**BỘ GIÁO DỤC ĐÀO TẠO BỘ QUỐC PHÒNG**

**HỌC VIỆN KỸ THUẬT QUÂN SỰ**

----------------

**CÔNG PHƯƠNG ĐÔNG**

**DƯƠNG VŨ THÁI CƯỜNG**

**TÌM HIỂU LẬP TRÌNH NHÚNG VÀ**

**PHÁT TRIỂN ỨNG DỤNG IoT**

**Chuyên ngành: Hệ thống thông tin**

**BÁO CÁO ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC**

**Hà Nội - Năm 2016**

**BỘ GIÁO DỤC ĐÀO TẠO BỘ QUỐC PHÒNG**

**HỌC VIỆN KỸ THUẬT QUÂN SỰ**

-----------------

**BÁO CÁO ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC**

**Chuyên ngành**: Hệ thống thông tin

**Mã số**: 001

**Ngày giao đồ án**:

**Ngày nộp đồ án**:

**Tên đề tài:**

Tìm hiểu lập trình nhúng và phát triển hệ thống IoT

**Học viên thực hiện:** Công Phương Đông

Dương Vũ Thái Cường

**Lớp:** Tin học 11A

**Cán bộ hướng dẫn:** Nguyễn Trung Thành

Tạ Minh Thanh

**Hà Nội – Năm 2016**

**ĐỀ CƯƠNG ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC**

**Tên đề tài**: Tìm hiểu lâp trình nhúng và phát triển hệ thống IoT

**Chuyên ngành**: Hệ thống thông tin

**Thời gian thực hiện**:

**1. Cơ sở khoa học và tính thực tiễn của đề tài**

+ Cơ sở khoa học:

- Lý thuyết về ứng dụng IoT (Internet of thing).

- Lý thuyết về các mạch sử dụng: arduino…

- Lý thuyết về công nghệ BLE (Bluetooth low energy).

- Lý thuyết về định vị GPS.

- Lý thuyết về GPRS.

- Lý thuyết về lập trình nhúng.

- Ngôn ngữ lập trình C++, Java, Android.

- Hệ điều hành Mbed.

- Tìm hiểu Android Studio, Git…

**2. Mục tiêu của đề tài**

+ Mục tiêu tổng quan: Thu được kiến thức nhất định về lập trình nhúng và IoT cũng như các mạch đã, đang và sẽ sử dụng trong quá trình thực hiện đồ án.

+ Mục tiêu chi tiết: Xây dựng được một ứng dụng IoT.

**3. Phương pháp nghiên cứu**

+ Về lý thuyết: Tìm hiểu các lý thuyết về các mạch, về IoT ở trên mạng.

+ Về thực nghiệm: Xây dựng các chương trình với các mạch đang có (Arduino) để từ đó biết được khả năng của mạch và khoanh vùng ứng dụng.

**4. Nội dung nghiên cứu**

+ Nghiên cứu tổng quan: Tìm hiểu, đọc các bài viết trên mạng về bo mạch đang có và về lập trình IoT.

+ Nghiên cứu của tác giả:

- Tìm hiểu về mạch Arduino.

- Tìm hiểu về mbed.

- Tìm hiểu cách viết ứng dụng IoT với Android.

**LỜI MỞ ĐẦU**

Thời đại vạn vật kết nối đang tăng tốc và sẽ thay đổi cách chúng ta sống, làm việc, giải trí bằng hàng triệu cách lớn nhỏ. Từ nông nghiệp đến phòng vệ, mua bán lẻ (bán hàng) đến chăm sóc sức khỏe, mọi thứ đều chịu ảnh hưởng bởi khả năng ngày càng phát triển của các nhà kinh doanh, chính phủ và người tiêu dùng trong việc kết nối và quản lí môi trường xung quanh

Không có một chút nghi ngờ nào khi thời đại vạn vật kết nối được gọi là “Cuộc cách mạng công nghiệp tiếp theo”. Cuộc cách mạng này rất lớn đến mức nó có thể tạo  ra nhiều nguồn thu nhập mới và các cơ hội cho công ty của bạn. Câu hỏi duy nhất là “Bạn đã thực sự bắt kịp với tốc độ phát triển của IoT chưa?”

Với kiến thức còn hạn chế, tuy em đã cố gắng nỗ lực hết mình nhưng đề  
tài của em vẫn không tránh khỏi những thiếu sót. Em mong nhận được những  
đóng góp về thiếu sót trong đề tài của em từ thầy cô và các bạn để em có thể rút  
kinh nghiệm và phát triển hệ thống một cách tốt nhất có thể.

*Em xin chân thành cảm ơn!*

**MỤC LỤC**

[**CHƯƠNG I: CƠ SỞ LÝ THUYẾT** 6](#_Toc470726475)

[1. Các mạch nhúng 6](#_Toc470726476)

[1.1. Mạch Arduino 6](#_Toc470726477)

[1.2. Mạch Nordic nRF51822 14](#_Toc470726478)

[2. Các công nghệ sử dụng 14](#_Toc470726479)

[2.1. Internet of things 14](#_Toc470726480)

[2.2. Bluetooth low energy 16](#_Toc470726481)

[2.3. GPRS 17](#_Toc470726482)

[2.4. GPS 18](#_Toc470726483)

[3. Hệ điều hành Mbed 20](#_Toc470726484)

[3.1. Tổng quan 20](#_Toc470726485)

[3.2. Làm việc với mbed 22](#_Toc470726486)

[3.3. APIs 24](#_Toc470726487)

[**CHƯƠNG II: PHÂN TÍCH HỆ THỐNG** 27](#_Toc470726488)

[1. Mô tả hệ thống 27](#_Toc470726489)

[1.1. Nhiệm vụ cơ bản 27](#_Toc470726490)

[1.2. Quy trình xử lý và quy tắc quản lý 27](#_Toc470726491)

[2. Phân tích chức năng nghiệp vụ 27](#_Toc470726492)

[2.1. Mô hình hóa chức năng nghiệp vụ 27](#_Toc470726493)

[2.2. Mô hình hóa tiến trình nghiệp vụ 27](#_Toc470726494)

[2.3. Đặc tả tiến trình nghiệp vụ 27](#_Toc470726495)

[3. Phân tích dữ liệu nghiệp vụ 27](#_Toc470726496)

[3.1. Mô hình dữ liệu ban đầu 27](#_Toc470726497)

[3.2. Chuẩn hóa dữ liệu 27](#_Toc470726498)

[3.3. Đặc tả dữ liệu 27](#_Toc470726499)

[**CHƯƠNG III: THIẾT KẾ HỆ THỐNG** 27](#_Toc470726500)

[1. Thiết kế UX 27](#_Toc470726501)

[2. Thiết kế cơ sở dữ liệu 27](#_Toc470726502)

[3. Thiết kế UI 27](#_Toc470726503)

[**CHƯƠNG IV: KẾT LUẬN** 27](#_Toc470726504)

# **CHƯƠNG I: CƠ SỞ LÝ THUYẾT**

## 1. Các mạch nhúng

### 1.1. Mạch Arduino

1.1.1 Lịch sử ra đời

+ Arduino được khởi động vào năm 2005 như là một dự án dành cho sinh viên trại Interaction Design Institute Ivrea (Viện thiết kế tương tác Ivrea) tại Ivrea, Italy. Vào thời điểm đó các sinh viên sử dụng một "BASIC Stamp" (con tem Cơ Bản) có giá khoảng $100, xem như giá dành cho sinh viên. Massimo Banzi, một trong những người sáng lập, giảng dạy tại Ivrea. Cái tên "Arduino" đến từ một quán bar tại Ivrea, nơi một vài nhà sáng lập của dự án này thường xuyên gặp mặt. Bản thân quán bar này có được lấy tên là Arduino, Bá tước của Ivrea, và là vua của Italy từ năm 1002 đến 1014.

+ Lý thuyết phần cứng được đóng góp bởi một sinh viên người Colombia tên là Hernando Barragan. Sau khi nền tảng Wiring hoàn thành, các nhà nghiên cứu đã làm việc với nhau để giúp nó nhẹ hơn, rẻ hơn, và khả dụng đối với cộng đồng mã nguồn mở. Trường này cuối cùng bị đóng cửa, vì vậy các nhà nghiên cứu, một trong số đó là David Cuarlielles, đã phổ biến ý tưởng này.

1.1.2. Các broad chính

+ Phần cứng Arduino gốc được sản xuất bởi công ty Italy tên là Smart Projects. Một vài board dẫn xuất từ Arduino cũng được thiết kế bởi công ty của Mỹ tên là SparkFun Electronics.

+ Sáu phiên bản phần cứng của Arduino cũng đã được sản xuất thương mại tính đến thời điểm hiện tại.

+ Các broad Arduino mẫu:

- Arduino Diecimila in Stoicheia

- Arduino Duemilanove (rev 2009b)

- Arduino UNO

- Arduino Leonardo

- Arduino Mega

- Arduino MEGA 2560 R3

- Arduino Nano

- Arduino Due (nền tảng ARM)

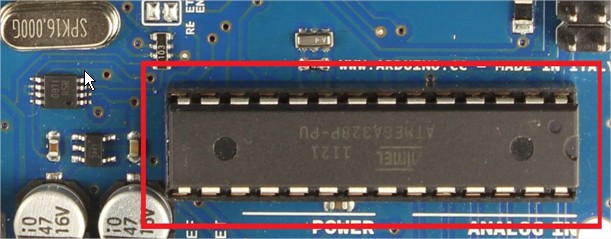
1.1.3. Arduino Uno R3

Nhắc tới dòng mạch Arduino dùng để lập trình, cái đầu tiên mà người ta thường nói tới chính là dòng Arduino UNO. Hiện dòng mạch này đã phát triển tới thế hệ thứ 3 (R3)

1.1.3.1. Thông số cơ bản

|  |  |
| --- | --- |
| Vi điều khiển | ATmega328 họ 8bit |
| Điện áp hoạt động | 5V DC (chỉ được cấp qua cổng USB) |
| Tần số hoạt động | 16 MHz |
| Dòng tiêu thụ | khoảng 30mA |
| Điện áp vào khuyên dùng | 7-12V DC |
| Điện áp vào giới hạn | 6-20V DC |
| Số chân Digital I/O | 14 (6 chân hardware PWM) |
| Số chân Analog | 6 (độ phân giải 10bit) |
| Dòng tối đa trên mỗi chân I/O | 30 mA |
| Dòng ra tối đa (5V) | 500 mA |
| Dòng ra tối đa (3.3V) | 50 mA |
| Bộ nhớ flash | 32 KB (ATmega328) với 0.5KB dùng bởi bootloader |
| SRAM | 2 KB (ATmega328) |
| EEPROM | 1 KB (ATmega328) |

1.1.3.2. Vi điều khiển



+ Arduino UNO có thể sử dụng 3 vi điều khiển họ 8bit AVR là ATmega8, ATmega168, ATmega328. Bộ não này có thể xử lí những tác vụ đơn giản như điều khiển đèn LED nhấp nháy, xử lí tín hiệu cho xe điều khiển từ xa, làm một trạm đo nhiệt độ - độ ẩm và hiển thị lên màn hình LCD… hay nhiều ứng dụng khác.

+ Thiết kế tiêu chuẩn của Arduino UNO sử dụng vi điều khiển ATmega328. Tuy nhiên nếu yêu cầu phần cứng của bạn không cao hoặc túi tiền không cho phép, bạn có thể sử dụng các loại vi điều khiển khác có chức năng tương đương nhưng rẻ hơn như ATmega8 (bộ nhớ flash 8KB hoặc ATmega168 (bộ nhớ flash 16KB).

+ Ngoài việc dùng cho board Arduino UNO, bạn có thể sử dụng những IC điều khiển này cho các mạch tự chế.

1.1.3.3. Năng lượng

+ Arduino UNO có thể được cấp nguồn 5V thông qua cổng USB hoặc cấp nguồn ngoài với điện áp khuyên dùng là 7-12V DC và giới hạn là 6-20V. Thường thì cấp nguồn bằng pin vuông 9V là hợp lí nhất nếu bạn không có sẵn nguồn từ cổng USB. Nếu cấp nguồn vượt quá ngưỡng giới hạn trên, bạn sẽ làm hỏng Arduino UNO

+ Các chân năng lượng:

**- GND (Ground)**: cực âm của nguồn điện cấp cho Arduino UNO. Khi bạn dùng các thiết bị sử dụng những nguồn điện riêng biệt thì những chân này phải được nối với nhau.

**- 5V**: cấp điện áp 5V đầu ra. Dòng tối đa cho phép ở chân này là 500mA.

**- 3.3V**: cấp điện áp 3.3V đầu ra. Dòng tối đa cho phép ở chân này là 50mA.

**- Vin (Voltage Input)**: để cấp nguồn ngoài cho Arduino UNO, bạn nối cực dương của nguồn với chân này và cực âm của nguồn với chân GND.

**- IOREF**: điện áp hoạt động của vi điều khiển trên Arduino UNO có thể được đo ở chân này. Và dĩ nhiên nó luôn là 5V. Mặc dù vậy bạn không được lấy nguồn 5V từ chân này để sử dụng bởi chức năng của nó không phải là cấp nguồn.

**- RESET**: việc nhấn nút Reset trên board để reset vi điều khiển tương đương với việc chân RESET được nối với GND qua 1 điện trở 10KΩ.

***\* Lưu ý:***

+ Arduino UNO không có bảo vệ cắm ngược nguồn vào. Do đó bạn phải hết sức cẩn thận, kiểm tra các cực âm – dương của nguồn trước khi cấp cho Arduino UNO. Việc làm chập mạch nguồn vào của Arduino UNO sẽ biến nó thành một miếng nhựa chặn giấy. Lời khuyến là nên dùng nguồn từ cổng USB nếu có thể.

+ Các chân 3.3V và 5V trên Arduino là các chân dùng để cấp nguồn ra cho các thiết bị khác, không phải là các chân cấp nguồn vào. Việc cấp nguồn sai vị trí có thể làm hỏng board. Điều này không được nhà sản xuất khuyến khích.

+ Cấp nguồn ngoài không qua cổng USB cho Arduino UNO với điện áp dưới 6V có thể làm hỏng board.

+ Cấp điện áp trên 13V vào chân RESET trên board có thể làm hỏng vi điều khiển ATmega328.

+ Cường độ dòng điện vào/ra ở tất cả các chân Digital và Analog của Arduino UNO nếu vượt quá 200mA sẽ làm hỏng vi điều khiển.

+ Cấp điệp áp trên 5.5V vào các chân Digital hoặc Analog của Arduino UNO sẽ làm hỏng vi điều khiển.

+ Cường độ dòng điện qua một chân Digital hoặc Analog bất kì của Arduino UNO vượt quá 40mA sẽ làm hỏng vi điều khiển. Do đó nếu không dùng để truyền nhận dữ liệu, bạn phải mắc một điện trở hạn dòng.

1.1.3.4. Bộ nhớ

Vi điều khiển Atmega328 tiêu chuẩn cung cấp cho người dùng:

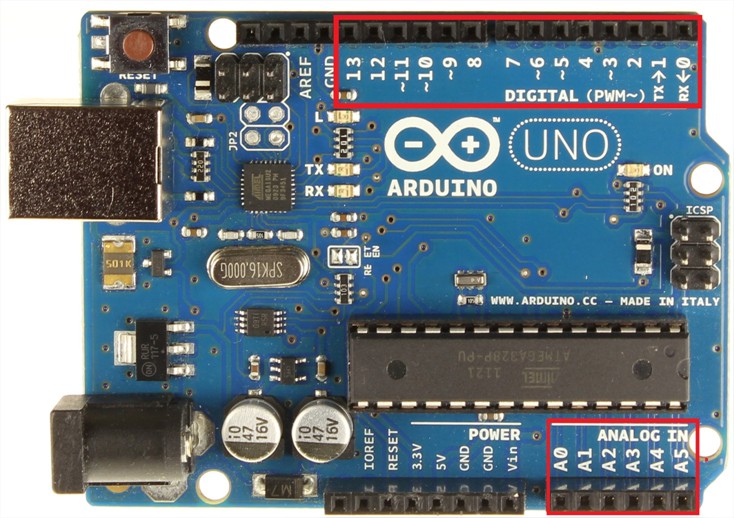
**+ 32KB bộ nhớ Flash**: những đoạn lệnh bạn lập trình sẽ được lưu trữ trong bộ nhớ Flash của vi điều khiển. Thường thì sẽ có khoảng vài KB trong số này sẽ được dùng cho bootloader nhưng đừng lo, bạn hiếm khi nào cần quá 20KB bộ nhớ này đâu.

**+ 2KB cho SRAM** (**S**tatic **R**andom **A**ccess **M**emory): giá trị các biến bạn khai báo khi lập trình sẽ lưu ở đây. Bạn khai báo càng nhiều biến thì càng cần nhiều bộ nhớ RAM. Tuy vậy, thực sự thì cũng hiếm khi nào bộ nhớ RAM lại trở thành thứ mà bạn phải bận tâm. Khi mất điện, dữ liệu trên SRAM sẽ bị mất.

**+ 1KB EEPROM (E**lectrically **E**raseble **P**rogrammable

**R**ead **O**nly **M**emory): đây giống như một chiếc ổ cứng mini – nơi bạn có thể đọc và ghi dữ liệu của mình vào đây mà không phải lo bị mất khi cúp điện giống như dữ liệu trên SRAM.

1.1.3.5. Các cổng vào ra



+ Arduino UNO có 14 chân digital dùng để đọc hoặc xuất tín hiệu. Chúng chỉ có 2 mức điện áp là 0V và 5V với dòng vào/ra tối đa trên mỗi chân là 40mA. Ở mỗi chân đều có các điện trở pull-up từ được cài đặt ngay trong vi điều khiển ATmega328 (mặc định thì các điện trở này không được kết nối).

+ Một số chân digital có các chức năng đặc biệt như sau:

* **2 chân Serial**: 0 (RX) và 1 (TX): dùng để gửi (transmit – TX) và nhận (receive – RX) dữ liệu TTL Serial. Arduino Uno có thể giao tiếp với thiết bị khác thông qua 2 chân này. Kết nối bluetooth thường thấy nói nôm na chính là kết nối Serial không dây. Nếu không cần giao tiếp Serial, bạn không nên sử dụng 2 chân này nếu không cần thiết
* **Chân PWM (~): 3, 5, 6, 9, 10, và 11**: cho phép bạn xuất ra xung PWM với độ phân giải 8bit (giá trị từ 0 → 28-1 tương ứng với 0V → 5V) bằng hàm analogWrite(). Nói một cách đơn giản, bạn có thể điều chỉnh được điện áp ra ở chân này từ mức 0V đến 5V thay vì chỉ cố định ở mức 0V và 5V như những chân khác.
* **Chân giao tiếp SPI:** 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK).  Ngoài các chức năng thông thường, 4 chân này còn dùng để truyền phát dữ liệu bằng giao thức SPI với các thiết bị khác.
* **LED 13**: trên Arduino UNO có 1 đèn led màu cam (kí hiệu chữ L). Khi bấm nút Reset, bạn sẽ thấy đèn này nhấp nháy để báo hiệu. Nó được nối với chân số 13. Khi chân này được người dùng sử dụng, LED sẽ sáng.

+ Arduino UNO có 6 chân analog (A0 → A5) cung cấp độ phân giải tín hiệu 10bit (0 → 210-1) để đọc giá trị điện áp trong khoảng 0V → 5V. Với chân **AREF** trên board, bạn có thể để đưa vào điện áp tham chiếu khi sử dụng các chân analog. Tức là nếu bạn cấp điện áp 2.5V vào chân này thì bạn có thể dùng các chân analog để đo điện áp trong khoảng từ 0V  → 2.5V với độ phân giải vẫn là 10bit.

+ Đặc biệt, Arduino UNO có 2 chân A4 (SDA) và A5 (SCL) hỗ trợ giao tiếp I2C/TWI với các thiết bị khác.

1.1.3.6. Ethernet Shield

Ethernet shield là một mạch mở rộng cho arduino, giúp arduino có thể kết nối với thế giới internet rộng lớn. Ứng dụng của shield này là truyền nhận thông tin giữa arduino với thiết bị bên ngoài sử dụng internet, shield này đặc biệt hữu ích cho các ứng dụng IoT, điều khiển và kiểm soát hệ thống vì internet luôn liên tục, dữ liệu truyền đi nhanh.

1.1.4. Lập trình cho Arduino

+ Các thiết bị dựa trên nền tảng Arduino được lập trình bằng ngôn riêng. Ngôn ngữ này dựa trên ngôn ngữ Wiring được viết cho phần cứng nói chung. Và Wiring lại là một biến thể của C/C++. Một số người gọi nó là Wiring, một số khác thì gọi là C hay C/C++. Riêng mình thì gọi nó là “ngôn ngữ Arduino”, và đội ngũ phát triển Arduino cũng gọi như vậy. Ngôn ngữ Arduino bắt nguồn từ C/C++ phổ biến hiện nay do đó rất dễ học, dễ hiểu.

+ Người dùng chỉ cần định nghĩa 2 hàm để tạo ra một chương trình vòng thực thi (cyclic executive) có thể chạy được:

- **setup():** hàm này chạy mỗi khi khởi động một chương trình, dùng để thiết lập các cài đặt

- **loop():** hàm này được gọi lặp lại cho đến khi tắt nguồn board mạch

+ Một chương trình điển hình cho một bộ vi điều khiển đơn giản chỉ là làm cho một bóng đèn Led sáng/tắt. Trong môi trường Arduino, ta sẽ phải viết một chương trình giống như sau:

#define LED\_PIN 13

void setup () {

pinMode (LED\_PIN, OUTPUT); *// Đặt chân 13 làm đầu ra digital*

}

void loop () {

digitalWrite (LED\_PIN, HIGH); *// Bật LED on*

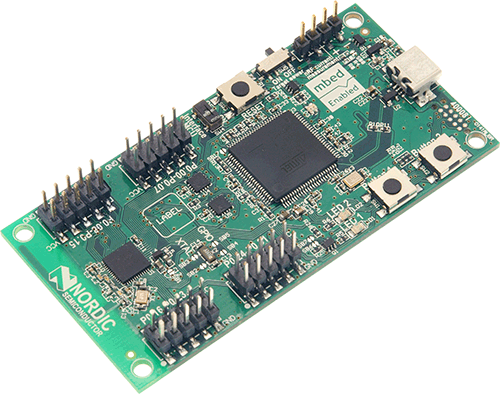
delay (1000); *// chờ trong 1 giây (1000 mili giây)*

digitalWrite (LED\_PIN, LOW); *// Tắt LED off*

delay (1000); *// chờ trong 1s*

}

### 1.2. Mạch Nordic nRF51822



+ Là một loại mạch ARM mbed với mức giá thấp, có tích hợp công nghệ bluetooth low energy. Bộ kit cho phép truy cập đến tất cả các chân GPIO qua các đầu cắm và tích hợp một ô chức pin coin – cell để phục vụ tính di động.

+ Đặc điểm:

- Bao gồm công nghệ BLE và bộ xử lý ARM Cortex – M0 trên một chip duy nhất.

- Có tất cả 31 chân GPIO.

- Có 4 chân GND.

- Có 2 đèn LED tích hợp ngay trên mạch.

- Nhận nguồn điện thông qua: USB, một cục pin coin – cell 2032 hoặc là nguồn điện bên ngoài.

## 2. Các công nghệ sử dụng

### 2.1. Internet of things

2.1.1. Internet of things là gì?

+ Internet of things (viết tắt là IoT) là một liên mạng của các thiết bị vật lý, các loại xe cộ, các tòa nhà, và các đồ vật khác mà ở đó, mỗi vật thể đều được gắn với các thiết bị điện tử, các phần mềm, các cảm biến và khả năng kết nối mạng nhằm mục đích giúp cho các vật thể này có thể thu thập và trao đổi dữ liệu với nhau. Nói đơn giản IoT là một tập hợp các thiết bị có khả năng kết nối với nhau, với Internet và với thế giới bên ngoài để thực hiện một công việc nào đó.

+ Một vật thể trong IoT có thể là một con người với một trái tim cấy ghép; một con vật ở trang trại với bộ chip sinh học; một chiếc xe với bộ cảm ứng tích hợp cảnh báo tài xế khi bánh xe xẹp hoặc bất kỳ vật thể tự nhiên hay nhân tạo nào mà có thể gán được một địa chỉ IP và cung cấp khả năng truyền dữ liệu thông qua mạng lưới.

+ Điểm quan trọng của IoT đó là các đối tượng phải có thể được nhận biết và định dạng. Nếu mọi đối tượng, kể cả con người, được “đánh dấu” để phân biệt bản thân đối tượng đó với những thứ xung quanh thì chúng ta có thể hoàn toàn quản lý được nó thông qua máy tính. Việc kết nối thì có thể thực hiện qua Wifi, mạng viễn thông băng rộng (3G, 4G), Bluetooth, hồng ngoại…

2.1.2. Ứng dụng của IoT

+ IoT có ứng dụng rộng vô cùng, có thể kể ra một số thứ sau:

- Quản lý chất thải.

- Quản lý và lập kế hoạch quản lý đô thị.

- Quản lý môi trường.

- Mua sắm thông minh.

- Nhà thông minh.

- Quản lý các thiết bị cá nhân.

+ Tác động của IOT rất đa dạng, trên các lĩnh vực: quản lý hạ tầng, y tế, xây dựng và tự động hóa, giao thông…. Cụ thể trong lĩnh vực y tế, Thiết bị IoT có thể được sử dụng để cho phép theo dõi sức khỏe từ xa và hệ thống thông báo khẩn cấp. Các thiết bị theo dõi sức khỏe có thể dao động từ huyết áp và nhịp tim màn với các thiết bị tiên tiến có khả năng giám sát cấy ghép đặc biệt, chẳng hạn như máy điều hòa nhịp hoặc trợ thính tiên tiến.

### 2.2. Bluetooth low energy

2.2.1. Tổng quan

+ Bluetooth low energy (BLE, tên trên thị trường là Bluetooth Smart) là một công nghệ mạng cá nhân không dây được thiết kế và tiếp thị bởi tập đoàn Bluetooth Special Interest nhắm vào các ứng dụng mới lạ, lôi cuốn trong các lĩnh vực như y tế, thể dục, beacon, an ninh và các ngành công nghiệp giải trí gia đình.

+ Khác với công nghệ bluetooth đời trước, BLE tiêu tốn ít năng lượng hơn khi duy trì kết nối trong một phạm vi tương đương với công nghệ cũ. Chính vì tính năng đặc biệt này nên nó trở nên phổ biến trong việc ứng dụng vào các ứng dụng IoT.

+ BLE được thiết kế cho các ứng dụng:

- Siêu tiết kiệm năng lượng, cho phép thiết bị hoạt động trong vòng vài tháng hoặc cũng có thể đến vài năm chỉ với một cục pin coin – cell.

- Khoảng cách ngắn, hoạt động ổn định trong phạm vi 10m.

- Dữ liệu truyền tải không lớn, thích hợp cho các ứng dụng điều khiển không liên tục, cảm biến.

2.2.2. Các điểm giới hạn của BLE

+ Thông lượng dữ liệu nhỏ: Tần số điều chế của sóng BLE trong không gian là 1Mbps. Đây là giới hạn trên của thông lượng theo lý thuyết. Tuy nhiên trong thực tế, tham số này nhỏ hơn do ảnh hưởng của rất nhiều yếu tố.

+ Khoảng cách gần: Khoảng cách truyền dữ liệu của BLE chịu ảnh hưởng bởi môi trường hoạt động, thiết kế anten, vật cản, hướng thiết bị … nên BLE tập trung vào các ứng dụng truyền thông trong phạm vi gần. Với BLE ta có:

- Khoảng cách lý thuyết: 100m (điều kiện tốt).

- Khoảng cách khả thi: 30m.

- Khoảng cách thường được sử dụng: 2 – 5m.

### 2.3. GPRS

2.3.1. GPRS là gì?

+ Dịch vụ vô tuyến gói tổng hợp (tiếng Anh: General Packet Radio Service (GPRS)) là một dịch vụ dữ liệu di động dạng gói dành cho những người dùng Hệ thống thông tin di động toàn cầu (GSM) và điện thoại di động IS-136.

+ Nó cung cấp dữ liệu ở tốc độ từ 56 đến 114 kbps.

+ GPRS có thể được dùng cho những dịch vụ như truy cập Giao thức Ứng dụng Không dây (WAP), Dịch vụ tin nhắn ngắn (SMS), Dịch vụ nhắn tin đa phương tiện (MMS), và với các dịch vụ liên lạc Internet như email và truy cập World Wide Web.

+ Các hệ thống di động 2G kết hợp với GPRS thường được gọi là **"2.5G"**, có nghĩa là, một công nghệ trung gian giữa thế hệ điện thoại di động thứ hai (2G) và thứ ba (3G). Nó cung cấp tốc độ truyền tải dữ liệu vừa phải, bằng cách sử dụng các kênh Đa truy cập theo phân chia thời gian (TDMA) đang còn trống, ví dụ, hệ thống GSM.

2.3.2. Phân loại các thiết bị sử dụng GPRS

Các thiết bị được sử dụng GPRS được chia thành 3 loại:

+ **Loại A:** Có thể kết nối vào dịch vụ GPRS và dịch vụ GSM (thoại, SMS) cùng lúc. Những thiết bị như vậy đã có mặt trên thị trường.

**+ Loại B:** Có thể kết nối vào dịch vụ GPRS và dịch vụ GSM (thoại, SMS) nhưng chỉ dùng một trong hai dịch vụ vào một thời điểm. Trong khi dùng dịch vụ GSM, dịch vụ GPRS bị ngưng, GPRS sau đó sẽ tự động được tiếp tục sau khi dịch GSM kết thúc. Phần lớn thiết bị di động GPRS thuộc Loại B.

**+ Loại C:** Được kết nối với hoặc dịch vụ GPRS hoặc dịch vụ GSM (thoại, SMS). Phải được chuyển bằng tay giữa hai dịch vụ.

### 2.4. GPS

2.4.1. GPS là gì?

+ **GPS**là viết tắt của "Global Positioning System" (hệ thống định vị toàn cầu), thực chất là một mạng lưới bao gồm 27 vệ tinh quay xung quanh trái đất. Trong số 27 vệ tinh này, 24 vệ tinh đang hoạt động, 3 vệ tinh còn lại đóng vai trò dự phòng trong trường hợp 1 trong số 24 vệ tinh chính bị hư hỏng.

+ Dựa vào cách sắp đặt của các vệ tinh này, khi đứng dưới mặt đất, bạn có thể nhìn được ít nhất là 4 vệ tinh trên bầu trời tại bất kì thời điểm nào.

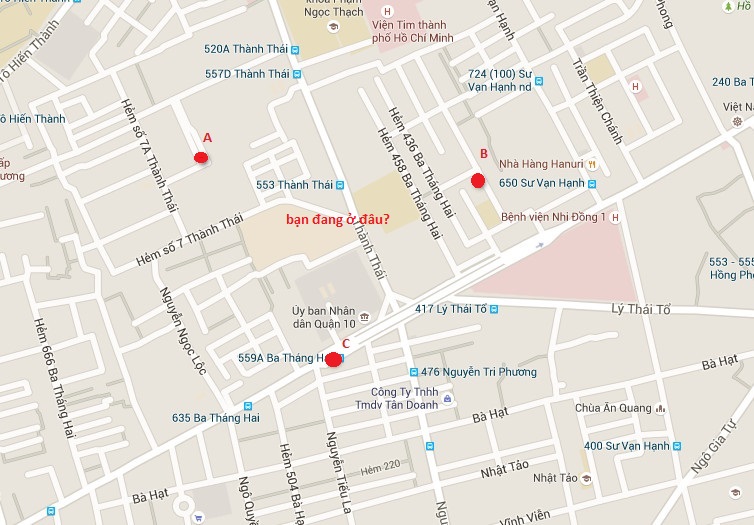
2.4.2. Cơ chế hoạt động của GPS

+ Các vệ tinh **GPS**bay hai vòng trong một ngày theo một quỹ đạo đã được tính toán chính xác và liên tục phát các tín hiệu có thông tin xuống Trái Đất. Các máy thu GPS nhận các tín hiệu này và giải mã bằng các phép tính lượng giác, qua đó sẽ tính toán và hiển thị được vị trí của người dùng.

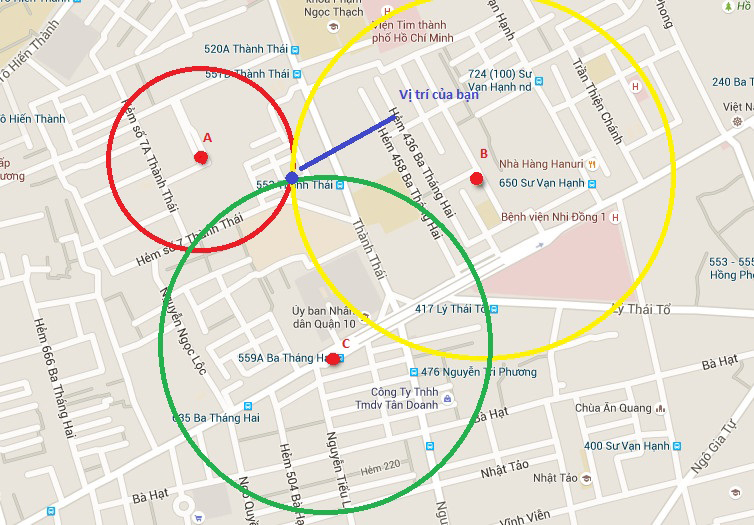
+ Các đầu thu GPS (các thiết bị smartphone…) thu dữ liệu từ các vệ tinh GPS ở trên bầu trời. Nói một cách đơn giản, mỗi vệ tinh cho bạn biết khoảng cách chính xác từ vị trí của bạn đến vệ tinh đó hoặc một điểm nào đó trên trái đất

+ Cơ chế hoạt động của GPS rất đơn giản, bạn có thể tưởng tượng như sau:

- Trên bản đồ có 3 điểm cố định A, B, C. Dữ liệu GPScho bạn biết khoảng cách lần lượt từ điểm A, B, C đến nơi bạn đứng là 1km, 3km, 2km



- Sau đó bạn vẽ 3 vòng tròn có tâm là A, B, C với bán kính lần lượt là 1km, 3km và 2km.



- Vị trí giao nhau của ba vòng tròn chính là vị trí của bạn

+ Thiết bị thu phải nhận được tín hiệu của ít nhất ba vệ tinh để cho ra vị trí hai chiều và để theo dõi được chuyển động của bạn. Khi nhận được tín hiệu của ít nhất bốn vệ tinh, máy sẽ cho ra được vị trí ba chiều. Một khi vị trí của bạn đã tính được thì thiết bị thu có thể tính các thông tin khác, như tốc độ di chuyển, hướng chuyển động, bám sát di chuyển, khoảng hành trình, khoảng cách đích đến và nhiều thứ khác nữa.

+ Để đưa ra vị trí chính xác, rất nhiều thiết bị GPS kết nối tới ít nhất là 4 vệ tinh. Đó là lý do vì sao đôi khi để tìm ra vị trí chính xác của bạn, hệ thống GPS lại mất nhiều thời gian tới vậy. Đó cũng là lý do vì sao đôi khi bạn bị mất sóng GPS: thiết bị di động của bạn có thể đã kết nối tới 1 hoặc 2 vệ tinh, song 2 vệ tinh vẫn là không đủ.

2.4.3. Độ chính xác của GPS

+ Các máy thu GPS ngày nay cực kì chính xác, nhờ vào thiết kế nhiều kênh hoạt động song song của chúng. Các máy thu 12 kênh song song (của Garmin) nhanh chóng khóa vào các quả vệ tinh khi mới bật lên và duy trì kết nối bền vững.

+ Trạng thái của khí quyển và các nguồn gây sai số khác có thể ảnh hưởng tới độ chính xác của máy thu GPS. Các máy thu GPS có độ chính xác trung bình trong vòng 15 mét.

+ Các máy thu mới hơn với khả năng WAAS (*Wide Area Augmentation System*) có thể tăng độ chính xác trung bình tới dưới 3 mét. Không cần thêm thiết bị hay mất phí để có được lợi điểm của WAAS bằng các máy phát hiệu. Để thu được tín hiệu đã sửa lỗi, người dùng phải có máy thu tín hiệu vi sai bao gồm cả ăn-ten để dùng với máy thu GPS của họ.

2.4.4. Ứng dụng của GPS

GPS đã được ứng dụng rất nhiều trong đời sống sản xuất. Có thể kể đến một số ứng dụng như:

+ Ứng dụng trong dân dụng: giám sát quản lý vận tải; theo dõi vị trí, tốc độ, hướng di chuyển…

+ Ứng dụng trong quân sự: bom thông minh JDAM, tên lửa không đối đất, tên lửa tấn công đất liền, tên lửa hành trình, tên lửa đất đối đất…

## 3. Hệ điều hành Mbed

### 3.1. Tổng quan

3.1.1. Khái niệm

+ Là nền tảng cung cấp hệ điều hành được thiết kế cho IoT.

+ mBed hỗ trợ hơn 100 bo mạch và hơn 400 các thành phần, công cụ để viết, xây dựng và kiểm thử ứng dụng. Nó còn hỗ trợ các công cụ server và client – side để giao tiếp với các thiết bị.

+ Ứng dụng được viết bằng ngôn ngữ C++ dùng để nhận thông tin từ phần cứng và gửi các lệnh tới nó.

3.1.2. Đặc điểm

+ Tính kết nối: Tính kết nối là yêu cầu cơ bản đối với các thiết bị IoT và mBed hoàn toàn đáp ứng được điều này. Những bo mạch do mbed phát triển và các dịch vụ liên quan tới những bo mạch này như là dịch vụ Device Connector Service đều hỗ trợ đầy đủ các phương thức đa giao tiếp. (BLE, 6LoWPAN).

+ Tính bảo mật: mbed cung cấp hai công cụ tập trung vào tính năng bảo mật tích hợp vào phát triển các thiết bị IoT. Hai công cụ này là: mbed TLS dùng cho việc mã hóa và khả năng SSL/TLS, mbed OS uVisor dùng cho việc bảo mật các tên miền.

3.1.3. Hệ điều hành

+ Hệ điều hành ARM mbed là hệ điều hành mã nguồn mở được thiết kế riêng cho các “đối tượng” thuộc IoT.

+ Hệ điều hành mbed hỗ trợ nhiều loại kết nối như: bluetooth, wifi, 6LoWPAN Sub-GHz Mesh, ethernet, cellular…

3.1.4. mbed Device Connector

+ Là một dịch vụ của mbed cho phép bạn kết nối các thiết bị IoT tới cloud mà không phải xây dựng hạ tầng.

+ Các đặc điểm chính:

- Bao gồm bảo mật end – to – end mạnh mẽ sử dụng các chuẩn bảo mật mở.

- Làm việc với REST APIs, làm cho nó trở nên dễ dàng trong việc tích hợp với hệ thống.

+ Lợi ích:

- Hoàn toàn tương thích với hệ điều hành mbed.

- Cho phép kết nối các thiết bị được hỗ trợ tới cloud đơn giản và an toàn.

- Hoàn toàn miễn phí với các lập trình viên IoT.

3.1.5. mbed Cloud

+ Cung cấp sự quản lý thiết bị IoT an toàn và cân bằng cho bất kỳ thiết bị, bất kỳ mạng hay bất kỳ cloud nào.

+ Đặc điểm:

- Hỗ trợ tất cả các thiết bị: mbed Cloud cung cấp sự vận chuyển và kết nối dựa trên các chuẩn đáng tin cậy, cùng với việc hỗ trợ tiết kiệm năng lượng cho các thiết bị hạn chế băng thông và năng lượng thấp.

- Quản lý từ xa.

- Kết nối dữ liệu: Kết nối được hỗ trợ bao gồm: CoAP/HTTP, TLS DTLS and OMALWM2M. mbed Cloud được xây dựng để cân bằng với các triển khai hàng loạt các nút IoT, sử dụng các cơ chế cache đặc biệt danh cho các thiết bị ít năng lượng.

- Bảo mật end – to – end.

- Tích hợp dễ dàng.

### 3.2. Làm việc với mbed

+ Công cụ phát triển:

- mbed CLI (Command Line Interface)

- IDE bên thứ 3: Nếu bạn đang dùng một công cụ lập trình khác, mBed hỗ trợ việc export cho một số công cụ sau:

* + - Keil uVision5
    - IAR Systems
    - GCC (ARM Embedded)
    - ARM Compiler 5
    - IAR ARM Compiler
    - Eclipse CDT using GCC (ARM Embedded)
    - ARM Compiler 5
    - IAR ARM Compiler
    - DS-5
    - LPCXpresso
    - Kinetis Design Studio
    - Simplicity Studio
    - Atmel Studio
    - SW4STM32 System Workbench for STM32
    - CooCox CoIDE
    - E2studio
    - Emblocks

- mbed online compiler: cho phép bạn viết và xây dựng các ứng dụng chỉ bằng trình duyệt web và kết nối USB.

+ Về việc import project hay tự tạo một chương trình và nhúng chương trình vào bo mạch bạn có thể xem chi tiết tại: <https://docs.mbed.com/docs/mbed-os-handbook/en/5.2/dev_tools/online_comp/>

+ Dưới đây là chương trình làm đèn led nhấp nháy viết trên mbed online compiler:

#include "mbed.h"

DigitalOut myled(LED1);

int main() {

while(1) {

myled = 1;

wait(0.2);

myled = 0;

wait(0.2);

}

}

### 3.3. APIs

3.3.1. Analog I/O

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **API** | **Công dụng** | **Chi tiết** |
| 1 | Analogin | Sử dụng để đọc một điện áp bên ngoài áp dụng cho một chân analog. Các chân analog bắt đầu bằng chữ A | <https://developer.mbed.org/handbook/AnalogIn> |
| 2 | Analogout | Sử dụng để thiết lập điện áp của một chân analog | <https://developer.mbed.org/handbook/AnalogOut> |

3.3.2. Digital I/O

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **API** | **Công dụng** | **Chi tiết** |
| 1 | DigitalIn | + Được dùng để đọc giá trị của một chân đầu vào số. DigitalIn có thể sử dụng trên bất kỳ chân nào có nhãn màu xanh dương.  + Chân đầu vào mang giá trị bằng “0” khi điện áp trên chân đó thấp hơn 0,8V và bằng “1” khi điện áp lớn hơn 2,0V. | <https://developer.mbed.org/handbook/DigitalIn> |
| 2 | DigitalOut | + Sử dụng để cấu hình và điều khiển một chân đầu ra số. DigitalOut có thể sử dụng trên bất kỳ chân nào có nhãn màu xanh dương và các đèn LED có sẵn ở trên bảng mạch.  + Gán giá trị bằng 0 để tắt và 1 để bật | <https://developer.mbed.org/handbook/DigitalOut> |
| 3 | DigitalInOut | + Dùng cho các chân số hai chiều (vừa đọc vừa ghi được).  + Bất kỳ chân nào có nhãn màu xanh dương được đánh số đều có thể sử dụng như là một DigitalInOut. | <https://developer.mbed.org/handbook/DigitalInOut> |
| 4 | BusIn | Tạo ra một vài chân DigitalIn có thể được đọc với cùng một giá trị | <https://developer.mbed.org/handbook/BusIn> |
| 5 | BusOut | Tạo ra một vài chân DigitalOut có thể được ghi với cùng một giá trị | <https://developer.mbed.org/handbook/BusOut> |

3.3.3. Timers

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **API** | **Công dụng** | **Chi tiết** |
| 1 | Timer | Sử dụng để create, start, stop và read một timer dùng để đo một khoản thời gian nhỏ. | <https://developer.mbed.org/handbook/Timer> |
| 2 | Timeout | Dùng để tạo ra một interrupt để gọi một hàm sau một khoảng thời gian delay | <https://developer.mbed.org/handbook/Timeout> |
| 3 | Ticker | Sử dụng để gọi một hàm lặp đi lặp lại tại một thời điểm cụ thể. | <https://developer.mbed.org/handbook/Ticker> |

3.3.4. Digital Interfaces

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **API** | **Công dụng** | **Chi tiết** |
| 1 | Serial | + Là một giao thức chung được dùng bởi các máy tính và các module điện tử để gửi và nhận dữ liệu và thông tin điều khiển.  + Serial link có hai kênh một chiều, một kênh dùng để nhận và một kênh dùng để gửi. Serial link  + Một trong các kết nối Serial đi qua cổng USB mbed, cho phép ta dễ dàng giao tiếp với máy tính.  + Serial có thể được sử dụng trên các chân được hỗ trợ và USBTX/USBRX | <https://developer.mbed.org/handbook/Serial> |

# **CHƯƠNG II: PHÂN TÍCH HỆ THỐNG**

## 1. Mô tả hệ thống

### 1.1. Nhiệm vụ cơ bản

### 1.2. Quy trình xử lý và quy tắc quản lý

## 2. Phân tích chức năng nghiệp vụ

### 2.1. Mô hình hóa chức năng nghiệp vụ

### 2.2. Mô hình hóa tiến trình nghiệp vụ

### 2.3. Đặc tả tiến trình nghiệp vụ

## 3. Phân tích dữ liệu nghiệp vụ

### 3.1. Mô hình dữ liệu ban đầu

### 3.2. Chuẩn hóa dữ liệu

### 3.3. Đặc tả dữ liệu

# **CHƯƠNG III: THIẾT KẾ HỆ THỐNG**

## 1. Thiết kế UX

## 2. Thiết kế cơ sở dữ liệu

## 3. Thiết kế UI

# **CHƯƠNG IV: KẾT LUẬN**

**5. Tài liệu tham khảo**

[1]. <https://www.arduino.cc/en/Guide/HomePage>

[2]. <https://en.wikipedia.org/wiki/Bluetooth_low_energy>

[3]. <https://www.mbed.com/en/>

[4]. <http://zedboard.org/product/microzed>

[5]. <https://www.raspberrypi.org/products/raspberry-pi-3-model-b/>

[6]. <http://www.webopedia.com/TERM/I/internet_of_things.html>

[7]. <https://spin.atomicobject.com/2015/08/19/learn-embedded-systems-programming/>

[8]. <http://arduino.vn/bai-viet/1181-vbluno-tutorial-2-tong-quan-ve-cong-nghe-ble-va-vi-du-minh-hoa-phan-1>

[9]. <https://vi.wikipedia.org/wiki/M%E1%BA%A1ng_l%C6%B0%E1%BB%9Bi_v%E1%BA%A1n_v%E1%BA%ADt_k%E1%BA%BFt_n%E1%BB%91i_Internet>

**6. Dự kiến kế hoạch**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **STT** | **Nội dung** | **Sản phẩm dự kiến** | **Tài liệu tham khảo** | **Thời gian thực hiện dự kiến** |
| 1 | Tìm hiểu về IoT, lập trình nhúng và công nghệ BLE | Tài liệu về đặc điểm; ưu, nhược điểm của IoT, lập trình nhúng và công nghệ BLE | [2], [6], [7] | 1 tuần |
| 2 | Tìm hiểu về GPRS và định vị GPS | Tài liệu về đặc điểm của GPRS, GPS |  | 1 tuần |
| 3 | Tìm hiểu các mạch | Tài liệu về đặc điểm mạch  arduino, nordic nRF51822 và cách thức làm việc với mỗi mạch | [1], [4], [5] | 2 tuần |
| 4 | Tìm hiểu về mbed | Tài liệu về các đặc điểm, công dụng của mbed. Hiểu rõ các API phục vụ cho việc phát triển ứng dụng | [3] | 1 tuần |
| 5 | Mô tả, phân tích hệ thống | + Mô tả về hệ thống, cách thức hoạt động của hệ thống.  + Đưa ra được các chức năng cần phải có của hệ thống. |  | 2 tuần |
| 6 | Thiết kế hệ thống | + Thiết kế UX cho ứng dụng  + Thiết kế UI cho ứng dụng |  | 1 tuần |
| 7 | Xây dựng, phát triển ứng dụng | Phát triển ứng dụng để đưa ra sản phẩm cuối cùng |  | 1 tháng |
| 8 | Kiểm thử và sửa lỗi | Sửa lỗi phát sinh trong ứng dụng |  | 1 tuần |
| 9 | Kết luận |  |  | 1 ngày |