**TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI**

**TRƯỜNG ĐIỆN – ĐIỆN TỬ**

**BỘ MÔN KỸ THUẬT ĐO VÀ TIN HỌC CÔNG NGHIỆP**

.......................\*\*\*........................

A picture containing text, sign

Description automatically generated

**BÁO CÁO ĐỒ ÁN I**

**Đề tài:**

|  |  |  |
| --- | --- | --- |
| **Sinh viên thực hiện**  Nguyễn Huy Hoàng  Nguyễn Văn Dũng | **MSSV**  20191855 | **Lớp**  Tự động hóa 06-K64 |

**Giảng viên hướng dẫn: TS.Nguyễn Thanh Hường**

**Hà Nội - 2022**

# **MỞ ĐẦU**

Thiết bị phục vụ yêu cầu học phần Đồ án I, kỳ 20212. Mục tiêu của thiết bị là thu thập dữ liệu về vị trí tọa độ từ vệ tinh sau đó gửi thông tin vị trí hiện tại tới điện thoại người dùng để theo dõi vị trí hiện tại của mình. Trong báo cáo này sẽ trình bày chi tiết quá trình phát triển từ ý tưởng tới sản phẩm cuối cùng. Các bước có thể kể tới như vẽ sơ đồ nguyên lý, PCB, lập trình firmware kiểm tra và sửa lỗi.

Báo cáo gồm 5 chương:

Chương 1: Xác định yêu cầu

Chương 2: Thiết kế phần cứng

Chương 3: Lập trình Firmware cho thiết bị

Chương 4: Kết quả và đánh giá

Chương 5: Link share thư mục lập trình

Trước hết chúng em xin gửi lời cảm ơn tới cô Nguyễn Thanh Hường đã hướng dẫn chỉ dạy để chúng em hoàn thành sản phẩm cũng như báo cáo học phần Đồ án I. Vì hiểu biết hạn hẹp và thời gian hạn chế chúng em còn nhiều sai sót và chúng em mong nhận được những góp ý, sửa đổi từ cô để sản phẩm của chúng em được hoàn thiện hơn.

Chúng em xin chân thành cảm ơn cô!

# **MỤC LỤC**

[**MỞ ĐẦU** 1](#_Toc105704449)

[**MỤC LỤC** 1](#_Toc105704450)

[**DANH MỤC CÁC KÝ HIỆU VÀ CÁC CHỮ VIẾT TẮT** 2](#_Toc105704451)

[**DANH MỤC CÁC HÌNH VẼ, ĐỒ THỊ** 2](#_Toc105704452)

[**CHƯƠNG 1: XÁC ĐỊNH YÊU CẦU** 3](#_Toc105704453)

[**1.1.** **Phân tích yêu cầu** 3](#_Toc105704454)

[**1.2.** **Lưu đồ thuật toán** 4](#_Toc105704455)

[**CHƯƠNG 2: THIẾT KẾ PHẦN CỨNG** 4](#_Toc105704456)

[**2.1. Nguyên lí các khối phần cứng** 4](#_Toc105704457)

[**2.2. Ngoại vi của STM32 cần sử dụng** 12](#_Toc105704458)

[**2.3. Sơ đồ ghép nối** 18](#_Toc105704459)

[**2.4. Thiết kế mạch PCB** 18](#_Toc105704460)

[**CHƯƠNG 3: LẬP TRÌNH FIRMWARE CHO THIẾT BỊ** 20](#_Toc105704461)

[**3.1. Giới thiệu công cụ hỗ trợ và phần mềm lập trình** 20](#_Toc105704462)

[**3.2. Cấu hình thiết bị trên phần mềm STM32CubeMX** 25](#_Toc105704463)

[**3.3. Lập trình Firmware** 34](#_Toc105704464)

[**CHƯƠNG 4: KẾT QUẢ VÀ ĐÁNH GIÁ** 35](#_Toc105704465)

[**4.1. Kết quả đạt được** 35](#_Toc105704466)

[**4.2. Phân tích và đánh giá** 35](#_Toc105704467)

[**4.3. Đánh giá thành viên và công việc được giao** 35](#_Toc105704468)

[**CHƯƠNG 5: LINK SHARE THƯ MỤC LẬP TRÌNH** 36](#_Toc105704469)

[**KẾT LUẬN** 36](#_Toc105704470)

[**DANH MỤC TÀI LIỆU THAM KHẢO** 36](#_Toc105704471)

# **DANH MỤC CÁC KÝ HIỆU VÀ CÁC CHỮ VIẾT TẮT**

|  |  |  |
| --- | --- | --- |
| **Từ viết tắt** | **Nghĩa tiếng anh** | **Nghĩa tiếng việt** |
| **MCU** | Microcontroller unit | Vi điều khiển |
| **GSM** | Global System for Mobile Communications | Hệ thống thông tin di động toàn cầu |
| **GPS** | Global Positioning System | Hệ thống Định vị Toàn cầu |
| **UART** | Universal Asynchronous Receiver / Transmitter | bộ truyền nhận dữ liệu nối tiếp bất đồng bộ |
| **ADC** | Analog-to-digital converter | Bộ chuyển dổi tương tự sang số |
| **GPIO** | General-purpose input/output | Vào, ra với mục đích chung |
| **EXTI** | External Interupts | Ngắt ngoài |
| **PCB** | Printed Circuit Board | Mạch in |
| **NVIC** | Nested vector interrupt control | Điều khiển vector ngắt |

# **DANH MỤC CÁC HÌNH VẼ, ĐỒ THỊ**

[Hình 1.1 : Sơ đồ khối giao tiếp giữa STM32 và các khối phần cứng và các ngoại vi 3](#_Toc105701911)

[Hình 2. 1: Vi điều khiển STM32F103C8T6 4](#_Toc105701912)

[Hình 2. 2: Sơ đồ nguyên lý của STM32F103C8T6 trên phần mềm Altium 5](#_Toc105701913)

[Hình 2. 3: Hình ảnh của SIM800C 5](#_Toc105701914)

[Hình 2. 4: Sơ đồ nguyên lí của Chip SIM800C trên phần mềm Altium 6](#_Toc105701915)

[Hình 2. 5: Sơ đồ nguyên lí của Sim Card trên phần mềm Altium 7](#_Toc105701916)

[Hình 2. 6: Hình ảnh của Quectel L70R 7](#_Toc105701917)

[Hình 2. 7: Sơ đồ nguyên lý của Quectel L70R trên phần mềm Altium 8](#_Toc105701918)

[Hình 2. 8: Hình ảnh còi chíp và Led 8](#_Toc105701919)

[Hình 2. 9: Sơ đồ nguyên lí của còi chíp và Led trên phần mềm Altium 9](#_Toc105701920)

[Hình 2. 10: Sơ đồ nguyên lí của mạch hạ áp 3.3V cấp cho MCU 10](#_Toc105701921)

[Hình 2. 11: Sơ đồ nguyên lí của mạch hạ áp 3.3V cấp cho GPS 10](#_Toc105701922)

[Hình 2. 12: Sơ đồ nguyên lí của mạch sạc cho Pin Lithium 4.2V có bộ bảo vệ sạc 11](#_Toc105701923)

[Hình 2. 13: Sơ đồ truyền thông UART 11](#_Toc105701924)

[Hình 2. 14: Định dạng gói tin của giao thức UART 12](#_Toc105701925)

[Hình 2. 14: Mô tả bộ biến đổi ADC 13](#_Toc105701926)

[Hình 2. 15: Cấu trúc bên trong các chân GPIO của STM32 14](#_Toc105701927)

[Hình 2. 16: Cấu trúc tín hiệu của các chân ngắt ngoài 16](#_Toc105701928)

[Hình 2. 17: Sơ đồ ghép nối giữa MCU với các Module phần cứng và các ngoại vi 17](#_Toc105701929)

[Hình 2. 18: Hình ảnh mạch 3D mặt trên 18](#_Toc105701930)

[Hình 2. 19: Hình ảnh mạch 3D mặt dưới 19](#_Toc105701931)

[Hình 3. 1: Giao diện của phần mềm STM32CubeMX 20](#_Toc105701932)

[Hình 3. 2: Giao diện phần mềm VsCode 21](#_Toc105701933)

[Hình 3. 3: Giao diện của Github 23](#_Toc105701934)

[Hình 3. 4: Cấu hình thạch anh ngoại trong STM32CubeMX 25](#_Toc105701935)

[Hình 3. 5: Cấu hình chế độ Debug trong STM32CubeMX 25](#_Toc105701936)

[Hình 3. 6: Cấu hình ADC trong STM32CubeMX 26](#_Toc105701937)

[Hình 3. 7: Cấu hình cho Timer 1 trong STM32CubeMX 27](#_Toc105701938)

[Hình 3. 8: Cấu hình cho Timer 2 trong STM32CubeMX 28](#_Toc105701939)

[Hình 3. 9: Cấu hình UART1 cho Debug trong STM32CubeMX 29](#_Toc105701940)

[Hình 3. 10: Cấu hình UART2 cho giao tiếp SIM800C trong STM32CubeMX 30](#_Toc105701941)

[Hình 3. 11: Cấu hình UART3 cho giao tiếp L70R trong STM32CubeMX 31](#_Toc105701942)

[Hình 3. 12: Sơ đồ các chân được sử dụng và chức năng 32](#_Toc105701943)

[Hình 3. 13: Cấu hình các chân GPIOs trong STM32CubeMX 32](#_Toc105701944)

[Hình 3. 14: Cấu hình các ngắt của ngoại vi trong STM32CubeMX 33](#_Toc105701945)

[Hình 3. 15: Lập trình firmware trên VSCode 34](#_Toc105701946)

# **CHƯƠNG 1: XÁC ĐỊNH YÊU CẦU**

## **Phân tích yêu cầu**

Thiết bị phải đảm bảo các yêu cầu sau đây:

* + 1. ***Chức năng thu thập vị trí tọa độ GPS***

Thiết bị thu thập vị trí tọa độ GPS thông qua chip L70R, thực hiện việc kết nối với vệ tinh từ đó có thể đọc được vị trí tọa độ hiện tại của GPS. Dữ liệu đo được sẽ được gửi tới MCU và được xử lí.

* + 1. ***Chức năng gửi tin nhắn SMS***

Thiết bị kết nối tới nhà mạng thông qua Sim Card, nhờ đó thiết bị có thể gửi tin nhắn SMS, gọi điện tới các số khác hoặc thiết bị khác có gắn Sim để thông báo cũng như gửi vị trí tọa độ GPS cho người dùng.

* + 1. ***Chức năng sạc và kiểm tra Pin***

Thiết bị sử dụng nguồn Pin Lithium 3.7V, 2000mAh có chức năng sạc lại Pin thông qua mạch sạc có tính năng bảo vệ quá áp, quá dòng và chai Pin. Bên cạch đó, thiết bị còn có chức năng kiểm tra mức Pin hiện tại thông qua nút bấm trên mạch.

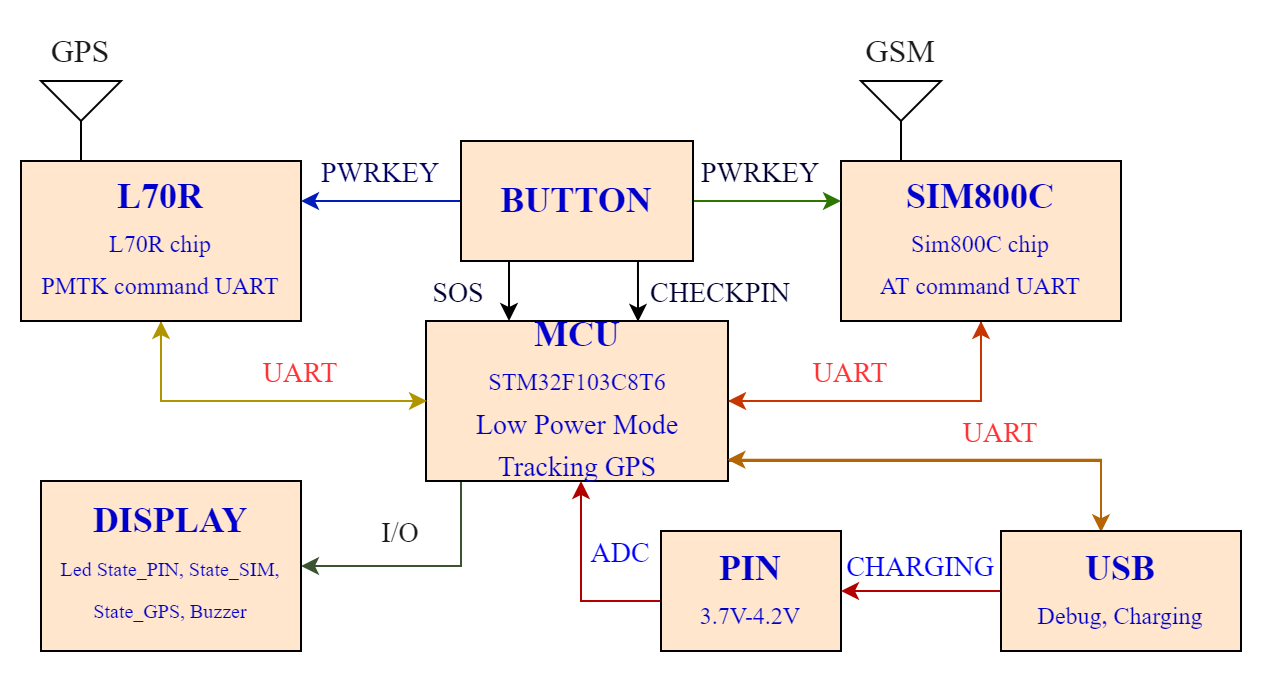
* + 1. ***Chức năng tiết kiệm Pin Sleep Mode***

Thiết bị khi chưa được sử dụng, dẫn đến trạng thái tiêu hao năng lượng không đáng có của khối MCU, khối GPS, khối GSM làm cho Pin mất nhiều năng lượng. Vì thế thiết bị có thêm chức năng Sleep Mode để khi thiết bị rảnh, không làm gì thì có thể tắt các khối đã nêu trên để rơi vào trạng thái ngủ tiết kiệm năng lượng.

* + 1. ***Chức năng update firmware từ xa OTA***

Thiết bị có chức năng cập nhật phần phềm từ xa mỗi khi cần nâng cấp hay sửa chữa. Khối SIM800C kết nối Internet tới OTA Drive để Download firmware về MCU và chạy chương trình đó.

## **Lưu đồ thuật toán**



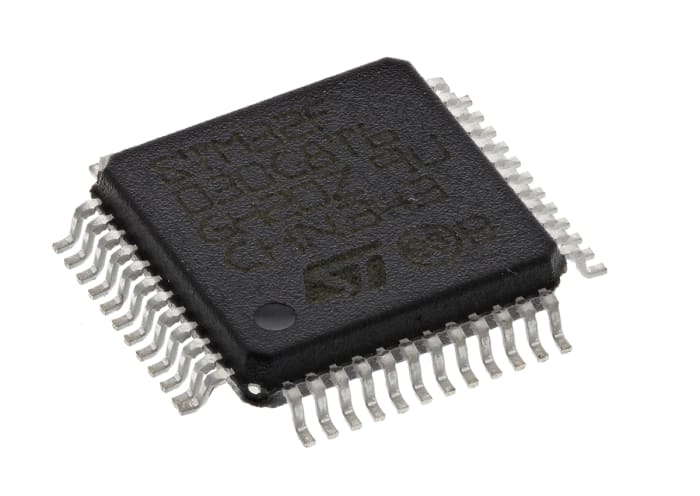
Hình 1.1 : Sơ đồ khối giao tiếp giữa STM32 và các khối phần cứng và các ngoại vi

# **CHƯƠNG 2: THIẾT KẾ PHẦN CỨNG**

## **2.1. Nguyên lí các khối phần cứng**

***2.1.1. Khối vi điều khiển MCU***

STM32 là một trong những dòng chip phổ biến của ST với nhiều họ thông dụng như F0,F1,F2,F3,F4….. Stm32f103 thuộc họ F1 với lõi là ARM COTEX M3. STM32F103 là vi điều khiển 32 bit, tốc độ tối đa là 72Mhz. Giá thành cũng khá rẻ so với các loại vi điều khiển có chức năng tương tự. Mạch nạp cũng như công cụ lập trình khá đa dạng và dễ sử dụng.



Hình 2. 1: Vi điều khiển STM32F103C8T6

* Chip: ARM 32 bits Cortex M3
* Điện áp hoạt động: 3,3V
* Tần số lớn nhất: 72MHz
* Bộ nhớ 64 128 Kb Flash, 20 Kb SRAM
* ADC 2x12 bit, tần số lấy mẫu 16MHz
* Timer: 7 bộ, 16 bits
* Kết nối: 2xI2C, 3xUSART, 2xSPI, CAN, USB 2.0 full speed
* Kiểu chân: LQFP48
* Watchdog: WWDG
* Chế độ tiết kiệm năng lượng: Sleep, Stop, Standby
* Hỗ trợ nạp xóa lên tới 100000 lần

Diagram, schematic

Description automatically generated

Hình 2. 2: Sơ đồ nguyên lý của STM32F103C8T6 trên phần mềm Altium

***2.1.2. Khối GSM - SIM800C***

Qr code

Description automatically generated

Hình 2. 3: Hình ảnh của SIM800C

Module SIM800C là loại module có khả năng nhắn tin SMS, nghe, gọi, GPRS,… như một điện thoại nhưng có kích thước nhỏ nhất trong các loại SIM (25 mm x 22 mm). Điều khiển SIM sử dụng bộ tập lệnh AT.

**Thông số kĩ thuật:**

* Nguồn cấp: 3.7 – 4.2VDC,
* Khe cắm SIM: MICROSIM
* Dòng khi ở chế độ chờ: 10mA
* Dòng khi hoạt động: 100mA đến 1A.
* Hỗ trợ 4 băng tần phổ biến.
* Kích thước: 25mm x 22mm

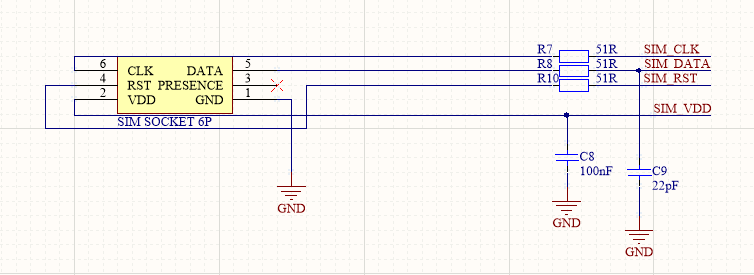
**Chức năng:**

* SIM800C giao tiếp với MCU thông qua giao tiếp UART
* Gửi tin nhắn SMS về vị trí tọa độ sử dụng API map tới số điện thoại người dùng
* Update Firmware từ xa sử dụng chức năng GPRS kết nối tới OTA Drive

Diagram, schematic

Description automatically generated

Hình 2. 4: Sơ đồ nguyên lí của Chip SIM800C trên phần mềm Altium



Hình 2. 5: Sơ đồ nguyên lí của Sim Card trên phần mềm Altium

***2.1.3. Khối GPS – Quectel L70R***

Qr code

Description automatically generated

Hình 2. 6: Hình ảnh của Quectel L70R

Module GPS L70R là module thu thập thông tin gửi về từ các vệ tinh GPS. Các thông tin này bao gồm: thời gian thực, vị trí, tọa độ, vận tốc,… của chính Module.

**Thông số kĩ thuật:**

* Nguồn cấp: 2.8V-4.3V
* Truyền thông: UART hỗ trợ tốc độ truyền Baud rate lên đến 115200bps
* Dòng khi hoạt động bình thường: 18mA, 3.3V
* Dòng khi nghỉ: 1mA, 3.3V
* Dải nhiệt độ: -40°C ~ +85°C
* Kích thước: 10.1±0.15 ×9.7±0.15 ×2.5±0.15mm

**Chức năng:**

* Quectel L70R giao tiếp với MCU thông qua giao tiếp UART
* Thu thập thông tin về tọa độ vị trí của người dùng từ vệ tinh

Diagram, schematic

Description automatically generated

Hình 2. 7: Sơ đồ nguyên lý của Quectel L70R trên phần mềm Altium

***2.1.4. Khối hiển thị***

A picture containing close

Description automatically generatedA picture containing tube

Description automatically generated

Hình 2. 8: Hình ảnh còi chíp và Led

Khối hiển thị bao gồm còi chíp và led có chức năng cảnh báo, hiển thị, tương tác với người dùng

**Thông số kĩ thuật:**

|  |  |
| --- | --- |
| **Còi chíp** | **Led** |
| Điện áp cấp: 1.5 ~ 12 VDC | Điện áp: 1.6V - 4V |
| Dòng điện tiêu thị: < 25mA | Công suất: 0,24W |
| Tần số: 2300 Hz (±500) | Cường độ dòng điện: 20mA |
| Biên độ âm thanh: > 80dB | Tuổi thọ: 50000 giờ |
| Nhiệt độ hoạt động: -20oC ~ 70oC | Nhiệt độ hoạt động: -20° – 60° |

**Chức năng:**

* Còi chíp được mỗi khi người dùng cần đọc vị trí tọa độ
* Led hiển thị mức Pin
* Led hiển thị trạng thái hoạt động của GPS và GSM

Diagram

Description automatically generatedDiagram, schematic

Description automatically generated

Hình 2. 9: Sơ đồ nguyên lí của còi chíp và Led trên phần mềm Altium

***2.1.5. Khối nguồn***

Thông số kĩ thuật:

|  |  |
| --- | --- |
| **Bộ hạ áp IC RT9013-33** | **Bộ sạc TP4056** |
| Dải điện áp vào: 2.2V-5.5V | Dải điện áp vào: 0.3V – 8V |
| Low Dropout : 250mV tại 500mA | Dòng điện: 1200mA |
| Điện áp ra: 3.3V | Điện áp sạc: 4.2V |

**Chức năng:**

* Sử dụng IC RT9013-33 để hạ áp điện áp từ Pin xuống 3.3V để cấp điện áp cho MCU và L70R
* Sử dụng bộ sạc TP4056 để sạc điện áp cho Pin lên tới 4.2V

Diagram, schematic

Description automatically generated

Hình 2. 10: Sơ đồ nguyên lí của mạch hạ áp 3.3V cấp cho MCU

Diagram, schematic

Description automatically generated

Hình 2. 11: Sơ đồ nguyên lí của mạch hạ áp 3.3V cấp cho GPS

Diagram, schematic

Description automatically generated

Hình 2. 12: Sơ đồ nguyên lí của mạch sạc cho Pin Lithium 4.2V có bộ bảo vệ sạc

## **2.2. Ngoại vi của STM32 cần sử dụng**

***2.2.1. Truyền thông nối tiếp UART***

UART tiếng anh là Universal Asynchronous Reciver/Transmister là một chuẩn giao tiếp không đồng bộ cho MCU và các thiết bị ngoại vi.

Chuẩn UART là chuẩn giao tiếp điểm và điểm, nghĩa là trong mạng chỉ có hai thiết bị đóng vai trò là transmister hoặc receiver.

A picture containing diagram

Description automatically generated

Hình 2. 13: Sơ đồ truyền thông UART

***Cách hoạt động của UART trong STM32:***

UART là giao thức truyền thông không đồng bộ, nghĩa là không có xung Clock, các thiết bị có thể hiểu được nhau nếu các

UART là truyền thông song công (Full duplex) nghĩa là tại một thời điểm có thể truyền và nhận đồng thời.

Trong đó quan trọng nhất là Baund rate (tốc độ Baund) là khoảng thời gian dành cho 1 bit được truyền và phải được cài đặt giống nhau ở gửi và nhận.

Định dạng gói tin như sau:

Graphical user interface, text

Description automatically generated

Hình 2. 14: Định dạng gói tin của giao thức UART

* **Start – Bit**

Start-bit còn được gọi là bit đồng bộ hóa được đặt trước dữ liệu thực tế. Nói chung, một đường truyền dữ liệu không hoạt động được điều khiển ở mức điện áp cao. Để bắt đầu truyền dữ liệu, truyền UART kéo đường dữ liệu từ mức điện áp cao (1) xuống mức điện áp thấp (0). UART thu được thông báo sự chuyển đổi này từ mức cao sang mức thấp qua đường dữ liệu cũng như bắt đầu hiểu dữ liệu thực. Nói chung, chỉ có một start-bit.

* **Stop – Bit**

Bit dừng được đặt ở phần cuối của gói dữ liệu. Thông thường, bit này dài 2 bit nhưng thường chỉ sử dụng 1 bit. Để dừng sóng, UART giữ đường dữ liệu ở mức điện áp cao.

* **Paratity Bit**

Bit chẵn lẻ cho phép người nhận đảm bảo liệu dữ liệu được thu thập có đúng hay không. Đây là một hệ thống kiểm tra lỗi cấp thấp & bit chẵn lẻ có sẵn trong hai phạm vi như Chẵn lẻ – chẵn lẻ cũng như Chẵn lẻ – lẻ. Trên thực tế, bit này không được sử dụng rộng rãi nên không bắt buộc.

* **Data Frame**

Các bit dữ liệu bao gồm dữ liệu thực được truyền từ người gửi đến người nhận. Độ dài khung dữ liệu có thể nằm trong khoảng 5 & 8. Nếu bit chẵn lẻ không được sử dụng thì chiều dài khung dữ liệu có thể dài 9 bit. Nói chung, LSB của dữ liệu được truyền trước tiên sau đó nó rất hữu ích cho việc truyền.

***2.2.2. Chuyển đổi tương tự - số ADC***

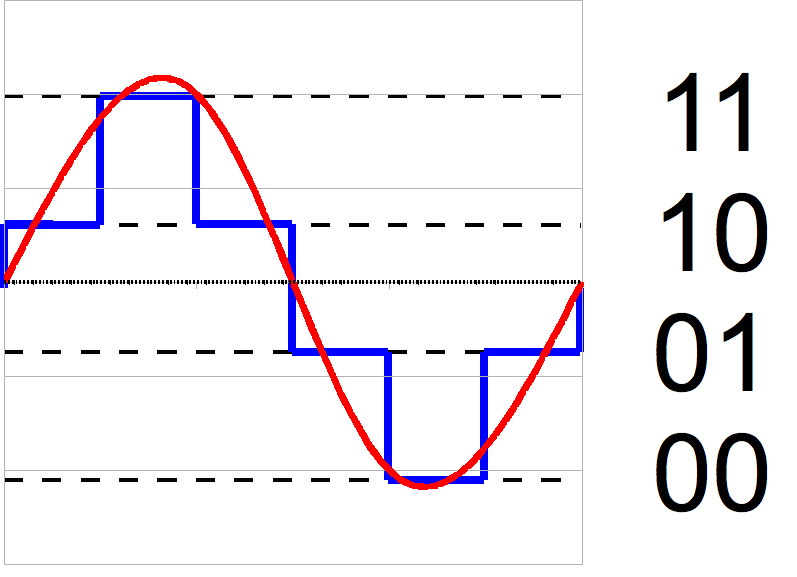
ADC Analog to Digital Convert là bộ chuyển đôi tương tự sang số. Đại lượng tương tự là Điện áp Vin được so sánh với điện áp mẫu Vref (giá trị lớn nhất), sau đó được chuyển đổi thành số lưu vào thanh ghi DATA của bộ chuyển đổi đó.

Có 2 tham số quan trọng của bộ ADC cần lưu ý:

* Tốc độ lấy mẫu (sampling) được tính theo số chu kì chuyển đổi
* Độ phân giải: Tính theo Bit bộ ADC có độ phân giải 12 Bit sẽ có 2^12 = 4096 giá trị

***Các chức năng chính của ADC trong STM32***

* Độ phân giải 12Bit
* Sinh ra ngắt tại các sự kiện End of convert, End of Injected, Analog Watchdog
* Chế độ Single hoặc Continuous
* Chế độ Scan tự động quét từ Kênh 0 đến Kênh n (mỗi bộ có 10 kênh tối đa)
* Có cơ chế cân chỉnh tay
* Data Alignment (Căn chỉnh Data) căn trái hoặc căn phải
* Cài đặt thời gian chuyển đổi đến từng Kênh
* Có thể kích hoạt bằng xung bên ngoài
* Chế độ Dual mode sử dụng cùng lúc 2 hoặc nhiều bộ ADC
* Hỗ trợ DMA
* Tần số chuyển đổi ADC được lấy từ bộ ABP2 thông qua ADC prescaler và phải nhỏ hơn 14MHz



Hình 2. 14: Mô tả bộ biến đổi ADC

***2.2.3. GPIO và ngắt ngoài***

***a.*** **GPIO:**

[*GPIO*](https://wiki.st.com/stm32mpu/wiki/GPIO_internal_peripheral) *(General Purpose Input/Output)* chính là các chân đầu ra hoặc đầu vào dùng chung.

***STM32 GPIO bao gồm 2 khối cơ bản:***

* Input Driver: Bao gồm thanh ghi Input Data (IDR), và 1 trigger. Tín hiệu Input ngoài việc được ghi vào IDR còn theo các đường Analog để vào bộ ADC, hoặc theo đường Alternate function input vào các ngoại vi khác
* Output Drive: Bao gồm thanh ghi Output Data (ODR), một khối output control để chọn tín hiệu ra là từ ODR hay từ các ngoại vi khác. Tiếp đến điều khiển 2 mosfet cho điện áp ra ở I/O pin

Diagram, schematic

Description automatically generated

Hình 2. 15: Cấu trúc bên trong các chân GPIO của STM32

***Chức năng của STM32 GPIO bao gồm:***

* **Input:**
* Input pull-up: Đầu vào có trở kéo lên (điện áp mặc định trên chân là Vcc)
* Input pull-down: Đầu vào có trở kéo xuống (điện áp mặc định trên chân là 0V)
* Input floating: Đầu vào thả nổi, điện áp không cố định dao động từ 0V tới Vcc
* Analog: Đầu vào tương tự, dùng để đo ADC
* **Output:**
* Ouput Push Pull: Đầu ra dạng đẩy kéo, tín hiệu sẽ chỉ có Vcc hoặc 0V tương ứng với Bit 1 và 0 ghi vào chân đó
* Ouput Opendrain: Đầu ra dạng cực máng hở. Chỉ có thể kéo về 0V bằng cách ghi bit 0, khi ghi bit 1, chân IO sẽ có điện áp tương ứng với nguồn nối vào IO đó
* Alternate function Push Pull: Đầu ra kểu đẩy kéo sử dụng trong các ngoại vi
* Alternate function Open Drain: Đầu ra dạng cực máng hở, sử dụng trong các ngoại vi (thường gặp trong I2C)

***b.*** **Ngắt Ngoài:**

[*EXTI*](https://wiki.st.com/stm32mpu/wiki/EXTI_internal_peripheral) *(External Interupts)*tạm dịch là ngắt ngoài hay ngắt sự kiện bên ngoài. Ngắt EXTI được kích hoạt khi có sự kiện từ bên ngoài tác động vào chân EXTI đó, tùy theo sự kiện đó có phù hợp với điều kiện ngắt không thì ngắt ngoài mới sảy ra.

*Ngắt ngoài STM32 được mô tả như sau:*

Các tham số (Main Features):

* Kích hoạt độc lập trên mỗi dòng ngắt (Line Interrupts)
* Truy cập đến từng Bit trong mỗi dòng ngắt
* Tạo ra tối đa 20 sự kiện/ngắt
* Tín hiệu phải có độ rộng xung thấp hơn chu kì xung nhịp của APB2 (vì APB2 cấp xung cho GPIO)

*Các Line ngắt ngoài được tổ chức như sau:*

Diagram

Description automatically generated

Hình 2. 16: Cấu trúc tín hiệu của các chân ngắt ngoài

Các chân PA0,PB0,…,PG0 sẽ chung là line EXTI0, Tiếp tục như vậy đến EXTI15 chúng ta có 16 ngắt.

*4 ngắt ngoài đặc biệt đó là:*

* EXTI line 16: dùng cho PVD output
* EXTI line 17: dùng cho sự kiện báo thức của bộ RTC
* EXTI line 18: dùng cho sự kiện thức dậy của USB
* EXTI line 19: dùng cho sự kiện thức dậy của ngoại vi Ethernet

## **2.3. Sơ đồ ghép nối**

Diagram

Description automatically generated

Hình 2. 17: Sơ đồ ghép nối giữa MCU với các Module phần cứng và các ngoại vi

## **2.4. Thiết kế mạch PCB**

Mạch được thiết kế trên phần mềm thiết kế mạch Altium, mạch được thiết kế từ các nguyên lý đã nêu ở trên. Sau khi sắp xếp hợp lý mạch được đặt in 2 lớp và hàn thủ công bằng thiếc hàn.

***2.4.1. Dạng 3D***

***A picture containing text, electronics, circuit

Description automatically generated***

Hình 2. 18: Hình ảnh mạch 3D mặt trên

***A screenshot of a computer

Description automatically generated with low confidence***

Hình 2. 19: Hình ảnh mạch 3D mặt dưới

***2.4.2. Mạch thực tế***

# **CHƯƠNG 3: LẬP TRÌNH FIRMWARE CHO THIẾT BỊ**

## **3.1. Giới thiệu công cụ hỗ trợ và phần mềm lập trình**

***3.1.1. STM32CubeMX***

STM32CubeMX là một công cụ đồ họa hỗ trợ cấu hình các dòng vi điều khiển STM32. Công cụ này còn có thể hỗ trợ sinh code dựa trên các cấu hình đã chọn theo nhiều các khác nhau tùy thuộc vào môi trường lập trình sử dụng. Việc sử dụng STM32CubeMX giúp tiết kiệm thời gian lập trình, dễ dàng thay đổi các cấu hình, cung cấp cho lập trình viên tầm nhìn tổng quan về trạng thái hoạt động của chip. Công cụ cung cấp đầy đủ cho toàn bộ dòng chip STM32 cùng các gói hỗi trợ. Mã nguồn sinh ra theo ngôn ngữ C, có thể sử dụng để sinh code cho phần mềm KeilC, hoặc sử dụng makefile và rất nhiều cách khác. Phần mềm này được sử dụng rộng rãi bởi các lập trình viên nhúng khi lập trình chip vi điều khiển STM32 vì tính đơn giản, dễ quản lý và tiết kiệm rất nhiều thời gian. Phần mềm có thể được tải xuống trên trang chủ của hãng ST.

*Các chức năng phần mềm bao gồm:*

- Lựa chọn các bộ vi điều khiển STM32 và các bộ vi điều khiển khác một cách trực quan

- Giao diện người dùng đồ họa phong phú, dễ sử dụng cho phép cấu hình: Sơ đồ chân với giải quyết xung đột tự động Thiết bị ngoại vi và chế độ chức năng phần mềm trung gian với xác nhận động các ràng buộc tham số cho cây ArmClock với xác nhận động cấu hình Chuỗi công suất với kết quả tiêu thụ ước tính

- Mã khởi tạo C, tuân thủ IAR Embedded Workbench®, MDK-ARM và STM32CubeIDE (trình biên dịch GCC) cho lõi Arm Cortex-M

- Tạo một Cây thiết bị Linux một phần cho lõi Arm Cortex-A (bộ vi xử lý STM32)

- Phát triển các Gói mở rộng STM32Cube nâng cao nhờ STM32PackCreator

- Tích hợp các gói Mở rộng STM32Cube vào dự án

- Khả dụng dưới dạng phần mềm độc lập chạy trên Windows, Linux và macOS (macOS là thương hiệu của Apple Inc. đã đăng ký tại Hoa Kỳ và các quốc gia khác.) Hệ điều hành và môi trường Java Runtime 64-bit

Graphical user interface, application

Description automatically generated

Hình 3. 1: Giao diện của phần mềm STM32CubeMX

***3.1.2. Visual Studio Code***

Visual Studio Code (VS Code) là một trình soạn thảo mã nguồn mở gọn nhẹ nhưng có khả năng vận hành mạnh mẽ trên 3 nền tảng là Windows, Linux và macOS được phát triển bởi Microsoft. Nó hỗ trợ cho JavaScript, Node.js và TypeScript, cũng như cung cấp một hệ sinh thái mở rộng vô cùng phong phú cho nhiều ngôn ngữ lập trình khác. Nó được xem là một sự kết hợp hoàn hảo giữa IDE và Code Editor. Visual Studio Code hỗ trợ chức năng debug, đi kèm với Git, có syntax highlighting, tự hoàn thành mã thông minh, snippets, và cải tiến mã nguồn.

A screenshot of a computer

Description automatically generated with medium confidence

Hình 3. 2: Giao diện phần mềm VsCode

Nhờ tính năng tùy chỉnh, Visual Studio Code cũng cho phép người dùng thay đổi theme, phím tắt, và các tùy chọn khác. Một số tính năng của VS code như:

Hỗ trợ nhiều ngôn ngữ lập trình: Visual Studio Code hỗ trợ nhiều ngôn ngữ lập trình như C/C++, C#, … Vì vậy, nó dễ dàng phát hiện và đưa ra thông báo nếu chương chương trình có lỗi.

Hỗ trợ đa nền tảng: Các trình viết code thông thường chỉ được sử dụng hoặc cho Windows hoặc Linux hoặc Mac Systems. Nhưng Visual Studio Code có thể hoạt động tốt trên cả ba nền tảng trên.

Cung cấp kho tiện ích mở rộng: Trong trường hợp lập trình viên muốn sử dụng một ngôn ngữ lập trình không nằm trong số các ngôn ngữ Visual Studio hỗ trợ, họ có thể tải xuống tiện ích mở rộng. Điều này vẫn sẽ không làm giảm hiệu năng của phần mềm, bởi vì phần mở rộng này hoạt động như một chương trình độc lập.

Hỗ trợ viết Code: Một số đoạn code có thể thay đổi chút ít để thuận tiện cho người dùng. Visual Studio Code sẽ đề xuất cho lập trình viên các tùy chọn thay thế nếu có.

Hỗ trợ Git: Visual Studio Code hỗ trợ kéo hoặc sao chép mã trực tiếp từ GitHub. Mã này sau đó có thể được thay đổi và lưu lại trên phần mềm.

**3.1.3. Công cụ lưu trữ và quản lý Github**

GitHub là một hệ thống quản lý dự án và phiên bản code, hoạt động giống như một mạng xã hội cho lập trình viên. GitHub được viết bằng Ruby on Rails và Erlang do Tom Preston-Werner, Chris Wanstrath, và PJ Hyett phát triển trang web được đưa ra và chạy chính thức vào tháng 4 năm 2008. Các lập trình viên có thể clone lại mã nguồn từ một repository và Github chính là một dịch vụ máy chủ repository công cộng, mỗi người có thể tạo tài khoản trên đó để tạo ra các kho chứa của riêng mình để có thể làm việc. Github có đầy đủ những tính năng của Git, ngoài ra nó còn bổ sung những tính năng về social để các developer tương tác với nhau. Git là công cụ giúp quản lý source code tổ chức theo dạng dữ liệu phân tán; Giúp đồng bộ source code của team lên 1 server; Hỗ trợ các thao tác kiểm tra source code trong quá trình làm việc (diff, check modifications, show history, merge source,…).

A screenshot of a computer

Description automatically generated with medium confidence

Hình 3. 3: Giao diện của Github

## **3.2. Cấu hình thiết bị trên phần mềm STM32CubeMX**

***3.2.1. Cấu hình xung Clock và Debug***

Đầu tiên, ta tiến hành cấu hình đồng hồ xung nhịp cho chip bằng cách vào mục System Core phần RCC, ta chọn chế độ sử dụng thạch anh ngoài High Speed Clock. Trong mục Clock Configuration, ta đặt tốc độ HCLK ở mức tối đa (72MHz) như hình 3.4.

Để cấu hình chế độ debug cho chip ta vào phần SYS và chọn chế độ debug Serial Wire sử dụng giao thức SW với ST-Link như hình 3.5.

Graphical user interface, text, application

Description automatically generated

Diagram

Description automatically generated

Hình 3. 4: Cấu hình thạch anh ngoại trong STM32CubeMX

Graphical user interface, application

Description automatically generated

Hình 3. 5: Cấu hình chế độ Debug trong STM32CubeMX

***3.2.2. Cấu hình ADC***

Để cấu hình cho bộ chuyển đổi tương tự - số ADC, ta vào mục Analog và vào phần ADC1 để chọn các kênh muốn sử dụng, trong đây chúng em sử dụng kênh 0. Các cấu hình của bộ ADC được giữ mặc định, ngoại trừ cho thời gian lấy mẫu được cấu hình là 1.5 chu kỳ xung của bộ ADC (đồng hồ của ADC được set ở tần số 12MHz) như hình 3.6.

Graphical user interface

Description automatically generated

Hình 3. 6: Cấu hình ADC trong STM32CubeMX

***3.2.3. Cấu hình Timer***

Để cấu hình Timer vào mục Timers cấu hình cho TIM1 và TIM2 là Internal Clock và Prescaler là 7200. Cấu hình Couter Period cho TIM1 là 30000; TIM2 là 2000. Timer được sử dụng để đếm thời gian kết thúc khi kiểm tra Pin cũng như nhận dữ liệu.

Graphical user interface

Description automatically generated with medium confidence

Hình 3. 7: Cấu hình cho Timer 1 trong STM32CubeMX

Graphical user interface

Description automatically generated

Hình 3. 8: Cấu hình cho Timer 2 trong STM32CubeMX

***3.2.4. Cấu hình UART***

Thiết bị sử dụng 3 bộ UART, UART1 để Debug; UART2 để giao tiếp với SIM800C; UART3 để giao tiếp với L70R. Để cấu hình cho ngoại vi UART, ta vào mục Connectivity, tích vào UART1, UART2, UART3, chọn chế độ Asynchronous. Chúng em sử dụng các cấu hình mặc định của ngoại vi, baudrate của UART1, UART2 được chỉnh lên 115200 Bits/s để tăng tốc độ truyền nhận giữa vi điều khiển và module SIM800L và với Debug, còn baudrate của L70R để 9600 Bit/s.

Graphical user interface

Description automatically generated with medium confidence

Hình 3. 9: Cấu hình UART1 cho Debug trong STM32CubeMX

Graphical user interface

Description automatically generated

Hình 3. 10: Cấu hình UART2 cho giao tiếp SIM800C trong STM32CubeMX

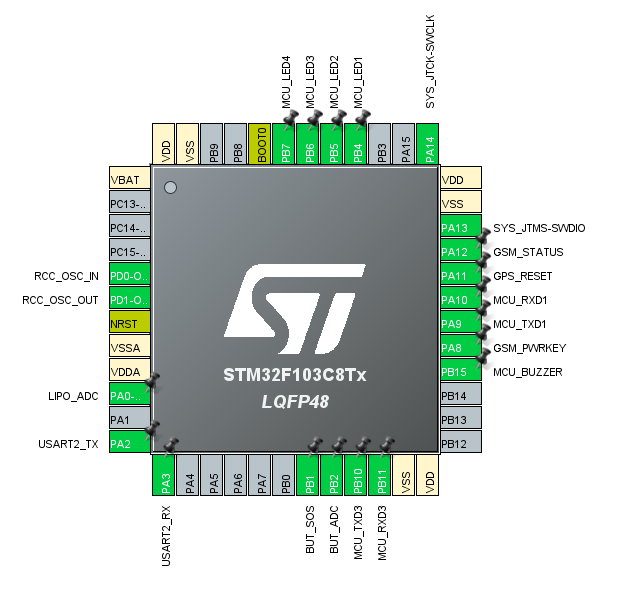
Graphical user interface

Description automatically generated

Hình 3. 11: Cấu hình UART3 cho giao tiếp L70R trong STM32CubeMX

***3.2.5. Cấu hình ngắt NVIC và GPIO***

Để cấu hình các chân GPIOs thực hiện các chức năng điều khiển, hiển thị, và cấu hình ngắt vào System Core và chọn GPIO. Trong đó PB1 cấu hình ngắt cho nút bấm SOS, PB2 cấu hình ngắt cho nút bấm ADC\_PIN, PA12 cấu hình input để đọc giá trị ADC của điện áp Pin, các chân GPIOs còn lại cấu hình Output Push Up.



Hình 3. 12: Sơ đồ các chân được sử dụng và chức năng

Table

Description automatically generated

Hình 3. 13: Cấu hình các chân GPIOs trong STM32CubeMX

Table

Description automatically generated

Hình 3. 14: Cấu hình các ngắt của ngoại vi trong STM32CubeMX

## **3.3. Lập trình Firmware**

Thư mục lập trình được STM32CubeMX sinh ra từ các cấu hình chúng em đã đặt trước, các ngoại vi được trình bày thành các cặp file .h và .c tương ứng. Chúng em sử dụng Makefile để biên dịch code thành file hex và file bin để nạp cho chip.

A screenshot of a computer

Description automatically generated with medium confidence

Hình 3. 15: Lập trình firmware trên VSCode

# **CHƯƠNG 4: KẾT QUẢ VÀ ĐÁNH GIÁ**

## **4.1. Kết quả đạt được**

## **4.2. Phân tích và đánh giá**

## **4.3. Đánh giá thành viên và công việc được giao**

**a. Nguyễn Huy Hoàng**

Nhận nhiệm vụ:

* Tìm hiểu và xây dựng chương trình
* Thiết kế mạch nguyên lí và PCB
* Xây dựng lưu đồ thuật toán cho chương trình
* Cấu hình phần mềm trên STM32CubeMX
* Code giao tiếp giữa Module phần cứng với MCU như: Khối hiển thị, ADC, GPIO, GSM - SIM800C, GPS - L70R
* Code chức năng tiết kiệm năng lượng Low Power Mode
* Code chức năng Update Firmware từ xa OTA
* Thử nghiệm và đánh giá kết quả

Nhận xét:

* Hoàn thành tốt nhiệm vụ
* Tích cực đóng góp tham gia sửa đổi và phát triển
* Đánh giá đóng góp 10/10

**b. Nguyễn Văn Dũng**

* Tìm hiểu và xây dựng chương trình
* Thiết kế mạch nguyên lí và PCB
* Xây dựng lưu đồ thuật toán cho chương trình
* Cấu hình phần mềm trên STM32CubeMX
* Code giao tiếp giữa Module phần cứng với MCU như: Khối hiển thị, ADC, GPIO, GSM - SIM800C, GPS - L70R
* Code chức năng Update Firmware từ xa OTA
* Code chức năng tiết kiệm năng lượng Low Power Mode
* Thử nghiệm và đánh giá kết quả

Nhận xét:

* Hoàn thành tốt nhiệm vụ
* Tích cực đóng góp tham gia sửa đổi và phát triển
* Đánh giá đóng góp 10/10

# **CHƯƠNG 5: LINK SHARE THƯ MỤC LẬP TRÌNH**

Link Github

<https://github.com/dungdz16/Mandevices_GPS_GSM>

Link OTA Drive

# **KẾT LUẬN**

# **DANH MỤC TÀI LIỆU THAM KHẢO**

[1] Các tài liệu đính kèm trong file REFERENCE

[2] “*TP4056 1A Standalone Linear Li-lon Battery Charger with Thermal Regulation in SOP-8*”, NanJing Top Power ASIC Corp

[3] “*DS9013-10”, Richtek Technology Corporation*, April 2011

[4] “*L70 Hardware Design*”, June 2014

[5] “*L70 GPS Protocol Specification*”, August 2015

[6] “*SIM800L Hardware Design V1.00*”, A company of SIM Tech

[7] “*SIM800 Series AT command manual V1.09*”, A company of SIM Tech

[8] “*RM0008 Reference manual*”, February 2021

[9] [*https://controllerstech.com/low-power-modes-in-stm32/*](https://controllerstech.com/low-power-modes-in-stm32/), Controllerstech