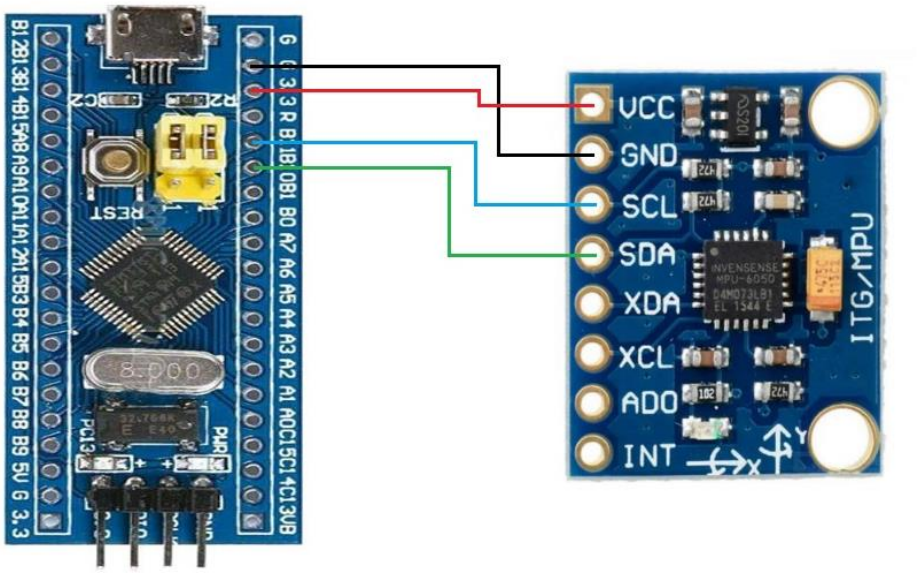
Vi điều khiển STM32f103c8t6 là một vi điều khiển thuộc kiếm trúc ARM với lõi vi xử lý ARM COTEX M3, tần số hoạt động tối đa 72MHz



Hình 1: Giao diện ghép nối I/O của STM32f103c8t6

3.1.2. Cảm biến đầu vào

Cảm biến gia tốc MPU 6050 được kết nối và giao tiếp với vi điều khiển STM32 thông qua giao thức I2C:

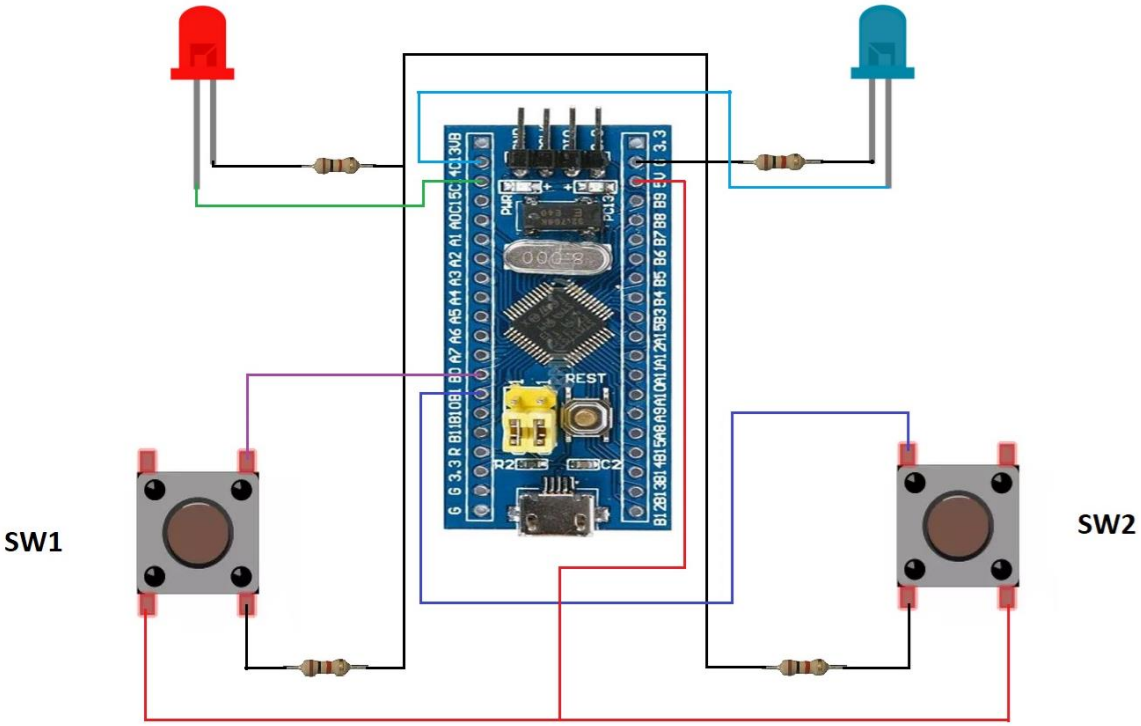


Hình 2: Sơ đồ kết nối chân cảm biến gia tốc MPU6050

MPU pin	STM32 Pin
VCC	3.3V
GND	G
SDA	B10
SCL	B11

Bảng 1: Kết nối cảm biến MPU6050 với bo mạch STM32f103c8t6

3.1.3. Các LED và SWITCH

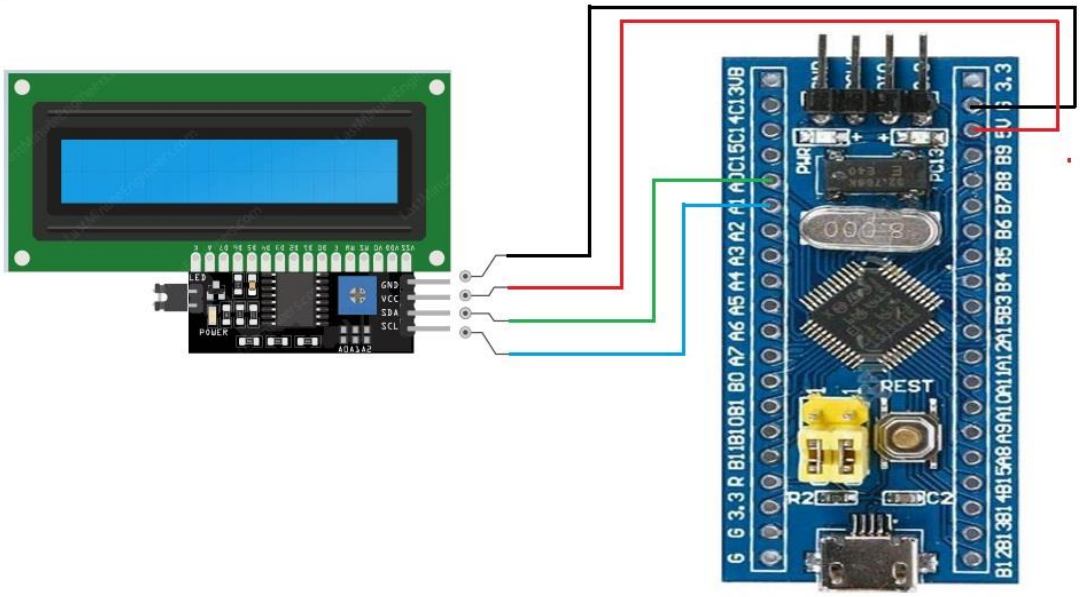


Hình 3: Sơ đồ kết nối chân các LED và các SWITCH

LED/SWITCH	STM32 Pin
Red	C14
Green	C13
SW1	B0
SW2	B1

Bảng 2: Kết nối các LED và SWITCH trên bo mạch STM32f103c8t6

3.1.4. LCD



Hình 4: Sơ đồ kết nối chân chân LCD

MODULE I2C	STM32 Pin
VCC	5V
GND	G
SDA	A0
SCL	A1

Bảng 3: Kết nối LCD sử dụng module chuyển đổi I2C trên bo mạch STM32f103c8t6

3.2. Thuật toán sử dụng

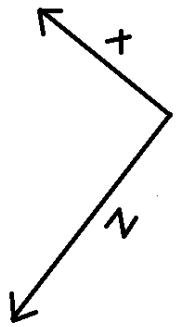
3.2.1. Ý tưởng thuật toán

Ta tính trung bình gia tốc đi thẳng từ lúc bắt đầu bước đi đến khi kết thúc bước đi. Nếu gia tốc đó nhỏ hơn x (ở đây ta lấy x = 0.2) hoặc time của bước đi quá ngắn thì sẽ không đếm, và ngược lại bước đếm sẽ tăng lên 1.

3.2.2. *Thể hiện ý tưởng*

Khi cầm máy ta có 2 hướng đo cần quan tâm là hướng do x và hướng do z (nếu để cảm biến trên mặt bàn thì z trùng với trục gia tốc trọng trường, hướng x là hướng đi thẳng của ta, và lúc này đo hướng z ta thu được G $\approx$ 9.8 và hướng x thu được G = 0)

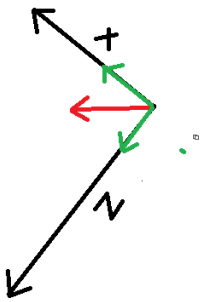
Khi ta cầm máy lên sẽ có 1 độ nghiêng lúc này hướng x và z sẽ có hình như sau:



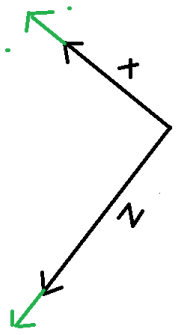
Lúc này khi ta đứng yên thì trục x và trục z đề bị tác động bởi gia tốc trọng trường và ta sẽ có Gx(giá trị gia tốc đo tại x) , Gz giá trị gia tốc đo tại z có liên hệ như sau

$$\sqrt{Gx^2 + Gz^2} = G\ trọng\ trường$$ (đặt là Gtb)

Bình thường nếu không di chuyển thì Gtb $\leq$ 9,9. Khi có sự di chuyển thì Gx và Gz đều bị ảnh hưởng bởi Gtb và Gia tốc di chuyển A, ta có liên hệ như hình vẽ



Khi tổng hợp các vecto lại sẽ có dạng:



Ta có mũi tên xanh là gia tốc sau khi hợp lại mũi tên đen là gia tốc lúc ban đầu khi chưa di chuyển và giờ Gtb > 0.99 (bắt đầu đi)

Từ đó để tìm gia tốc A ta có công thức:

$$\sqrt{(Gx\ đo\ được\ -\ Gx\ đứng\ im\ )^2 + (Gz\ đo\ được - Gz\ đứng\ im)^2} = A$$

Từ đó mỗi 1 đơn vị thời gian ta lại cập nhật A và cộng chúng lại với nhau đến khi Gtb < 0.99 (trở lại về trạng thái đứng)

Tổng đó ta đặt là Asum

Từ đó Atb = Asum / tổng đơn vị thời gian đi.

So sánh Atb với 0.2 nếu Atb ≥ 0.2 và thời gian đi phải đủ dài ở đây ta lấy (50ms) thì mới tính bước chân tăng lên 1 còn không thì giữ nguyên

Lưu ý:

Vì khi đo sẽ có độ nhiễu nên thi thoảng Gtb sẽ nháy quá mức nên ta sẽ giới hạn nó lại (Gtb < 2) và dùng cách tính trung bình (Atb) cũng sẽ làm giảm nhiễu đưa ra thông số chính xác

3.3. Lập trình phần mềm (chi tiết tại repo github)

3.3.1. Khởi tạo các LED

- **Khởi tạo đối tượng cấu hình chân GPIO**

  GPIO_InitTypeDef gpioInit;

- **Cấp xung và cài đặt chân cho các led đỏ - C14, xanh – C13:**

```
RCC_APB2PeriphClockCmd(RCC_APB2Periph_GPIOC, ENABLE);
gpioInit.GPIO_Mode = GPIO_Mode_Out_PP;
gpioInit.GPIO_Pin = GPIO_Pin_13 | GPIO_Pin_14;
gpioInit.GPIO_Speed = GPIO_Speed_50MHz;
GPIO_Init(GPIOC, &gpioInit);
```

3.3.2. Khởi tạo các Switch

- **Cấp xung clock cho cổng GPIO, thiết lập ngắt và set thứ tự ưu tiên của ngắt, SW1 – B0 có chức năng bật tắt bộ đếm, nên sẽ có độ ưu tiên cao hơn SW2 – B1.**

  interrupt_init(RCC_APB2Periph_GPIOB, GPIO_Pin_0, GPIO_PortSourceGPIOB, GPIO_PinSource0, EXTI_Line0, EXTI0_IRQn, 0x01);

  interrupt_init(RCC_APB2Periph_GPIOB, GPIO_Pin_1, GPIO_PortSourceGPIOB, GPIO_PinSource1, EXTI_Line1, EXTI1_IRQn, 0x00);

3.3.3. Khởi tạo LCD

- **Cấu hình các chân GPIO của module chuyển đổi I2C**

  i2c_setup(GPIOA, GPIO_Pin_0, GPIO_Pin_1, RCC_APB2Periph_GPIOA);

- **Khởi tạo, cấu hình phần hiển thị của LCD: cài đặt các chế độ hiển thị, xóa màn hình và hiển thị con trỏ**

```
uint8_t i;
I2C_LCD_Delay_Ms(50);
My_I2C_Init();
for (i = 0; i < 8; ++i) {
  u8LCD_Buff[i] = 0;
}
I2C_LCD_FlushVal();
u8LCD_Buff[LCD_RS] = 0;
I2C_LCD_FlushVal();
u8LCD_Buff[LCD_RW] = 0;
I2C_LCD_FlushVal();
I2C_LCD_Write_4bit(0x03);
I2C_LCD_Delay_Ms(5);
I2C_LCD_Write_4bit(0x03);
I2C_LCD_Delay_Ms(1);
I2C_LCD_Write_4bit(0x03);
I2C_LCD_Delay_Ms(1);
I2C_LCD_Write_4bit(MODE_4_BIT >> 4);
I2C_LCD_Delay_Ms(1);
I2C_LCD_WriteCmd(MODE_4_BIT);
I2C_LCD_WriteCmd(DISP_ON);
I2C_LCD_WriteCmd(CURSOR_ON);
I2C_LCD_WriteCmd(CLR_SCR);
```

3.3.4. Khởi tạo Timer

- **Tần số hoạt động của STM32f103c8t6 là 72MHz, với giá trị *Prescaler* = 71 ta dễ dàng tính toán được tần số của Timer là:**

$$F_{timer} = 72MHz / (Prescaler + 1) = 1Mhz$$

TIM_TimeBaseInitTypeDef timerInit;
RCC_APB1PeriphClockCmd(RCC_APB1Periph_TIM2, ENABLE);
timerInit.TIM_CounterMode = TIM_CounterMode_Up;
timerInit.TIM_Period = 0xFFFF;
timerInit.TIM_Prescaler = 72 - 1;
TIM_TimeBaseInit(TIM2, &timerInit);

3.3.5. Khởi tạo, đọc giá trị MPU6050

- **Setup MPU6050: Cấp xung và thiết lập các chân GPIO cho MPU6050 để sử dụng cho giao thức I2C**

  i2c_setup(GPIOB, GPIO_Pin_10, GPIO_Pin_11, RCC_APB2Periph_GPIOB);

- **Khởi tạo I2C: Cấu hình các thanh ghi của MPU6050**

  i2c_init();

  CTR_Single_Write(MPU6050_ADDR, MPU6050_SMPLRT_DIV, 0x00);

  CTR_Single_Write(MPU6050_ADDR, MPU6050_CONFIG, 0x00);

  CTR_Single_Write(MPU6050_ADDR, MPU6050_GYRO_CONFIG, 0x08);

  CTR_Single_Write(MPU6050_ADDR, MPU6050_ACCEL_CONFIG, 0x02);

  CTR_Single_Write(MPU6050_ADDR, MPU6050_PWR_MGMT_1, 0x01);

- **Đọc dữ liệu gia tốc từ MPU6050 và chuyển đổi thành đơn vị chuẩn**

  CTR_MPU6050_BUF[0] = CTR_Single_Read(MPU6050_ADDR,ACCEL_XOUT_H);

  CTR_MPU6050_BUF[1] = CTR_Single_Read(MPU6050_ADDR,ACCEL_XOUT_L);

  a =(uint16_t)(CTR_MPU6050_BUF[0] <<8)|CTR_MPU6050_BUF[1];

  mpuData[0] = a / 16384.0;

  CTR_MPU6050_BUF[2] = CTR_Single_Read(MPU6050_ADDR,ACCEL_YOUT_H);

  CTR_MPU6050_BUF[3] = CTR_Single_Read(MPU6050_ADDR,ACCEL_YOUT_L);

  a = (uint16_t)(CTR_MPU6050_BUF[2] <<8)|CTR_MPU6050_BUF[3];

  mpuData[1] = a /16384.0;

  CTR_MPU6050_BUF[4] = CTR_Single_Read(MPU6050_ADDR,ACCEL_ZOUT_H);

  CTR_MPU6050_BUF[5] = CTR_Single_Read(MPU6050_ADDR,ACCEL_ZOUT_L);

  a = (uint16_t)(CTR_MPU6050_BUF[4] <<8)|CTR_MPU6050_BUF[5];

  mpuData[2] = a/16384.0;

- **Đọc dữ liệu giao tốc góc từ MPU6050**

  CTR_MPU6050_BUF[0] = CTR_Single_Read(MPU6050_ADDR,GYRO_XOUT_H);

  CTR_MPU6050_BUF[1] = CTR_Single_Read(MPU6050_ADDR,GYRO_XOUT_L);

  a =(uint16_t)(CTR_MPU6050_BUF[0] <<8)|CTR_MPU6050_BUF[1];

  mpuData[0] = a ;

  CTR_MPU6050_BUF[2] = CTR_Single_Read(MPU6050_ADDR,GYRO_YOUT_H);

  CTR_MPU6050_BUF[3] = CTR_Single_Read(MPU6050_ADDR,GYRO_YOUT_L);

a = (uint16_t)(CTR_MPU6050_BUF[2] <<8)|CTR_MPU6050_BUF[3];

mpuData[1] = a ;

CTR_MPU6050_BUF[4] = CTR_Single_Read(MPU6050_ADDR,GYRO_ZOUT_H);

CTR_MPU6050_BUF[5] = CTR_Single_Read(MPU6050_ADDR,GYRO_ZOUT_L);

a = (uint16_t)(CTR_MPU6050_BUF[4] <<8)|CTR_MPU6050_BUF[5];

mpuData[2] = a ;

3.3.6. Chương trình điều khiển (Hàm main())

- **Kiểm tra trạng thái hoạt động của bộ đếm (SW1) thông qua đọc giá trị ngắt**

  logic_led = read_interrupt();

- **Trường hợp bộ đếm đang hoạt động**

  - **Thực hiện tắt đèn đỏ và đọc giá trị cảm biến MPU6050**

```
GPIO_ResetBits(GPIOC, GPIO_Pin_14);
I2C_MPU6050_Setup();
CTR_MPU6050_getvalue = CTR_READ_ACCEL_MPU3050();
G_sum = sqrt(pow(CTR_MPU6050_getvalue[0], 2) + pow(CTR_MPU6050_getvalue[2],2));
```

  - **Sử dụng thuật toán đã trình bày ở phần 3.2 để xử lý dữ liệu đọc được từ cảm biến, đếm bước chân và hiển thị lên LCD**

```
if (start == 0 && G_sum < 0.99 ) {
    Gx = CTR_MPU6050_getvalue[0];
    Gz = CTR_MPU6050_getvalue[2];
    if (start == 0 && 0.99 <= G_sum && G_sum <= 2)
      start =1;
    if (start == 1 && 0.99 <= G_sum && G_sum <= 2 )
    {
      index1 ++;
      gtb_X = sqrt(pow(CTR_MPU6050_getvalue[0] - Gx, 2) + pow(CTR_MPU6050_getvalue[2] - Gz,2)) + gtb_X;
    }
    if ( start == 1 && G_sum < 0.99 )
    {
      gtb_X = gtb_X / index1;
      if (gtb_X >= 0.2 && index1 >= 45 )
      {
        count ++;
        I2C_LCD_Setup();
        I2C_LCD_Clear();
        sprintf(CTR_datasend,"Buoc chan : %d", count);
        I2C_LCD_Puts(CTR_datasend);
      }
      index1 = 0;
      gtb_X=0;
      start = 0;
    }
}
```

- **Nháy LED xanh với tần số 1kHz, thực hiện đếm 1Ms mỗi vòng lặp để bộ đếm có thể hoạt động liên tục với trễ thấp**

```
i++;
Delay_Ms(1);
if (i == 500)
  GPIO_SetBits(GPIOC, GPIO_Pin_13);
if (i == 1000)
{
  GPIO_ResetBits(GPIOC, GPIO_Pin_13);
  I2C_LCD_Setup();
  sprintf(CTR_datasend,"Z=%.4f ",CTR_MPU6050_getvalue[2]);
  I2C_LCD_NewLine();
  I2C_LCD_Puts(CTR_datasend);
  i = 0;
}
```

- **Trường hợp bộ đếm dừng hoạt động, bật đèn đỏ, tắt đèn xanh**

  GPIO_SetBits(GPIOC, GPIO_Pin_14);

  GPIO_ResetBits(GPIOC, GPIO_Pin_13);

4. Kiểm chứng

- Hình ảnh sản phẩm

Hình 5: Hình ảnh sản phẩm

- Video kiểm chứng

  https://tinyurl.com/2zvwafau

5. Kết luận

Kết quả thực nghiệm cho thấy hệ thống đếm bước chân hoạt động một cách chính xác. Chúng em đã thử nghiệm trên nhiều bước chân và thu được kết quả đáng tin cậy. Đồng thời, việc tích hợp cảm biến gia tốc MPU6050 và vi điều khiển STM32f103c8t6 vào một thiết bị nhỏ gọn và có thể đeo được đã mang lại sự tiện lợi cho người dùng.

Tuy nhiên, còn một số hạn chế cần cải tiến. Ví dụ, cải thiện thuật toán đếm bước chân để giảm thiểu sai số và tối ưu hóa việc sử dụng tài nguyên của vi điều khiển. Ngoài ra, có thể cân nhắc thêm tính năng như theo dõi khoảng cách đi được và tiêu thụ năng lượng.

6. Lời cảm ơn

Chúng em xin chân thành cảm ơn thầy TS. Nguyễn Kiêm Hùng đem đến cho chúng em kiến thức môn học vô cùng hữu ích để chúng em áp dụng vào dự án trên. Bản báo cáo của chúng em có điều gì sai sót, rất mong được sự góp ý, nhận xét từ thầy. Chúng em xin chân thành cảm ơn!

Danh sách hình ảnh

Hình 1: Giao diện ghép nối I/O của STM32f103c8t6. 4
Hình 2: Sơ đồ kết nối chân cảm biến gia tốc MPU6050 5
Hình 3: Sơ đồ kết nối chân các LED và các SWITCH.....6
Hình 4: Sơ đồ kết nối chân chân LCD.....7
Hình 5: Hình ảnh sản phẩm 13

Danh sách Bảng

Bảng 1: Kết nối cảm biến MPU6050 với bo mạch STM32f103c8t6 5
Bảng 2: Kết nối các LED và SWITCH trên bo mạch STM32f103c8t6 6
Bảng 3: Kết nối LCD sử dụng module chuyển đổi I2C trên bo mạch STM32f103c8t6 7