

MICRO:BIT KẾT NỐI VẬN VẬT



Lê Trọng Nhân - Nguyễn Văn Hạnh
Lê Phương Nam

Mục lục

Chương 1. Tạo tài khoản trên ThingSpeak	7
1 Giới thiệu	8
2 Tạo tài khoản trên ThingSpeak	8
3 Tạo đồ thị hiển thị dữ liệu trên ThingSpeak	11
4 Lấy thông tin Key API	13
5 Câu hỏi ôn tập	15
Chương 2. Gửi dữ liệu lên ThingSpeak	17
1 Giới thiệu	18
2 Kết nối phần cứng	18
3 Lập trình gửi dữ liệu lên Server	20
3.1 Kết nối mạng Wifi	22
3.2 Gửi dữ liệu lên ThingSpeak	22
3.3 Cài tiến chương trình	23
4 Theo dõi kết quả từ xa	24
4.4 Theo dõi dữ liệu bằng trình duyệt web	24
4.5 Theo dõi dữ liệu bằng ứng dụng trên di động	24
5 Câu hỏi ôn tập	26
Chương 3. Dịch vụ cảnh báo IFTTT	27
1 Giới thiệu	28
2 Đăng ký tài khoản trên IFTTT	29
3 Cấu hình Webhook	30
4 Cấu hình gửi email	31
5 Kiểm tra hệ thống	35
6 Câu hỏi ôn tập	38
Chương 4. ThingHTTP và React trên ThingSpeak	39
1 Giới thiệu	40
2 Tạo ứng dụng ThingHTTP	41
3 Tạo ứng dụng React	42
4 Kiểm tra quy trình	44
5 Câu hỏi ôn tập	45
Chương 5. Cảm biến quan trắc tích hợp	47
1 Giới thiệu	48
2 Kết nối với DHT11	48
3 Nguyên lý hoạt động của DHT11	50

4	Lập trình với DHT11	50
5	Các phiên bản nâng cấp của DHT11	52
	5.1 Cảm biến DHT22	52
	5.2 Cảm biến AM2305	52
6	Câu hỏi ôn tập	54
Chương 6. Cảm biến tương tự Analog		55
1	Giới thiệu	56
2	Nguyên lý thiết kế cảm biến Analog	57
3	Cảm biến khí Gas	58
4	Đọc dữ liệu từ cảm biến khí Gas	59
5	Cảm biến Analog ChiPi	60
6	Câu hỏi ôn tập	62
Chương 7. Tạo tài khoản trên Adafruit IO		63
1	Giới thiệu	64
2	Tạo tài khoản trên Adafruit IO	64
3	Tạo kênh dữ liệu (Feed)	66
4	Chia sẻ Feed ở dạng Public	68
5	Câu hỏi ôn tập	70
Chương 8. Thiết kế Dashboard trên Adafruit IO		71
1	Giới thiệu	72
2	Tạo Dashboard mới	73
3	Thiết kế giao diện cho Dashboard	75
4	Chỉnh sửa nút nhấn	78
5	Kiểm tra kết nối giữa Feed và Dashboard	79
6	Câu hỏi ôn tập	81
Chương 9. Đồ thị trên Dashboard		83
1	Giới thiệu	84
2	Tạo Feed dữ liệu mới	85
3	Thêm đồ thị cho Dashboard	85
4	Kiểm tra tương tác giữa Feed và Dashboard	88
5	Câu hỏi ôn tập	90
Chương 10. Kết nối Microbit và Adafruit IO		91
1	Giới thiệu	92
	1.1 Username và Active Key	92
	1.2 Key của feed	92
2	Chương trình trên Microbit	93
	2.3 Kết nối với Adafruit IO	93
	2.4 Gửi dữ liệu lên Server	94
	2.5 Nhận dữ liệu từ server	95
3	Đồng hồ Internet	96
4	Câu hỏi ôn tập	98

Chương 11. Giao tiếp tầm xa LoRa	99
1 Giới thiệu	100
2 Kết nối SX1278 với máy tính	101
3 Cấu hình giao tiếp LoRa	101
4 Kiểm tra việc gửi nhận trên máy tính	103
5 Câu hỏi ôn tập	105
Chương 12. Giao tiếp tầm xa LoRa	107
1 Giới thiệu	108
2 Kết nối phần cứng với MicroBit	109
3 Gửi nhận dữ liệu qua LoRa	109
3.1 Gửi dữ liệu	109
3.2 Nhận dữ liệu	110
4 Đổi cổng kết nối UART	110
5 Câu hỏi ôn tập	112
Chương 13. Xây dựng mạng cảm biến LoRa	113
1 Giới thiệu	114
2 Chương trình gửi dữ liệu	115
3 Chương trình nhận dữ liệu	115
4 Câu hỏi ôn tập	117

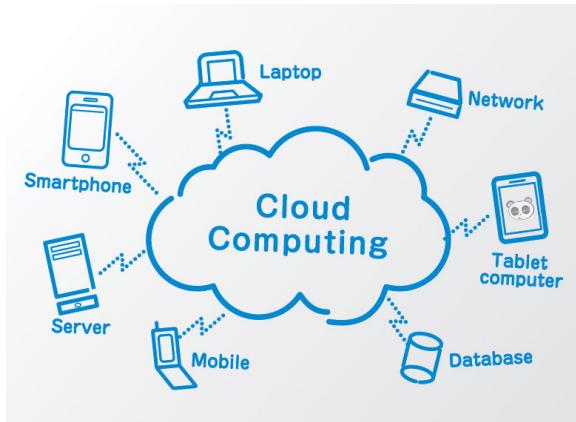
CHƯƠNG 1

Tạo tài khoản trên ThingSpeak



1 Giới thiệu

Một trong những nét đặc trưng của các ứng dụng kết nối vạn vật, là việc chia sẻ dữ liệu giữa các thiết bị tham gia vào mạng. Để làm được điều đó, dữ liệu của từng thiết bị sẽ được lưu trữ lên các điện toán đám mây hoặc các máy chủ.



Hình 1.1: Cấu trúc điện toán đám mây

Điện toán đám mây được hình thành bởi việc liên kết nhiều máy chủ, và dữ liệu sẽ được phân tán khắp toàn cầu, dựa vào mạng Internet. Do đó, đóng vai trò là người dùng, chúng ta có thể xem được dữ liệu của một thiết bị. Tất cả những gì mà chúng ta cần, là một thiết bị, từ máy tính, máy tính bảng hay điện thoại di động, với khả năng truy cập vào mạng Internet, như minh họa ở hình bên trên.

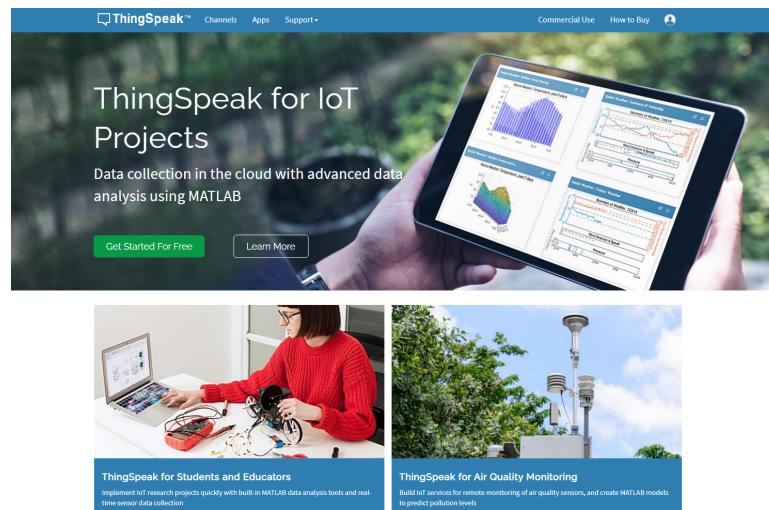
Trong bài này, chúng tôi sẽ hướng dẫn bạn đọc tạo một tài khoản trên máy chủ ThingSpeak, một trong những máy chủ thông dụng bậc nhất cho các ứng dụng liên quan đến kết nối vạn vật. Không chỉ miễn phí, máy chủ này còn cung cấp một giao diện thân thiện cho người dùng thao tác.

Đi sâu hơn về góc độ kĩ thuật, ThingSpeak là một nền tảng đám mây dựa trên giao thức HTTP. Giao thức này hiển nhiên rất thân thuộc với chúng ta, bởi nó là giao thức mỗi khi chúng ta sử dụng trình duyệt và truy cập vào một trang web. Giao thức này tương đối ổn định và rất phù hợp cho người mới bắt đầu hiện thực các ứng dụng giám sát thông tin từ xa. Hiện tại, ThingSpeak đã được sở hữu bởi The MathWorks, Inc., là tập đoàn sáng tạo ra phần mềm MathLab nổi danh toàn cầu trong tất cả lĩnh vực kĩ thuật, kinh tế. Các kĩ năng đặt được trong bài hướng dẫn này như sau:

- Tạo được một tài khoản trên ThingSpeak
- Cấu hình được trên máy chủ ThingSpeak
- Tạo được giao diện hiển thị đồ thị trên máy chủ

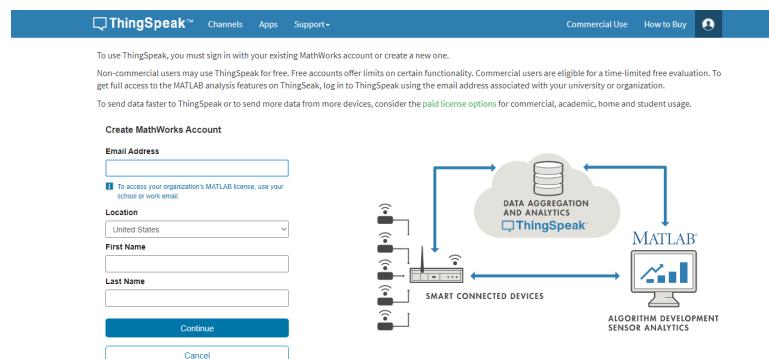
2 Tạo tài khoản trên ThingSpeak

Để tạo một tài khoản, chúng ta vào trang web <https://thingspeak.com/>, với giao diện của trang web như hình bên dưới.



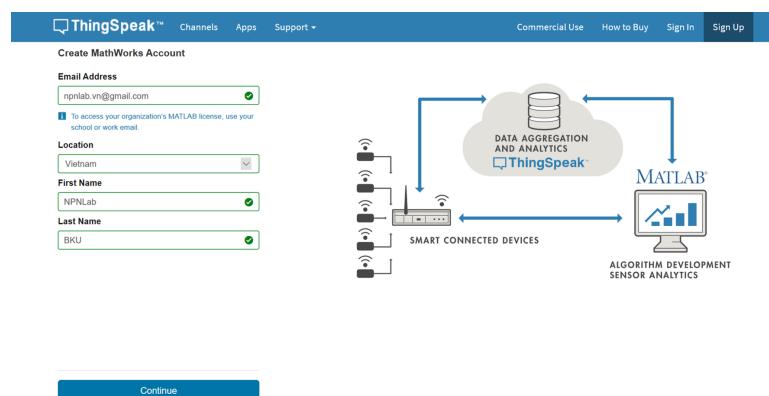
Hình 1.2: Giao diện trang web ThingSpeak.

Tiếp theo, chúng ta sẽ nhấp vào nút Sign Up, để tạo một tài khoản, nằm ở góc trên bên phải màn hình, giao diện sau đây sẽ được hiện ra. **Khi nhập thông tin cá nhân, chúng ta không được phép nhập tiếng Việt có dấu.**



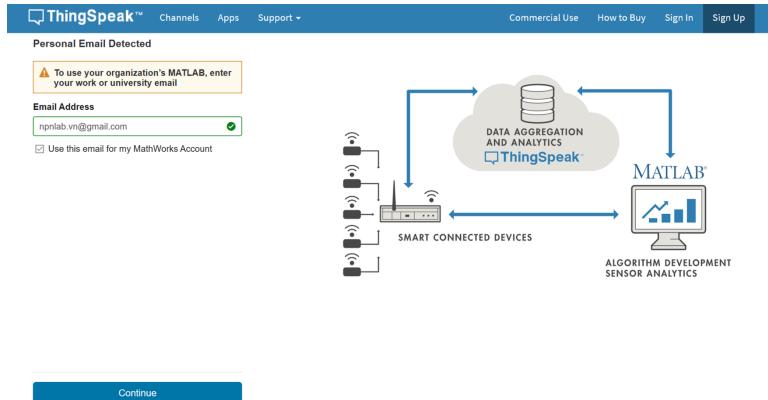
Hình 1.3: Tạo một tài khoản trên ThingSpeak.

Chúng ta điền đầy đủ thông tin, và nhấn nút Continue, ở phía cuối trang, như hình ảnh bên dưới:



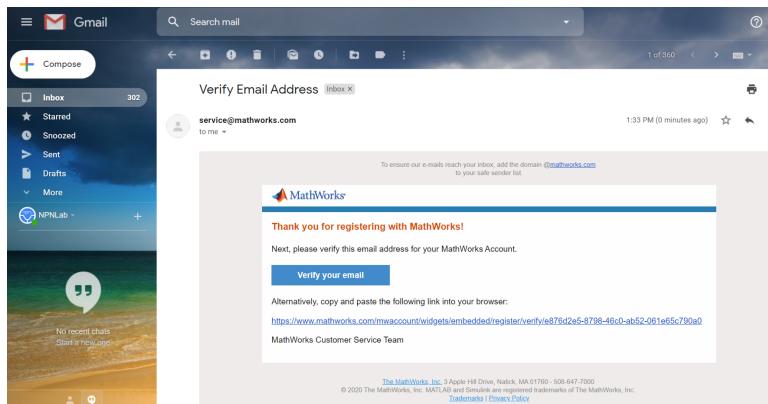
Hình 1.4: Điền đầy đủ thông tin và nhấn nút Continue.

Với chính sách hiện tại trên ThingSpeak, bạn cần phải xác thực email cá nhân, cũng như dùng tài khoản này cho dịch vụ MATLAB. Bạn hãy check chọn vào mục **Use this email for my MathWorks Account**, rồi nhấn tiếp **Continue**, như hình bên dưới.



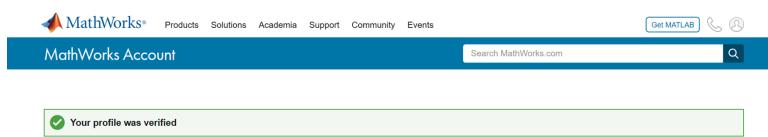
Hình 1.5: Check vào tùy chọn cho dịch vụ MathLab và nhấn Continue.

Đây là bước mà bạn đọc có thể làm không hợp lệ. **Trước khi nhấn vào nút Continue, bạn phải mở hộp mail của mình để xác nhận email hợp lệ.** Bạn hãy đợi khoảng 5 phút, sau đó mở email của mình, nhấn vào đường liên kết được gửi tới email hoặc nhấn vào nút Verify your email, như kết quả ở hình bên dưới:



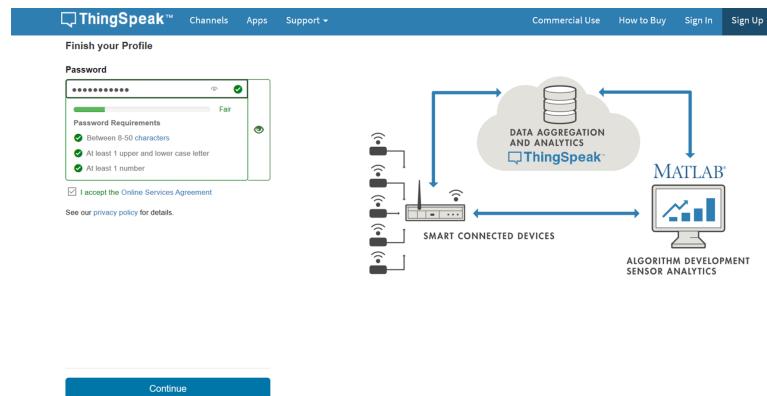
Hình 1.6: Xác thực email bằng cách nhấn vào Verify your email.

Khi mọi bước thành công, giao diện sau đây hiện lên, báo hiệu việc xác thực email thành công:



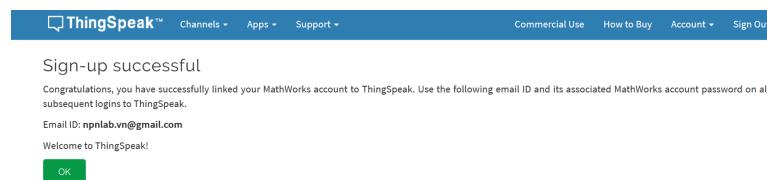
Hình 1.7: Xác thực email thành công.

Lúc này, giao diện trên trang ThingSpeak sẽ yêu cầu bạn nhập thêm thông tin mật khẩu, là một chuỗi có từ 8 đến 50 ký tự, có ít nhất 1 ký tự số và ít nhất 1 ký tự viết hoa.



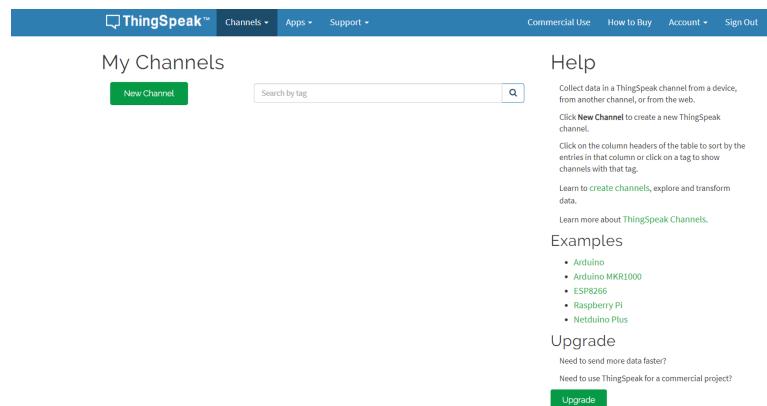
Hình 1.8: Nhập thông tin mật khẩu cho tài khoản.

Sau khi chọn được mật khẩu hợp lệ, nút Continue sẽ cho phép bạn nhấn, để hoàn thành việc đăng kí tài khoản trên ThingSpeak. Thông báo sau đây sẽ xuất hiện khi việc đăng kí tài khoản thành công.



Hình 1.9: Đăng kí tài khoản thành công.

Cuối cùng, chúng ta nhấn nút OK để đến với kênh dữ liệu của mình.



Hình 1.10: Kênh dữ liệu của người dùng trên ThingSpeak.

3 Tạo đồ thị hiển thị dữ liệu trên ThingSpeak

Ban đầu, kênh dữ liệu của chúng ta chưa có giao diện hiển thị, chúng ta bắt đầu tạo các đồ thị thống kê cho server của mình, bằng cách nhấn vào nút New Channel ở Hình 1.10. Giao diện bên dưới sẽ hiện ra để chúng ta tiếp tục các bước hình cho các kênh dữ liệu của mình.

New Channel

Name: IoT - Microbit - Raspberry Pi

Description: Hệ thống giám sát thông tin nhiệt độ và độ ẩm

Field 1	Nhiệt Độ	<input checked="" type="checkbox"/>
Field 2	Ánh Sáng	<input checked="" type="checkbox"/>
Field 3		<input type="checkbox"/>
Field 4		<input type="checkbox"/>
Field 5		<input type="checkbox"/>
Field 6		<input type="checkbox"/>
Field 7		<input type="checkbox"/>
Field 8		<input type="checkbox"/>

Channel Settings

- Percentage complete: Calculated based on data entered into the various fields of a channel. Enter the name, description, location, URL, video, and tags to complete your channel.
- Channel Name: Enter a unique name for the ThingSpeak channel.
- Description: Enter a description of the ThingSpeak channel.
- Fields: Check the box to enable the field, and enter a field name. Each ThingSpeak channel can have up to 8 fields.
- Metadata: Enter information about channel data, including JSON, XML, or CSV data.
- Tags: Enter keywords that identify the channel. Separate tags with commas.
- Link to External Site: If you have a website that contains information about your ThingSpeak channel, specify the URL.
- Show Channel Location:
 - Latitude: Specify the latitude position in decimal degrees. For example, the latitude of the city of London is 51.5072.

Hình 1.11: Cấu hình các kênh dữ liệu trên server.

Với phiên bản miễn phí, ThingSpeak chỉ hỗ trợ tối đa 8 kênh dữ liệu mà thôi. Trong ứng dụng của chúng ta, thông tin về nhiệt độ và cường độ ánh sáng được thu thập. Do đó, chúng ta sẽ check chọn hiển thị 2 kênh, Field 1 và Field 2. Bạn có thể đặt tên cho nó, cũng như điền các thông tin khác, như minh họa ở Hình 1.11.

Cuối cùng, chúng ta kéo xuống cuối trang, và nhấn nút Save Channel, như trình bày ở hình bên dưới:

Elevation

Show Channel Location

Latitude: 0.0

Longitude: 0.0

Show Video

YouTube

Vimeo

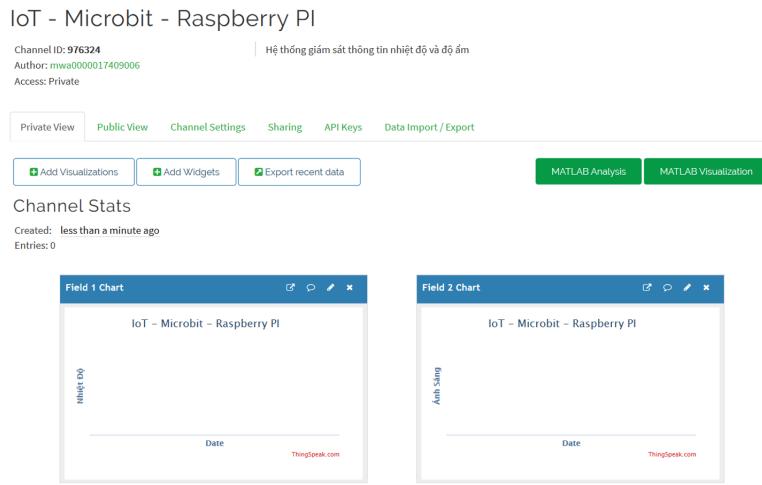
Video URL: http://

Show Status

Save Channel

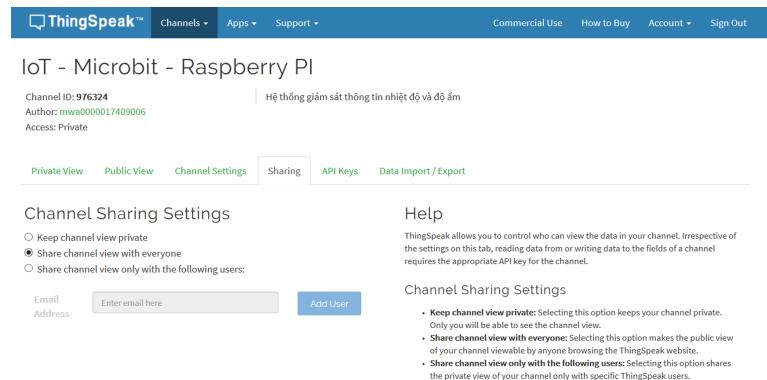
Hình 1.12: Lưu cấu hình của kênh thu thập dữ liệu ThingSpeak.

Sau khi kết thúc quá trình cấu hình kênh của chúng ta, giao diện sau đây sẽ xuất hiện, với 2 đồ thị thu thập dữ liệu nhiệt độ và ánh sáng:



Hình 1.13: Giao diện hiển thị trên server ThingSpeak.

Tuy nhiên, dữ liệu hiển thị này chỉ có chúng ta mới xem được. Muốn chia sẻ dữ liệu này với cộng đồng, chúng ta nhấp vào tab **Sharing**, và chọn vào tùy chọn thứ 2, **Share channel view with everyone**, như hướng dẫn ở hình bên dưới.



Hình 1.14: Chia sẻ kênh dữ liệu với cộng đồng.

4 Lấy thông tin Key API

Sau khi cấu hình server xong, chúng ta sẽ tìm đến mục API Keys. Mục này sẽ xuất hiện rất nhiều vị trí trên trang web, ngay khi đăng nhập vào tài khoản, chúng ta cũng sẽ tìm thấy mục API Keys dễ dàng, như trình bày ở hình bên dưới:

My Channels

Name	Created	Updated
IoT - Microbit - Raspberry Pi	2020-01-30	2020-01-30 06:44

Private Public Settings Sharing API Keys Data Import / Export

Hình 1.15: Giao diện khi đăng nhập lại vào server.

Nhấn vào mục API Keys, giao diện sau đây sẽ hiện ra:

Write API Key

Key: ZDC919EE3WYRYGSE

[Generate New Write API Key](#)

Read API Keys

Key: JBUNWD2MI0063BWM

Note:

[Save Note](#) [Delete API Key](#)

[Add New Read API Key](#)

Help

API keys enable you to write data to a channel or read data from a private channel. API keys are auto-generated when you create a new channel.

API Keys Settings

- **Write API Key:** Use this key to write data to a channel. If you feel your key has been compromised, click [Generate New Write API Key](#).
- **Read API Keys:** Use this key to allow other people to view your private channel feeds and charts. Click [Generate New Read API Key](#) to generate an additional read key for the channel.
- **Note:** Use this field to enter information about channel read keys. For example, add notes to keep track of users with access to your channel.

API Requests

Write a Channel Feed
GET https://api.thingspeak.com/update?api_key=ZDC919EE3WYRYGSE&field1=1

Read a Channel Feed
GET <https://api.thingspeak.com/channels/976324/feeds.json?results=2>

Read a Channel Field
GET <https://api.thingspeak.com/channels/976324/fields/1.json?results=2>

Hình 1.16: Giao diện quản lý API Keys.

Thông tin ở mục Key cần được lưu lại, chuẩn bị cho việc lập trình trên Microbit cho bài tiếp theo.

5 Câu hỏi ôn tập

1. Để kết nối với mạng WIFI, các thông tin nào là cần thiết cho module ESP8266?
 - A. SSID và KEY
 - B. SSID và API KEY
 - C. KEY và API KEY
 - D. API KEY
2. Trong dự án, dữ liệu được gửi lên server cloud nào?
 - A. IoT Server
 - B. Adafruit IO Server
 - C. ThingSpeak Server
 - D. Tất cả các server trên
3. Để kết nối lên server ThingSpeak, ta sử dụng khái niệm nào?
 - A. ESP8266_IoT
 - B. NPNBitKit
 - C. NPNLCD
 - D. Radio
4. Trong câu lệnh **set data to send ThingSpeak**, API key để truyền vô câu lệnh được lấy ở đâu?
 - A. Trên tài khoản server ThingSpeak
 - B. Trên trang web Microbit
 - C. Tìm miễn phí trên Google
 - D. Mua từ trang ThingSpeak
5. Phần mềm trên di động để xem dữ liệu trực tuyến trên ThingSpeak là gì?
 - A. ThingSpeak
 - B. ThingSpeak IoT
 - C. ThingView
 - D. Tất cả các phần mềm trên
6. Để gửi dữ liệu thành công lên server ThingSpeak, khoảng thời gian đợi giữa 2 lần gửi liên tiếp nên là?
 - A. 0.5s
 - B. 1s
 - C. 2s
 - D. 30s
7. Server ThingSpeak hỗ trợ gửi tối đa bao nhiêu trường dữ liệu lên server?
 - A. 2
 - B. 4
 - C. 6
 - D. 8

Đáp án

1. A 2. C 3. A 4. A 5. C 6. D 7. D

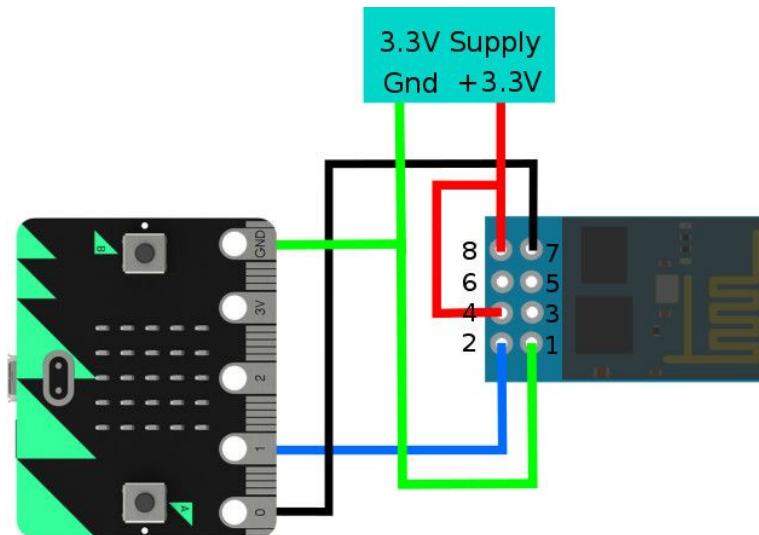
CHƯƠNG 2

Gửi dữ liệu lên ThingSpeak



1 Giới thiệu

Sau khi đã tạo xong tài khoản trên server ThingSpeak, trong bài này chúng ta có thể sử dụng mạch Microbit để gửi dữ liệu lên ThingSpeak. Tuy nhiên, bản thân mạch Microbit không có khả năng kết nối vào mạng Internet. Do đó, chúng ta cần phải có 1 thiết bị tích hợp để hỗ trợ kết nối Wifi cho mạch Microbit. Thiết bị giới thiệu trong bài này có tên gọi là ESP8266, một thiết bị hỗ trợ kết nối Wifi vô cùng phổ biến trong các ứng dụng vi điều khiển. Không khó để bạn đọc có thể tìm thấy những hướng dẫn liên quan đến Microbit và ESP8266, kể cả những hướng dẫn kết nối từ chân của Microbit, như minh họa ở hình bên dưới:



Hình 2.1: Một ví dụ để kết nối giữa MicroBit và ESP8266

Thành tựu của ngành Cơ Vi Điện Tử, cho phép ra đời những sản phẩm ngày càng tích hợp và giá thành rẻ, như ESP8266 là một ví dụ. Nhờ thiết bị này, gần như bất kì thiết bị vi điều khiển nào cũng có khả năng kết nối lên mạng và giao tiếp dữ liệu. Đây cũng là một thực tế, hình thành nên thế giới mạng mới, gọi là Kết Nối Vạn Vật, khi bây giờ, không chỉ riêng máy tính, rất nhiều các thiết bị khác có thể kết nối vào mạng Internet. Các mục tiêu chính trong bài hướng dẫn này như sau:

- Kết nối được mạch Microbit và mô đun Wifi ESP8266
 - Lập trình gửi dữ liệu từ Microbit lên Server ThingSpeak
 - Xem kết quả từ xa trên máy tính và thiết bị di động

Bạn đọc cần lưu ý rằng, trong bài này, chúng ta sẽ sử dụng thông tin **Write API Key** ở bài trước cho việc lập trình.

2 Kết nối phần cứng

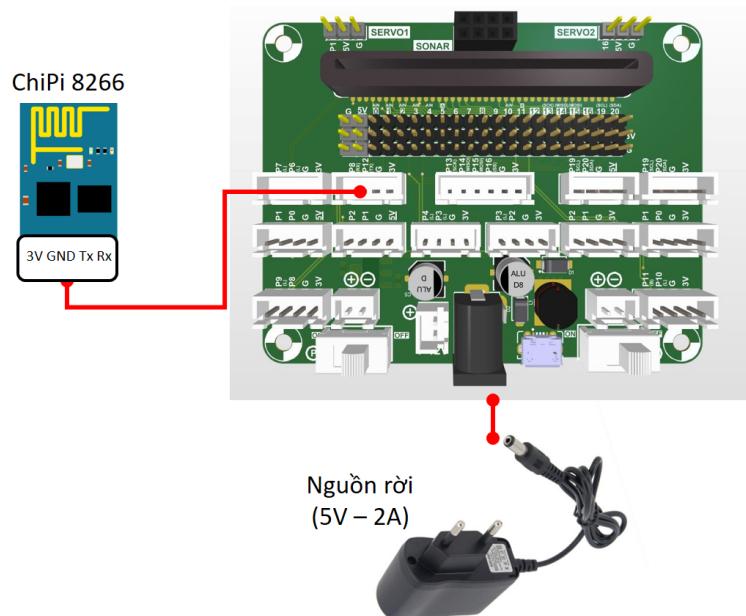
Đầu tiên, với bất kì thiết bị điện tử nào kết nối thêm vào mạch Microbit, bạn đọc cần phải lưu ý về nguồn của nó. Bên cạnh mức điện áp, công suất của nguồn cũng rất quan trọng để duy trì độ ổn định của thiết bị. Nếu công suất không đủ, sẽ xảy ra hiện tượng sụt áp (điện áp nguồn bị giảm), làm hệ thống không hoạt động được

do thiếu điện áp nguồn.

Với ESP8266, mặc dù điện áp cần để thiết bị hoạt động là 3V3, tuy nhiên bạn không thể lấy nguồn trực tiếp từ chân 3V của mạch Microbit được, vì dòng của nó không đảm bảo, hay nói cách khác, công suất cấp nguồn là không đủ. Do đó, một mạch mở rộng và một nguồn ngoài (adapter) sẽ được sử dụng để cấp nguồn cho cả hệ thống, và ESP8266 sẽ lấy nguồn từ adapter rời, như minh họa ở Hình 2.1.

Tiếp theo, về chân tín hiệu của thiết bị, ESP8266 cần kết nối tối thiểu 2 chân với Microbit, bao gồm Tx và Rx. Đây là 2 chân đặc trưng cho việc gửi lệnh giữa Microbit và ESP8266. Tx, chân truyền dữ liệu, và Rx, chân nhận dữ liệu, cũng là 2 kí hiệu đặc trưng của giao tiếp nối tiếp Uart Serial. Đây là 2 chân quan trọng sẽ được sử dụng cho việc lập trình sắp tới. Cuối cùng, chân EN của ESP8266 sẽ được nối lên nguồn. Chân này cho phép mạch ESP8266 nhận lệnh. Trong các ứng dụng đơn giản, chúng ta không cần phải điều khiển chân EN, mà chỉ luôn nối nó lên nguồn.

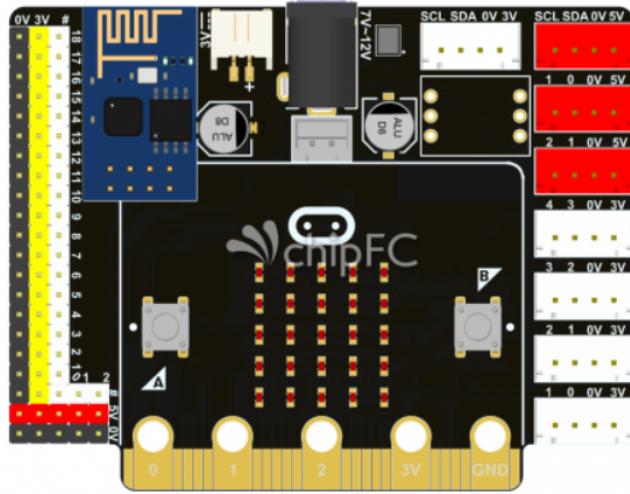
Đối với thiết bị được hỗ trợ bởi The Dariu Foudation, kết nối đã được chuẩn hóa thành 4 chân cắm. Bạn chỉ đơn giản là cắm vào cổng có kí hiệu UART trên mạch mở rộng, với 2 chân kết nối quan trọng là **P8** và **P12**. Cũng tương tự như trên, bạn cũng cần một nguồn rời để cấp cho mạch mở rộng và MicroBit. Kết nối giữa Microbit và ESP8266 thông qua mạch mở rộng ChiPi sẽ như sau:



Hình 2.2: Thêm thư viện trên MakeCode

Mô đun ESP8266 cũng được thiết kế lại để ra chuẩn nối 4 dây, tương thích hoàn toàn với mạch mở rộng ChiPi Base Shield. Điều này vô cùng thuận tiện cho người mới bắt đầu lập trình với các thiết bị ngoại vi.

Bên cạnh đó, chúng tôi cũng phát triển một mạch tích hợp gồm Microbit, mạch mở rộng và thiết bị WIFI ESP8266, gọi là mạch Microbit V3, có hình ảnh như sau: Bên cạnh việc tích hợp mạch mở rộng và thiết bị ESP8266, các khe cắm trên mạch V3 cũng được thiết kế theo hướng dễ kết nối hơn với các thiết bị trên thị trường.

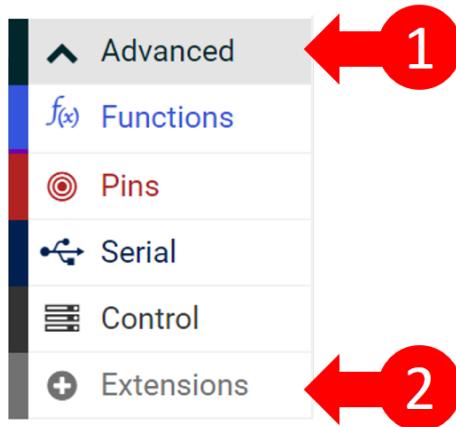


Hình 2.3: Mạch Microbit tích hợp WiFi phiên bản 3

Đặc biệt, mạch có thêm các khe cắm màu đỏ, có nguồn cấp 5V rất thuận tiện cho các thiết bị ngoại vi cần điện áp cao và dòng tiêu thụ lớn.

3 Lập trình gửi dữ liệu lên Server

Để hỗ trợ cho việc lập trình trong bài này, cũng như các bài còn lại trong toàn bộ giáo trình, thư viện IoTBitKit sẽ được thêm vào. Trình tự thêm thư viện này như sau: từ màn hình lập trình chọn vào **Advanced**, chọn tiếp **Extensions**, như hướng dẫn ở hình bên dưới:

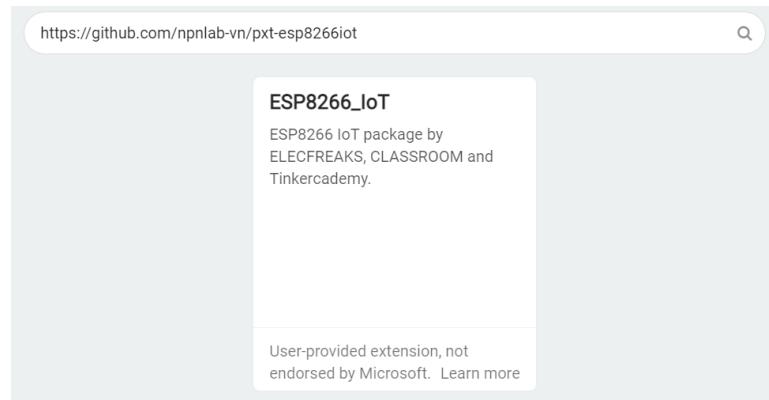


Hình 2.4: Thêm thư viện trên MakeCode

Với giao diện mới hiện ra, chúng ta nhập và đường dẫn sau đây:

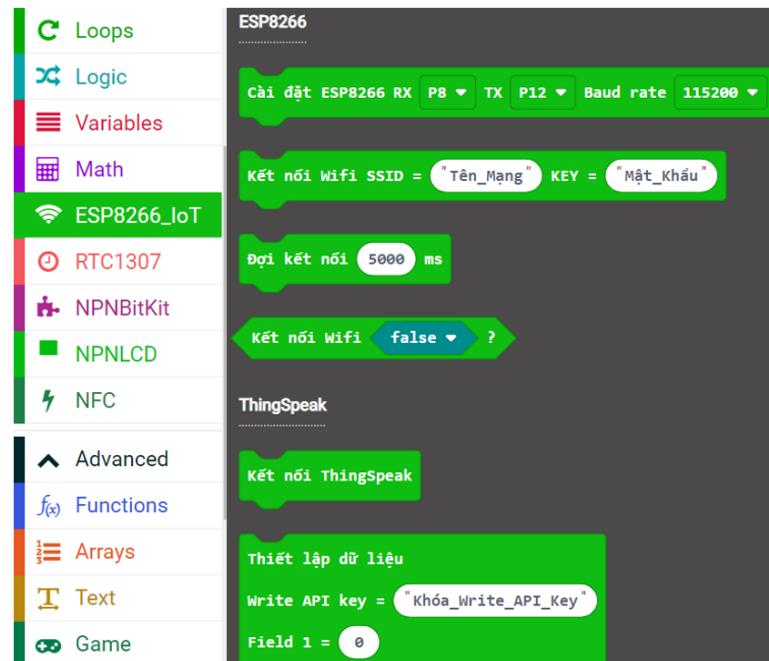
<https://github.com/npnlab-vn/pxt-esp8266iot>

Cuối cùng, nhấn vào nút tìm kiếm, chúng ta sẽ có kết quả như minh họa ở hình bên dưới:



Hình 2.5: Thêm thư viện ESP8266 IoT

Thư viện hỗ trợ cho việc kết nối Wifi và gửi dữ liệu lên server là **ESP8266_IoT**. Tiếp tục nhấp vào kết quả tìm được, thư viện lập trình sẽ được thêm vào như minh họa ở hình bên dưới:



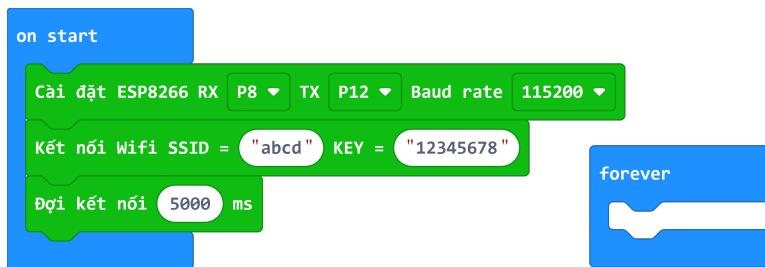
Hình 2.6: Các câu lệnh thuộc nhóm ESP8266_IoT

Trong thư viện ESP8266_IoT, hỗ trợ 4 nhóm câu lệnh là ESP8266 và ThingSpeak. Với nhóm đầu tiên, là các tác vụ cấu hình và kết nối với mạng Wifi. Bạn cũng có thể dùng điện thoại để tạo ra điểm phát Wifi trong trường hợp không có mạng. Nhóm thứ 2 là các câu lệnh liên quan đến việc gửi dữ liệu định kì lên ThingSpeak.

Nhóm lệnh thứ 3 và 4 sẽ là các dịch vụ cao cấp hơn cho các ứng dụng kết nối vạn vật. Trong khi nhóm lệnh 3 sẽ dành cho 1 server mới là Adafruit IO, nhóm 4 sẽ hỗ trợ cho bạn đọc về thời gian hiện tại của hệ thống. Nhóm lệnh 3 và 4 sẽ được trình bày từ bài 7 của giáo trình này.

3.1 Kết nối mạng Wifi

Đầu tiên, để kết nối với Wifi, chúng ta sẽ hiện thực các câu lệnh sau trong khôi **on start**, như sau:



Hình 2.7: Thiết lập kết nối mạng WiFi

Để cho dễ gọi nhớ, đây là 3 câu lệnh đầu tiên trong mục ESP8266. Chức năng của 3 câu lệnh này như sau:

- Câu lệnh 1: Chỉ định việc kết nối giữa ESP8266 và mạch MicroBit ở chân P8 và P12, tốc độ giao tiếp là 115200. Đây là các trạng thái mặc định của câu lệnh đầu tiên và bạn không cần phải thay đổi gì. Trong trường hợp bạn kết nối với chân khác, bạn cần phải thay đổi cho đúng vị trí chân kết nối.
- Câu lệnh 2: Bạn sẽ thay đổi 2 thông số ở câu lệnh này, bao gồm tên và mật khẩu của mạng WiFi. Trong ví dụ ở đây, mạng có tên là **abcd** và mật khẩu là **12345678**.
- Câu lệnh 3: Đợi thêm 5 giây cho việc kết nối vào mạng WiFi. Trong điều kiện bình thường, khoảng 2 giây là hệ thống đã có thể kết nối vào mạng WiFi.

3.2 Gửi dữ liệu lên ThingSpeak

Định kì mỗi 30 giây, chúng ta sẽ gửi thông tin nhiệt độ và cường độ ánh sáng từ mạch Microbit lên server. Bằng cách sử dụng các câu lệnh trong nhóm **ThingSpeak**, chương trình gợi ý cho bạn đọc như sau:



Hình 2.8: Gửi dữ liệu lên server mỗi 30 giây

Chức năng của 3 câu lệnh liên quan đến việc gửi dữ liệu lên server được tóm tắt như sau:

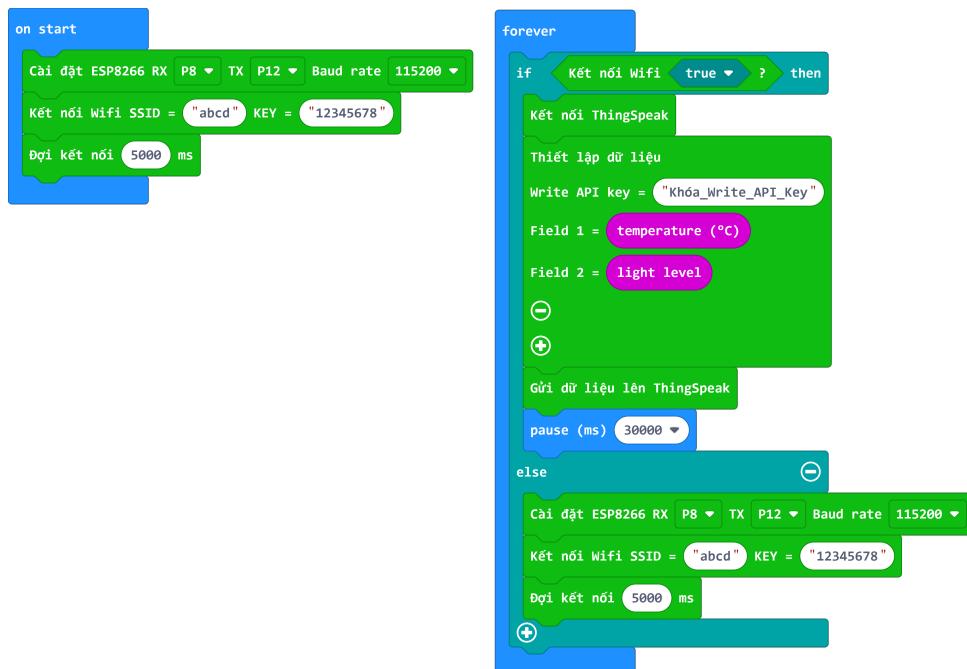
- Câu lệnh 1: Kết nối với server
- Câu lệnh 2: Khai báo APIKey và các trường dữ liệu gửi lên server.
- Câu lệnh 3: Bắt đầu tải dữ liệu lên server

Trong 3 câu lệnh liệt kê ở trên, quan trọng nhất là câu lệnh thứ 2. Bạn phải chỉ định Write API key cho chính xác thì dữ liệu mới có thể tải lên thành công được. Thông tin này được sinh ra khi bạn tạo tài khoản trên server ThingSpeak. Cuối cùng, vì server ThingSpeak là miễn phí, rất nhiều ứng dụng đang tải dữ liệu lên nó. Do đó, ThingSpeak phải giới hạn việc gửi dữ liệu lên một cách liên tục. Trong trường hợp này, bạn nên để **cách nhau khoảng 30 giây** trở lên cho 2 lần tải dữ liệu liên tiếp lên server. Chương trình được chia sẻ ở đường dẫn sau đây:

https://makecode.microbit.org/_aLgf9C03f1oc

3.3 Cải tiến chương trình

Thực ra, các bước hiện thực trình bày ở trên chỉ là những tác vụ cơ bản nhất. Để chương trình có thể hoàn thiện hơn, bạn đọc có thể sử dụng thêm các điều kiện được hỗ trợ sẵn trong thư viện. Chẳng hạn như việc gửi dữ liệu lên server chỉ được thực hiện khi có kết nối Wifi. Ngược lại, chương trình sẽ phải kết nối lại Wifi. Tính năng này có thể được hiện thực như sau:



Hình 2.9: Kiểm tra kết nối WiFi trước khi gửi dữ liệu

Bạn thậm chí cũng có thể kiểm tra việc gửi dữ liệu lên server có thành công hay không. Thậm chí, bạn có thể loại bỏ các câu lệnh đợi và thay thế nó bằng các kĩ thuật đã được hướng dẫn trong giáo trình Microbit - Nhà Thông Minh. Phần mở rộng này xem như bài tập nâng cao cho bạn đọc. Chương trình bên trên được chia sẻ ở đường dẫn sau:

4 Theo dõi kết quả từ xa

Khi dữ liệu đã được tải lên server thành công, có nhiều cách để có thể theo dõi dữ liệu từ xa, được trình bày bên dưới.

4.4 Theo dõi dữ liệu bằng trình duyệt web

Đầu tiên là sử dụng trình duyệt web, chẳng hạn như máy tính. Bạn có thể vào trang ThingSpeak, đăng nhập bằng tài khoản của mình rồi xem dữ liệu ở chế độ Public View, như sau:

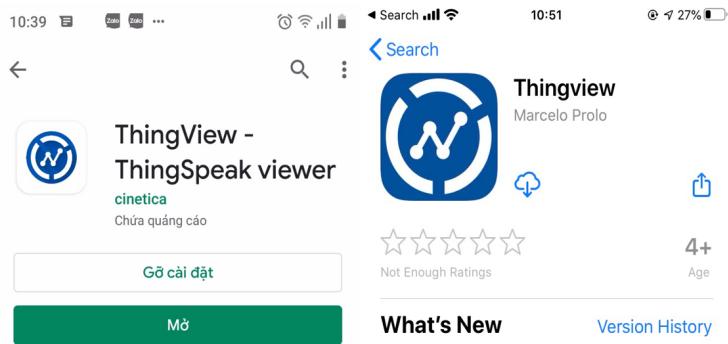


Hình 2.10: Xem dữ liệu bằng trình duyệt web

Một lợi thế của giải pháp này là hầu như trình duyệt web đều có sẵn trên máy tính và các thiết bị di động (điện thoại thông minh hay máy tính bảng). Tuy nhiên chúng ta cần phải đăng nhập vào trang web ThingSpeak trước khi có thể xem được dữ liệu.

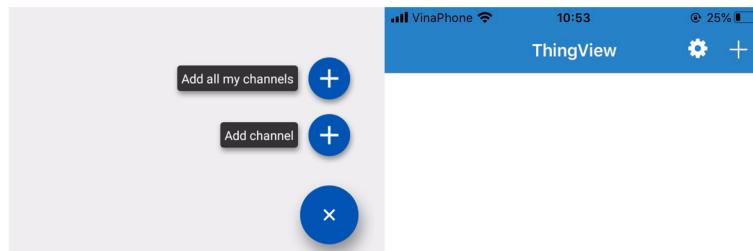
4.5 Theo dõi dữ liệu bằng ứng dụng trên di động

Trong phương pháp thứ 2, bạn cần phải cài đặt thêm phần mềm, và chỉ hỗ trợ cho các thiết bị di động (điện thoại thông minh và máy tính bảng). Phần mềm này có tên là ThingView, được hỗ trợ trên cả Android và iOS, như trình bày ở hình bên dưới:



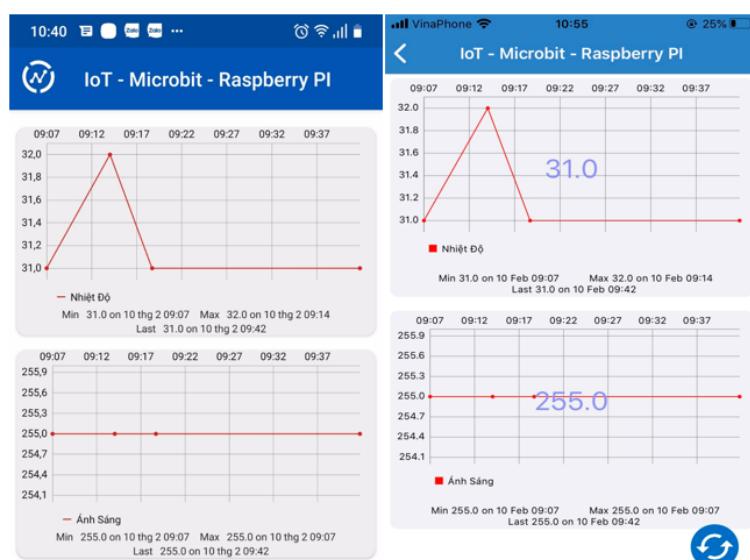
Hình 2.11: Cài đặt phần mềm ThingView

Sau khai cài đặt và khởi động chương trình, bạn hãy nhấn dấu + (ở phía dưới cho Android và ở phía trên cho iOS), và chọn Add Channel:



Hình 2.12: Chọn tiếp Add channel trên Android và iOS

Giao diện sau đây sẽ hiện ra để chúng ta nhập ID của kênh ThingSpeak. Thông tin về ID bạn có thể tìm thấy ở góc bên khi chuyển sang chế độ Public View. Tiếp theo, chúng ta nhấn nút Search, rồi nối tiếp nút SAVE. Bây giờ, chúng ta đã có thể xem dữ liệu của server ThingSpeak trên điện thoại di động, như hình minh họa dưới đây:



Hình 2.13: Kết quả hiển thị trên di động

5 Câu hỏi ôn tập

1. Để kết nối với mạng WIFI, các thông tin nào là cần thiết cho module ESP8266?
 - A. SSID và KEY
 - B. SSID và API KEY
 - C. KEY và API KEY
 - D. API KEY
2. Trong dự án, dữ liệu được gửi lên server cloud nào?
 - A. IoT Server
 - B. Adafruit IO Server
 - C. ThingSpeak Server
 - D. Tất cả các server trên
3. Để kết nối lên server ThingSpeak, ta sử dụng khái lệnh nào?
 - A. ESP8266_IoT
 - B. NPNBitKit
 - C. NPNLCD
 - D. Radio
4. Trong câu lệnh **set data to send ThingSpeak**, API key để truyền vô câu lệnh được lấy ở đâu?
 - A. Trên tài khoản server ThingSpeak
 - B. Trên trang web Microbit
 - C. Tìm miễn phí trên Google
 - D. Mua từ trang ThingSpeak
5. Phần mềm trên di động để xem dữ liệu trực tuyến trên ThingSpeak là gì?
 - A. ThingSpeak
 - B. ThingSpeak IoT
 - C. ThingView
 - D. Tất cả các phần mềm trên
6. Để gửi dữ liệu thành công lên server ThingSpeak, khoảng thời gian đợi giữa 2 lần gửi liên tiếp nên là?
 - A. 0.5s
 - B. 1s
 - C. 2s
 - D. 30s
7. Server ThingSpeak hỗ trợ gửi tối đa bao nhiêu trường dữ liệu lên server?
 - A. 2
 - B. 4
 - C. 6
 - D. 8

Đáp án

1. A 2. C 3. A 4. A 5. C 6. D 7. D

CHƯƠNG 3

Dịch vụ cảnh báo IFTTT



1 Giới thiệu

Việc gửi dữ liệu lên server ThingSpeak ở bài hướng dẫn trước, thực chất chúng ta đang sử dụng giao thức Internet truyền thống, giao thức http, và bản nâng cấp của nó, https. Hạn chế của giao thức này thì bạn đọc cũng có thể thấy rõ, độ đáp ứng của nó khá chậm. Điều này có thể dễ dàng nhận thấy khi chúng ta không thể gửi dữ liệu một cách thường xuyên lên server được. Chu kì gửi dữ liệu chỉ dừng lại tối đa là 30 giây. Tuy nhiên, với nhiều ứng dụng không cần phải gửi liên tục, chúng ta vẫn có thể ứng dụng được.



Hình 3.1: Dịch vụ cảnh báo IFTTT

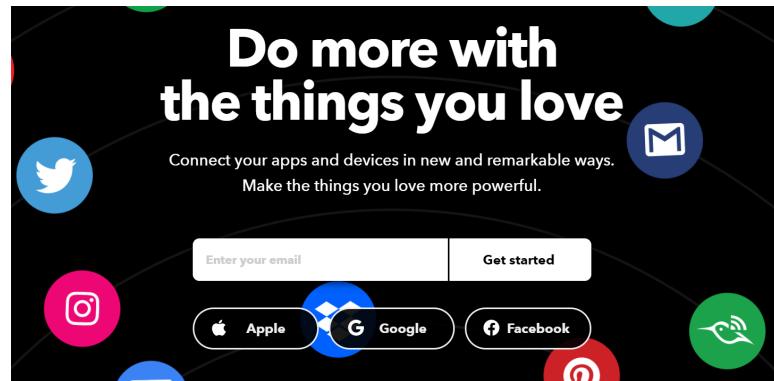
Trong bài này, chúng ta sẽ sử dụng một dịch vụ mới, được thiết kế chuyên dụng hơn cho các ứng dụng mang tính chất cảnh báo, chẳng hạn như khi nhiệt độ quá một ngưỡng cho phép chẳng hạn. Dịch vụ này có tên là IFTTT, viết tắt của thuật ngữ **If This Then That**. Dịch nôm na sang tiếng Việt, có nghĩa là nếu một điều kiện xảy ra, hệ thống sẽ thực thi một tác vụ nào đó. Nói một cách khác, nó cũng giống như bạn đang xây dựng một luật vận hành cho hệ thống, và nó sẽ tự động làm hết cho chúng ta.

Ở bài hướng dẫn này, chúng ta sẽ gửi một email cảnh báo cho người dùng khi nhiệt độ vượt một ngưỡng nhất định. Các mục tiêu chính của bài hướng dẫn này như sau:

- Đăng ký một tài khoản trên IFTTT
- Cấu hình dịch vụ WebHook trên IFTT
- Cấu hình email trên IFTTT
- Cấu hình nội dung email gửi đi
- Kiểm tra tính năng IFTTT đã cấu hình

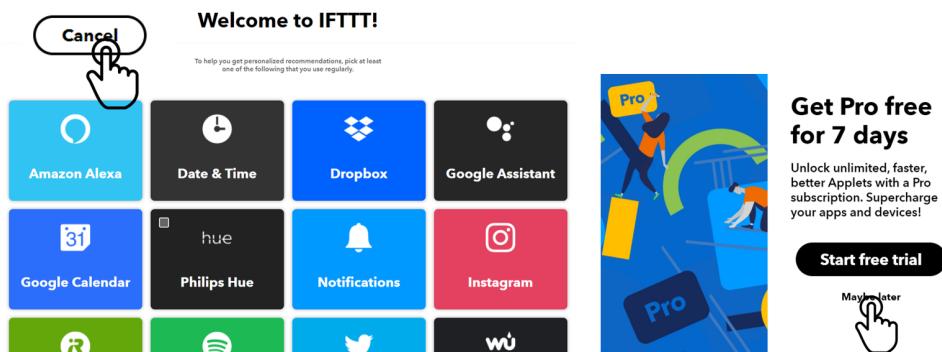
2 Đăng ký tài khoản trên IFTTT

Để đăng ký tài khoản, bạn cần vào trang chủ của tại <https://ifttt.com/>, giao diện sau đây sẽ hiện ra:



Hình 3.2: Trang chủ IFTTT

Sau khi sử dụng tài khoản Google để đăng nhập (hoặc một tài khoản khác), một số giao diện quảng cáo về dịch vụ IFTTT có thể hiện ra như hình bên dưới. Bạn chỉ đơn giản là bỏ qua, bằng cách chọn vào **Cancel** hoặc **Maybe Later**, như minh họa ở hình bên dưới:



Hình 3.3: Chọn bỏ qua một số màn hình quảng cáo

Cuối cùng, chúng ta sẽ tới được giao diện chính của IFTTT để bắt đầu làm việc, như sau:

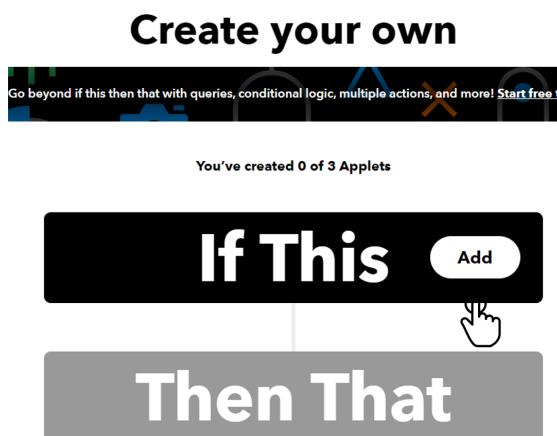


Hình 3.4: Màn hình chính của IFTTT

Bằng cách nhấn vào chức năng **Create**, chúng ta bắt đầu cấu hình một dịch vụ IFTTT (gọi là một Applet). Có 2 bước cấu hình quan trọng cho tính năng gửi email cảnh báo ở bài này, bao gồm cấu hình WebHook và Email, sẽ được trình bày ở các phần tiếp theo.

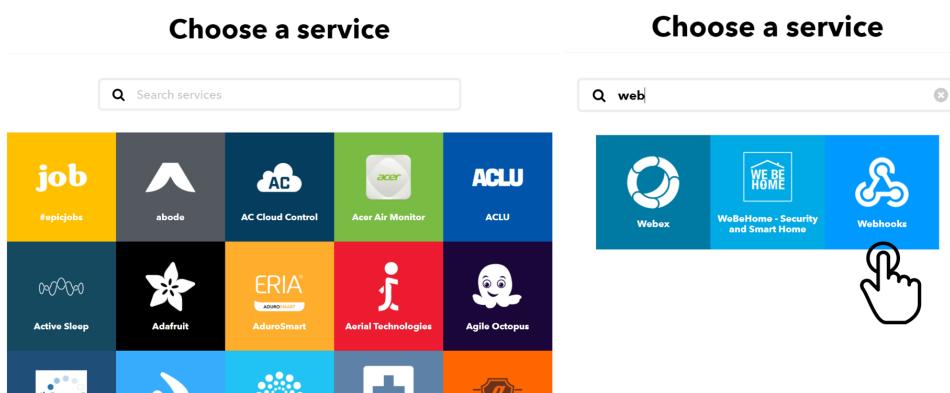
3 Cấu hình Webhook

Về mặt cơ bản, Webhook là một tính năng cho phép website tự động thông báo và gửi dữ liệu thời gian thực đến các hệ thống khi có một sự kiện nào đó phát sinh trên website (ví dụ như khách hàng đăng ký, điền form, mua hàng, hay gửi email) Webhook sẽ giúp hệ thống của bạn chủ động hơn trong việc vận hành cũng như trao đổi thông tin. Webhook là công nghệ sẽ được sử dụng trong phần **This** của IFTTT.



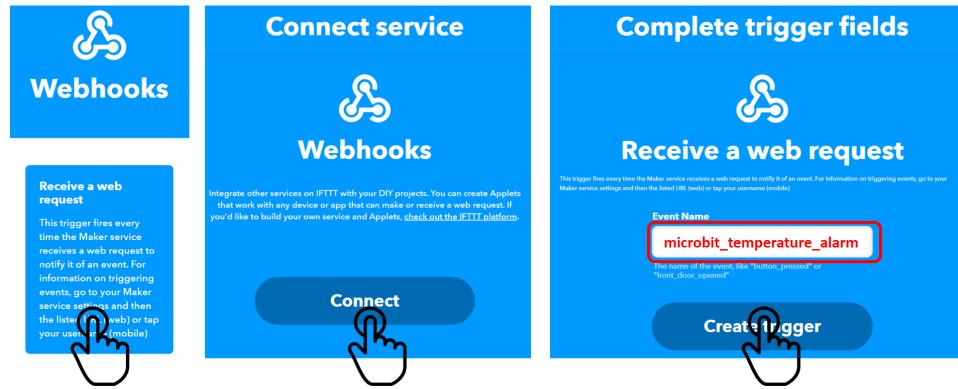
Hình 3.5: Tạo Webhook bằng cách nhấn vào Add

Sau khi nhấn vào nút **Add**, giao diện sau đây sẽ hiện ra:



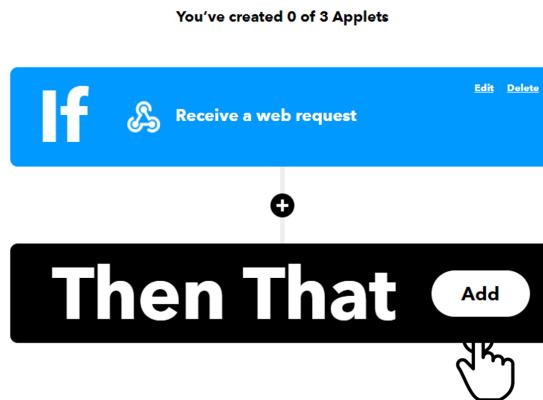
Hình 3.6: Chọn dịch vụ Webhook cho phần This

Với rất nhiều dịch vụ được hỗ trợ, bạn đọc hãy lọc lại nó bằng từ khóa **webhook**, và chọn chính xác dịch vụ này, như minh họa ở hình bên trên.



Hình 3.7: Các bước cấu hình sự kiện cho Webhook

Ba bước tiếp theo được trình bày như hình bên trên. Riêng ở màn hình thứ 3, bạn cần đặt tên cho sự kiện Webhook trước khi nhấn vào **Create Trigger**. Ở đây chúng tôi đặt là **microbit_temperature_alarm**, với ý nghĩa là sự kiện cảnh báo quá nhiệt. Sau khi tạo sự kiện thành công, màn hình sau đây sẽ xuất hiện.

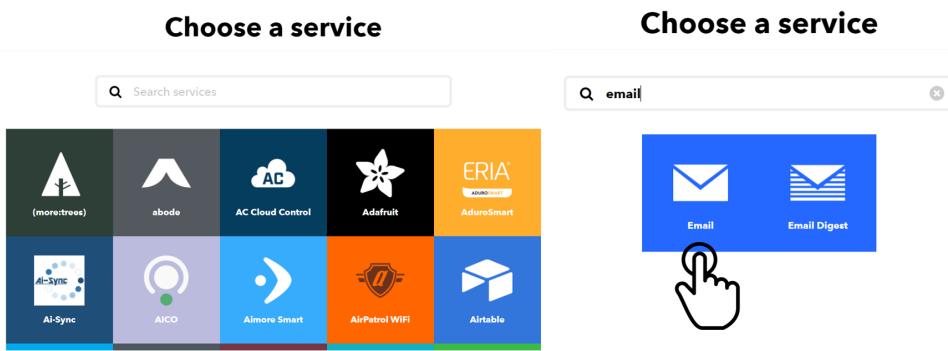


Hình 3.8: Tạo sự kiện Webhook thành công

Trong trường hợp muốn thay đổi thông tin của Webhook, bạn có thể nhấn vào **Edit**. Bây giờ, nút **Add** đã xuất hiện ở phần thứ 2 của IFTTT. Trong phần thứ 2 này, chúng ta sẽ cấu hình cho việc gửi email. Chi tiết của quá trình này được trình bày ở phần tiếp theo.

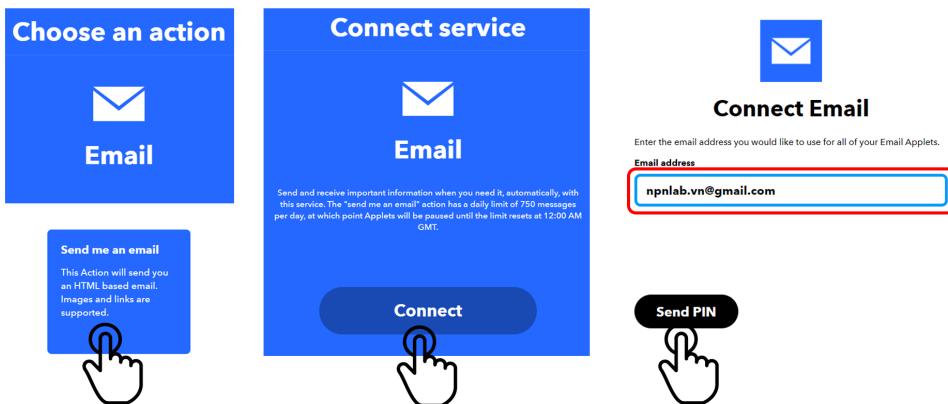
4 Cấu hình gửi email

Đây là dịch vụ thứ 2 của IFTTT. Sau khi tạo thành công Webhook ở bước trước, nhấn vào nút **Add**, giao diện về cái dịch vụ của IFTT sẽ hiện ra, như sau:



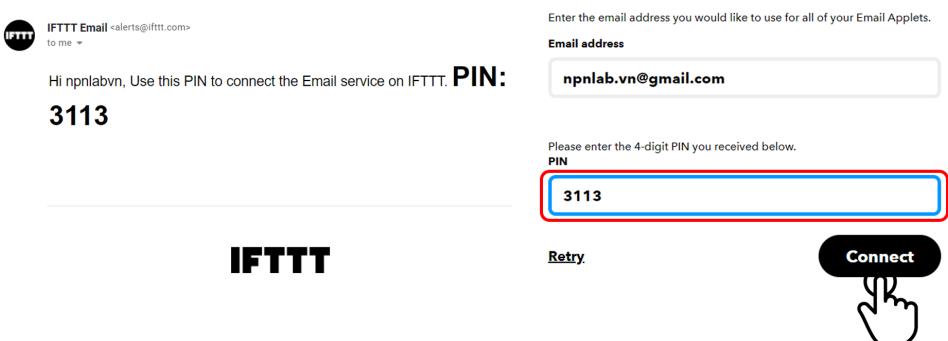
Hình 3.9: Tìm và chọn dịch vụ Email

Lần này, chúng ta sẽ tìm kiếm dịch vụ email cho phần thứ 2 của IFTTT. Các bước tiếp theo cũng khá tương tự với việc cấu hình Webhook. Chúng ta cũng sẽ có 3 bước cơ bản cho việc cấu hình email, như minh họa ở hình bên dưới:



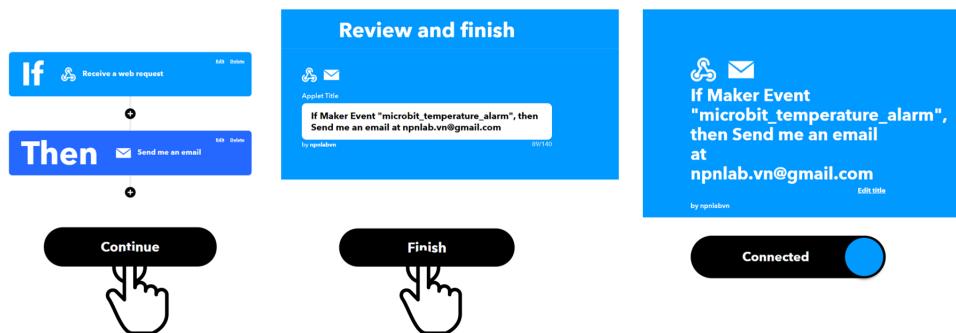
Hình 3.10: Tìm và chọn dịch vụ email

Ở bước cuối cùng, chúng ta sẽ nhập vào địa chỉ email để nhận thông báo từ IFTTT, sau đó nhấn vào nút **Send PIN**. Một mật mã 4 số sẽ được gửi tới địa chỉ mail khai báo, và sẽ được dùng để hoàn tất quá trình đăng ký email, như minh họa ở hình bên dưới:



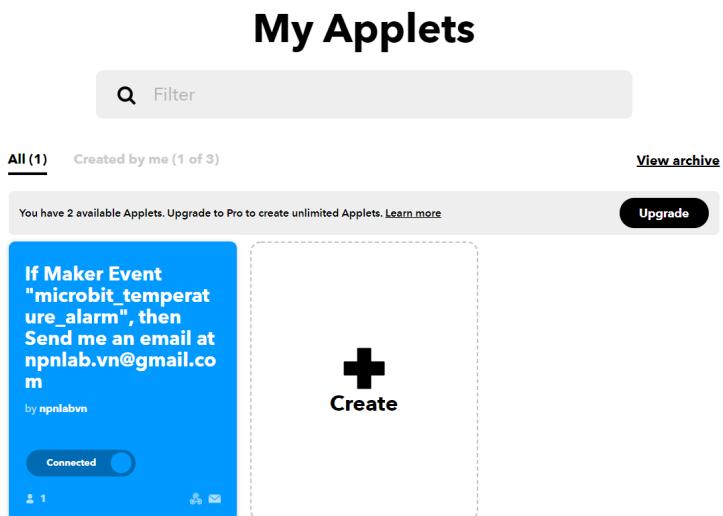
Hình 3.11: Nhập mã PIN và hoàn tất quá trình đăng kí email

Các bước cuối cùng để xem lại việc cài đặt, cũng như kết quả của hệ thống, được trình bày như hình bên dưới:



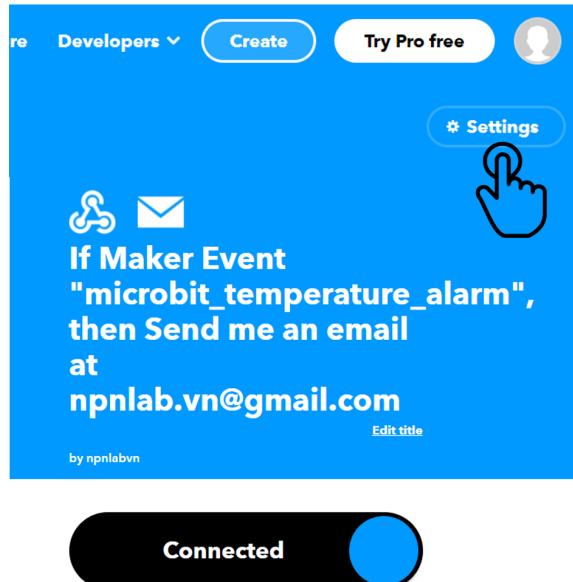
Hình 3.12: Xem lại và xác nhận kết quả trên IFTTT

Đến bước này, bạn có thể tắt trình duyệt và vào lại trang chủ IFTTT tại đường dẫn <https://ifttt.com/>. Sở dĩ chúng ta phải làm bước này, là vì những giao diện ở trên chỉ xuất hiện lần đầu tiên khi bạn chưa có tài khoản trên IFTTT. Sau khi đã có tài khoản và cấu hình các dịch vụ trên (gọi là 1 Applet), chúng ta sẽ thấy giao diện sau đây:



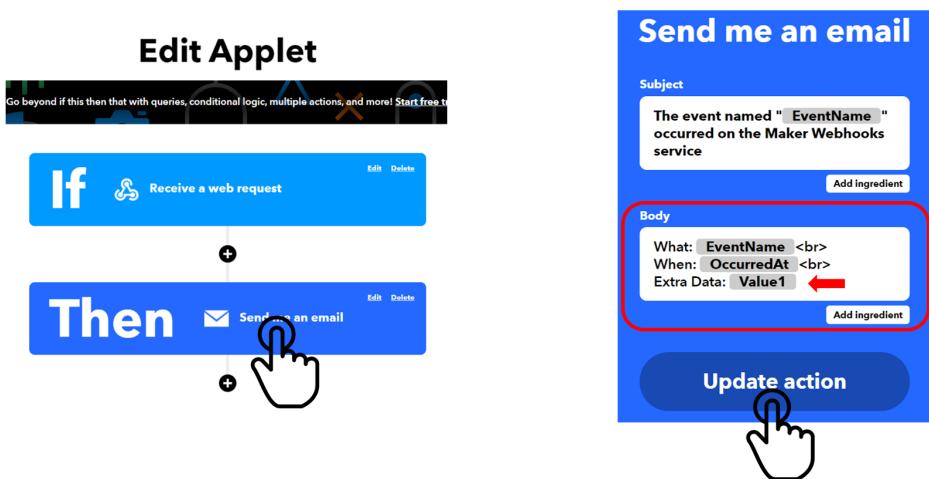
Hình 3.13: Trang chủ IFTTT sau khi đã có tài khoản

Từ thời điểm này, nếu muốn tạo mới 1 Applet, bạn sẽ nhấn vào nút **Create** ở bên phải. Trong trường hợp muốn cập nhật, hoặc thậm chí là xóa Applet cũ, bạn nhấn trực tiếp vào nó để đến giao diện sau đây:



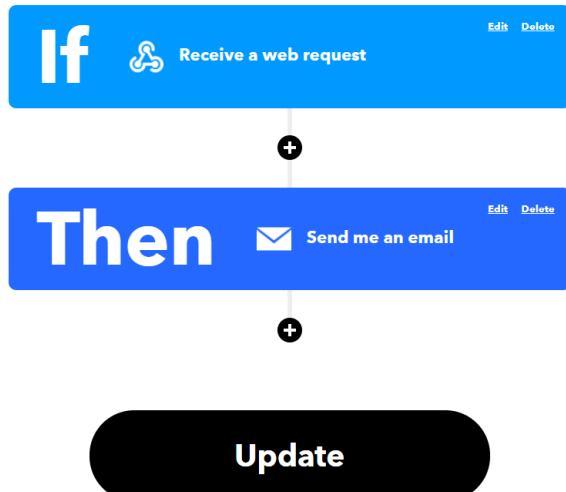
Hình 3.14: Trang cấu hình cho một Applet

Chúng ta sẽ tiếp tục cấu hình cho dịch vụ gửi email của hệ thống. Bằng cách nhấn vào **Setting**, chúng ta sẽ quy định nội dung được gửi trong email là gì. Các bước tiếp theo được trình bày như hình bên dưới:



Hình 3.15: Cấu hình nội dung gửi qua email

Ở màn hình bên phải, bạn đọc cần lưu ý, ở đây chúng ta chỉ gửi 1 giá trị của nhiệt độ đến người dùng mà thôi. Để gửi nhiều thông tin hơn, mức độ phức tạp trong các bước tiếp theo sẽ cao. Do đó, bạn đọc hãy xóa các thông tin mặc định và giữ lại 1 thông tin, **Value1** mà thôi. Về tiêu đề của email (phần **Subject**), bạn đọc có thể giữ mặt định mà không cần phải thay đổi gì cả. Chúng ta nhấn vào **Update action** để kết thúc việc cài đặt nội dung cho email. Giao diện sẽ quay lại màn hình như sau:

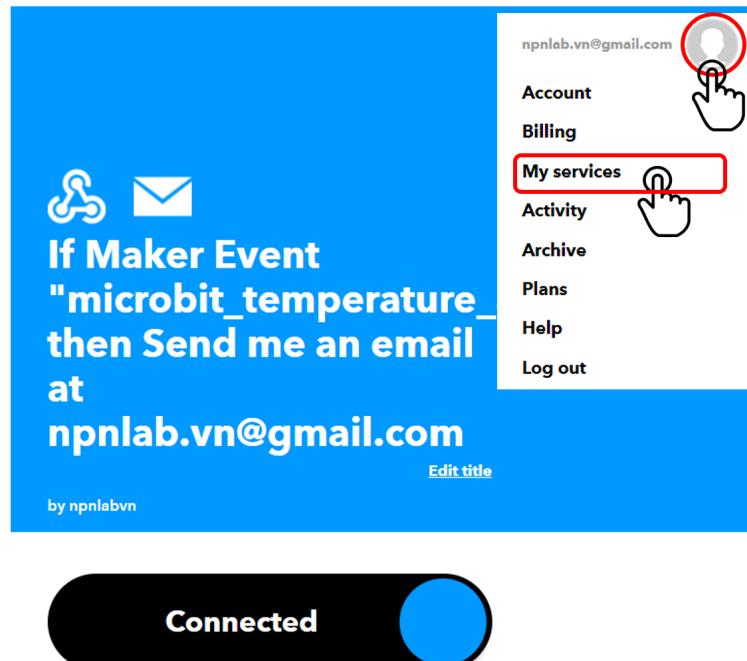


Hình 3.16: Lưu những thay đổi bằng cách nhấn vào Update

Chúng ta cần nhấn thêm nút **Update** để lưu lại những thay đổi đã chỉnh sửa trong phần nội dung của email. Giao diện sẽ trở lại màn hình trang chủ của IFTTT.

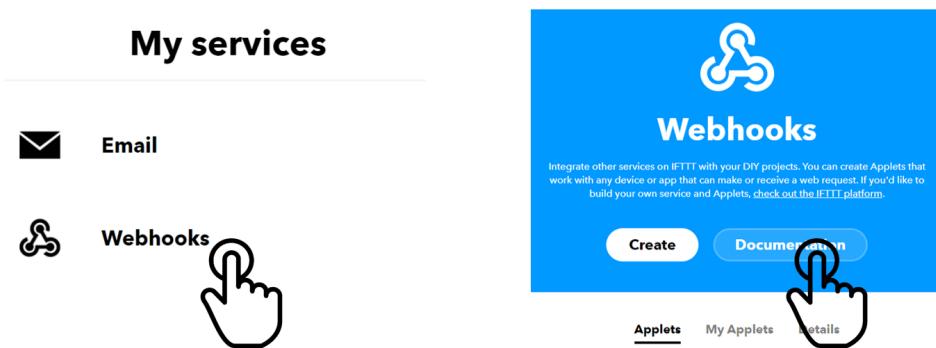
5 Kiểm tra hệ thống

Ở bước cuối cùng của bài này, chúng ta sẽ kiểm tra dịch vụ IFTTT, trước khi tích hợp nó vào server ThingSpeak ở bài sau. Từ màn hình trang chủ ở bước trước (hoặc vào lại trang chủ tại địa chỉ <https://ifttt.com/>), chúng ta chọn vào biểu tượng tài khoản, sau đó chọn tiếp **My services**, như minh họa ở hình bên dưới.



Hình 3.17: Kiểm tra dịch vụ bằng cách vào Services

Giao diện sau đây sẽ hiện ra, bạn chọn và **Webhooks** rồi chọn tiếp **Documentation**, như minh họa bên dưới.



Hình 3.18: Các bước để đến dịch vụ kiểm tra

Màn hình để chúng ta kiểm tra dịch vụ sẽ hiện ra như bên dưới:

Your key is: **dTR3mCg_CAQ6J7fOTNAj4g**

To trigger an Event

Make a POST or GET web request to:

`https://maker.ifttt.com/trigger/microbit_temperature_alarm/with/key/dTR3mCg_CAQ6J7fOTNAj4g`

With an optional JSON body of:

`{ "value1" : 45, "value2" : "_____", "value3" : "_____"}`

The data is completely optional, and you can also pass `value1`, `value2`, and `value3` as query parameters or form variables. This content will be passed on to the action in your Applet.

You can also try it with `curl` from a command line.

`curl -X POST -H "Content-Type: application/json" -d '{"value1":"45"}'
https://maker.ifttt.com/trigger/microbit_temperature_alarm/with/key/dTR3mCg_CAQ6J7fOTNAj4g`

Please read our [FAQ](#) on using Webhooks for more info.

Test It

Hình 3.19: Giao diện để kiểm tra dịch vụ

Đầu tiên, bạn cần thay đổi sự kiện event thành **microbit_temperature_alarm**, là tên sự kiện mà cũng ta đã đặt ở phần **This** của IFTTT. Tiếp theo, nhập một giá trị cho phần **value1**. Cuối cùng, nhấp vào nút **Test it**. Một email sẽ được gửi đến hộp mail đăng ký trong phần **That** của IFTTT, như sau:



Hình 3.20: Email cảnh báo được gửi đến hộp mail

Email này sẽ được gửi gần như ngay lập tức khi chúng ta nhấn vào nút **Test it**. Nội dung của email bao gồm tên sự kiện, thời gian xảy ra và dữ liệu khi xảy ra sự kiện. Nhờ dịch vụ này, cảnh báo mỗi khi quá nhiệt độ sẽ gửi gần như tức thì tới người quản lý để có hướng xử lý.

Trước khi kết thúc bài này, bạn cần sao chép lại toàn bộ dòng chứa sự kiện **microbit_temperature_alarm** cũng như mục **Your key is**. Hai thông tin này sẽ được dùng để cấu hình trên server ThingSpeak ở bài sau.

6 Câu hỏi ôn tập

1. Phát biểu nào là đúng về IFTTT?
 - A. Một dịch vụ trên nền tảng mạng kết nối vạn vật
 - B. Có thể xây dựng ứng dụng gửi tin nhắn cảnh báo
 - C. Webhook và Email là 2 trong số các dịch vụ của IFTTT
 - D. Tất cả đều đúng
2. Trong dịch vụ gửi tin nhắn cảnh báo khi quá nhiệt, dịch vụ nào được sử dụng cho phần cảnh báo?
 - A. IFTTT
 - B. Webhook
 - C. Email
 - D. Tất cả các dịch vụ trên
3. Trong dịch vụ gửi tin nhắn cảnh báo khi quá nhiệt, dịch vụ nào được sử dụng cho phần gửi email?
 - A. IFTTT
 - B. Webhook
 - C. Email
 - D. Tất cả các dịch vụ trên
4. Các tài khoản nào sau đây có thể được dùng để đăng nhập vào IFTTT?
 - A. Google
 - B. Facebook
 - C. Apple
 - D. Tất cả các tài khoản trên
5. Một quy trình IFTTT còn được gọi là gì?
 - A. IFTTT Service
 - B. Webhook
 - C. Applet
 - D. Tất cả đều đúng
6. Nội dung cảnh báo nào sẽ được gửi trong email?
 - A. Sự kiện cảnh báo
 - B. Thời gian xảy ra
 - C. Giá trị cảm biến
 - D. Tất cả các thông tin trên
7. Các thông tin quan trọng cần lưu lại cho việc cấu hình trên ThingSpeak?
 - A. Your key
 - B. Đường dẫn chứa sự kiện
 - C. Cả 2 thông tin trên
 - D. Không cần lưu thông tin nào

Đáp án

1. D 2. B 3. C 4. D 5. C 6. D 7. C



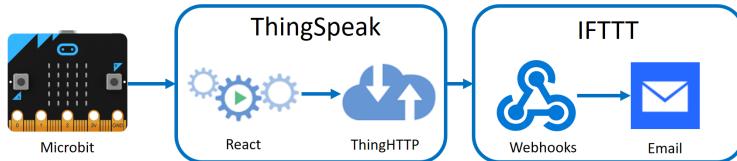
CHƯƠNG 4

ThingHTTP và React trên ThingSpeak



1 Giới thiệu

Trong bài hướng dẫn này, chúng ta sẽ liên kết việc gửi dữ liệu lên ThingSpeak từ mạch Microbit để kích hoạt dịch vụ gửi email mỗi khi quá nhiệt. Toàn bộ quy trình này được minh họa như hình bên dưới, với 5 khâu chức năng như sau:



Hình 4.1: Quy trình gửi dữ liệu và cảnh báo email

Để tạo ra 1 sự kiện, mỗi khi quá nhiệt (chẳng hạn như nhiệt độ lớn hơn $30^{\circ}C$), một ứng dụng có sẵn trên ThingSpeak sẽ phải thường xuyên giám sát giá trị của nhiệt độ gửi lên từ mạch Microbit. Ứng dụng này có tên là React, một cơ chế giám sát dữ liệu gửi lên ThingSpeak hoàn toàn tự động. Tiếp theo đó, một ứng dụng khác, gọi là ThingHTTP sẽ được kích hoạt để liên kết với dịch vụ Webhooks bên IFTTT. Từ đây, email cảnh báo sẽ được gửi đến người dùng.

Mục tiêu chính của bài này là tạo ra 2 ứng dụng mới trên ThingSpeak để hoàn thiện quy trình 5 bước như trình bày ở hình trên. Trước khi bắt đầu, bạn cần vào lại phần API Key để lấy ra 2 thông tin quan trọng, như minh họa ở hình bên dưới:

The screenshot shows the ThingSpeak Channel settings page for a channel named "IoT - Microbit - Raspberry PI".
1. Channel ID: 976324
Author: mwia0000017409006
Access: Public
Tabs: Private View, Public View, Channel Settings, Sharing, API Keys, Data Import / Export
API Keys section:

- Write API Key: Key ZDC919EE3WYRYGSE, Button: Generate New Write API Key
- Read API Keys: Key JBUNWD2MI0063BW, Note: (empty)
- API Requests: GET https://api.thingspeak.com/update?api_key=ZDC919EE3WYRYGSE&field1=

Hình 4.2: Thông tin về kênh dữ liệu trên ThingSpeak

Đầu tiên là mã kênh, ở mục **Channel ID**. Để có được mã kênh này, bạn phải đảm bảo rằng kênh đã được chia sẻ công cộng. Chi tiết về quy trình này đã được trình bày ở bài đầu tiên của giáo trình. Tiếp theo, là đường link dùng để kiểm tra việc gửi dữ liệu trên server, nằm ở mục **Write a Channel Feed**. Chúng ta sẽ sao chép lại đường dẫn này (bỏ từ khóa GET), và sửa thông tin cho Field 1 (giá trị nhiệt độ) thành 50, chẳng hạn như sau:

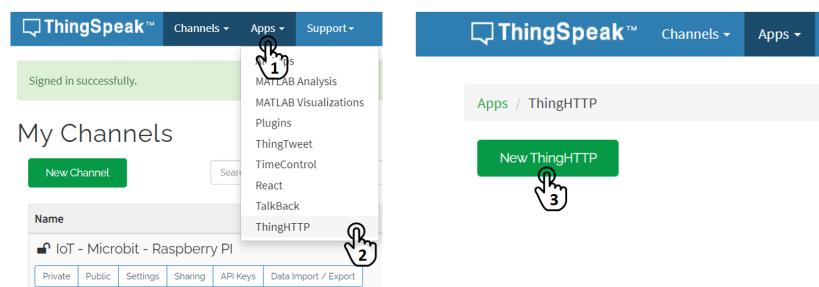
https://api.thingspeak.com/update?api_key=ZDC919EE3WYRYGSE&field1=50

Các mục tiêu của bài hướng dẫn này như sau:

- Tạo và cấu hình ứng dụng ThingHTTP
- Tạo và cấu hình ứng dụng React
- Kiểm tra toàn bộ quy trình gửi dữ liệu và cảnh báo

2 Tạo ứng dụng ThingHTTP

Bước 1: Sau khi vào lại trang ThingSpeak, chọn vào mục **Apps** và chọn vào **ThingHTTP**.



Hình 4.3: Tạo ứng dụng ThingHTTP trên ThingSpeak

Bạn chọn tiếp vào **New ThingHTTP**, để đến giao diện ở bước 2.

Bước 2: Điền các thông tin cấu hình cho ThingHTTP trong giao diện sau.

The form has seven numbered fields:
1. Name: Temperature Alarm
2. API Key: 4FUJE1CBI3FN556S
3. URL: https://maker.ifttt.com/trigger/microbit_temperature_alarm/with/key/dTR3mCg_CAQ6J7fOTNAj4g
4. Method: POST
5. Content Type: application/json
6. HTTP Version: 1.1
7. Body: {"value1": "%channel_976324_field_1%"}
A green 'Save ThingHTTP' button is at the bottom.

Hình 4.4: Điền thông tin cấu hình cho ThingHTTP

Thông tin cho từng mục được mô tả như bên dưới:

- Name: Tên của ứng dụng ThingHTTP, bạn đọc có thể chọn một tên tùy ý, không nên dùng tên tiếng Việt có dấu.
- URL: Đường liên kết với Webhooks ở bài trước, đây là thông tin quan trọng nhất đối với ThingHTTP
- Method: Phương thức liên kết giữa ThingHTTP và Webhooks, ở đây là POST
- Content Type: Định dạng thông tin trao đổi, bạn đọc cần ghi chính xác là **application/json**
- HTTP Version: Phiên bản của giao thức, ở đây bạn chọn là 1.1
- Body: Định dạng của chuỗi json cho bước 4. Bạn phải gõ chính xác thông tin là `{"value1": "%channel_xxxxxx_field_1%"}, với xxxxxx là mã kênh của bạn, có 6 kí tự số.`

Trong các thông tin trên, **sai sót có thể xảy ra ở trường Body**. Bạn đọc có thể sử dụng các trình soạn thảo đơn giản như Notepad để hoàn thiện thông tin này, sau đó chép vào giao diện cấu hình trên ThingSpeak. Từng dấu nháy kép, gạch dưới và phần trăm đều phải chính xác thì ThingHTTP mới có thể giao tiếp được với IFTTT Webhooks. Ở đây, chúng ta đang sử dụng dữ liệu từ Field 1, trường nhiệt độ trên ThingSpeak để gửi qua IFTTT.

Cuối cùng, nhấn vào **Save ThingHTTP** và trở lại giao diện ThingHTTP, chúng ta sẽ có 1 ứng dụng được tạo ra, như sau:

The screenshot shows the ThingSpeak 'Apps' section. A green button labeled 'New ThingHTTP' is visible. Below it is a table with one row, showing the 'Name' column with 'Temperature Alarm' and the 'Created' column with '2021-07-11'. At the bottom of the table are two buttons: 'View' and 'Edit'.

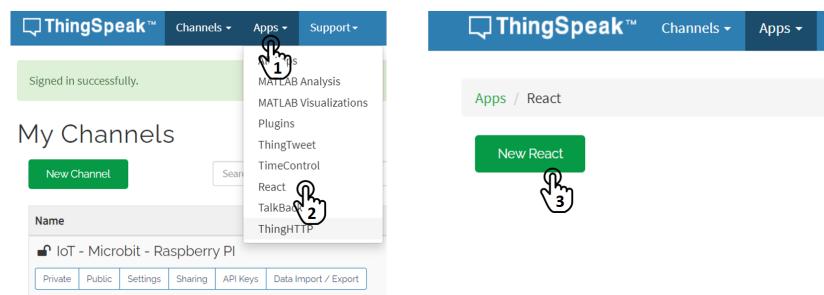
Name	Created
Temperature Alarm	2021-07-11

Hình 4.5: Một ứng dụng ThingHTTP đã được tạo

Trong trường hợp muốn xóa 1 ứng dụng ThingHTTP, bạn chọn vào **Edit**. Một giao diện sẽ hiện ra, bạn chọn tiếp **Delete ThingHTTP** nằm ở cuối trang web.

3 Tạo ứng dụng React

Bước 1: Khá tương tự với việc tạo một ThingHTTP, chúng ta cũng chọn vào **Apps** và chọn tiếp vào **React**, như minh họa sau:



Hình 4.6: Tạo ứng dụng React trên ThingSpeak

Giao diện mới hiện ra, chọn tiếp vào **New React**, để đến bước 2.

Bước 2: Điền các thông tin cấu hình cho React, như hướng dẫn ở hình sau đây:

React Name	Alarm 1
Condition Type	Numeric 2
Test Frequency	On Data Insertion
Condition	If channel IoT - Microbit - Raspberry Pi (976324)
field	1 (Nhiệt Độ) 3
	is greater than 4
	30 5
Action	ThingHTTP
	then perform ThingHTTP Temperature Alarm 6
Options	<input type="radio"/> Run action only the first time the condition is met <input checked="" type="radio"/> Run action each time condition is met 7
Save React 8	

Hình 4.7: Điền thông tin cấu hình cho React

Bạn đọc cần đảm bảo mình điền đúng thông tin như hướng dẫn ở trên. Một số thông tin quan trọng được tóm tắt lại như sau:

- React Name: Tên của ứng dụng React, không nên đặt tên tiếng Việt có dấu.
- Condition Type: Phải chọn sang dạng số (Numeric) để có thể cấu hình khi nhiệt độ lớn hơn 1 giá trị nào đó.

- Field: Các thông tin trong mục này có ý nghĩa là Filed 1 (nhiệt độ) lớn hơn 30.
- Action: Khi điều kiện xảy ra cảnh báo, nó sẽ liên kết với ThingHTTP. Trong trường hợp bạn đang có nhiều ứng dụng ThingHTTP, bạn phải chọn cho đúng.
- Options: Cách thức kích hoạt React, bạn chọn vào **Run action each time condition is met**. Điều này có ý nghĩa là cứ mỗi khi mạch Microbit gửi dữ liệu lên, và nó lớn hơn 30 thì sẽ gửi cảnh báo. Lựa chọn đầu tiên có ý nghĩa là cảnh báo chỉ gửi 1 lần mà thôi, không đúng với yêu cầu ứng dụng của chúng ta.

Sau khi đã hoàn tất các bước trên, chúng ta nhấn vào **Save React** để kết thúc việc cấu hình. Trong giao diện về các React, chúng ta sẽ thấy như sau:

The screenshot shows a ThingSpeak interface. At the top, there's a navigation bar with 'Apps / React'. Below it is a green button labeled 'New React'. A table lists the created Reacts. The first row has columns for 'Name', 'Created', and 'Last Ran'. The first entry is 'Alarm' with a checked checkbox, created on '2021-07-11'. Below the table are two buttons: 'View' and 'Edit'.

Name	Created	Last Ran
✓ Alarm	2021-07-11	

Hình 4.8: Một ứng dụng React đã được tạo

Các thao tác với React cũng tương tự với ThingHTTP. Trong trường hợp bạn muốn chỉnh sửa hoặc thậm chí là xóa React này, hãy nhấn vào **Edit**.

4 Kiểm tra quy trình

Cho đến bước này, chúng ta đã gần như hoàn thiện hệ thống cảnh báo nhiệt độ, mỗi khi nó quá 30 độ, như cấu hình trong phần React. Tuy nhiên, trước khi sử dụng mạch Microbit gửi dữ liệu lên tự động bằng mạch Microbit, bạn hoàn toàn có thể dùng trình duyệt web (chrome chẳng hạn) để kiểm tra việc này trước. Bằng cách sử dụng đường liên kết ở phần 1 của bài hướng dẫn này, giá trị 50 sẽ được gửi lên ThingSpeak. Trình duyệt web sẽ **hiện một số lớn hơn 0**, báo hiệu việc gửi dữ liệu lên ThingSpeak là thành công. Nếu như một email được gửi tới tài khoản của bạn, tức là mạch Microbit đã có thể hoạt động và gửi dữ liệu lên server định kì.

5 Câu hỏi ôn tập

1. Dịch vụ nào dùng để kiểm tra dữ liệu gửi lên từ Microbit?
 - A. React
 - B. ThingHTTP
 - C. Webhook
 - D. Email
2. Dịch vụ nào trên ThingSpeak dùng để kích hoạt IFTTT?
 - A. React
 - B. ThingHTTP
 - C. Webhook
 - D. Email
3. Khi cấu hình ứng dụng cảnh báo quá nhiệt, trường Condition Type được cấu hình là gì?
 - A. Luận lý (boolean)
 - B. Chuỗi (string)
 - C. Số (Numeric)
 - D. Tất cả đều đúng
4. Để gửi cảnh báo mỗi khi quá nhiệt, cấu hình so sánh nào sẽ được dùng?
 - A. is less than
 - B. is equal than
 - C. is greater than
 - D. Tất cả đều sai
5. Trong việc cấu hình ứng dụng ThingHTTP, Content Type là gì?
 - A. raw
 - B. application/raw
 - C. application/json
 - D. Tất cả đều sai
6. Cơ chế giao tiếp giữa ThingHTTP và Webhook là gì?
 - A. GET
 - B. POST
 - C. UPDATE
 - D. Tất cả các thông tin trên
7. Khi dùng trình duyệt web để kiểm tra việc gửi dữ liệu lên ThingSpeak, khi gửi thành công, thông tin nào sẽ hiện ra?
 - A. Một số âm
 - B. Chữ Success
 - C. Một số dương
 - D. Tất cả đều sai

Đáp án

1. A 2. B 3. C 4. C 5. B 6. B 7. C

CHƯƠNG 5

Cảm biến quan trắc tích hợp



1 Giới thiệu

Một trong những ứng dụng mạnh mẽ mà nền tảng kết nối vạn vật mang lại, là các ứng dụng quan trắc môi trường. Nhờ khả năng kết nối vào mạng Internet, gửi dữ liệu, gửi cảnh báo mà các ứng dụng này được phát triển ngày càng nhiều, góp phần vào việc nâng tầm chất lượng cuộc sống, thúc đẩy các ứng dụng nghiên cứu lần triển khai, hướng đến các dịch vụ của thành phố thông minh trong tương lai.

Đặc điểm chung của các ứng dụng quan trắc là thu thập các thông tin từ cảm biến đặt rải rác trong môi trường quan trắc. Các thông tin này sẽ được gửi lên một điện toán đám mây, để phục vụ cho việc giám sát từ xa. Cảm biến vì vậy, có thể được xem là phần đầu vào (còn gọi là **input**), thành phần vô cùng quan trọng của một hệ thống. Do đó, việc lựa chọn cảm biến cho một ứng dụng nói chung, cần phải được xem xét tỉ mỉ.



Hình 5.1: Cảm biến Nhiệt độ và độ ẩm DHT11

Trong bài hướng dẫn này, chúng tôi sẽ trình bày một cảm biến tích hợp, là cảm biến nhiệt độ - độ ẩm không khí DHT11, như trình bày ở hình trên. Sở dĩ gọi là tích hợp, bởi nó có thể cung cấp nhiều thông tin đồng thời, ở đây là 2 thông tin về môi trường. DHT11 là cảm biến khá đơn giản cho người mới bắt đầu, nhưng các phiên bản cao cấp của nó như DHT22 hay AM2305 là những sản phẩm có thể được dùng trong các ứng dụng thực tế. Và trên hết, các cảm biến này hoàn toàn tương thích về mặt chương trình, chỉ cần đổi thiết bị phần cứng, chúng ta sẽ có những thông tin về môi trường chính xác hơn và độ bền của thiết bị tốt hơn.

Các mục tiêu hướng dẫn trong bài này như sau:

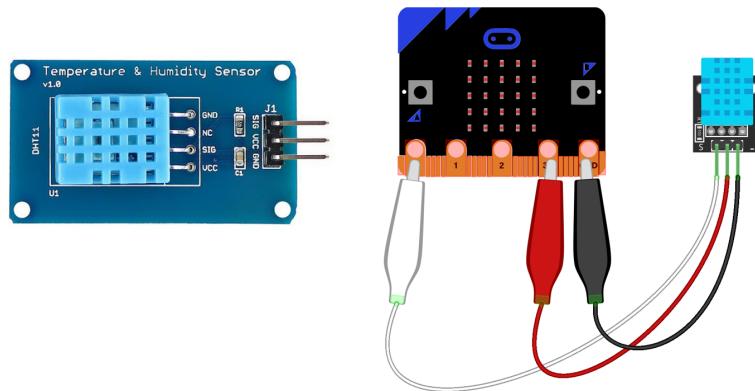
- Kết nối với cảm biến DHT11
- Lập trình lấy dữ liệu từ DHT11
- Gửi dữ liệu lên server ThingSpeak
- Tìm hiểu các cảm biến cao cấp DHT22 và AM2305

2 Kết nối với DHT11

Để kết nối với DHT11, chúng ta chỉ cần 3 chân kết nối, như sau:

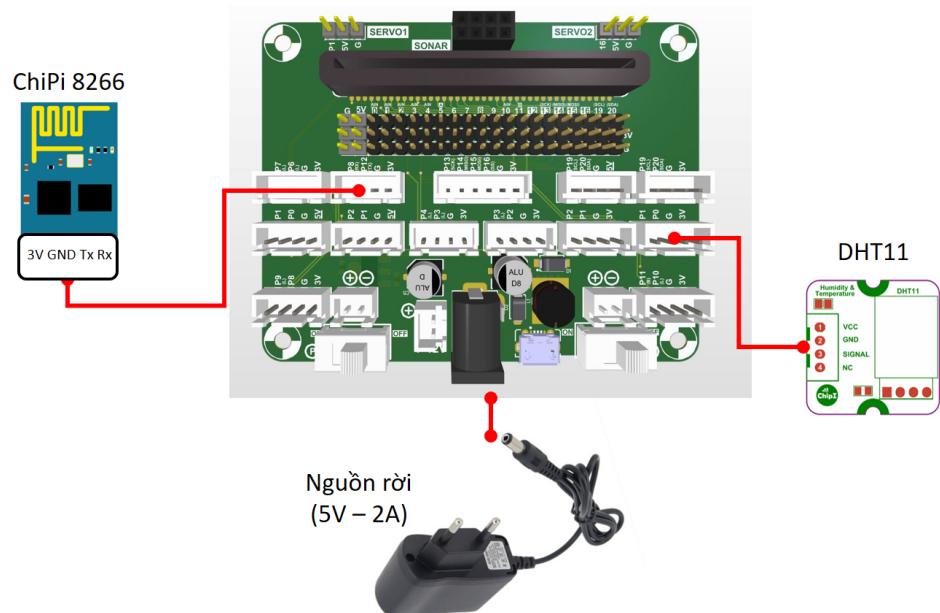
- Nguồn: Từ 3V đến 5V
- Đất: 0V
- Chân tín hiệu

Rõ ràng, với việc tương thích điện áp rộng, cảm biến DHT11 được sử dụng nhiều trên các thiết bị lập trình phần cứng chung, và mạch Microbit nói riêng. Không khó để bạn đọc có thể kiểm các dự án liên quan giữa Microbit và cảm biến DHT11 trên mạng, kèm theo các hướng dẫn kết nối, như minh họa ở hình bên dưới:



Hình 5.2: Kết nối với DHT11 bằng 3 chân tín hiệu

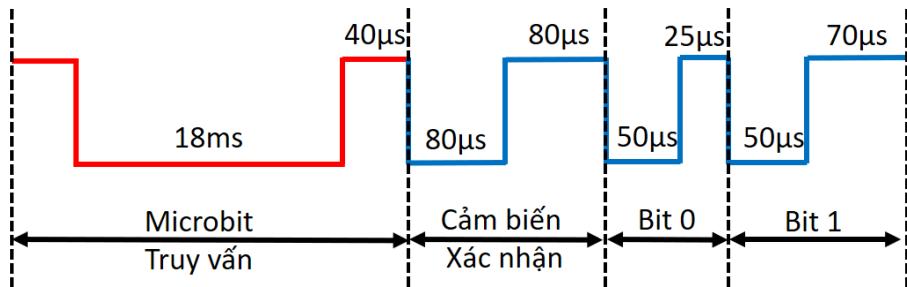
Trong bài hướng dẫn này, chúng tôi sử dụng thiết bị DHT11 đã được chuẩn hóa lại kết nối cho tương thích với mạch mở rộng ChiPi Base Shield. Chúng ta chỉ đơn giản là nối 1 dây có 4 chân vào mạch mở rộng. Khi kết nối vào mạch, bạn cần lưu ý chân tín hiệu của DHT11 đang kết nối với chân nào của Microbit. Thông tin này sẽ được dùng cho việc lập trình sắp tới. Như kết nối ở hình dưới đây, cảm biến đang được nối với chân P0 của mạch Microbit.



Hình 5.3: Kết nối với ChiPi DHT11 với P0 của Microbit

3 Nguyên lý hoạt động của DHT11

DHT11 là một dạng cảm biến tích hợp đơn giản và khá phổ biến với các ứng dụng dùng vi điều khiển nói chung, và Microbit nói riêng. Để có thể đọc dữ liệu từ nó, mạch Microbit phải gửi tín hiệu **truy vấn** (query). Khi nhận được tín hiệu này, cảm biến mới bắt đầu tính toán và gửi dữ liệu trả về cho mạch Microbit. Cũng chính vì lý do này, nếu như mạch Microbit quên gửi tín hiệu truy vấn, dữ liệu mà nó nhận được là dữ liệu cũ và không hợp lệ.



Hình 5.4: Nguyên lý giao tiếp với DHT11

Mặc dù là cảm biến tích hợp đơn giản bậc nhất, việc lấy dữ liệu từ nó phức tạp hơn chúng ta nghĩ. Để bắt đầu việc giao tiếp, tín hiệu truy vấn từ Microbit gửi tới cảm biến là 1 xung mức thấp, kéo dài trong 18ms. Sau đó, Microbit sẽ nâng tín hiệu lên mức cao và chờ phản hồi từ DHT11. Sau khoảng 40µs, cảm biến sẽ xác nhận việc nhận lệnh bằng xung 80µs ở mức thấp, theo sau là xung 80µs nhưng ở mức cao. Cuối cùng, 40 bit dữ liệu sẽ được gửi lên Microbit, với bit 0 có hình xung là 50µs mức thấp và 25µs ở mức cao, còn bit 1 có hình xung là 50µs mức thấp và 70µs ở mức cao. Bạn đọc cần lưu ý về đơn vị thời gian cho quá trình giao tiếp này, được minh họa ở hình bên trên.

Cuối cùng, sau khi giải mã ra 40 bit dữ liệu, mạch Microbit phải tiếp tục xử lý để lấy ra thông tin cần thiết cho ứng dụng, với 16 bit đầu là thông tin cho độ ẩm, 16 bit tiếp theo là thông tin về nhiệt độ, và 8 bit cuối cùng là thông tin kiểm tra lỗi ($40 = 16 + 16 + 8$).

Vì tính chất phức tạp của cảm biến tín hiệu, chủ yếu là xử lý tín hiệu xung, đa số các ứng dụng trên Microbit sẽ sử dụng các thư viện lập trình hỗ trợ. Nhờ các thư viện này, việc lập trình sẽ đơn giản hơn rất nhiều và thuận tiện cho bạn đọc để tập trung xây dựng ứng dụng, hơn là việc can thiệp sâu vào hệ thống cho các tác vụ liên quan đến xử lý tín hiệu.

4 Lập trình với DHT11

Trong hướng dẫn này, các câu lệnh cần thiết cho việc lập trình lấy dữ liệu từ DHT11 được tích hợp sẵn trong nhóm lệnh NPNBitKit, như trình bày ở hình bên dưới:



Hình 5.5: Các câu lệnh liên quan đến DHT11 trong NPNBitKit

Hãy lưu ý rằng để nhận được giá trị cập nhật từ cảm biến, chúng ta sẽ phải truy vấn nó trước bằng câu lệnh **DHT11: Đọc cảm biến tại P0**. Trong trường hợp cảm biến được kết nối với chân khác, bạn đọc cần lựa chọn lại cho đúng chân kết nối. Sau khi truy vấn, chúng ta đã có thể sử dụng giá trị từ cảm biến này với 2 khối lập trình **DHT11: nhiệt độ** và **DHT11: độ ẩm**.

Sử dụng lại chương trình đã phát triển ở bài trước, chúng ta chỉ đơn giản là đổi 2 giá trị gửi lên ThingSpeak bằng các giá trị đọc được từ cảm biến DHT11. Chương trình gợi ý cho bạn đọc như sau:



Hình 5.6: Chương trình đọc cảm biến DHT11

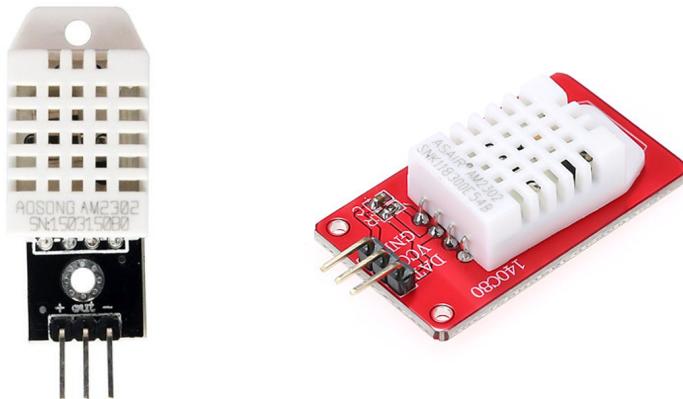
Chương trình được chia sẻ ở đường dẫn sau đây:

https://makecode.microbit.org/_KMTdYfcKMCCs

5 Các phiên bản nâng cấp của DHT11

5.1 Cảm biến DHT22

Đây là cảm biến có hình dạng rất giống với DHT11, nhưng thông thường nó sẽ có màu trắng để bạn đọc dễ phân biệt. Giá thành của nó thông thường cũng đắt hơn gấp đôi so với DHT11. Do đó, độ chính xác của nó hiển nhiên là tốt hơn DHT11, với chỉ khoảng 2% sai số khi đọc độ ẩm, so với 5% sai số của DHT11. Bên cạnh đó, nhiệt độ từ DHT22 có độ sai lệnh tối đa 0.5°C so với sai số 2°C của DHT11.

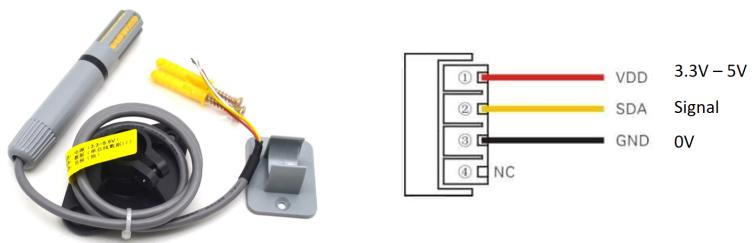


Hình 5.7: Cảm biến DHT22 trên thị trường

Do đó, đối với các ứng dụng mà việc đo nhiệt độ là thông tin quan trọng để vận hành hệ thống, chẳng hạn như máy ấp trứng hoặc các máy sấy kích thước lớn, việc sử dụng DHT22 sẽ mang lại kết quả tốt hơn nhiều so với DHT11. Với kết nối chân hoàn toàn tương thích với DHT11, việc thay thế DHT22 vào hệ thống cũ sử dụng DHT11 là vô cùng dễ dàng và không có thay đổi nào về kết nối phần cứng lẫn chương trình trên phần mềm.

5.2 Cảm biến AM2305

Cảm biến nhiệt độ - độ ẩm này có đóng gói cứng cáp hơn và có độ bền tốt hơn so với DHT22. Thông thường, thiết bị này được sử dụng cho các sản phẩm trong công nghiệp đòi hỏi độ bền cao và chịu được nhiệt lớn. Hình ảnh của sản phẩm này được trình bày ở bên dưới.



Hình 5.8: Cảm biến AM2305 với 3 chân kết nối

Khi sử dụng sản phẩm này, bạn đọc cần lưu ý số lượng chân kết nối của cảm biến. Phiên bản nâng cấp của AM2305 là AM2315, có đến 4 chân kết nối, thay vì 3 chân như AM2305. Với chân nguồn tương thích với dãy điện áp rộng, từ 3.3V đến 5V, nên rất thích hợp với mạch Microbit. Tuy nhiên, trong các ứng dụng lớn, nguồn 5V nên được sử dụng cho cảm biến để đảm bảo độ ổn định. Độ chính xác của cảm biến AM2305 là tốt hơn hẳn so với DHT11, khi chỉ sai số trong khoảng 0.1°C và 1% sai số của độ ẩm.

6 Câu hỏi ôn tập

1. Phát biểu nào sau đây là đúng về cảm biến DHT11?
 - A. Cảm biến tích hợp
 - B. Đo được nhiệt độ và độ ẩm không khí
 - C. Là một dạng thiết bị đầu vào (input)
 - D. Tất cả đều đúng
2. Trước khi truy xuất giá trị của cảm biến DHT11, chúng ta cần làm gì?
 - A. Kéo chân cảm biến lên cao
 - B. Kéo chân cảm biến xuống thấp
 - C. Truy vấn cảm biến
 - D. Tất cả đều đúng
3. Nhóm lệnh hỗ trợ cho việc lập trình với DHT11 và LCD giới thiệu trong bài là gì?
 - A. DHT11
 - B. LCD
 - C. NPNBitKit
 - D. Tất cả đều đúng
4. Số chân kết nối giữa DHT11 và mạch mở rộng Microbit là bao nhiêu?
 - A. 1
 - B. 2
 - C. 3
 - D. 4
5. DHT11 hỗ trợ kết nối với nguồn có điện áp bao nhiêu Volt?
 - A. 3.3V
 - B. 4.1V
 - C. 5V
 - D. Tất cả đều đúng
6. Cảm biến nào sau đây là tương thích với chuẩn cắm dây với DHT11?
 - A. DHT22
 - B. AM2305
 - C. AM2315
 - D. Hai cảm biến đầu tiên
7. Cảm biến nào sau đây có độ chính xác cao nhất?
 - A. DHT11
 - B. DHT22
 - C. AM2305
 - D. Không xác định được

Đáp án

1. D 2. C 3. C 4. C 5. D 6. D 7. C



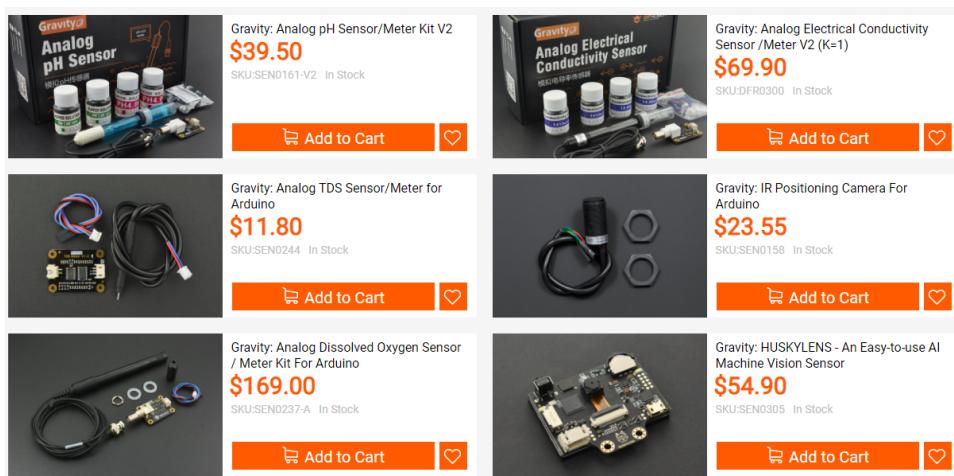
CHƯƠNG 6

Cảm biến tương tự Analog



1 Giới thiệu

Khác với cảm biến tích hợp giới thiệu ở bài trước, cảm biến tương tự, còn gọi là cảm biến Analog giới thiệu ở bài này, chỉ hỗ trợ đo một thông số duy nhất. Tuy nhiên, đầu ra của cảm biến là rất đơn giản, gọi là đầu ra tương tự, bạn đọc có thể dễ dàng đọc được dữ liệu từ cảm biến mà không cần sự hỗ trợ của các thư viện phức tạp. Việc đọc dữ liệu từ cảm biến tương tự còn được gọi là đọc ADC (Analog to Digital Converter). Và đúng như tên gọi ADC của nó, là chuyển từ tương tự sang số, giá trị mà chúng ta nhận được là một con số có giá trị từ 0 đến 1023, khi lập trình trên mạch Microbit.



Hình 6.1: Các cảm biến ADC từ DFRobot

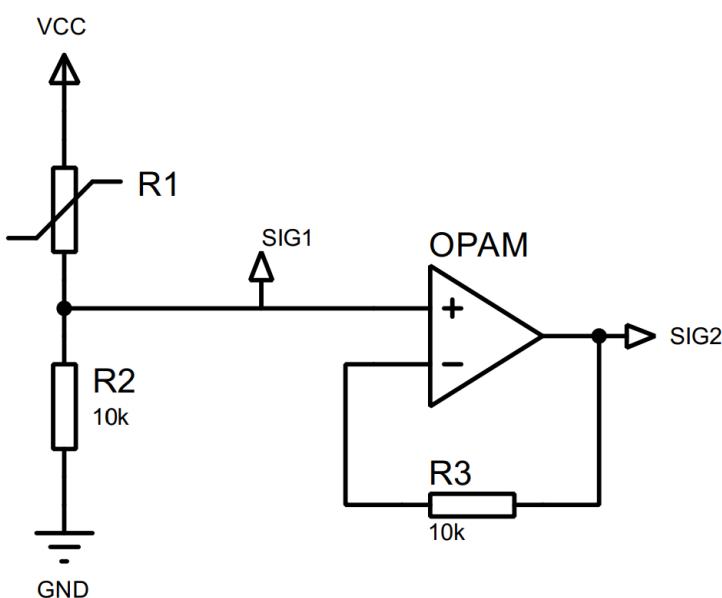
Do tính chất đơn giản của nó, các hệ thống cảm biến cho các dự án nói chung, và đặc thù cho mục đích giáo dục, đều có hỗ trợ chuẩn đầu ra là ADC. Thậm chí những cảm biến có độ phức tạp cao, vẫn có hỗ trợ chuẩn đầu ra là ADC, chẳng hạn như hệ thống cảm biến Analog Gravity từ DFRobot, với hàng trăm cảm biến hỗ trợ chuẩn đầu ra ADC. Một trong những yếu tố quan trọng của dạng cảm biến này, là bạn phải cung cấp đủ nguồn cho nó, bởi đơn giản, điện áp là thông tin quan trọng cho hoạt động của cảm biến.

Trong bài hướng dẫn này, chúng tôi sẽ giới thiệu về nguyên lý thiết kế của cảm biến ADC và minh họa trên cảm biến đo khí gas. Chúng tôi cũng sẽ giới thiệu về hệ thống cảm biến ChiPi, một hệ thống cảm biến đơn giản, với một chuẩn kết nối đồng bộ và rất phù hợp cho giáo dục ở lứa tuổi cấp 2. Các mục tiêu trong bài hướng dẫn này như sau:

- Hiểu nguyên lý thiết kế của cảm biến Analog.
- Kết nối được cảm biến khí Gas.
- Hiện thực chương trình đọc dữ liệu từ cảm biến.
- Tìm hiểu các cảm biến ADC của hệ thống ChiPi.

2 Nguyên lý thiết kế cảm biến Analog

Cảm biến Analog khi kết nối với các hệ thống vi điều khiển khá đơn giản, bởi độ phức tạp của nó đã dồn qua việc thiết kế mạch. Thêm nữa, việc lập trình để đọc dữ liệu từ cảm biến cũng vô cùng thuận tiện. Các cảm biến loại này đều dựa vào đặc điểm là khi điều kiện giám sát thay đổi, điện trở của nó sẽ thay đổi theo. Ví dụ, cảm biến đo nhiệt độ sử dụng linh kiện, mà điện trở của nó sẽ thay đổi khi nhiệt độ thay đổi. Linh kiện này còn gọi là **nhiệt điện trở**. Tương tự như vậy, cảm biến ánh sáng sẽ sử dụng **quang trở**, thiết bị có điện trở thay đổi khi cường độ ánh sáng thay đổi. Nguyên lý kết nối các cảm biến này được trình bày như hình bên dưới:



Hình 6.2: Nguyên lý kết nối cảm biến ADC

Đầu tiên, bạn hãy để ý cầu điện trở ở bên trái, gồm 2 điện trở R1 và R2, nối từ nguồn xuông đất. Ở đây, điện trở R1 tượng trưng cho thiết bị cảm biến. Giá trị của R1 sẽ thay đổi tùy thuộc vào điện kiện của môi trường, trong khi đó, R2 sẽ có giá trị cố định. Dựa vào định luật Ohm, điện áp tại chân SIG1 sẽ được tính như sau:

$$SIG1 = \frac{VCC * R2}{R1 + R2}$$

Rõ ràng, khi R1 thay đổi, SIG1 sẽ thay đổi theo. Ví dụ như cảm biến độ ẩm đất ở bài trước, khi đất khô trở kháng R1 sẽ có giá trị lớn. Do giá trị R1 đang ở mấu số, giá trị điện áp ở chân SIG1 sẽ giảm. Ngược lại, khi đất ẩm, trở kháng R1 nhỏ, làm cho điện áp tại chân SIG1 tăng cao. Hành vi của hệ thống sẽ đảo ngược khi chúng ta đổi vị trí của R1 và R2 trên mạch điện trên.

Dựa vào nguyên lý này, rất nhiều thiết bị trong hệ thống ChiPi được thiết kế như trên. Do đó, đầu ra của cảm biến cũng chỉ cần 3 chân là VCC, GND và SIG. Tuy nhiên, trong hệ thống ChiPi, một mạch khuếch đại thuật toán OPAM được tích hợp thêm ở phía bên phải để tín hiệu đầu ra SIG2 được ổn định hơn. Mạch này được gọi là mạch hồi tiếp âm, do đầu ra được nối ngược lại vào chân âm của OPAM.

Cách măc này đảm bảo tín hiệu điện áp SIG2 và SIG1 là như nhau, nhưng với khả năng cách ly của OPAM, sẽ không có dòng điện đi vào mạch Microbit. Do đó, mạch Microbit sẽ rất bền và không bị nóng trong quá trình hoạt động. Trong trường hợp không có OPAM, sẽ có 1 dòng nhỏ đi từ SIG1 vào chân Microbit. Và trong trường hợp xấu, khi SIG1 tăng cao, do nhiễu hoặc do thiết bị cảm biến bị lỗi, một dòng điện cao có thể làm hư mạch Microbit.

Với các cảm biến có đầu ra là điện áp, việc lập trình trên Microbit từ phía người sử dụng là vô cùng đơn giản, với câu lệnh **analog read**. Kết quả của phép đọc này chỉ là một con số, có tầm giá trị từ 0 đến 1023 (gọi là ADC 10 bit). Kết quả này không có đơn vị. Do đó, để chuyển nó sang đơn vị % của độ ẩm đất, hay **lux** của cường độ ánh sáng, thậm chí là **ppm** của nồng độ CO₂, chúng ta cần phải xây dựng thêm một hàm chuyển đổi nữa. Việc xây dựng hàm này sẽ tốn kém bởi chúng ta cần một thiết bị thương mại để có thể ánh xạ từ giá trị điện áp sang thông tin cần thiết. Trong phạm vi của giáo trình này, cũng như tính chất ứng dụng hiện tại còn đơn giản, chỉ cần phân biệt được có/không hoặc vượt ngưỡng, nên chúng tôi không đi vào chi tiết của hàm ánh xạ này.

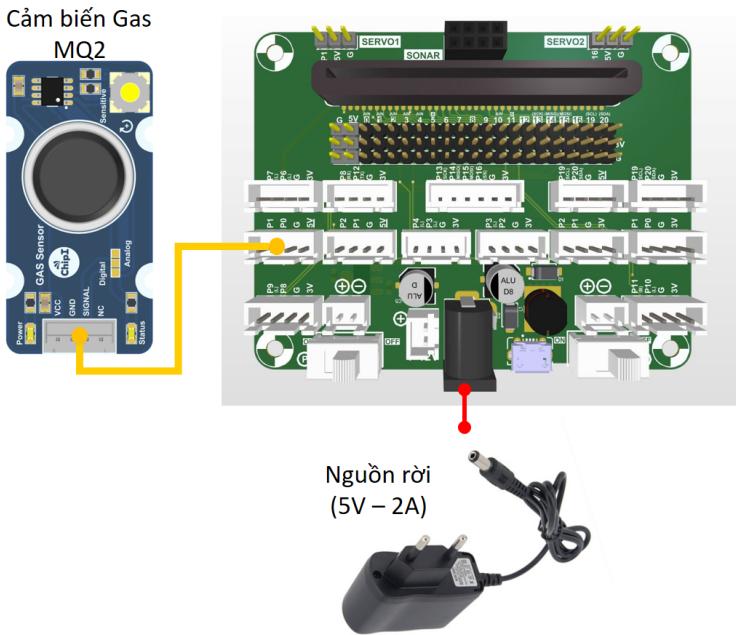
3 Cảm biến khí Gas

Cảm biến khí gas sử dụng trong bài hướng dẫn này có tên là MQ2, thuộc họ MQx, một công nghệ cảm biến khí đơn giản. Cảm biến loại này hoạt động dựa vào nguyên lý điện hóa. Chính vì vậy, nó cần một dòng điện có công suất cao, thì mới có thể phân tách được khí cần đo và trả về kết quả cho chúng ta. Cũng vì lý do này, mà MQ2 cần nguồn ngoài 5V-1A trở lên, và khi hoạt động, nó sẽ hơi ấm.

Trên thị trường, có rất nhiều cảm biến không khí được hiện thực bằng công nghệ điện hóa như MQ2, có thể tóm tắt ra một số thiết bị thông dụng như sau:

- MQ3: Alcohol, Ethanol, khói thuốc
- MQ4: Khí Methane
- MQ7: Carbon Monoxide CO
- MQ8: Khí hydro
- MQ9: Khí CO và khí độc hại

Do đó, hướng dẫn trong bài này có thể dễ dàng mở rộng ra nhiều ứng dụng khác nhau khi thay thế cảm biến MQ2 thành các cảm biến khác cùng họ MQx. Tuy nhiên, điều quan trọng là cảm biến này phải được cấp nguồn điện 5V, nên bạn đọc phải hết sức lưu ý thông tin này khi làm việc trên mạch Microbit và mạch mở rộng của nó. Trong hướng dẫn này, chúng tôi sử dụng mạch mở rộng ChiPi Based Shield V2, với sự hỗ trợ của 2 khe cắm 5V. đương nhiên, để có điện áp 5V này, một nguồn ngoài (còn gọi là adapter) sẽ được sử dụng. Lúc này, cảm biến MQ2 kết nối với mạch mở rộng thông qua một dây 4 chân, như minh họa ở hình bên dưới.



Hình 6.3: Kết nối với cảm biến khí Gas MQ2

Theo kết nối ở trên, cảm biến khí Gas đang được kết nối với chân P0 của mạch Microbit. Để nhận ra vị trí chân kết nối bạn đọc cần chú ý màu dây đang nối với chân SIGNAL của cảm biến và mạch mở rộng ChiPi Base Shield V2.

4 Đọc dữ liệu từ cảm biến khí Gas

Việc lập trình lấy dữ liệu thô từ cảm biến ADC thực sự đơn giản, với một câu lệnh **analog read pin** trong mục **Pins** được sử dụng, chương trình gọi ý như sau:

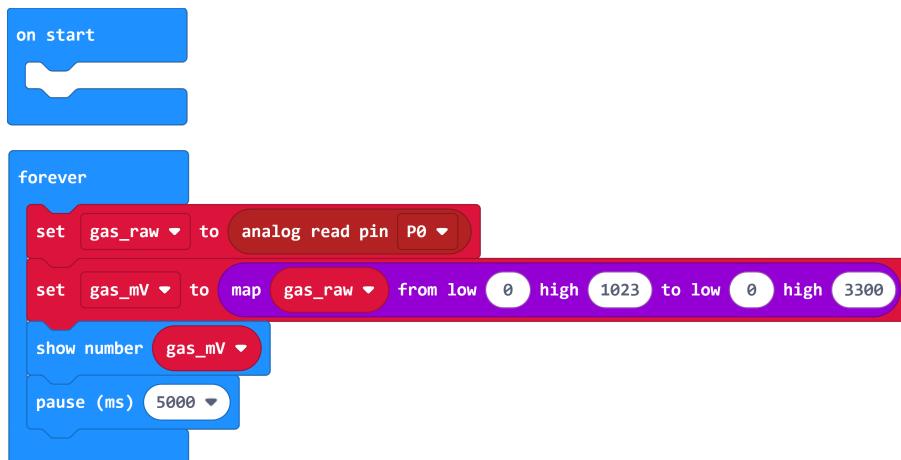


Hình 6.4: Đọc dữ liệu thô từ cảm biến ADC

Với các ứng dụng đơn giản, đoạn chương trình trên sẽ dùng để thống kê giá trị nhận được từ cảm biến. Bạn đọc cần chọn một ngưỡng so sánh để đưa ra các cảnh báo khi giá trị của cảm biến vượt ngưỡng cho phép. Chẳng hạn như với cảm biến khí Gas, một ngưỡng cần được chọn lựa kĩ càng, để đưa ra cảnh báo khi nồng độ khí Gas là nhiều hơn mức cho phép. Bạn đọc cũng có thể chủ động gửi dữ liệu này lên server ThingSpeak và kích hoạt tính năng gửi email cảnh báo, vốn đã hướng dẫn ở các bài trước, và sẽ không trình bày lại ở bài này.

Giá trị mà chúng ta nhận được từ mọi cảm biến ADC đều có giá trị từ 0 cho đến 1023, khi lập trình với Microbit. Nói một cách khác, bộ chuyển đổi ADC trên Microbit có 10 bit. Do đó, với điện áp đọc vào từ cảm biến, miền giá trị từ 0V đến 3.3V

sẽ được ánh xạ tuyến tính trên miền giá trị số 10 bit, từ 0 cho đến 1023 ($2^{10} - 1$). Do đó, trong trường hợp muốn tính ra chính xác giá trị của từng thông tin, chúng ta sẽ phải xem xét công thức chuyển đổi của từng thiết bị. Chúng tôi ví dụ rằng, **giá trị của khí Gas tính theo đơn vị ppm, là 2 lần của điện áp được tính ở đơn vị mili-Volt**. Với định nghĩ như thế, chúng ta cần đổi giá trị đọc được sang mV, bằng cách sử dụng khối lập trình **map** trong mục **Math**. Chương trình gợi ý cho việc chuyển đổi này như sau:



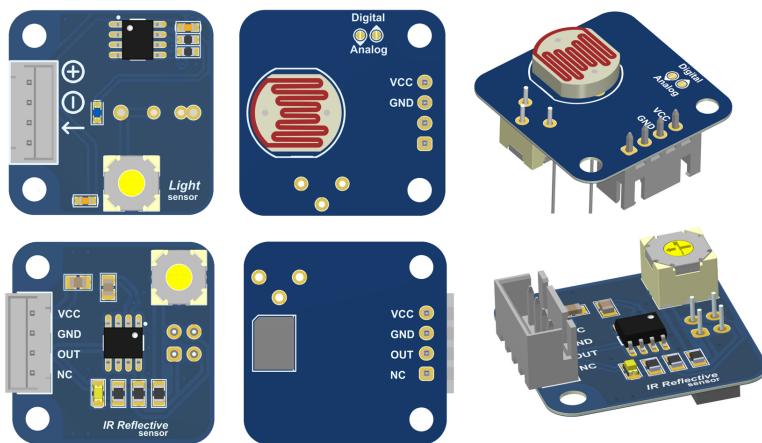
Hình 6.5: Chuyển đổi dữ liệu thô thành mV

Trong chương trình trên, 2 biến số đã được khai báo để lưu tạm các kết quả xử lý, bao gồm **gas_raw** để đọc dữ liệu thô và **gas_mV** là biến lưu giá trị sau khi đã chuyển sang mV. Do miền giá trị của **gas_raw** là từ 0 đến 1023, tương ứng với miền điện áp từ 0 đến 3300 mV (3.3V), nên các tham số của khối map được chỉnh lại như chương trình gợi ý ở trên.

Sau khi đã chuyển dữ liệu về đơn vị điện áp, bạn đọc sẽ phải sử dụng công thức chuyển đổi thêm 1 lần nữa, để tính ra đơn vị mình mong muốn. Công thức này sẽ phụ thuộc vào thiết bị cảm biến mà chúng ta sử dụng, nên sẽ không được hướng dẫn chi tiết ở bài này. Khi hiện thực công thức thứ 2 này, bạn đọc sẽ phải xài các câu lệnh tính toán trong nhóm Math. Phần này sẽ không được trình bày chi tiết trong hướng dẫn ở đây.

5 Cảm biến Analog ChiPi

Với mục đích tiện lợi cho bạn đọc mới bắt đầu với hệ thống cảm biến, vốn cần nhiều kinh nghiệm về điện tử, các cảm biến trong hệ thống ChiPi được thiết kế để đơn giản nhất cho bạn đọc với 1 kiểu kết nối duy nhất. Các khe cắm được thiết kế để bạn đọc không thể cắm nhầm. Tất cả những gì bạn đọc cần là xác định chân cảm cho chính xác, để lập trình lấy dữ liệu hợp lệ từ cảm biến. Một ví dụ về 2 thiết bị cảm biến khác, cũng ra dữ liệu dạng ADC, chẳng hạn như cảm biến cường độ ánh sáng và cảm biến vật cản hồng ngoại, như sau:



Hình 6.6: Một số cảm biến ADC trong chuẩn ChiPi

Đối với cảm biến vật cản hồng ngoại, khi vật thể ở gần, điện áp nhận được sẽ tăng (do ánh sáng phản xạ nhiều). Ngược lại, khi vật cản đi xa hoặc không có vật cản, điện áp sẽ giảm. Bên cạnh đó, còn có các cảm biến như độ ẩm đất, cảm biến cường độ âm thanh cũng thuộc nhóm này. Một số điểm lưu ý quan trọng khi kết nối với Microbit như sau:

- Cảm biến dạng analog chỉ hỗ trợ được cho các chân P0, P1, P2, P3, P4 và P10. Tức là chúng ta chỉ có thể kết nối được tối đa 6 cảm biến dạng analog vào một mạch Microbit. Trong trường hợp muốn kết nối nhiều hơn, bạn có thể sử dụng thêm 1 mạch Microbit khác và trao đổi dữ liệu giữa chúng thông qua giao tiếp không dây Radio.
- Trong trường hợp kết nối với các chân P3, P4 và P10, trong khôi **on start** cần sử dụng câu lệnh **led enable false** để lấy quyền điều khiển của 3 chân này. Khi đó, câu lệnh hiển thị ra màn hình của Microbit sẽ không sử dụng được nữa.
- Việc lựa chọn ngưỡng cần phải thực hiện tỉ mỉ. Khi không còn sử dụng được 25 đèn hiển thị trên Microbit (do câu lệnh **led enable false**), bạn có thể sử dụng như gửi dữ liệu lên máy tính hoặc tính năng Data Streamer của Excel.

Rất nhiều các cảm biến cao cấp, chẳng hạn như đo chất lượng nước và chất lượng không khí đều có hỗ trợ trên thị trường với chuẩn đầu ra là ADC, mà bạn đọc có thể tích hợp vào hệ thống sử dụng mạch Microbit. Khi đó, bạn đọc cần lưu ý điện áp nguồn cung cấp cho thiết bị để đảm bảo cảm biến hoạt động đúng chức năng.

6 Câu hỏi ôn tập

1. Cảm biến CO₂ trong bài có đầu ra là dạng tín hiệu gì?
 - A. digital
 - B. analog
 - C. uart
 - D. Tất cả đều đúng
2. Điện áp đầu ra của cảm biến analog được dựa trên nguyên lý gì?
 - A. Cầu phân áp dùng 2 điện trở
 - B. Biến trở
 - C. Điện trở cố định
 - D. Tất cả đều đúng
3. Thiết kế cách ly đối với cảm biến analog, thiết bị nào thường được sử dụng?
 - A. Điện trở
 - B. Biến trở
 - C. OPAM
 - D. Tất cả đều đúng
4. Mạch Microbit hỗ trợ tối đa bao nhiêu chân cho kết nối với cảm biến analog?
 - A. 1
 - B. 2
 - C. 6
 - D. Tất cả đều sai
5. Câu lệnh để hỗ trợ cho việc chuyển đổi từ giá trị thô sang điện áp được hỗ trợ trên MakeCode là gì?
 - A. Math
 - B. map
 - C. linear
 - D. Tất cả đều sai
6. Khi kết nối cảm biến với chân P3, cần phải thực thi câu lệnh nào trong khối on start?
 - A. led enable true
 - B. led enable false
 - C. led turn on
 - D. led turn off
7. Khi tắt quyền hiển thị đèn trên mạch Microbit, các câu lệnh nào sẽ không còn tác dụng?
 - A. show number
 - B. show icon
 - C. show string
 - D. Tất cả các câu lệnh trên

Đáp án

1. B 2. A 3. C 4. B 5. B 6. B 7. D



CHƯƠNG 7

Tạo tài khoản trên Adafruit IO



1 Giới thiệu

Với những gì được hỗ trợ trên server ThingSpeak, việc gửi dữ liệu quan trắc lên server có thể được thực hiện dễ dàng. Tuy nhiên, trong nhiều ứng dụng, nhu cầu điều khiển ngược lại là rất cần thiết. Chẳng hạn như, từ một nút nhấn trên điện thoại, người sử dụng muốn điều khiển một động cơ đang kết nối với mạch Micro-bit.



Hình 7.1: Giao diện quan trắc và điều khiển trên Adafruit IO

Với sự hỗ trợ của server ThingSpeak, chúng ta cũng có thể hiện thực được chức năng trên, nhưng nó là không dễ dàng. Do vậy, từ bài hướng dẫn này, chúng tôi sẽ sử dụng một server khác chuyên dụng hơn cho các ứng dụng quan trắc và điều khiển từ xa. Mặc dù tính năng điều khiển ngược lại từ phía server tới mạch Microbit chưa được hiện thực một cách trọn vẹn, Adafruit server cung cấp những giao diện đẹp hơn và theo kinh nghiệm của chúng tôi, nó dễ tương tác hơn so với ThingSpeak. Các nội dung chính trong bài hướng dẫn này sẽ như sau:

- Tạo tài khoản trên Adafruit IO Server
- Tạo kênh lưu trữ dữ liệu trên Adafruit IO
- Chia sẻ kênh ở dạng Public

2 Tạo tài khoản trên Adafruit IO

Bước 1: Vào trang web chính tại địa chỉ <https://io.adafruit.com/>. Giao diện sau đây ở hiện ra.



Hình 7.2: Trang chủ của Adafruit IO

Bạn đọc nhấn vào nút **Sign In** để đăng nhập vào hệ thống nếu như đã có tài khoản. Tuy nhiên, nút này cũng sẽ dẫn chúng ta đến trang tạo mới tài khoản ở bước tiếp theo.

Bước 2: Vì chúng ta chưa có tài khoản, nên ở bước này, chúng ta sẽ chọn tiếp **Sign Up** để đăng kí tài khoản.

SIGN IN

Your Adafruit account grants you access to all of Adafruit, including the shop, learning system, and forums.

EMAIL OR USERNAME

PASSWORD [Forget your password?](#)

SIGN IN

ORDER STATUS

Did you check out as a guest? Or do you just want to check your order status without signing in?

EMAIL ADDRESS

ORDER NUMBER [Where do I find this?](#)

CHECK ORDER STATUS

NEED AN ADAFRUIT ACCOUNT?

SIGN UP



Hình 7.3: Tạo tài khoản trên Adafruit IO

Bước 3: Cung cấp thông tin cá nhân(FIRST NAME và LAST NAME), Email, Tên đăng nhập và Mật khẩu. Riêng với tên đăng nhập, bạn không được sử dụng kí tự đặc biệt. Cuối cùng, chúng ta nhấn vào nút **CREATE ACCOUNT** để tạo tài khoản, như hướng dẫn bên dưới.

SIGN UP

The best way to shop with Adafruit is to create an account which allows you to shop faster, track the status of your current orders, review your previous orders and take advantage of our other member benefits.

FIRST NAME
NPNLab

LAST NAME
BKU

EMAIL
nplabvn@gmail.com ✓

USERNAME
nplablyn ✓

Username is viewable to the public on the forums, Adafruit IO, and elsewhere.

PASSWORD
..... ✓

CREATE ACCOUNT

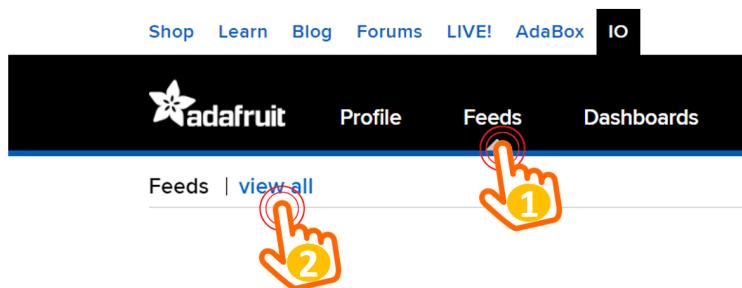
Hình 7.4: Giao diện đăng ký tài khoản

Sau khi tạo tài khoản thành công, hệ thống sẽ tự đăng nhập. Tuy nhiên, ở lần làm việc tiếp theo, bạn đọc hoàn toàn có thể đăng nhập vào hệ thống với chức năng **Sign In**.

3 Tạo kênh dữ liệu (Feed)

Để có thể lưu dữ liệu trên server, chúng ta cần phải phân loại cho nó. Thông thường, chúng ta gọi là một **kênh dữ liệu, hay feed**. Mỗi đối tượng trong hệ thống cũng sẽ thường có 1 kênh dữ liệu cho riêng nó. Chẳng hạn như để lưu trạng thái của một bóng đèn, chúng ta cần một kênh dữ liệu, tên là **BBC_LED** chẳng hạn. Sau khi đăng nhập vào hệ thống thành công, chúng ta bắt đầu tạo một kênh dữ liệu đầu tiên, với các bước hướng dẫn bên dưới.

Bước 1: Mở danh sách kênh dữ liệu, bằng cách nhấn vào Feeds, như minh họa ở hình bên dưới:



Hình 7.5: Truy cập vào các kênh dữ liệu

Tiếp theo đó, chọn tiếp vào tính năng **View all** để liệt kê tất cả các kênh dữ liệu đang có trong tài khoản, như minh họa ở hình sau đây:



Hình 7.6: Danh sách các kênh có sẵn

Trong minh họa ở hình trên, tài khoản của chúng ta chưa có một kênh dữ liệu nào cả, do nó mới được đăng ký lần đầu tiên. Một kênh dữ liệu sẽ tạo ra bằng cách nhấn vào nút **New Feed**.

Bước 2: Điền các thông tin cho kênh dữ liệu, như minh họa ở hình bên dưới.

Name	BBC_LED
Maximum length: 128 characters. Used: 7	
Description	Kênh dữ liệu cho đèn LED
<input type="button" value="Cancel"/> <input type="button" value="Create"/>	

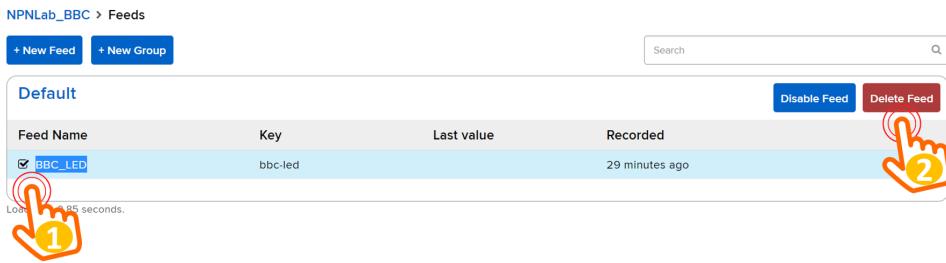
Hình 7.7: Điền thông tin cần thiết cho kênh dữ liệu

Quan trọng nhất là trường **Name** của kênh dữ liệu. Bạn đọc nên đặt tên cho nó đi kèm với 1 tiếp đầu ngữ, để có thể dễ dàng phân biệt kênh dữ liệu này là dành cho dự án nào. Trong hướng dẫn này, chúng tôi minh họa cho dữ liệu từ một đèn hiển thị trên Microbit, và đặt tên cho kênh là **BBC_LED**. Phần mô tả **Description** chỉ là tùy chọn, bạn đọc có thể ghi thêm các thông tin chú thích. Cuối cùng nhấn vào nút **Create**. Một kênh mới sẽ được tạo ra và giao diện của chúng ta bây giờ sẽ như sau:



Hình 7.8: Kênh dữ liệu mới được tạo thành công

Trong trường hợp muốn xóa một kênh dữ liệu cũ, bạn đọc chỉ cần chọn nó và nhấn nút **Delete Feed**, như minh họa ở hình bên dưới:

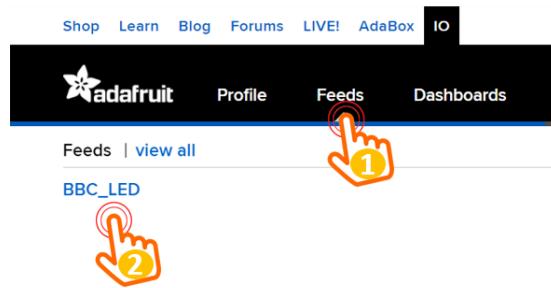


Hình 7.9: Xóa một kênh dữ liệu đã có

4 Chia sẻ Feed ở dạng Public

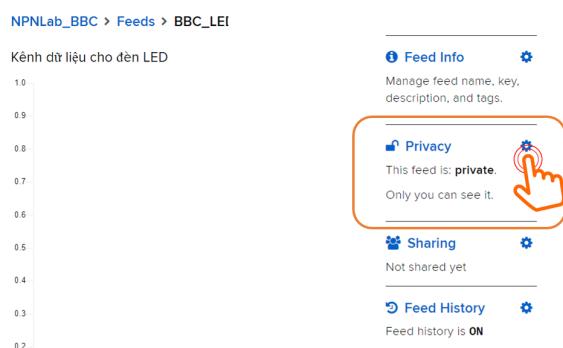
Khi một kênh dữ liệu (feed) được tạo ra, mặc định của nó là ở chế độ **private**, chỉ có tài khoản đăng nhập mới có thể truy cập và gửi dữ liệu lên nó. Để đơn giản hóa việc lập trình từ các thiết bị IoT gửi lên feed, chúng ta sẽ cấu hình cho nó là dạng **Public**.

Để làm được việc này, chúng ta cần phải truy xuất trực tiếp vào feed, bằng cách nhấp chuột trực tiếp vào tên feed, ở đây là **BBC_LED**. Có nhiều cách để bạn đọc có thể tìm thấy kênh của mình. Theo quy trình hiện tại, bạn đọc có thể nhấp trực tiếp vào **BBC_LED** ở Hình 7.9. Trong trường hợp bạn vào lại tài khoản của mình, chỉ cần chọn Feeds, kênh dữ liệu này sẽ xuất hiện, như minh họa ở hình bên dưới:



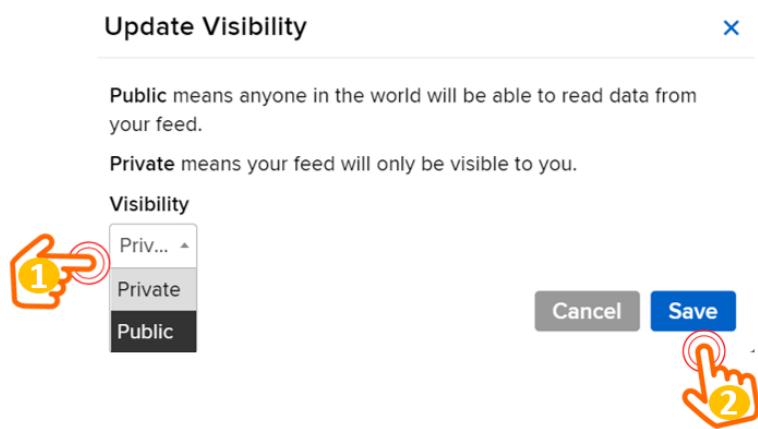
Hình 7.10: Một cách khác để truy cập vào feed

Thông tin chi tiết của feed dữ liệu sẽ được hiện ra như hình bên dưới:



Hình 7.11: Thông tin của feed dữ liệu

Chúng ta hãy để ý ở mục **Privacy** nằm ở khung bên phải, hiện tại nó đang ở chế độ **private**. Chúng ta nhấp vào biểu tượng cài đặt, để tới giao diện sau đây:



Hình 7.12: Cài đặt chia sẻ cho feed

Trong giao diện trên, chúng ta chọn **Public** ở phần **Visibility** và cuối cùng, nhấn vào nút **Save** để hoàn tất việc chỉnh kênh ở chế độ chia sẻ.

Bây giờ, thông tin ở mục **Privacy** đã thay đổi, với thêm thông tin chỉ dẫn **Anyone can see it at this link**. Bạn có thể chia sẻ kênh dữ liệu của mình với người khác bằng cách gửi đi đường liên kết này. Tuy nhiên, bước này chỉ cần thiết trong việc kiểm tra kênh dữ liệu có giao tiếp được tức thì hay không. Ngoài ra, nó cũng không thực sự là tính năng có ích trong các ứng dụng mà chúng ta sắp sửa hiện thực. Thông thường, bạn sẽ có nhu cầu che giấu kênh của mình để bảo vệ dữ liệu của hệ thống. Mục đích của chúng ta khi chỉnh kênh thành Public chỉ để đơn giản việc lập trình trong tương lai.

5 Câu hỏi ôn tập

1. Server được giới thiệu trong bài hướng dẫn có tên là gì?
 - A. ThingSpeak
 - B. Google
 - C. Amazon
 - D. Adafruit IO
2. Một kênh để lưu dữ liệu trên server còn được gọi là gì?
 - A. Client
 - B. Server
 - C. Feed
 - D. Channel
3. Kiến trúc ứng dụng kết nối vạn vật được đề xuất bởi Timothy Chou có mấy lớp?
 - A. 2
 - B. 3
 - C. 4
 - D. 5
4. Lớp các thiết bị như cảm biến, máy bơm, các mạch công suất, thuộc lớp nào trong mô hình IoT?
 - A. Things
 - B. Connect
 - C. Collect
 - D. Learn
5. Gateway IoT thuộc lớp nào trong các lớp dưới đây?
 - A. Things
 - B. Connect
 - C. Collect
 - D. Learn
6. Để tiện lợi cho việc lập trình trong tương lai, các thao tác nào là cần thiết?
 - A. Tạo kênh dữ liệu có tên gợi nhớ
 - B. Nên có tiếp đầu ngữ cho mỗi kênh dữ liệu
 - C. Chính kênh ở chế độ Public
 - D. Tất cả các thao tác trên
7. Để truy cập vào thông tin chi tiết của một feed, các thao tác cần thiết là gì?
 - A. Chọn Feeds, chọn kênh dữ liệu (tên feed)
 - B. Chọn Feeds, chọn view all, chọn kênh dữ liệu (tên feed)
 - C. Cả 2 thao tác trên đều được
 - D. Tất cả đều đúng

Đáp án

1. D 2. C 3. D 4. A 5. C 6. D 7. D

CHƯƠNG 8

Thiết kế Dashboard trên Adafruit IO

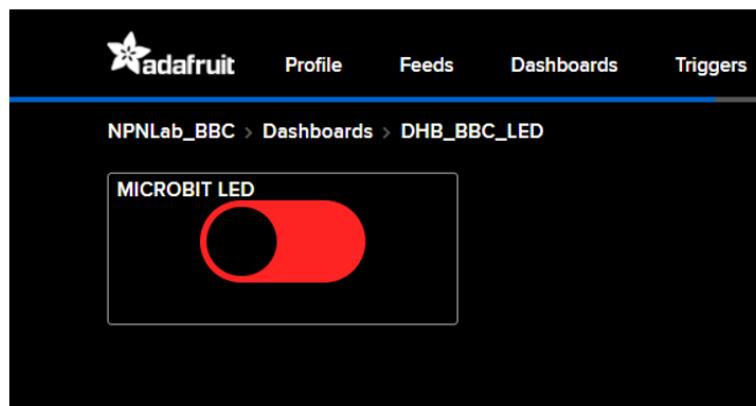


1 Giới thiệu

Dashboard có thể được hiểu là một bảng tổng hợp, hiển thị các thông tin cần thiết của một hệ thống. Trên dashboard, thông tin có thể được tổng hợp để hiển thị dưới dạng đồ thị với các dữ liệu lịch sử, thông tin hiện tại cũng như thống kê các giá trị nhỏ nhất, lớn nhất hay các giá trị trung bình. Ngoài ra, Dashboard còn có thể được sử dụng như một bản điều khiển thân thiện, để người dùng có thể tương tác và vận hành hệ thống từ xa. Tùy vào đặc thù của ứng dụng và nơi áp dụng, các yêu cầu của Dashboard có thể khác nhau. Trong các công ty, Dashboard đưa ra một cái nhìn tổng quát về năng suất của từng bộ phận, các xu hướng, các hoạt động, các chỉ số KPI (Key Performance Indicator – hay còn gọi là chỉ số đánh giá thực hiện công việc).

Với mục đích sử dụng khác nhau hoàn toàn, kênh dữ liệu Feed giới thiệu ở bài trước, thường dành cho người quản lý, để kiểm tra dữ liệu thô của hệ thống. Trong khi đó, Dashboard sẽ là một giao diện đẹp và thân thiện đối với người sử dụng. Hai đối tượng này, cũng thường được gọi là **Back End** dành cho Feed và **Front End** dành cho Dashboard. Và hiển nhiên, 2 đối tượng này sẽ liên kết chặt chẽ với nhau: Mỗi khi có dữ liệu gửi lên Feed, giao diện trên Dashboard sẽ được cập nhật tương ứng, và ngược lại, mỗi khi có tương tác trên Dashboard, thông tin cũng sẽ được lưu lại trên Feed.

Ở bài này, bạn đọc sẽ được hướng dẫn để tạo một Dashboard đơn giản trên Adafruit IO. Dashboard này là dùng để kết nối với Feed ta đã tạo ở bài trước và có một nút nhấn, để người dùng có thể bật tắt đèn trên mạch Microbit. Giao diện tương tác của người dùng trên Dashboard như sau:



Hình 8.1: Giao diện Dashboard điều khiển đèn

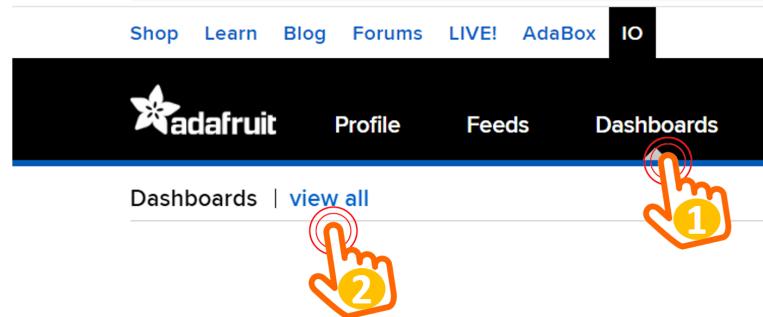
Các mục tiêu hướng dẫn trong bài này được tóm tắt như sau:

- Tạo một Dashboard với một nút nhấn
- Liên kết Dashboard và Feed dữ liệu
- Kiểm tra tương tác giữa Dashboard và Feed

2 Tạo Dashboard mới

Trong phần này, hướng dẫn sẽ trình bày để bạn đọc có thể tạo ra 1 giao diện đơn giản, với một nút nhấn trên Dashboard, dùng để bật/tắt một thiết bị, chẳng hạn như là đèn hiển thị trên mạch Microbit. Giao diện và thao tác để tạo mới một Dashboard cũng khá tương tự với việc tạo mới một Feed ở bài trước, được trình bày chi tiết từng bước như dưới đây.

Bước 1: Sau khi đăng nhập vào Adafruit IO, bạn đọc chọn vào **Dashboard**, và chọn tiếp **View all**, như hướng dẫn bên dưới:



Hình 8.2: Truy cập vào Dashboard của tài khoản

Bước 2: Nhấn nút **+New Dashboard** để tạo mới một Dashboard trên Adafruit, như minh họa ở hình sau đây:



Hình 8.3: Tạo mới một Dashboard

Sau đó, một cửa sổ sẽ hiện ra cho ta ghi tên và mô tả cho Dashboard của mình. Lưu ý là trường **Name** thông tin bắt buộc, trong khi đó, trường **Description** (mô tả) là tùy chọn. Khi đặt tên cho Dashboard, bạn cũng nên thêm những tiếp đầu ngữ để dễ quản lý trong tương lai, như minh họa ở hình bên dưới:

Create a new Dashboard X

Name	<input type="text" value="DHB_BBC_LED"/>
Description	<input type="text" value=""/>
Cancel Create 	

Hình 8.4: Hoàn thiện thông tin cho Dashboard

Cuối cùng, nhấn vào nút **Create**, một Dashboard mới sẽ xuất hiện trong trang chủ Adafruit IO của bạn, như sau:

NPNLab_BBC > Dashboards

Dashboards		
Name	Key	Created At
DHB_BBC_LED	dhb-bbc-led	August 9, 2021

Loaded in 1.35 seconds.

Hình 8.5: Một Dashboard mới đã được tạo

Trong trường hợp muốn xóa Dashboard, bạn đọc cần chọn nó trong danh sách ở trên, và nhấn vào nút **Delete Dashboard**, như minh họa ở hình sau:

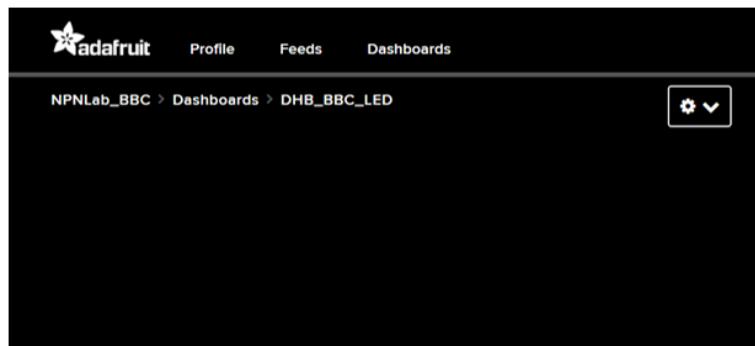
NPNLab_BBC > Dashboards

Dashboards		
Name	Key	Created At
DHB_BBC_LED	dhb-bbc-led	August 9, 2021

1 2

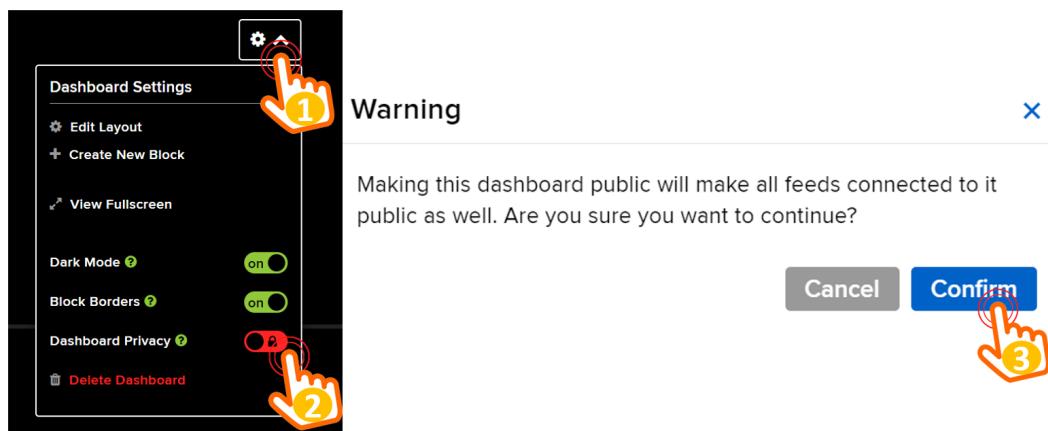
Hình 8.6: Xóa một Dashboard trong danh sách

Sau khi tạo thành công Dashboard ở các bước hướng dẫn bên trên, bạn đọc có thể truy xuất vào nó (bằng cách nhấp chuột trực tiếp vào tên Dashboard), một giao diện khởi tạo của Dashboard sẽ hiện ra, như sau:



Hình 8.7: Giao diện khởi tạo cho Dashboard

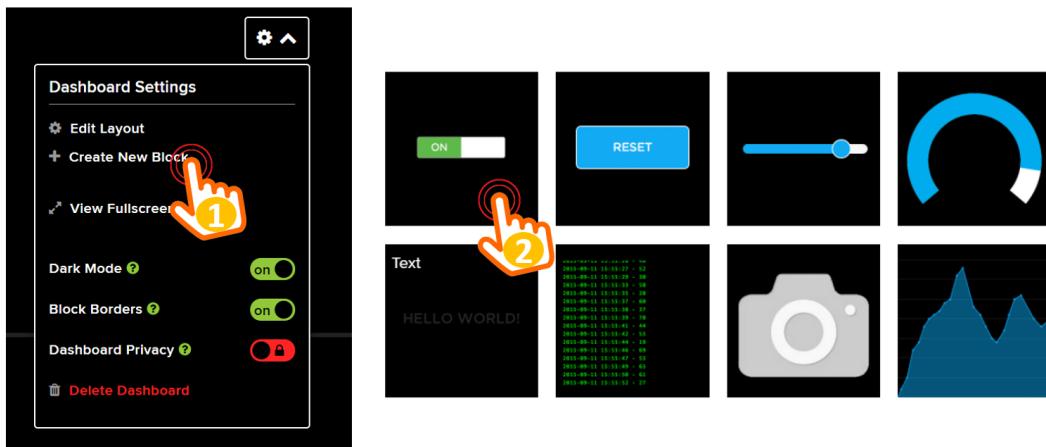
Trước khi thiết kế các đối tượng giao diện trên Dashboard (trong bài này là một nút nhấn), chúng ta cần cấu hình cho Dashboard ở dạng **Public**, để tiện chia sẻ trên nhiều thiết bị khác nhau, chẳng hạn như điện thoại hoặc máy tính bảng, để điều khiển và giám sát hệ thống từ xa. Bằng cách chọn vào biểu tượng cài đặt ở phía bên phải, chúng ta chọn tiếp vào **Dashboard Privacy**, như hướng dẫn sau đây:



Hình 8.8: Tùy chỉnh cho Dashboard ở chế độ Public

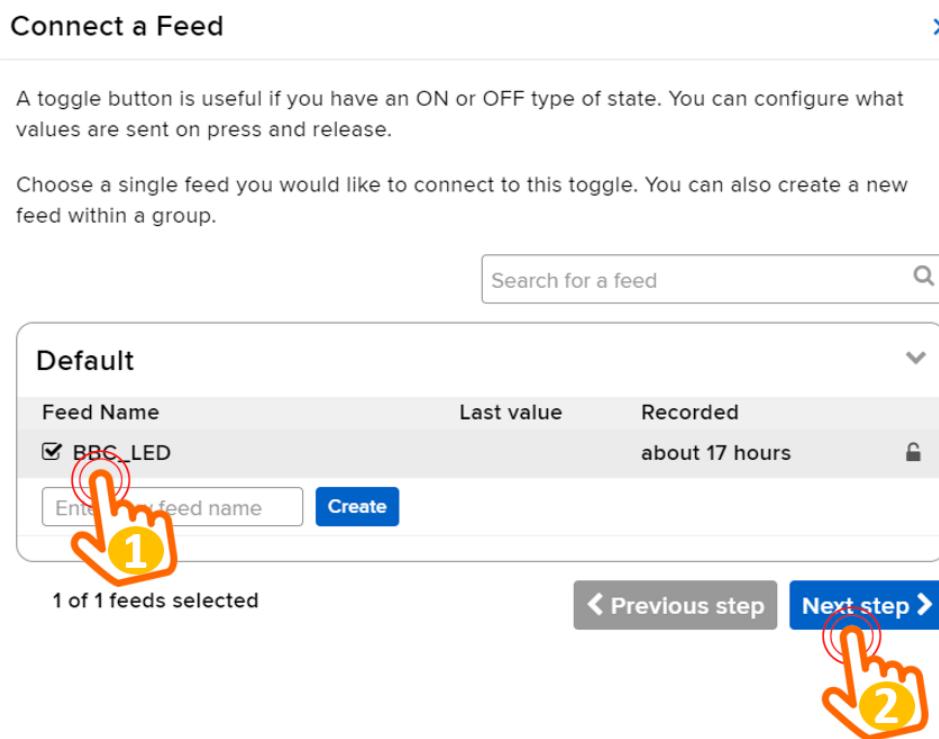
3 Thiết kế giao diện cho Dashboard

Giao diện mặc định của Dashboard khi mới được tạo ra là chưa có phần tử giao diện nào, chúng ta sẽ bắt đầu thêm một nút nhấn trên giao diện. Bằng cách vào lại biểu tượng cài đặt, và chọn **Create New Block**, như sau:



Hình 8.9: Thêm đối tượng giao diện vào Dashboard

Đối tượng giao diện hỗ trợ trên Dashboard là rất phong phú. Tuy nhiên trong bài hướng dẫn này, chúng ta sẽ chọn vào đối tượng đầu tiên. Đối tượng này có tên gọi là **Toggle Button**, và nó hoàn toàn phù hợp cho một ứng dụng điều khiển Bật/Tắt một thiết bị trong bài này. Sau khi chọn, giao diện sau đây sẽ hiện ra.



Hình 8.10: Kết nối với feed đã tạo

Đây là bước quan trọng nhất trong thiết kế giao diện cho Dashboard, khi chúng ta cần phải chọn liên kết cho nó với một Feed dữ liệu. Cho đến bài này, vì chúng ta chỉ có 1 feed dữ liệu (là BBC_LED), nên việc lựa chọn rất đơn giản. Trong trường hợp có nhiều feed dữ liệu, bạn đọc cần phải lựa chọn cho đúng. Sau đó chọn vào **Next step** và tiếp tục tùy chỉnh các thông tin cài đặt cho giao diện bên dưới.

Block settings

X

In this final step, you can give your block a title and see a preview of how it will look. Customize the look and feel of your block with the remaining settings. When you are ready, click the "Create Block" button to send it to your dashboard.

Block Title (optional)

MICROBIT LED

Button On Text

ON

Button On Value (uses On Text if blank)

1

Button

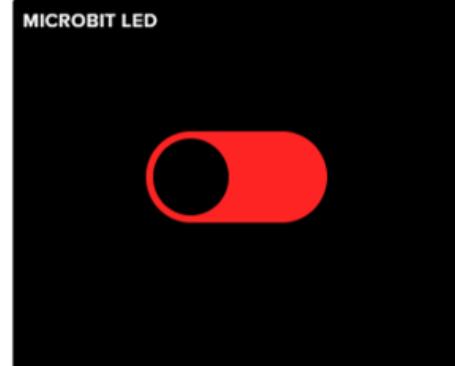
OFF

Button Off Value (uses Off Text if blank)

0



Block Preview



Toggle A toggle button is useful if you have an ON or OFF type of state. You can configure what values are sent on press and release.

Test Value



Publish

0 bytes

◀ Previous step

Create block

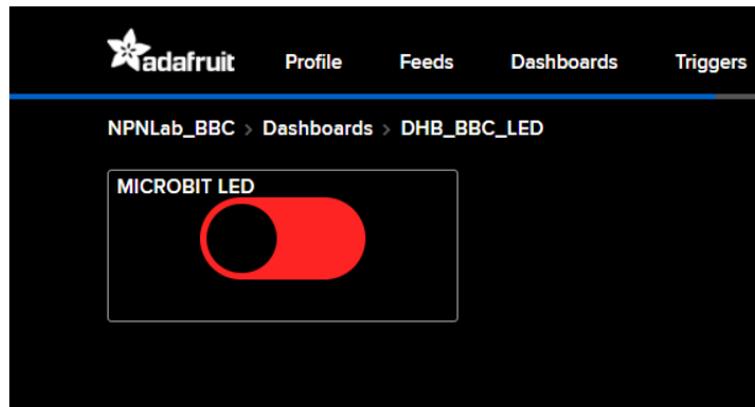


Hình 8.11: Các chức năng cài đặt cho đối tượng nút nhấn

Chức năng của các trường thông tin được tóm tắt như sau:

- **Block Title:** Trường này là tùy chọn thêm, không bắt buộc. Ở đây ta có thể đặt tên cho nút nhấn.
- **Button On Text:** Ở đây ta sẽ để những ký tự sẽ hiện lên trên nút nếu nút mở. Ở trạng thái mặc định không tùy chỉnh thì khi được mở, nút sẽ hiện từ "ON".
- **Button On Value:** Trường này cho ta tùy chỉnh dữ liệu được gửi đi khi nút này được bấm để trở thành trạng thái Bật. Để cho đơn giản, **chúng ta sẽ quy định dữ liệu cho nó là 1.**
- **Button Off Text:** Ở đây ta sẽ để những ký tự sẽ hiện lên trên nút nếu nút tắt. Ở trạng thái mặc định không tùy chỉnh thì khi được tắt, nút sẽ hiện từ "OFF".
- **Button Off Value:** Trường này cho ta tùy chỉnh dữ liệu được gửi đi khi nút này được bấm để trở thành trạng thái tắt. **Chúng ta sẽ quy định dữ liệu cho trường này là 0.**

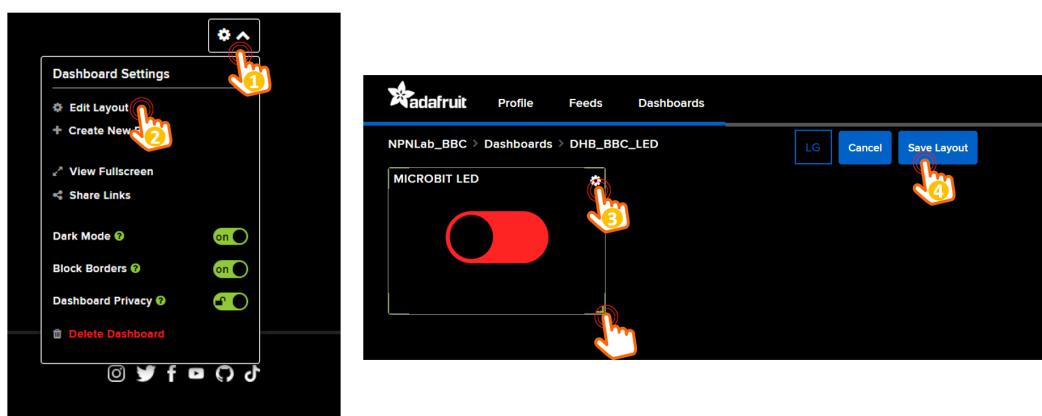
Ngoài ra, bạn đọc có thể kiểm tra việc thay đổi giao diện của nút nhấn, mỗi khi có dữ liệu mới được gửi lên Feed, bằng cách nhập giá trị 0 và 1 trong ô **Test Value**. Chức năng này đang mô phỏng hình ảnh của nút nhấn, khi vận hành trong thực tế. Cuối cùng, ta chọn **Create block** và một nút nhấn mới sẽ xuất hiện trong Dashboard của bạn, như sau.



Hình 8.12: Nút nhấn mới được tạo trên Dashboard

4 Chính sửa nút nhấn

Trong một số trường hợp, chúng ta sẽ có nhu cầu cài đặt lại cho đối tượng nút nhấn, chẳng hạn như làm cho kích thước của nó to hơn, thay đổi các giá trị cho trạng thái Bật/Tắt, hoặc thậm chí là xóa nút nhấn này. Tính năng hiệu chỉnh có thể được kích hoạt, bằng cách nhấn vào biểu tượng cài đặt, và chọn tiếp vào **Edit Layout**, như hướng dẫn bên dưới:



Hình 8.13: Chính sửa đối tượng nút nhấn

Các đối tượng giao diện trên Dashboard lúc này sẽ xuất hiện thêm một biểu tượng cài đặt riêng của nó. Khi nhấn vào biểu tượng này, và chọn tiếp **Edit Block**, giao diện cũ trình bày ở Hình 8.11 sẽ xuất hiện để bạn đọc thay đổi các thông tin cấu hình. Tính năng xóa đối tượng giao diện cũng sẽ xuất hiện trong danh sách lựa chọn, khi chúng ta nhấn vào biểu tượng cài đặt này.

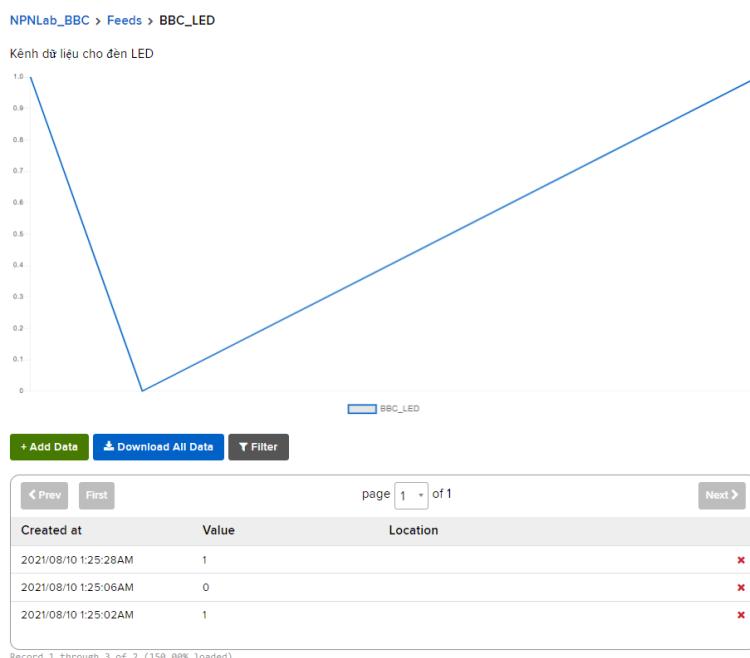
Trong trường hợp muốn thay đổi kích thước của đối tượng giao diện, bạn đọc chỉ đơn giản là kéo thả 4 góc của nó để thay đổi. Cuối cùng, nhấn vào nút **Save Layout**.

5 Kiểm tra kết nối giữa Feed và Dashboard

Đây là bước kiểm tra cuối cùng các cấu hình trên Adafruit IO, trước khi chúng ta bắt đầu lập trình gửi dữ liệu từ Gateway lên Feed. Quy trình kiểm tra sẽ theo trình tự như sau:

Bước 1: Từ giao diện Dashboard, bạn hãy nhấn thử nút nhấn trên giao diện, ghi nhớ lại số lần nhấn, tương ứng với các trạng thái ON và OFF của nút nhấn.

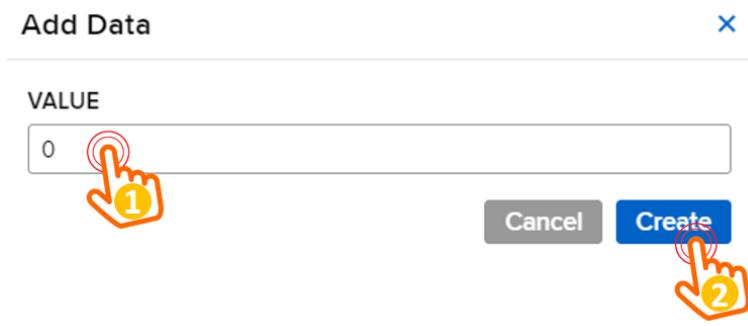
Bước 2: Mở lại giao diện của kênh dữ liệu Feed, bạn sẽ thấy giá trị thô, là 0 và 1 sẽ được gửi lên Feed, như minh họa ở hình sau.



Hình 8.14: Dữ liệu được gửi lên Feed khi tương tác trên Dashboard

Mặc dù một đồ thị được vẽ ra trên kênh dữ liệu, bạn đọc cần lưu ý dữ liệu thô của nó, được liệt kê trong một bảng bên dưới. Mỗi khi chúng ta nhấn nút, trạng thái của nó sẽ được lưu lại, cùng với thông số về thời gian. Thông tin thô này, sẽ được gửi xuống Gateway trong tương lai.

Bước 3: Tại trang dữ liệu Feed, hãy chọn và nút **+ Add Data**. Tính năng này đang mô phỏng việc gửi dữ liệu từ Gateway lên Feed trong tương lai. Giao diện sau đây sẽ hiện ra:



Hình 8.15: Thêm dữ liệu trên Feed bằng tay

Bạn đọc hãy điền những giá trị hợp lệ, trong trường hợp này là 0 hoặc 1, sau đó nhấn vào nút **Create**. Hiển nhiên, 1 dòng dữ liệu mới sẽ được thêm vào Feed. Nhưng song song đó, giao diện của Dashboard cũng được thay đổi tương ứng.

Bước 4: Kiểm tra độ trễ của hệ thống. Đây sẽ điều mới mẻ mà Adafruit IO server mang lại. Bạn đọc hãy thử chia sẻ đường link Dashboard của mình với người khác, để 2 bên có thể kiểm tra từ xa. Chúng ta sẽ thấy rằng, việc giao tiếp dữ liệu giữa Feed và Dashboard có độ trễ rất thấp.

Với Gateway IoT sẽ hiện thực ở bài sau, mỗi khi tương tác trên Dashboard, dữ liệu sẽ được gửi lên Feed. Bước tiếp theo đó, Feed sẽ tự động gửi xuống cho Gateway IoT để thực thi lệnh này. Trong trường hợp ngược lại, khi dữ liệu được gửi lên Feed từ Gateway IoT, giao diện của Dashboard cũng sẽ tự động cập nhật theo. Nhờ cơ chế này, mà server Adafruit IO mới phù hợp cho các ứng dụng quan trắc và điều khiển từ xa qua mạng.

6 Câu hỏi ôn tập

1. Đối tượng nào sau đây có thể được sử dụng để giám sát và điều khiển hệ thống?
 - A. Gateway IoT
 - B. Feed
 - C. Dashboard
 - D. Adafruit IO
2. Đối tượng nào có vai trò giống như một Back-End?
 - A. Gateway IoT
 - B. Feed
 - C. Dashboard
 - D. Adafruit IO
3. Đối tượng nào có vai trò giống như một Front-End?
 - A. Gateway IoT
 - B. Feed
 - C. Dashboard
 - D. Adafruit IO
4. Khi cấu hình một nút nhấn trên Dashboard, giá trị cho 2 trạng thái của nút nhấn như thế nào là phù hợp?
 - A. 0 và 1
 - B. 1 và 2
 - C. ON và OFF
 - D. Khác nhau là được
5. Độ trễ của việc giao tiếp dữ liệu giữa Feed và Dashboard là:
 - A. Nhanh
 - B. Chậm
 - C. Trung bình
 - D. Không xác định được
6. Độ trễ của việc giao tiếp dữ liệu giữa Feed và Gateway IoT là:
 - A. Nhanh
 - B. Chậm
 - C. Trung bình
 - D. Không xác định được
7. Độ trễ của việc giao tiếp dữ liệu giữa Dashboard và Gateway IoT là:
 - A. Nhanh
 - B. Chậm
 - C. Trung bình
 - D. Không xác định được

Đáp án

1. C 2. B 3. C 4. D 5. A 6. A 7. A

CHƯƠNG 9

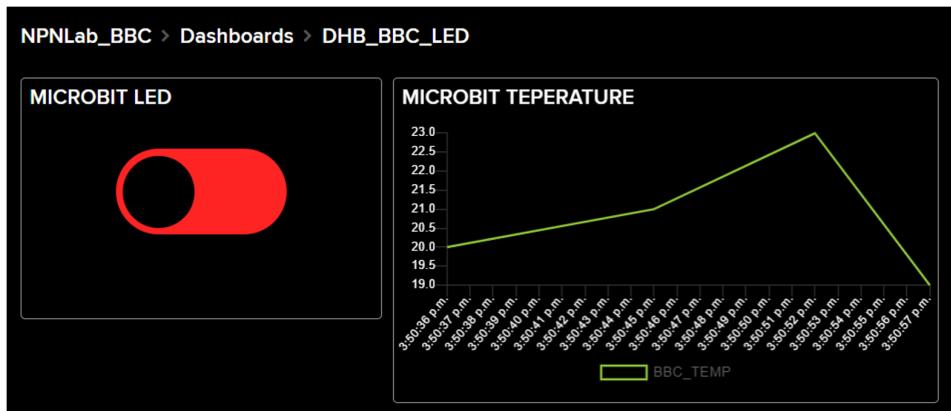
Đồ thị trên Dashboard



1 Giới thiệu

Ngược lại với bài trước, khi tín hiệu sẽ được gửi từ nút nhấn xuống mạch Microbit để điều khiển thiết bị, như một bóng đèn chẳng hạn, bài này sẽ thiết kế một đồ thị để nhận giá trị cảm biến từ mạch Microbit gửi lên. Nhu cầu này sẽ cần khi bạn đọc muốn xây dựng một ứng dụng giám sát định kì dữ liệu của hệ thống từ xa. Chẳng hạn như giám sát nhiệt độ của máy ấp trứng mỗi 30 giây. Thông tin này sẽ được vẽ thành đồ thị trên Dashboard. Khi nhiệt độ đạt tới ngưỡng mong muốn, bạn đọc có thể tắt bộ phận gia nhiệt để tiết kiệm điện chẳng hạn. Bạn đọc có thể nghĩ rằng, một hệ thống thông minh có thể tự tắt khi đã đạt được nhiệt độ mong muốn. Tuy nhiên, với các hệ thống thực tế, nhu cầu điều khiển bằng tay vẫn luôn là cần thiết, cho những trường hợp ngoại lệ có thể xảy ra. Do đó, chúng ta thường sẽ kết hợp cả 2 tính năng này trên một hệ thống giám sát và điều khiển từ xa.

Trong bài này, chúng ta sẽ thiết kế thêm một giao diện trên Dashboard để hiện thị thông tin nhiệt độ. Tất nhiên bạn đọc hoàn toàn có thể gửi thông tin khác, tùy thuộc vào ứng dụng mà mình muốn triển khai. Giao diện trên Dashboard sẽ như minh họa sau đây:



Hình 9.1: Giao diện đồ thị trên Dashboard

Với nhiều hỗ trợ hơn từ server Adafruit, mỗi khi mạch Microbit gửi thông tin, nó sẽ được cập nhật lên đồ thị ngay lập tức, với thời gian trễ rất thấp. Đây là điểm rất khác biệt nếu so sánh với các server truyền thống như ThingSpeak. Thông thường, ThingSpeak có độ trễ rất lớn, khoảng hơn 5 giây dữ liệu mới được cập nhật trên đồ thị. Tuy nhiên, do chúng ta đang xài tài khoản miễn phí từ Adafruit IO, nên cũng có giới hạn về số lần gửi dữ liệu lên server. Hiện tại, chúng ta chỉ có thể gửi tối đa 1 gói tin trong 1 giây. Do vậy, đối với các ứng dụng quan trắc, vốn không nhất thiết phải gửi dữ liệu liên tục, bạn hãy gửi với chu kỳ 30 giây hoặc thậm chí là 60 giây. Chúng ta sẽ để dành tài nguyên mạng cho các nút nhấn điều khiển ngược lại thiết bị kết nối với mạch Microbit (như đèn hay máy bơm chẳng hạn).

Mục tiêu hướng dẫn của bài được tóm tắt như sau:

- Tạo thêm một Feed dữ liệu
- Tạo giao diện đồ thị trên Dashboard
- Kiểm tra dữ liệu giữa Dashboard và Feed

2 Tạo Feed dữ liệu mới

Trước khi có thể tạo giao diện đồ thị trên Dashboard, bạn đọc cần tạo kênh dữ liệu (Feed) cho nó trước. Quy trình tạo thêm 1 kênh dữ liệu hoàn toàn tương tự như hướng dẫn ở bài trước. Trong bài này, bạn sẽ tạo thêm 1 kênh dữ liệu mới, đặt tên cho nó là **BBC_TEMP**. Bạn đọc cần vào lại mục **Feeds**, chọn tiếp vào **View All** và cuối cùng là **New Feed**.

The screenshot shows the Adafruit IO Feeds interface. At the top, there are buttons for '+ New Feed' and '+ New Group'. A search bar is on the right. Below is a table titled 'Default' with columns: 'Feed Name', 'Key', 'Last value', and 'Recorded'. Two entries are listed:

Feed Name	Key	Last value	Recorded
BBC_LED	bbc-led	0	about 16 hours ago
BBC_TEMP	bbc-temp	19	30 minutes ago

Hình 9.2: Tạo thêm một Feed trong tài khoản

Sau khi tạo Feed mới thành công, kết quả sẽ được hiển thị như hình bên trên. Tuy nhiên bạn đọc cần truy xuất trực tiếp vào Feed để chỉnh chế độ trung cập của nó thành **Public**. Thao tác này đã được trình bày ở bài trước và sẽ không được trình bày chi tiết ở đây.

3 Thêm đồ thị cho Dashboard

Sau khi tạo mới kênh dữ liệu cho dự án, bạn đọc có thể bắt đầu thiết lập Dashboard, với đối tượng đồ thị được thêm vào. Từ menu chính, chọn và **Dashboard** để truy cập vào chi tiết của Dashboard hiện tại (DHB_BBC_LED), như sau:

The screenshot shows the Adafruit IO Dashboard Settings menu. On the left is a preview of a dashboard block titled 'MICROBIT LED' with a red switch icon. On the right is a sidebar with 'Dashboard Settings' containing options like 'Edit Layout', '+ Create New Block' (which has a large orange circle with the number 1 over it), 'View Fullscreen', 'Share Links', 'Dark Mode', 'Block Borders', 'Dashboard Privacy', and 'Delete Dashboard'. The '+ Create New Block' option is highlighted with a red circle and a large orange hand cursor icon pointing at it.

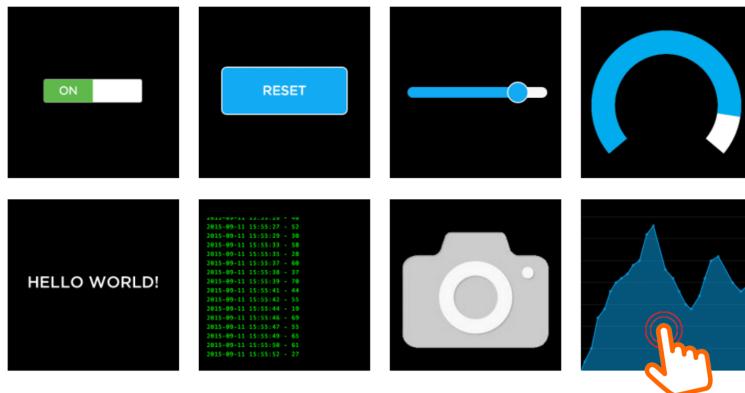
Hình 9.3: Thiết kế giao diện cho Dashboard

Từ biểu tượng cài đặt ở góc bên phải, bạn đọc chọn vào tính năng thêm đối tượng giao diện mới bằng cách nhấn chuột vào **+ Create New Block**. Với các đối tượng giao diện hỗ trợ trên Adafruit IO, chúng ta sẽ chọn vào đối tượng đồ thị, như hướng dẫn bên dưới:

Create a new block

X

Click on the block you would like to add to your dashboard. You can always come back and switch the block type later if you change your mind.



Hình 9.4: Chọn đồ thị giao diện

Tiếp theo, một giao diện sẽ được hiện ra, để chúng ta liên kết đồ thị với Feed dữ liệu. Ở đây, chúng ta sẽ chọn vào Feed **BBC_TEMPERATURE**, như minh họa ở hình bên dưới:

Connect Feeds

X

The line chart is used to graph one or more feeds.

Choose multiple feeds you would like to connect to this line chart. You can also create a new feed within a group.

Search for a feed

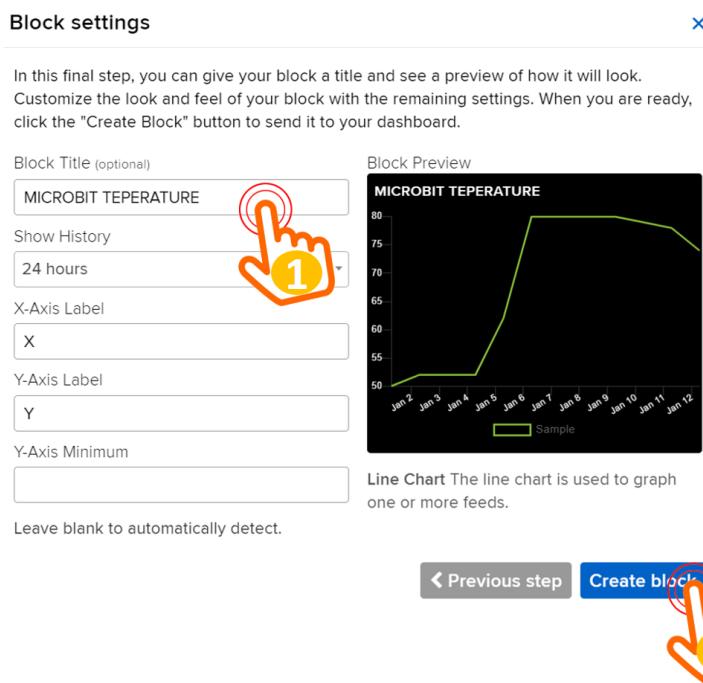
Default			
Feed Name	Last value	Recorded	
<input type="checkbox"/> BBC_LED	0	about 15 hours	🔒
<input checked="" type="checkbox"/> BBC_TEMP	30	less than a min...	🔒

Enter new feed

1 of 5 feeds selected 2

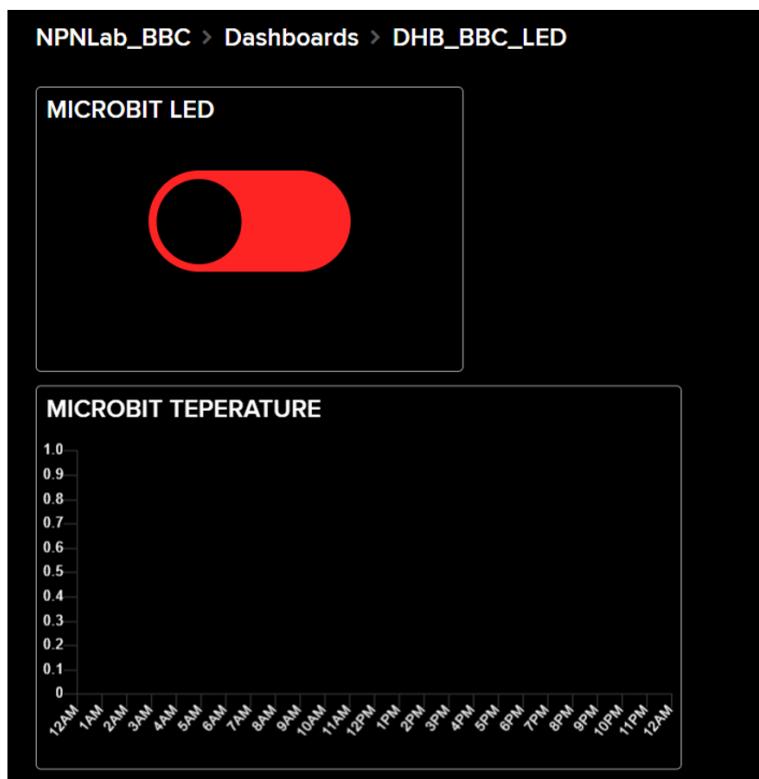
Hình 9.5: Chọn Feed dữ liệu liên kết với đồ thị

Nhấn vào nút **Next step** để cấu hình cho việc hiển thị của đồ thị. Giao diện bên dưới sẽ hiện ra:



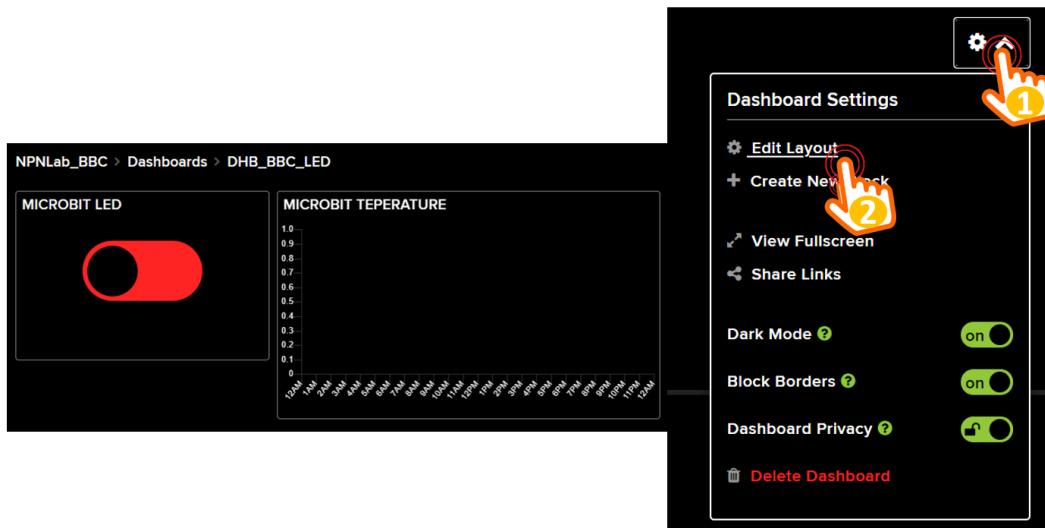
Hình 9.6: Cấu hình hiển thị trên đồ thị

Có rất nhiều thông tin mà bạn đọc có thể thay đổi. Tuy nhiên các thông tin cấu hình đều đang ở dạng mặc định mà bạn không cần phải thay đổi gì. Ở đây, chúng tôi chỉ đơn giản là thêm thông tin **Block Title**, và nhấn vào nút **Create block** ở cuối giao diện cài đặt này. Kết quả hiện tại, một đồ thị sẽ được thêm vào Dashboard như sau:



Hình 9.7: Cấu hình hiển thị trên đồ thị

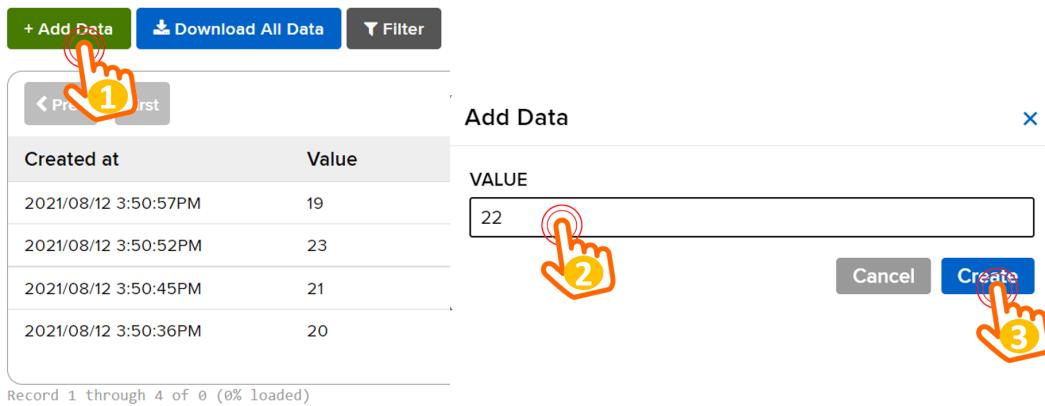
Theo chế độ mặc định, đồ thị mới thêm vào sẽ được xếp theo chiều dọc. Trong trường hợp bạn muốn sắp xếp lại nó ở vị trí khác, hay thậm chí thay đổi kích thước của đồ thị, thao tác cũng tương tự như khi chúng ta làm với nút nhấn: Chọn vào biểu tượng cài đặt, còn tiếp vào **Edit Layout**, sau đó kéo thả đồ vật trên Dashboard để thay đổi. Cuối cùng, nhấn vào nút **Save Layout** để kết thúc.



Hình 9.8: Thay đổi layout của giao diện trên Dashboard

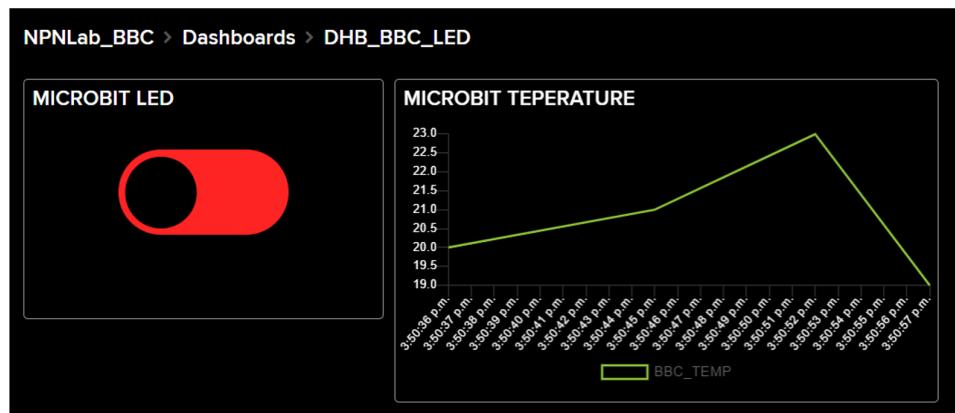
4 Kiểm tra tương tác giữa Feed và Dashboard

Đây là giai đoạn luôn luôn cần thiết, trước khi bắt đầu lập trình cho Gateway. Chúng ta cần phải đảm bảo liên kết giữa Feed và Dashboard là hoàn thiện. Việc này được thực hiện dễ dàng bằng cách thêm dữ liệu bằng tay cho Feed dữ liệu **BBC_TEMP**, như hướng dẫn bên dưới:



Hình 9.9: Thêm dữ liệu bằng tay trên Feed

Khi đã thêm một vài dữ liệu kiểm tra, bạn đọc có thể mở lại Dashboard. Nếu việc cài đặt trên Dashboard là thành công, một đồ thị sẽ được vẽ ra, tương ứng với các giá trị thêm bằng tay trên Feed, như sau:



Hình 9.10: Giao diện đồ thị trên Dashboard

Như vậy, hiện tại trên server chúng ta đã có 2 đối tượng giao diện cơ bản cho 1 ứng dụng. Một nút nhấn để điều khiển một thiết bị trên mạch Microbit và một giao diện đồ thị để biến diễn thông tin theo thời gian.

Các giao diện cao cấp khác trên Adafruit IO bạn đọc có thể tự tìm hiểu thêm và tích hợp vào các ứng dụng của mình. Chẳng hạn như để điều khiển tốc độ của một thiết bị, bạn đọc có thể sử dụng Thanh trượt (Slider), hoặc biểu đồ Gauge cũng là một đối tượng giao diện phổ biến để hiển thị giá trị hiện tại của nhiệt độ và độ ẩm.

5 Câu hỏi ôn tập

1. Đối tượng giao diện nào sau đây là phù hợp để biểu diễn dữ liệu theo thời gian?
 - A. Nút nhấn
 - B. Thanh trượt
 - C. Biểu đồ Gauge
 - D. Đồ thị (Graph)
2. Phát biểu nào sau đây là sai?
 - A. Nút nhấn dùng để gửi dữ liệu xuống mạch Microbit
 - B. Đồ thị dùng để điều khiển thiết bị theo thời gian
 - C. Thanh trượt có thể dùng để biểu diễn tốc độ của một động cơ
 - D. Biểu đồ Gauge có thể dùng để biểu diễn giá trị hiện tại
3. Dữ liệu cảm biến được gửi lên với chu kỳ bao nhiêu là phù hợp nhất?
 - A. 1 giây
 - B. 2 giây
 - C. 5 giây
 - D. Tùy ứng dụng, nhưng chu kỳ thường là đơn vị phút.
4. Khi cài đặt giao diện đồ thị, thông tin nào sau đây thường được thay đổi nhất?
 - A. Block title
 - B. Show history
 - C. X Label
 - D. Y Label
5. Trình tự để thay đổi vị trí của đồ thị trên Dashboard là gì?
 - A. Dashboard Setting, Create Layout, Save Layout
 - B. Dashboard Setting, Edit Layout, Save Layout
 - C. Tất cả đều đúng
 - D. Tất cả đều sai
6. Để thêm một dữ liệu vào đồ thị, trình tự thao tác nào sau đây là chính xác?
 - A. Dashboard Setting, Create Layout, Save Layout
 - B. Dashboard Setting, Edit Layout, Save Layout
 - C. Chọn vào feed, chọn Add Data
 - D. Tất cả đều sai

Đáp án

1. D 2. B 3. D 4. A 5. B 6. C



CHƯƠNG 10

Kết nối Microbit và Adafruit IO

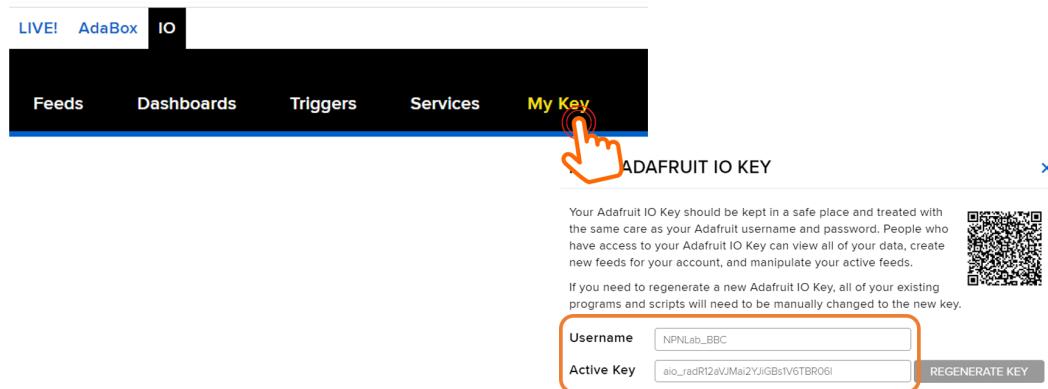


1 Giới thiệu

Sau khi đã tạo xong feed và các giao diện trên Dashboard, chúng ta đã có thể bắt đầu lập trình trên mạch Microbit để nhận dữ liệu điều khiển hoặc gửi dữ liệu lên server. Tuy nhiên, trước khi bắt đầu việc lập trình, chúng ta cần phải ghi lại một số thông tin quan trọng sau đây.

1.1 Username và Active Key

Khá giống ý nghĩa với API Key trên server ThingSpeak, 2 thông tin này sẽ được mạch Microbit sử dụng để đăng nhập vào server Adafruit IO. Từ trang chủ Adafruit, thông tin này được tìm thấy ở phần **My Key**, như minh họa sau đây:

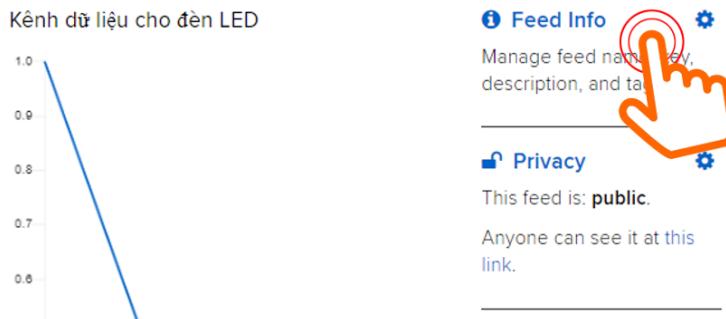


Hình 10.1: Thông tin để xác thực tài khoản

Với giao diện hiện ra khi nhấn vào **My Key**, chúng ta cần lưu lại thông tin về **Username** và **Active Key**.

1.2 Key của feed

Tiếp theo, bạn cần kiểm tra Feed dữ liệu khi truy xuất nó dưới dạng **Active Key**. Tên này sẽ được sinh ra tự động bởi hệ thống, và khác với tên chúng ta đặt khi tạo mới một Feed ở bài trước. Để truy xuất thông tin này, bạn chọn và **Feeds** trên thanh công cụ và nhấp chuột vào Feed dữ liệu đã tạo, trong hướng dẫn này là **BBC_LED**, giao diện sau đây sẽ hiện ra:



Hình 10.2: Kiểm tra thông tin của Feed

Sau khi nhấp vào mục **Feed Info**, một giao diện nữa sẽ hiện ra. Tại đây, chúng ta sẽ có tên của kênh dữ liệu để sử dụng cho phần lập trình, như minh họa ở hình bên dưới:

The form is titled 'Create a new Feed'. It has a 'Name' field containing 'BBC_LED' and a 'Key' field containing 'bbc-led'. Below the key field is a note: 'Changing the key will change API URLs and MQTT subscription topics. The only characters we permit are lower case english letters.'

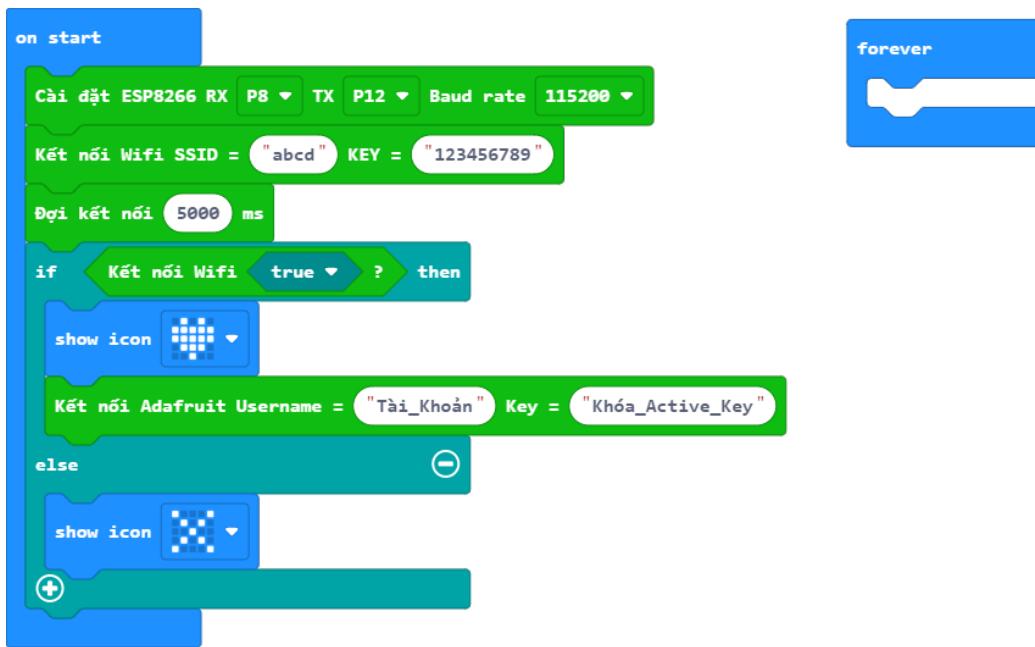
Hình 10.3: Tên truy cập của Feed dữ liệu

Bạn đọc cần làm tương tự để lưu lại thông tin cho Feed lưu giữ thông tin nhiệt độ đã tạo ra ở bài trước. Như vậy, các thông tin cần lưu lại trước khi lập trình, bao gồm tên Feed ở dạng truy cập bằng khóa (thường là tên của Feed, nhưng ở dạng viết thường), Username và Active Key. Các thông tin này sẽ được sử dụng khi hiện thực chương trình trên mạch Microbit.

2 Chương trình trên Microbit

2.3 Kết nối với Adafruit IO

Các tác vụ này cần được thực hiện khi khởi động hệ thống. Do đó, phần kết nối với server Adafruit sẽ được hiện thực trong phần **on start** của mạch Microbit. Sau khi kết nối vào mạng Internet, chúng ta sẽ sử dụng 2 thông tin là Username và Active Key để kết nối vào server. Chương trình gợi ý cho phần này như sau:



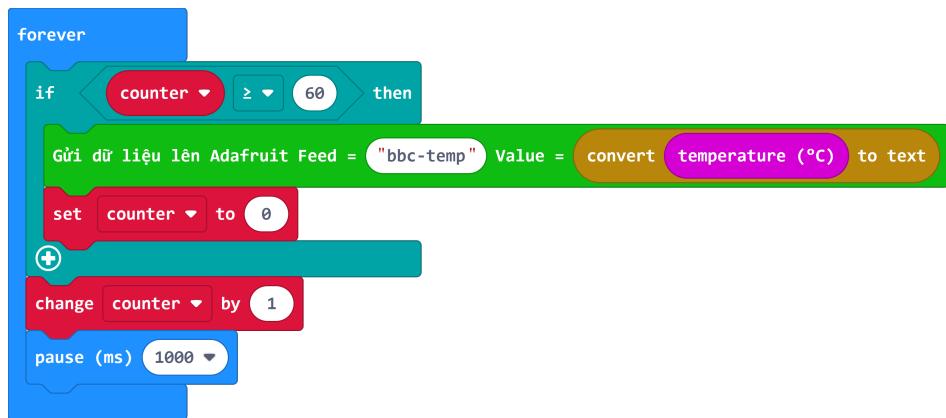
Hình 10.4: Kết nối WiFi và Adafruit IO

Trong chương trình trên, bạn đọc cần thay thế 2 thông tin **Tài_Khoản** và **Active_Key** cho phù hợp với thông tin của mình. Đây là 2 thông tin dùng để đăng nhập vào Server trước khi có thể gửi dữ liệu hoặc nhận dữ liệu từ Server.

2.4 Gửi dữ liệu lên Server

Đây là tính năng thông dụng bậc nhất cho các ứng dụng về quan trắc dựa trên nền tảng kết nối vạn vật. Thông tin đọc được từ các cảm biến được gửi định kì lên server. Tùy vào nhu cầu của ứng dụng, tuy nhiên thông thường chu kì này thường tính theo phút, chẳng hạn như 1 phút, là 60 giây.

Tuy nhiên, chúng ta lại không thể sử dụng câu lệnh đợi với thời gian dài trong chương trình. Điều này sẽ làm chậm tín hiệu gửi từ server xuống mạch Microbit. Do đó, chúng ta sẽ chỉ sử dụng một câu lệnh đợi có thời gian ngắn là **pause(1000)** và sử dụng thêm biến số để tạo ra hiệu ứng đợi 60 giây (1 phút). Cấu trúc chương trình gửi dữ liệu lên server sau mỗi phút sẽ như sau:



Hình 10.5: Kết nối WiFi và Adafruit IO

Do dữ liệu được gửi lên server định kì, tính năng này sẽ được hiện thực trong khối **forever**. Một biến **counter** được tạo ra để hiện thực tính năng 60 giây gửi dữ liệu lên server.

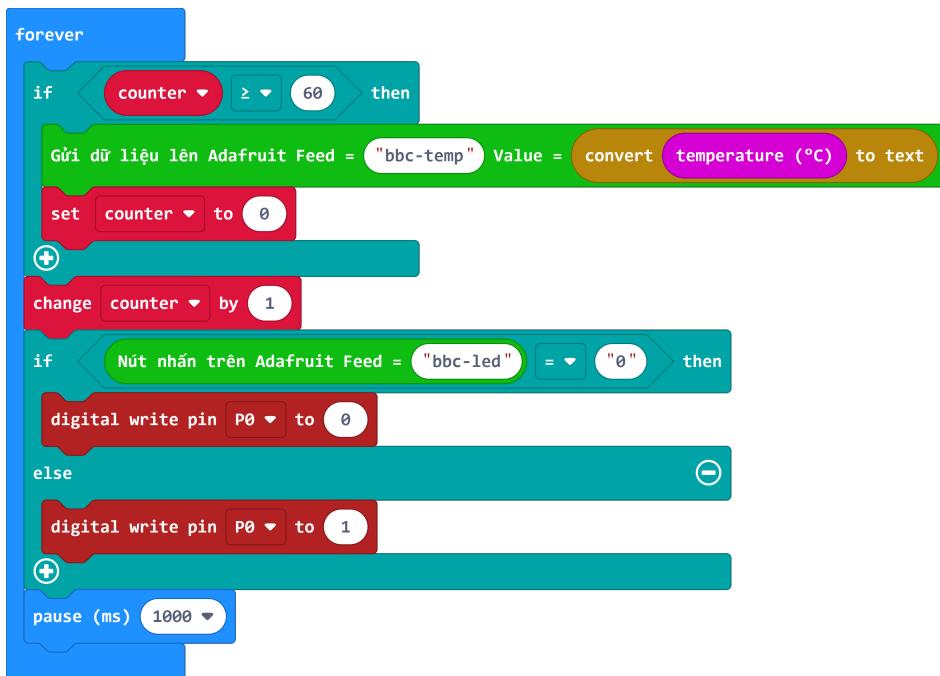
Tiếp theo, khóa **key** của feed dữ liệu sẽ được sử dụng trong câu lệnh **Gửi dữ liệu lên Adafruit Feed**. Bạn đọc cần phải hết sức chú ý thông tin này, nó không phải là tên của Feed.

Cuối cùng, dữ liệu gửi lên server là dạng kiểu chuỗi. Do đó, dữ liệu đọc từ cảm biến, đa phần là kiểu số, cần phải được chuyển đổi bằng câu lệnh **convert to text** trong mục **Advanced, Text**.

Cấu trúc chương trình bày ở trên gần như là kiến trúc bắt buộc để hiện thực các dự án lớn trên Microbit. Chúng ta sẽ chỉ có 1 câu lệnh đợi duy nhất trong khối forever. Sau đó, bắt kì tính năng nào cũng sẽ được hiện thực bằng cách khai báo thêm biến số và sử dụng câu lệnh **if** một cách hiện quả nhất.

2.5 Nhận dữ liệu từ server

Tính năng này là điểm nổi bật của Adafruit IO so với ThingSpeak. Khi chúng ta nhấn một nút trên Dashboard, dữ liệu của nó sẽ được mạch Microbit nhận ra và thực hiện một chức năng nào đó. Chúng ta sẽ sử dụng câu lệnh **if** để định kì kiểm tra dữ liệu của một nút nhấn trên server với câu lệnh được hỗ trợ sẵn trong mục **ESP8266_IoT**. Chương trình minh họa lần này sẽ như sau:



Hình 10.6: Kiểm tra nút nhấn trên Dashboard

Một lần nữa, dữ liệu từ server là dạng chuỗi, nên phép toán so sánh trong câu lệnh if sẽ là so sánh chuỗi. Dữ liệu từ nút nhấn có 2 trạng thái, tùy thuộc vào việc bạn thiết lập cho nó khi tạo ra nút nhấn này trên giao diện. Trong hướng dẫn này, chúng ta đang có 1 nút nhấn với 2 giá trị là **0 khi tắt và 1 khi mở**.

Bên trong câu lệnh if, chúng ta thực hiện việc bật tắt một động cơ thông qua công tắc điện tử (thường được gọi là Relay) bằng câu lệnh đơn giản là **digital write pin**. Câu lệnh trong hướng dẫn này chỉ mang tính chất minh họa. Tùy vào kết nối phần cứng trong hệ thống mà bạn đọc cần chọn lại chân cho đúng.

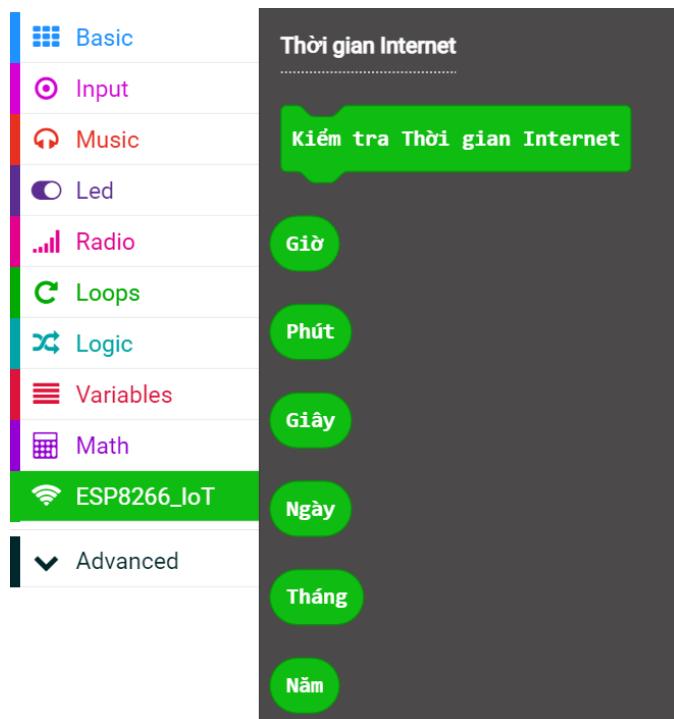
Một lần nữa, key của feed nút nhấn (bbc-led) được sử dụng để lập trình, thay vì tên của nó. Bạn đọc cần phải hết sức lưu ý về thông tin này.

Chương trình được chia sẻ ở đường dẫn sau đây:

https://makecode.microbit.org/_3dAaaVCdP5Ta

3 Đồng hồ Internet

Chức năng này được chúng tôi trình bày thêm để làm phong phú hơn cho các ứng dụng liên quan đến thời gian hiện tại (ngày tháng năm, giờ và phút). Với khả năng kết nối Internet, hệ thống của chúng ta có khả năng biết chính xác thời gian hiện tại. Đây cũng là nguyên lý cho tính năng cập nhật thời gian tự động đổi với các thiết bị thông minh hiện tại, như điện thoại, máy tính bảng và cả máy tính cá nhân.



Hình 10.7: Các khối lệnh liên quan tới thời gian

Trước khi sử dụng các khối lệnh liên quan đến thời gian, bạn đọc cần thực hiện câu lệnh đầu tiên là **Kiểm tra thời gian Internet**. Sau đó, các thông tin về thời gian hiện tại sẽ được cập nhật trong các khối lệnh tương ứng. Với sự hỗ trợ của đồng hồ Internet, các ứng dụng như tưới cây đúng giờ, đánh trống đúng giờ có thể được thực hiện rất dễ dàng và thuận tiện.

Ưu điểm cuối cùng, nhưng cũng là nhược điểm của hệ thống giờ Internet, là nó chỉ cần có mạng là cập nhật được thời gian. Cho dù hệ thống mất nguồn cũng không là vấn đề. Sau khi khởi động lại, kết nối lại vào mạng là thông tin thời gian sẽ tự động cập nhật lại.

4 Câu hỏi ôn tập

1. Thông tin cần cho việc kết nối với Adafruit IO là gì?
 - A. Tên và mật khẩu của mạng WIFI
 - B. Tài khoản Username và Active Key
 - C. Tên Feed
 - D. Tất cả đều đúng
2. Để gửi dữ liệu lên Feed Adafruit, thông tin nào sau đây sẽ được sử dụng?
 - A. Tên và mật khẩu của mạng WIFI
 - B. Tài khoản Username và Active Key
 - C. Tên Feed
 - D. Key của Feed
3. Để kiểm tra dữ liệu của nút nhấn trên Adafruit, thông tin nào sau đây sẽ được sử dụng trong câu lệnh nếu?
 - A. Tên và mật khẩu của mạng WIFI
 - B. Tài khoản Username và Active Key
 - C. Tên Feed
 - D. Key của Feed
4. Dữ liệu từ nút nhấn có kiểu dữ liệu là gì?
 - A. Kiểu chuỗi
 - B. Kiểu số
 - C. Kiểu luận lý
 - D. Kiểu digital
5. Đồng hồ Internet hoạt động dựa vào nguyên lý nào?
 - A. Có kết nối Internet sẽ có giờ
 - B. Có điện sẽ có giờ
 - C. Tất cả đều đúng
 - D. Tất cả đều sai

Đáp án

1. B 2. D 3. D 4. A 5. A



CHƯƠNG 11

Giao tiếp tầm xa LoRa



1 Giới thiệu

Trong các ứng dụng dựa trên kết nối vạn vật, một đặc điểm vô cùng đặc trưng là các phần tử trong ứng dụng sẽ giao tiếp không dây để chia sẻ dữ liệu. Mặc dù mạch MicroBit đã có sẵn chức năng giao tiếp không dây (nhóm lệnh Radio), khoảng cách của nó khá hạn chế, khi chỉ đạt tầm 15m ở môi trường có vật cản và có thể lên được 50m trong môi trường không có vật cản.

Để có thể chia sẻ dữ liệu xa hơn, có 2 cách để giải quyết vấn đề này. Một là giao tiếp qua các nốt trung gian, được đặt ở giữa 2 nốt đầu cuối. Giải pháp này sẽ tốn kém vì cần phải có nhiều nốt truyền dữ liệu. Thêm nữa, việc hiện thực sẽ không dễ dàng dành cho người bắt đầu. Giải pháp thứ 2 là tăng khoảng cách giao tiếp giữa 2 nốt với nhau, mà LoRa là một giải pháp. Với cơ chế mã hóa đặc biệt, LoRa (viết tắt của Long Range) có thể hỗ trợ giao tiếp từ xa lên đến hàng kilometer.

Với lợi thế của việc giao tiếp ở khoảng cách xa, LoRa có thể áp dụng để truyền dữ liệu trong một diện tích quan sát rộng, nhưng các dự án bảo vệ rừng, dự án liên quan đến quan trắc trong nông nghiệp hay nước thải, đều cần khoảng cách giao tiếp xa.

Trong bài hướng dẫn này, chúng tôi sẽ giới thiệu một thiết bị LoRa tích hợp khá phổ biến, có tên là SX1278. Hình ảnh của thiết bị này được minh họa như bên dưới:



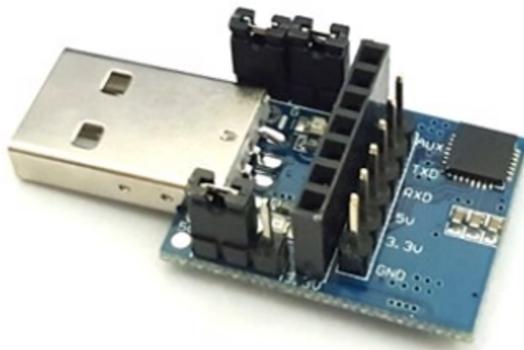
Hình 11.1: Thiết bị tích hợp LoRa SX1278

Tuy nhiên, trước khi có thể kết nối nó với mạch Microbit, chúng ta cần phải cấu hình thiết bị cho đúng. Cụ thể, chúng ta sẽ sử dụng máy tính, để cấu hình nó như 1 kênh kết nối UART. Khi việc cấu hình thành công, thiết bị này mới có thể kết nối với mạch Microbit để lập trình gửi nhận dữ liệu LoRa. Các mục tiêu chính của bài hướng dẫn này như sau:

- Kết nối SX1278 với máy tính
- Cấu hình SX1278 bằng phần mềm trên máy tính
- Kiểm tra việc gửi nhận dữ liệu trên máy tính

2 Kết nối SX1278 với máy tính

Để có thể kết nối với máy tính, chúng ta sẽ cần một mạch chuyển đổi từ LoRa sang USB, gọi là **mạch chuyển giao tiếp USB UART Lora SX1278**, cũng khá thông dụng trên thị trường, có hình ảnh như bên dưới:



Hình 11.2: Mạch chuyển giao tiếp USB UART Lora SX1278

Mặc định, thiết bị này có 2 jumper cầu hình. Khi kết nối với mạch LoRa SX1278 cho mục đích cầu hình bằng máy tính, **chúng ta phải gỡ 2 jumper này ra**, như minh họa ở hình bên dưới:



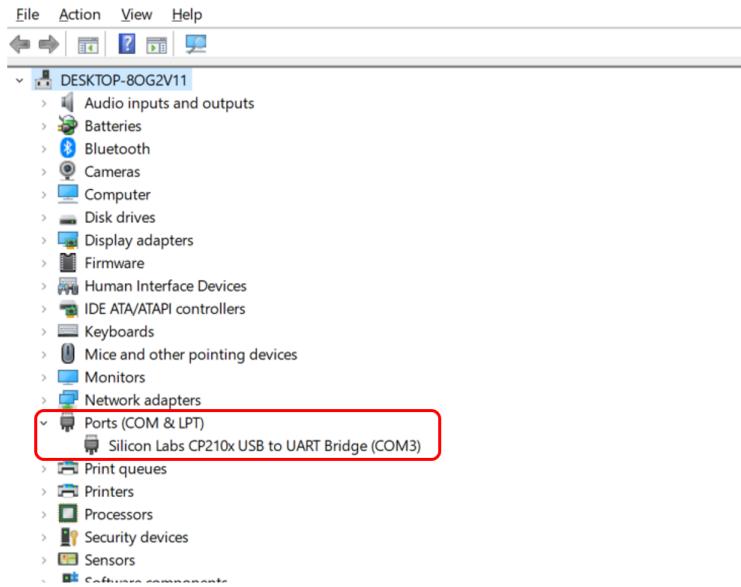
Hình 11.3: Kết nối mạch chuyển đổi và thiết bị LoRa SX1278

Sau khi 2 jumper phía trên đã được gỡ ra, chúng ta có thể kết nối nó với bất kì cổng USB nào của máy tính. Với cái máy tính thông dụng hiện tại, nó sẽ được tự động nhận thiết bị USB này. Trong trường hợp các máy cũ hơn, bạn đọc cần phải tự cài driver cho máy của mình bằng cách tìm từ khóa **Driver USB to UART CP2102**.

3 Cấu hình giao tiếp LoRa

Mục tiêu của phần này, là cấu hình SX1278 thành một thiết bị UART không dây LoRa. Việc lập trình nó từ mạch Microbit chỉ đơn giản là giao tiếp nối tiếp bằng các câu lệnh trong nhóm Serial. LoRa SX1278 sẽ tự động chuyển đổi điều chế tín hiệu, để có thể truyền dữ liệu không dây đến thiết bị nhận. Phía đầu nhận còn lại, quy trình đọc dữ liệu cũng chỉ đơn giản là sử dụng khối lệnh nhận dữ liệu trong nhóm Serial. Các bước để cấu hình thiết bị này được trình bày chi tiết như sau.

Bước 1: Cắm thiết bị ở phần trên vào cổng USB của máy tính. Kiểm tra xem cổng COM đang kết nối với thiết bị trong Device Manager của máy tính, như minh họa ở hình bên dưới:



Hình 11.4: Kết nối mạch chuyển đổi và thiết bị LoRa SX1278

Như hình minh họa ở trên, thiết bị LoRa đang được kết nối với COM3.

Bước 2: Tải phần mềm cấu hình từ đường dẫn sau đây:

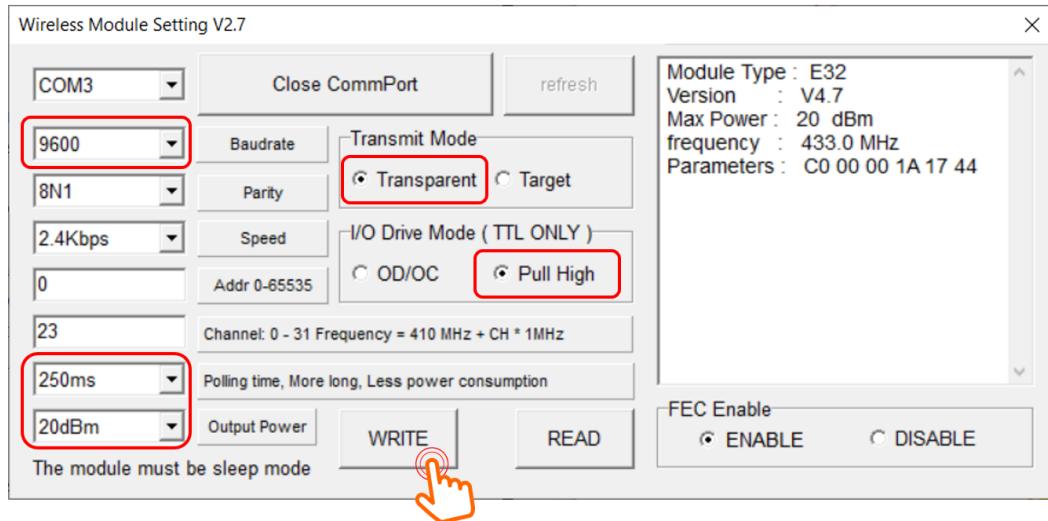
<https://ubc.sgp1.cdn.digitaloceanspaces.com/DARIU/LoRaSetting.exe>



Hình 11.5: Mở cổng kết nối để cấu hình

Khởi động phần mềm, lựa chọn **COM3** và chọn tiếp vào nút **Open CommPort**. Cuối cùng, chọn tiếp vào nút **Press to Read**.

Bước 3: Với giao diện hiện ra bên dưới, bạn đọc có thể xem theo để lựa chọn cho đúng.



Hình 11.6: Cài đặt các thông tin cấu hình

Trong các thông tin trên, quan trọng nhất là tốc độ giao tiếp UART (mục **Baudrate**), mặc định là 9600. Đổi với công suất truyền (mục **Output Power**), giá trị càng lớn sẽ gửi được càng xa. Cuối cùng là các cấu hình trong phần **Transmit Mode**, với 2 lựa chọn là **Transparent** và **Pull High**. Sau khi đã kết thúc việc cấu hình, nhấn nút **WRITE** để ghi lại thông tin cần thiết.

4 Kiểm tra việc gửi nhận trên máy tính

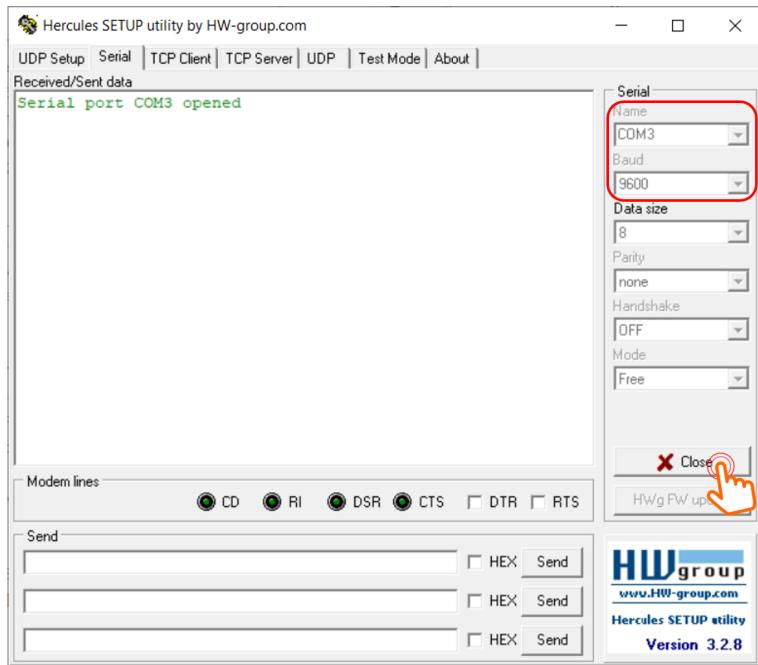
Trước khi tiến hành việc lập trình với mạch Microbit, chúng ta nên kiểm tra việc gửi nhận LoRa bằng máy tính, thông qua kết nối UART từ cổng USB. Tuy nhiên, để kiểm tra việc gửi nhận bằng máy tính, chúng ta cần **2 Mạch chuyển giao tiếp USB UART Lora SX1278**, như đã trình bày ở Hình 11.2. Nếu như không có đủ 2 mạch chuyển này, các bước tiếp theo sẽ không thể thực hiện được. Trình tự kiểm tra được trình bày từng bước như bên dưới.

Bước 1: 2 jumper cấu hình cần được gắn lại, để thiết bị LoRa có thể hoạt động. Cắm **một thiết bị vào máy tính** và kiểm tra cổng COM đang kết nối trong Device Manager.

Bước 2: Tải và mở phần mềm sau đây:

<https://ubc.sgp1.cdn.digitaloceanspaces.com/DARIU/Hercules.exe>

Sau khi mở phần mềm lên, chuyển sang tab **Serial**, lựa chọn cổng COM, tốc độ giao tiếp (9600) và sau đó nhấn nút **Open**. Khi việc kết nối là thành công, nút Open sẽ chuyển thành Close, và sẽ xuất hiện thông báo **Serial port COM opened**, như minh họa ở hình bên dưới.



Hình 11.7: Cấu hình cổng COM kết nối với LoRa

Bước 3: Đổi với mạch LoRa thứ 2, bạn sẽ cắm nó vào cổng USB thứ 2 của máy tính. Các jumper cấu hình cũng được gắn vào đầy đủ như bước 1. Nếu như máy tính không có đủ USB, bạn có thể cắm nó vào máy tính khác để kiểm tra. Chúng ta cũng ghi nhận lại tên cổng COM kết nối với thiết bị.

Bước 4: Khởi động một lần nữa phần mềm ở bước 2. Nếu trên cùng 1 máy tính, chúng ta sẽ có 2 cửa sổ khác nhau cho phần mềm này. Hiển nhiên, việc lựa chọn cổng COM lần này, sẽ dành cho thiết bị thứ 2.

Sau khi kết nối thành công, chúng ta đã có thể nhập thông tin và nhấn nút **Send**. Phần mềm thứ 2 sẽ nhận được dữ liệu và hiển thị lên màn hình. Chúng ta cũng nên kiểm tra chiều ngược lại để đảm bảo cả 2 thiết bị đều có khả năng gửi nhận dữ liệu, trước khi tích hợp vào việc lập trình ở bài tiếp theo.

5 Câu hỏi ôn tập

1. Đối với những ứng dụng có khoảng cách xa trên 1km, giao tiếp nào sau đây là phù hợp?
 - A. Radio
 - B. Wifi
 - C. Bluetooth
 - D. LoRa
2. Đối với các ứng dụng có khoảng cách ngắn dưới 15m, giao tiếp nào sau đây là phù hợp nhất với mạch Microbit?
 - A. Radio
 - B. Wifi
 - C. Bluetooth
 - D. LoRa
3. Chuẩn giao tiếp với thiết bị LoRa SX1278 là gì?
 - A. USB
 - B. USB UART
 - C. PS2
 - D. SPI
4. Cấu hình jumper cho M0 và M1 của thiết bị SX1278 được gỡ ra khi nào?
 - A. Khi mạch cần hoạt động gửi nhận dữ liệu
 - B. Khi mạch cần cấu hình thông số hoạt động
 - C. Khi cần nạp chương trình microbit cho mạch
 - D. Tất cả đều đúng
5. Cấu hình jumper cho M0 và M1 của thiết bị SX1278 được gắn khi nào?
 - A. Khi mạch cần hoạt động gửi nhận dữ liệu
 - B. Khi mạch cần cấu hình thông số hoạt động
 - C. Khi cần nạp chương trình microbit cho mạch
 - D. Tất cả đều đúng
6. Tốc độ (Baudrate) cho giao tiếp UART nào là phù hợp?
 - A. 600
 - B. 900
 - C. 9000
 - D. 9600
7. Trong chế độ Trasmit Mode, cấu hình nào là chính xác cho SX1278?
 - A. Transparent
 - B. Target
 - C. OD/OC
 - D. Tất cả đều đúng

Đáp án

1. D 2. A 3. B 4. B 5. A 6. D 7. A

CHƯƠNG 12

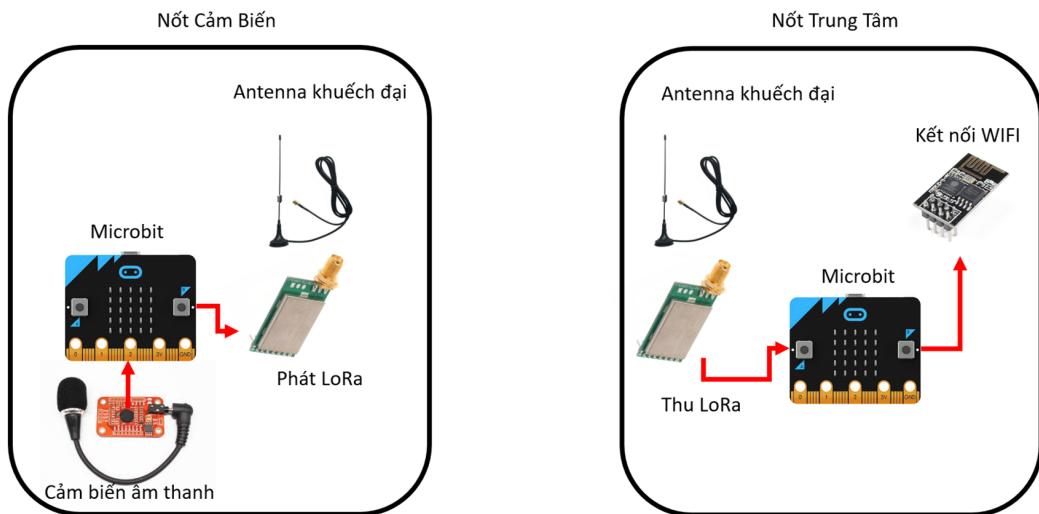
Giao tiếp tầm xa LoRa



1 Giới thiệu

Sau khi cấu hình thiết bị LoRa SX1278, nó sẽ được kết nối với mạch Microbit theo chuẩn kết nối UART. Đây là chuẩn kết nối đơn giản nhất giữa các thiết bị phần cứng với nhau. Bên cạnh 4 chân kết nối cơ bản, bao gồm VCC-GND-Tx-Rx, các chân cấu hình của thiết bị LoRa (chân M0 và M1) cần nối với nguồn VCC.

Trong bài hướng dẫn này, chúng ta sẽ sử dụng 2 mạch Microbit, mỗi mạch được kết nối với thiết bị LoRa SX1278 để gửi nhận dữ liệu. Kiến trúc hệ thống trong bài hướng dẫn này sẽ như sau:



Hình 12.1: Gửi nhận dữ liệu giữa 2 thiết bị LoRa

Nốt cảm biến, thông thường đóng vai trò là nốt gửi dữ liệu. Thông tin cảm biến có thể là thông tin môi trường, được gửi định kì, hoặc một thông tin sự kiện (chẳng hạn như cháy rừng) sẽ được gửi khi có sự kiện xảy ra. Nốt cảm biến có thể kết nối với nhiều cảm biến khác nhau trong mô hình này.

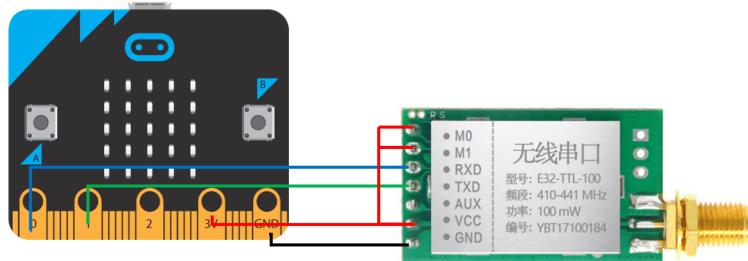
Trong khi đó, nốt trung tâm, thường là nốt nhận dữ liệu, sẽ không có cảm biến. Tuy nhiên, nó sẽ thường kết nối với một thiết bị hỗ trợ kết nối mạng Internet, chẳng hạn như là mạch ESP8266. Vấn đề ở nốt trung tâm, là nó sẽ sử dụng một giao tiếp nối tiếp cho cả 2 thiết bị, là LoRa SX1278 và ESP8266. Do vậy, các câu lệnh ở nốt trung tâm cần phải được tổ chức hợp lý để luân chuyển qua lại giữa 2 thiết bị cùng sử dụng giao tiếp UART này.

Trong bài hướng dẫn này, chúng tôi hướng dẫn việc gửi một giá trị số từ nốt cảm biến đến nốt trung tâm. Việc thay đổi giá trị số này thành thông tin cảm biến, sẽ để dành cho bạn đọc tự tùy chỉnh theo ứng dụng của mình. Một gợi ý cho việc chuyển đổi giao tiếp UART ở nốt trung tâm cũng sẽ được trình bày. Các mục tiêu chính trong bài hướng dẫn này như sau:

- Kết nối phần cứng giữa Microbit và SX1278
- Gửi và nhận dữ liệu LoRa
- Chuyển đổi cổng kết nối UART ở nốt trung tâm

2 Kết nối phần cứng với MicroBit

Quan trọng nhất trong giao tiếp với SX1278 là 2 chân kết nối với RXD và TXD. Chân kết nối với RXD sẽ là chân **gửi dữ liệu** và chân kết nối với TXD sẽ là chân **nhận dữ liệu** không dây qua giao tiếp LoRa. Một kết nối được minh họa như sau:



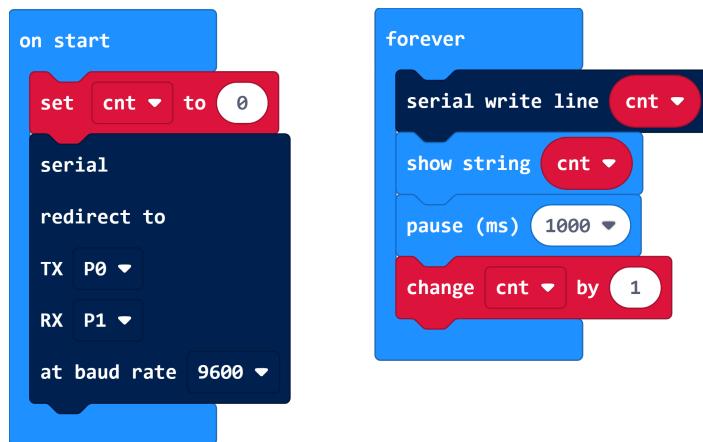
Hình 12.2: Thiết bị tích hợp LoRa SX1278

Cần lưu ý là 2 chân M0 và M1 của SX1278 cần được nối lên nguồn (3V) thì thiết bị này mới có thể hoạt động được. Đây là kết nối chéo, nên chân P0 của mạch MicroBit sẽ đóng vai trò là chân gửi (nối với RXD của SX1278) và chân P1 sẽ đóng vai trò là chân nhận (nối với TXD của SX1278).

3 Gửi nhận dữ liệu qua LoRa

3.1 Gửi dữ liệu

Chúng ta sẽ hiện thực chương trình gửi dữ liệu cho nốt cảm biến trước. **Dữ liệu sẽ định kì gửi mỗi giây một lần**. Một chương trình ví dụ cho việc gửi dữ liệu không dây qua kênh LoRa như sau:



Hình 12.3: Chương trình gửi dữ liệu LoRa.

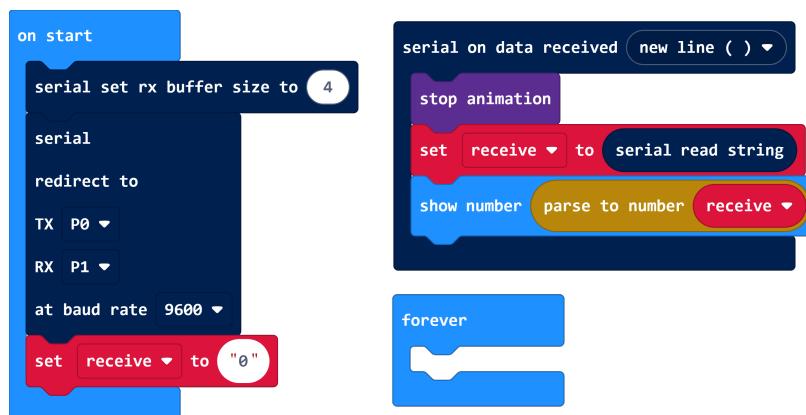
Một biến đếm, có tên là **cnt** được tạo ra và tăng dần sau mỗi lần gửi (cách đều nhau 1s). Đầu tiên, trong khối **on start**, biến cnt được gán ban đầu bằng 0. Tiếp theo, là khai báo cổng Serial để kết nối với khôi LoRa, ở đây là P0 và P1. Với khai báo này, chân P0 của Microbit sẽ nối với TXD của LoRa, chân P1 sẽ nối với RXD của LoRa.

Việc nối chân chéo giữa 2 thiết bị cần phải được kiểm tra kĩ lưỡng, đảm bảo sự chính xác giữa lập trình và kết nối mạch điện.

Cuối cùng, việc gửi dữ liệu sẽ được hiện thực định kì trong khối **forever**. Câu lệnh **serial write line** sẽ được dùng để gửi thông tin ra thiết bị LoRa SX1278. Sau đó, thông tin sẽ được mã hóa và gửi đi không dây. Cần lưu ý là khi dùng câu lệnh này, thông tin gửi đi sẽ được tự động thêm kí tự xuống dòng (kí tự **new line**).

3.2 Nhận dữ liệu

Phần cấu hình cho nốt nhận (trong câu lệnh **on start**) cũng khá tương tự với nốt gửi. Tuy nhiên, câu lệnh set rx buffer size cần phải định thiết lập ban đầu để tránh hiện tượng tràn vùng nhớ trong mạch Microbit. Tùy kích thước dữ liệu bên nốt gửi, bên nốt nhận sẽ khai báo kích thước lớn hơn một chút là được. Trong ví dụ này, nốt gửi đang gửi 1 con số, nên kích thước của vùng nhớ tối đa là 4 bytes mà thôi. Chương trình cho nốt nhận được gửi ý như bên dưới:



Hình 12.4: Chương trình nhận dữ liệu LoRa.

Với câu lệnh gửi có kí tự xuống dòng, chúng ta phải chọn lựa cho tương ứng với câu lệnh nhận. Do trong khối nhận dữ liệu, chúng ta hiển thị ra đèn để kiểm tra, nên câu lệnh **stop animation** được sử dụng thêm để tránh tình trạng nốt nhận dữ liệu khi đang hiển thị. Trong trường hợp gửi dữ liệu thưa (30 giây một lần), câu lệnh này có thể không cần phải sử dụng.

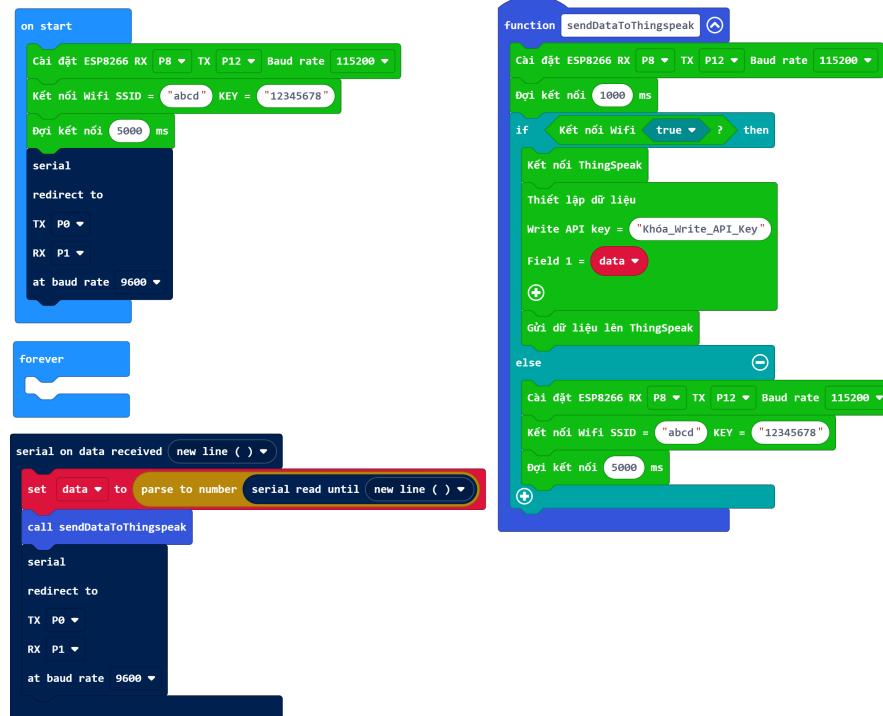
4 Đổi cổng kết nối UART

Nhu cầu này xảy ra khi trong một hệ thống kết nối với MicroBit, chúng ta có trên 2 thiết bị cùng kết nối theo chuẩn UART. Điều này sẽ rất thường xuyên xảy ra, do UART là chuẩn kết nối đơn giản mà nhiều thiết bị sẽ hỗ trợ, từ kết nối Wifi, Bluetooth cho tới GPS hoặc thậm chí là thiết bị gửi nhận tin nhắn SMS.

Như đã trình bày ở phần giới thiệu, nốt trung tâm sẽ có thể kết nối cùng lúc với LoRa SX1278 để nhận dữ liệu từ một nốt cảm biến. Sau đó, dữ liệu này sẽ được gửi lên ThingSpeak nhờ kết nối Wifi. Do đó, chúng ta phải kéo léo xử lý phần cấu hình kết nối UART để tránh bị đụng độ. Cụ thể hơn, chúng ta phải cấu hình lại ESP8266

kết nối vào chân P8 và P12 mỗi khi gửi dữ liệu lên ThingSpeak và cấu hình cho LoRa là chân P0 và P1 trước khi muốn gửi dữ liệu LoRa.

Cuối cùng, để đảm bảo cho mọi tác vụ trở nên ổn định, mỗi lần cấu hình lại, chúng ta cũng nên thêm 1 khoảng thời gian đợi khoảng 1 giây. Điều này cũng không ảnh hưởng nhiều đến hiệu suất của hệ thống vì các ứng dụng quan trắc thường cũng không đòi hỏi việc gửi dữ liệu liên tục. Một chương trình gợi ý cho việc đổi cổng kết nối giữa ESP8266 và LoRa như sau:



Hình 12.5: Chuyển đổi kết nối UART trên MicroBit

Trong chương trình trên, trong khối **on start** chúng ta sẽ ưu tiên kết nối Wifi trước. Sau đó, chương trình đổi cấu hình UART qua thiết bị LoRa SX1278 để chờ nhận dữ liệu từ LoRa. Khi có dữ liệu từ LoRa, chương trình sẽ xử lý dữ liệu này: chuyển đổi nó qua dữ liệu kiểu số trước khi gọi hàm **sendDataToThingSpeak**. Sau khi kết thúc các tác vụ liên quan đến ESP8266, cổng kết nối UART sẽ chuyển về lại P0 và P1 để chờ nhận dữ liệu từ LoRa. Chương trình được chia sẻ ở đường dẫn sau đây:

https://makecode.microbit.org/_ehqh4X3TpaYM

5 Câu hỏi ôn tập

1. Để quan trắc trong 1 diện tích rộng (tầm km), nhu cầu gửi dữ liệu trong khoảng cách xa thì ta nên sử dụng công nghệ nào?
 - A. Wifi
 - B. 3G/4G/5G
 - C. LoRa
 - D. Bluetooth
2. Để gửi dữ liệu không dây qua LoRa, mạch Microbit sử dụng kĩ thuật gì?
 - A. Giao tiếp LoRa
 - B. Giao tiếp nối tiếp UART
 - C. Giao tiếp nối tiếp SPI
 - D. Giao tiếp I2C
3. Nếu nhiều giao tiếp UART với nhiều thiết bị. Phát biểu nào sau đây là đúng?
 - A. Mạch Microbit có thể giao tiếp đồng thời với nhiều thiết bị
 - B. Tại một thời điểm, chỉ có thể giao tiếp được với 1 thiết bị
 - C. Microbit không hỗ trợ giao tiếp UART
 - D. Tất cả đều sai
4. Để đổi kênh giao tiếp UART, câu lệnh nào sau đây sẽ được sử dụng?
 - A. serial update
 - B. serial redirect to
 - C. serial config
 - D. serial setup
5. Để nhận dữ liệu từ kênh truyền UART, câu lệnh nào được sử dụng?
 - A. serial on received
 - B. serial on data
 - C. serial on data received
 - D. serial on data received string
6. Câu lệnh để thiết lập kích thước vùng nhớ cho nốt nhận là:
 - A. set tx buffer size
 - B. set rx buffer length
 - C. set rx buffer size
 - D. set tx buffer length
7. Ví dụ kích thước vùng nhớ của nốt nhận được thiết lập bằng câu lệnh **set rx buffer size 4**, thì vùng nhớ tối đa là:
 - A. 4MB (Megabytes)
 - B. 4KB (Kilobytes)
 - C. 4 bytes
 - D. 4 bits

Đáp án

1. C 2. B 3. B 4. B 5. C 6. C 7. C

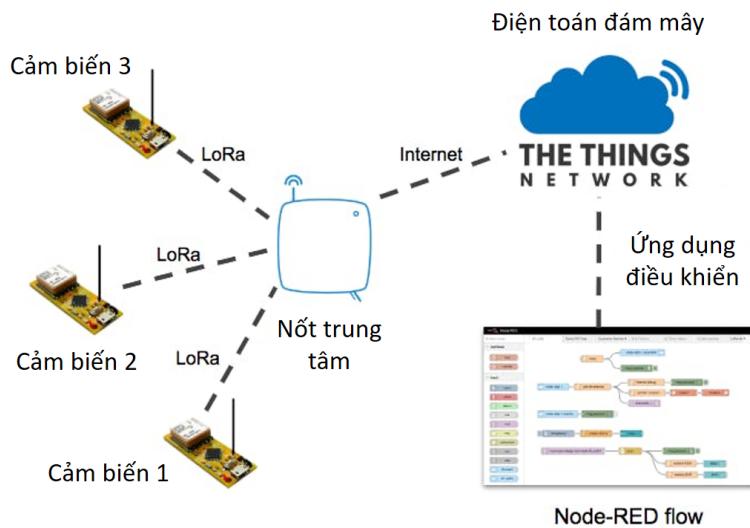
CHƯƠNG 13

Xây dựng mạng cảm biến LoRa



1 Giới thiệu

Một trong những ứng dụng lớn nhất của nền tảng kết nối vạn vật là mạng cảm biến không dây. Trong mô hình mạng này, sẽ không phải chỉ có 1 nốt cảm biến như minh họa ở bài trước, mà nhiều nốt cảm biến sẽ được phân bố xung quanh nốt trung tâm. Mỗi nốt cảm biến sẽ có dữ liệu riêng của nó, được gửi về nốt trung tâm thông qua giao tiếp không dây, trước khi dữ liệu này có thể được gửi lên server.



Hình 13.1: Mạng cảm biến không dây sử dụng LoRa

Khi có nhiều nốt cảm biến cùng gửi dữ liệu về nốt trung tâm, sẽ có nhiều vấn đề cần phải xử lý. Bỏ qua các vấn đề sâu về giao tiếp trong tín hiệu, chẳng hạn như dung độ dữ liệu chẳng hạn, một trong những vấn đề là làm sao nốt trung tâm biết được dữ liệu này có ý nghĩa là gì (nhiệt độ hay độ ẩm) và được gửi từ nốt nào. Để giải quyết vấn đề này, dữ liệu gửi đi cần phải có thêm thông tin, thì nốt trung tâm mới có thể xử lý được.

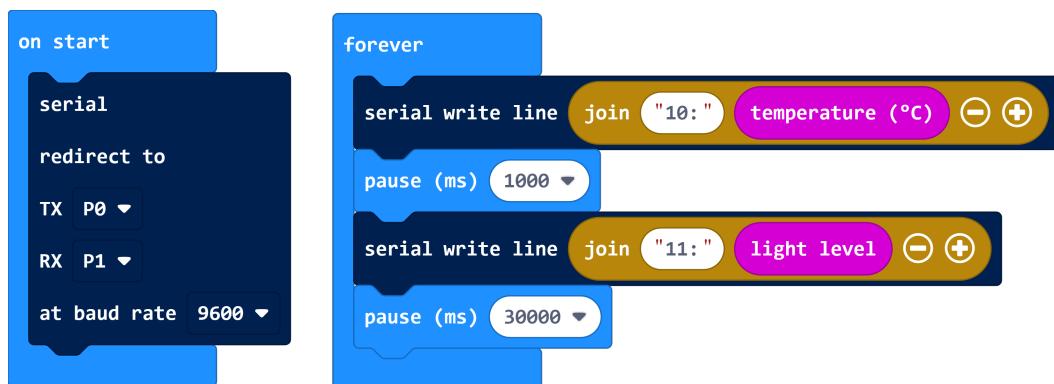
Để có thể minh họa điều này, chúng tôi giả sử rằng hệ thống có 2 nốt cảm biến, mỗi nốt có thể gửi 2 thông tin khác nhau là nhiệt độ và độ ẩm (từ cảm biến DHT11 chẳng hạn) qua giao tiếp LoRa đến nốt trung tâm. Chúng ta sẽ quy định trường thông tin (field) là một số có 2 chữ số. ví dụ 10 là nốt số 1, gửi thông tin nhiệt độ và 11 là nốt số 1, gửi thông tin độ ẩm. Như vậy, đối với nốt số 2, nó sẽ là 20 và 21. Thông tin field này sẽ được ghép với giá trị cảm biến, cách nhau bằng dấu hai chấm (:). Với quy ước như vậy, tại nốt trung tâm, nó sẽ nhận được các giá trị "10:30" hoặc "21:59", với ý nghĩa rằng thông tin nhiệt độ từ nốt 1 là 30°C và độ ẩm từ nốt 2 là 59%.

Với các ràng buộc như trên, sẽ không quá khó khăn để bạn đọc hiệu chỉnh lại chương trình cho nốt gửi. Tuy nhiên, ở nốt trung tâm, đóng vai trò nhận dữ liệu, sẽ cần phải xử lý để phân tích dữ liệu nhận được. Quy trình này sẽ không đơn giản, đặc biệt là khi phải hiện thực trên mạch Microbit. Trong bài hướng dẫn này, chúng tôi sẽ đưa ra một số giải pháp đơn giản cho việc gửi và nhận dữ liệu trên mạng không dây nhiều nốt nói chung, và LoRa nói riêng. Các mục tiêu trong bài này như sau:

- Cải tiến chương trình gửi dữ liệu
- Phân tách dữ liệu tại nốt nhận
- Tổng hợp dữ liệu và gửi lên server

2 Chương trình gửi dữ liệu

Trong chương trình của nốt gửi, chúng tôi chỉ minh họa bằng cảm biến có sẵn trên mạch Microbit, là nhiệt độ và cường độ ánh sáng. Trong từng ứng dụng, bạn đọc cần thay đổi thông tin này cho phù hợp. Các kĩ thuật đọc dữ liệu từ cảm biến tích hợp hay cảm biến ADC đã được trình bày ở các bài trước, sẽ không được trình bày lại ở bài này.



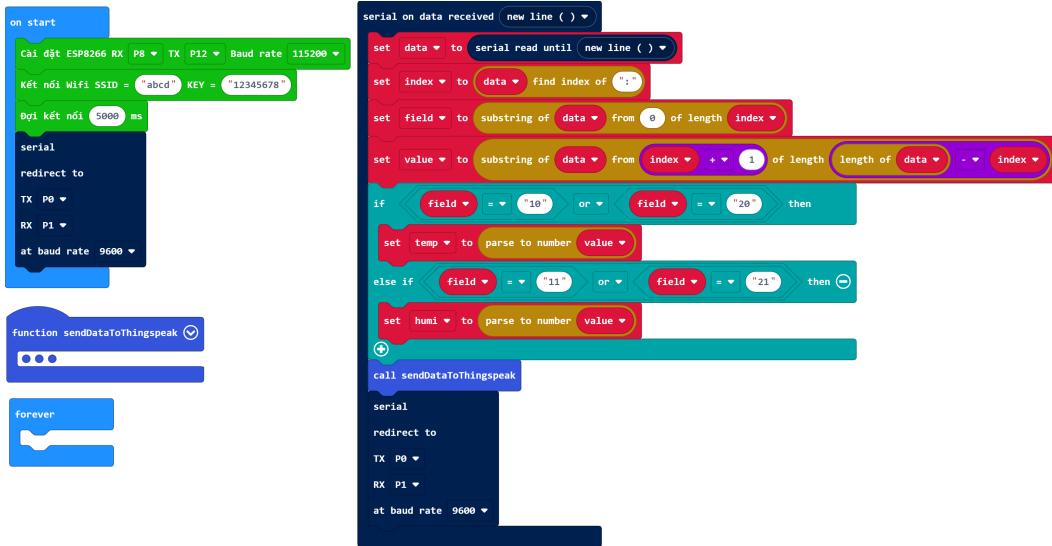
Hình 13.2: Dữ liệu được đóng gói theo định dạng trước khi gửi đi

Như chương trình gợi ý ở trên, chức năng chính khi xây dựng mạng cảm biến không dây đối với nốt gửi, là việc phải đóng gói dữ liệu làm sao cho đúng. Trong trường hợp một nốt có hơn 10 cảm biến khác nhau, bạn đọc có thể quy định trường thông tin field là một số có 3 chữ số chẳng hạn. Chương trình trên được chia sẻ ở đường dẫn sau đây, để bạn đọc tiện tham khảo:

https://makecode.microbit.org/_V9j8P42pPF9T

3 Chương trình nhận dữ liệu

Đối với nốt trung tâm (đôi khi còn được gọi là nốt gateway), đóng vai trò thu thập dữ liệu từ nhiều nốt cảm biến. Việc xử lý dữ liệu sẽ được hiện thực trong khôi nhận dữ liệu từ cổng giao tiếp nối tiếp. Ý tưởng ở đây là tìm vị trí dấu hai chấm, được sử dụng như là một kí tự phân tách thông tin. Sự phức tạp của chương trình ở chỗ tất cả các thao tác đều thao tác trên dữ liệu chuỗi, vốn là kiểu dữ liệu khá phức tạp trong các ngôn ngữ lập trình. Chương trình xử lý gợi ý như sau:



Hình 13.3: Phân tách dữ liệu nhận từ LoRa

Tùy theo giá trị của biến **field**, được sử dụng để lưu phần đầu của thông tin, chúng ta sẽ có những thao tác tương ứng bằng các câu lệnh **if**. Thực ra, việc xử lý tiếp theo vẫn còn có thể rất phức tạp, tùy thuộc vào yêu cầu của ứng dụng. Với 2 thông tin nhiệt độ khác nhau từ 2 nốt, chúng ta có thể tính giá trị trung bình, hoặc gửi lên server ở 2 đồ thị tách biệt. Trong chương trình minh họa ở trên, chúng tôi chỉ đơn giản là lưu nó vào 1 biến, có tên là **temp**. Bạn đọc có thể chủ động ở phần xử lý này theo mục đích thiết kế của mình. Chương trình được chia sẻ ở đường dẫn sau đây:

https://makecode.microbit.org/_FJbi7rP7sYxf

4 Câu hỏi ôn tập

1. Vấn đề trong mạng cảm biến là gì?
 - A. Nhiều nốt cùng gửi dữ liệu về nốt trung tâm
 - B. Nhiều thông tin khác nhau từ một nốt gửi về trung tâm
 - C. Có thể xảy ra đụng độ, mất dữ liệu
 - D. Tất cả các vấn đề trên
2. Kí tự nào sau đây có thể được dùng làm kí tự phân cách giữa 2 thông tin gửi từ nốt cảm biến?
 - A. Hai chấm (:)
 - B. Xuống dòng
 - C. Số 1
 - D. Tất cả đều có thể
3. Dữ liệu nhận được từ khôi lệnh serial on data received có kiểu là gì?
 - A. Dữ liệu số nguyên
 - B. Dữ liệu số thực
 - C. Dữ liệu chuỗi
 - D. Không xác định được
4. Câu lệnh dùng để tìm vị trí kí tự hai chấm trong 1 chuỗi là gì?
 - A. find index of
 - B. substring of
 - C. parse to number
 - D. Tất cả đều đúng
5. Câu lệnh dùng để lấy chuỗi con trong 1 chuỗi là gì?
 - A. find index of
 - B. substring of
 - C. parse to number
 - D. Tất cả đều sai
6. Câu lệnh để chuyển một chuỗi sang 1 số là gì?
 - A. find index of
 - B. substring of
 - C. parse to number
 - D. text to number
7. Khi gửi dữ liệu lên server ThingSpeak, các giá trị trong trường field có kiểu dữ liệu là gì?
 - A. Số nguyên hoặc số thực
 - B. Chỉ số nguyên mà thôi
 - C. Chuỗi
 - D. Dữ liệu nào cũng được

Đáp án

1. D 2. A 3. C 4. A 5. B 6. C 7. A