**Lập trình nhúng và Internet of Things (IoT)   
với Arduino IDE và ESP8266 Wemos D1 R2**

***Bài 1: Tìm hiểu tổng quan và thiết lập môi trường làm việc***

1. **Giới thiệu về Arduino và Arduino IDE**

Arduino là một board mạch vi xử lý, nhằm xây dựng các ứng dụng tương tác với nhau hoặc với môi trường được thuận lợi hơn. Phần cứng bao gồm một board mạch nguồn mở được thiết kế trên nền tảng vi xử lý AVR Atmel 8bit, hoặc ARM Atmel 32-bit. Những Model hiện tại được trang bị gồm 1 cổng giao tiếp USB, 6 chân đầu vào analog, 14 chân I/O kỹ thuật số tương thích với nhiều board mở rộng khác nhau.

Được giới thiệu vào năm 2005, Những nhà thiết kế của Arduino cố gắng mang đến một phương thức dễ dàng, không tốn kém cho những người yêu thích, sinh viên và giới chuyên nghiệp để tạo ra những thiết bị có khả năng tương tác với môi trường thông qua các cảm biến và các cơ cấu chấp hành. Những ví dụ phổ biến cho những người yêu thích mới bắt đầu bao gồm các robot đơn giản, điều khiển nhiệt độ và phát hiện chuyển động. Đi cùng với nó là một môi trường phát triển tích hợp (IDE) chạy trên các máy tính cá nhân thông thường và cho phép người dùng viết các chương trình cho Aduino bằng ngôn ngữ C hoặc C++.

Giá của các board Arduino dao động xung quanh €20 - $27, nếu được "làm giả" thì giá có thể giảm xuống thấp hơn $9. Các board Arduino có thể được đặt hàng ở dạng được lắp sẵn hoặc dưới dạng các kit tự-làm-lấy. Thông tin thiết kế phần cứng được cung cấp công khai để những ai muốn tự làm một mạch Arduino bằng tay có thể tự mình thực hiện được (mã nguồn mở). Người ta ước tính khoảng giữa năm 2011 có trên 300 ngàn mạch Arduino chính thức đã được sản xuất thương mại, và vào năm 2013 có khoảng 700 ngàn mạch chính thức đã được đưa tới tay người dùng.

## Lịch sử

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

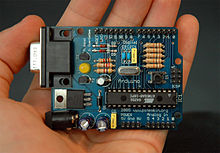
Lý thuyết phần cứng được đóng góp bởi một sinh viên người Colombia tên là Hernando Barragan. Sau khi nền tảng Wiring hoàn thành, các nhà nghiên cứu đã làm việc với nhau để giúp nó nhẹ hơn, rẻ hơn, và khả dụng đối với cộng đồng mã nguồn mở. Trường này cuối cùng bị đóng cửa, vì vậy các nhà nghiên cứu,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

Giá hiện tại của board mạch này dao động xung quanh $30 và được làm giả đến mức chỉ còn $9. Một mạch bắt chước đơn giản Arduino Mini Pro có lẽ được xuất phát từ Trung Quốc có giá rẻ hơn $4, đã trả phí bưu điện.

## Phần cứng

[](https://vi.wikipedia.org/wiki/T%E1%BA%ADp_tin:Arduino_top-1.jpg)

Một mạch Arduino Uno chính thức với các mô tả về các cổng I/O

[](https://vi.wikipedia.org/wiki/T%E1%BA%ADp_tin:Arduino316.jpg)

Một board Arduino đời đầu gồm một cổng giao tiếp RS-232 (góc phía trên-bên trái) và một chip vi xử lý Atmel ATmega8 (màu đen, nằm góc phải-phía dưới); 14 chân I/O số nằm ở phía trên và 6 chân analog đầu vào ở phía đáy.

Một mạch Arduino bao gồm một vi điều khiển AVR với nhiều linh kiện bổ sung giúp dễ dàng lập trình và có thể mở rộng với các mạch khác. Một khía cạnh quan trọng của Arduino là các kết nối tiêu chuẩn của nó, cho phép người dùng kết nối với CPU của board với các module thêm vào có thể dễ dàng chuyển đổi, được gọi là *shield*. Vài shield truyền thông với board Arduino trực tiếp thông qua các chân khách nhau, nhưng nhiều shield được định địa chỉ thông qua serial bus I²C-nhiều shield có thể được xếp chồng và sử dụng dưới dạng song song. Arduino chính thức thường sử dụng các dòng chip megaAVR, đặc biệt là ATmega8, ATmega168, ATmega328, ATmega1280, và ATmega2560. Một vài các bộ vi xử lý khác cũng được sử dụng bởi các mạch Aquino tương thích. Hầu hết các mạch gồm một bộ điều chỉnh tuyến tính 5V và một thạch anh dao động 16 MHz (hoặc bộ cộng hưởng ceramic trong một vài biến thể), mặc dù một vài thiết kế như LilyPad chạy tại 8 MHz và bỏ qua bộ điều chỉnh điện áp onboard do hạn chế về kích cỡ thiết bị. Một vi điều khiển Arduino cũng có thể được lập trình sẵn với một boot loader cho phép đơn giản là upload chương trình vào bộ nhớ flash on-chip, so với các thiết bị khác thường phải cần một bộ nạp bên ngoài. Điều này giúp cho việc sử dụng Arduino được trực tiếp hơn bằng cách cho phép sử dụng 1 máy tính gốc như là một bộ nạp chương trình.

Theo nguyên tắc, khi sử dụng ngăn xếp phần mềm Arduino, tất cả các board được lập trình thông qua một kết nối RS-232, nhưng cách thức thực hiện lại tùy thuộc vào đời phần cứng. Các board Serial Arduino có chứa một mạch chuyển đổi giữa RS232 sang TTL. Các board Arduino hiện tại được lập trình thông qua cổng USB, thực hiện thông qua chip chuyển đổi USB-to-serial như là FTDI FT232. Vài biến thể, như Arduino Mini và Boarduino không chính thức, sử dụng một board adapter hoặc cáp nối USB-to-serial có thể tháo rời được, Bluetooth hoặc các phương thức khác. (Khi sử dụng một công cụ lập trình vi điều khiển truyền thống thay vì ArduinoIDE, công cụ lập trình AVR ISP tiêu chuẩn sẽ được sử dụng.)

Board Arduino sẽ đưa ra hầu hết các chân I/O của vi điều khiển để sử dụng cho những mạch ngoài. Diecimila, Duemilanove, và bây giờ là Uno đưa ra 14 chân I/O kỹ thuật số, 6 trong số đó có thể tạo xung PWM (điều chế độ rộng xung) và 6 chân input analog, có thể được sử dụng như là 6 chân I/O số. Những chân này được thiết kế nằm phía trên mặt board, thông qua các header cái 0.10-inch (2.5 mm). Nhiều shield ứng dụng plug-in cũng được thương mại hóa. Các board Arduino Nano, và Arduino-compatible Bare Bones Board và Boarduino có thể cung cấp các chân header đực ở mặt trên của board dùng để cắm vào các breadboard.

Có nhiều biến thể như Arduino-compatible và Arduino-derived. Một vài trong số đó có chức năng tương đương với Arduino và có thể sử dụng để thay thế qua lại. Nhiều mở rộng cho Arduino được thực thiện bằng cách thêm vào các driver đầu ra, thường sử dụng trong các trường học để đơn giản hóa các cấu trúc của các 'con rệp' và các robot nhỏ. Những board khác thường tương đương về điện nhưng có thay đổi về hình dạng-đôi khi còn duy trì độ tương thích với các shield, đôi khi không. Vài biến thể sử dụng bộ vi xử lý hoàn toàn khác biệt, với các mức độ tương thích khác nhau.

## Các board chính thức

Phần cứng Arduino gốc được sản xuất bởi công ty Italy tên là Smart Projects. Một vài board dẫn xuất từ Arduino cũng được thiết kế bởi công ty của Mỹ tên là SparkFun Electronics. Sáu phiên bản phần cứng của Arduino cũng đã được sản xuất thương mại tính đến thời điểm hiện tại.

**Các board Arduino mẫu**

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên** | **Ảnh** | **Tên** | **Ảnh** |
| Arduino Diecimila in [Stoicheia](https://vi.wikipedia.org/w/index.php?title=Stoicheia&action=edit&redlink=1) | [https://upload.wikimedia.org/wikipedia/commons/thumb/9/96/Arduino_Diecimila_6.jpg/120px-Arduino_Diecimila_6.jpg](https://vi.wikipedia.org/wiki/T%E1%BA%ADp_tin:Arduino_Diecimila_6.jpg) | Arduino MEGA 2560 R3 (mặt trước) | [https://upload.wikimedia.org/wikipedia/commons/thumb/8/80/Arduino_MEGA_2560_R3%2C_front_side.jpg/120px-Arduino_MEGA_2560_R3%2C_front_side.jpg](https://vi.wikipedia.org/wiki/T%E1%BA%ADp_tin:Arduino_MEGA_2560_R3,_front_side.jpg) |
| Arduino Duemilanove (rev 2009b) | [https://upload.wikimedia.org/wikipedia/commons/thumb/0/05/Arduino_Duemilanove_2009b.jpg/120px-Arduino_Duemilanove_2009b.jpg](https://vi.wikipedia.org/wiki/T%E1%BA%ADp_tin:Arduino_Duemilanove_2009b.jpg) | Arduino MEGA 2560 R3 (mặt sau) | [https://upload.wikimedia.org/wikipedia/commons/thumb/0/0d/Arduino_MEGA_2560_R3%2C_back_side.jpg/120px-Arduino_MEGA_2560_R3%2C_back_side.jpg](https://vi.wikipedia.org/wiki/T%E1%BA%ADp_tin:Arduino_MEGA_2560_R3,_back_side.jpg) |
| Arduino UNO  (AVR 8-bit ATmega328P) | [https://upload.wikimedia.org/wikipedia/commons/thumb/3/3a/Arduino_UNO_unpacked.jpg/120px-Arduino_UNO_unpacked.jpg](https://vi.wikipedia.org/wiki/T%E1%BA%ADp_tin:Arduino_UNO_unpacked.jpg) | Arduino Nano | [https://upload.wikimedia.org/wikipedia/commons/thumb/8/8d/Arduino_Nano.jpg/55px-Arduino_Nano.jpg](https://vi.wikipedia.org/wiki/T%E1%BA%ADp_tin:Arduino_Nano.jpg) |
| Arduino Leonardo | [https://upload.wikimedia.org/wikipedia/commons/thumb/3/38/Arduino_Leonardo.jpg/120px-Arduino_Leonardo.jpg](https://vi.wikipedia.org/wiki/T%E1%BA%ADp_tin:Arduino_Leonardo.jpg) | Arduino Due (nền tảng ARM) | [https://upload.wikimedia.org/wikipedia/commons/thumb/1/16/Arduino_Due.jpg/90px-Arduino_Due.jpg](https://vi.wikipedia.org/wiki/T%E1%BA%ADp_tin:Arduino_Due.jpg) |
| Arduino Mega | [https://upload.wikimedia.org/wikipedia/commons/thumb/0/01/Arduino_Mega.jpg/120px-Arduino_Mega.jpg](https://vi.wikipedia.org/wiki/T%E1%BA%ADp_tin:Arduino_Mega.jpg) | LilyPad Arduino (rev 2007) | [https://upload.wikimedia.org/wikipedia/commons/thumb/e/ec/LilyPad_Arduino_Main_Board.JPG/120px-LilyPad_Arduino_Main_Board.JPG](https://vi.wikipedia.org/wiki/T%E1%BA%ADp_tin:LilyPad_Arduino_Main_Board.JPG) |

## Shield (Bo mạch mở rộng)

Các board Arduino và Arduino-compatible sử dụng các *shield*— các board mạch in mở rộng được dùng bằng cách cắm vào các chân header của Arduino. Các shield có thể là module điều khiển cho động cơ, GPS, ethernet, LCD, hoặc cũng có thể là breadboard. Một số lượng lớn các shield cũng có thể được chế tạo bởi DIY (những người thích tự làm lấy các ứng dụng cho riêng họ).

**Các shield Arduino điển hình**

|  |  |
| --- | --- |
| [https://upload.wikimedia.org/wikipedia/commons/thumb/d/df/Arduino_Protoboard_Shields.jpg/120px-Arduino_Protoboard_Shields.jpg](https://vi.wikipedia.org/wiki/T%E1%BA%ADp_tin:Arduino_Protoboard_Shields.jpg) Nhiều shield có thể được xếp chồng lên nhau. Trong ví dụ này shield ở trên cùng có chứa một breadboard chưa hàn | [https://upload.wikimedia.org/wikipedia/commons/thumb/f/f7/Adafruit_Motor_Shield_-_ARSH-02-MS_01.jpg/120px-Adafruit_Motor_Shield_-_ARSH-02-MS_01.jpg](https://vi.wikipedia.org/wiki/T%E1%BA%ADp_tin:Adafruit_Motor_Shield_-_ARSH-02-MS_01.jpg)Shield Adafruit Motor với các đầu domino dùng để kết nối với động cơ |
| [https://upload.wikimedia.org/wikipedia/commons/thumb/7/72/Wingshield_on_Arduino_-_ARSH-05-WI.jpg/120px-Wingshield_on_Arduino_-_ARSH-05-WI.jpg](https://vi.wikipedia.org/wiki/T%E1%BA%ADp_tin:Wingshield_on_Arduino_-_ARSH-05-WI.jpg) Shield này sử dụng các đầu domino bắt vít dùng để đấu các đầu dây vào | [https://upload.wikimedia.org/wikipedia/commons/thumb/e/ee/ARSH-09-DL_03.jpg/120px-ARSH-09-DL_03.jpg](https://vi.wikipedia.org/wiki/T%E1%BA%ADp_tin:ARSH-09-DL_03.jpg)Shield Adafruit Datalogging với một khe chứa [thẻ nhớ SD](https://vi.wikipedia.org/w/index.php?title=Th%E1%BA%BB_nh%E1%BB%9B_SD&action=edit&redlink=1) và chip clock Real-Time |

## Software

|  |  |
| --- | --- |
| **Arduino Software IDE** | |
| [**Phát triển bởi**](https://vi.wikipedia.org/wiki/Nh%C3%A0_ph%C3%A1t_tri%E1%BB%83n_ph%E1%BA%A7n_m%E1%BB%81m) | Arduino Software |
| [**Repository**](https://vi.wikipedia.org/w/index.php?title=Repository_(version_control)&action=edit&redlink=1) | [github.com/arduino/Arduino](https://github.com/arduino/Arduino) |
| **Được viết bằng** | [Java](https://vi.wikipedia.org/w/index.php?title=Java_(programming_language)&action=edit&redlink=1), [C](https://vi.wikipedia.org/w/index.php?title=C_(programming_language)&action=edit&redlink=1) và [C++](https://vi.wikipedia.org/wiki/C%2B%2B) |
| [**Hệ điều hành**](https://vi.wikipedia.org/wiki/H%E1%BB%87_%C4%91i%E1%BB%81u_h%C3%A0nh) | [Cross-platform](https://vi.wikipedia.org/wiki/Cross-platform) |
| [**Thể loại**](https://vi.wikipedia.org/w/index.php?title=Danh_s%C3%A1ch_th%E1%BB%83_lo%E1%BA%A1i_ph%E1%BA%A7n_m%E1%BB%81m&action=edit&redlink=1) | [Integrated development environment](https://vi.wikipedia.org/w/index.php?title=Integrated_development_environment&action=edit&redlink=1) |
| [**Giấy phép**](https://vi.wikipedia.org/wiki/Gi%E1%BA%A5y_ph%C3%A9p_ph%E1%BA%A7n_m%E1%BB%81m) | [LGPL](https://vi.wikipedia.org/wiki/GNU_Lesser_General_Public_License) or [GPL](https://vi.wikipedia.org/wiki/GNU_General_Public_License) license |
| **Trang mạng** | [arduino.cc](http://arduino.cc/en/Main/Software) |

Môi trường phát triển tích hợp ([IDE](https://vi.wikipedia.org/wiki/IDE)) của Arduino là một ứng dụng [cross-platform](https://vi.wikipedia.org/wiki/Cross-platform) (nền tảng) được viết bằng [Java](https://vi.wikipedia.org/w/index.php?title=Java_(programming_language)&action=edit&redlink=1), và từ IDE này sẽ được sử dụng cho [Ngôn ngữ lập trình xử lý](https://vi.wikipedia.org/w/index.php?title=Ng%C3%B4n_ng%E1%BB%AF_l%E1%BA%ADp_tr%C3%ACnh_x%E1%BB%AD_l%C3%BD&action=edit&redlink=1) (Processing programming language) và project [Wiring](https://vi.wikipedia.org/w/index.php?title=Wiring_(development_platform)&action=edit&redlink=1). Nó được thiết kế để dành cho những người mới tập làm quen với lĩnh vực phát triển phần mềm. Nó bao gồm một chương trình code editor với các chức năng như đánh dấu cú pháp, tự động [brace matching](https://vi.wikipedia.org/w/index.php?title=Brace_matching&action=edit&redlink=1), và tự động canh lề, cũng như compile(biên dịch) và upload chương trình lên board chỉ với 1 cú nhấp chuột. Một chương trình hoặc code viết cho Arduino được gọi là một *sketch*.

Các chương trình Arduino được viết bằng [C](https://vi.wikipedia.org/w/index.php?title=C_(programming_language)&action=edit&redlink=1) hoặc [C++](https://vi.wikipedia.org/wiki/C%2B%2B). Arduino IDE đi kèm với một [thư viện phần mềm](https://vi.wikipedia.org/w/index.php?title=Th%C6%B0_vi%E1%BB%87n_ph%E1%BA%A7n_m%E1%BB%81m&action=edit&redlink=1) được gọi là "[Wiring](https://vi.wikipedia.org/w/index.php?title=Wiring_(development_platform)&action=edit&redlink=1)", từ project Wiring gốc, có thể giúp các thao tác input/output được dễ dàng hơn. Người dùng chỉ cần định nghĩa 2 hàm để tạo ra một chương trình [vòng thực thi](https://vi.wikipedia.org/w/index.php?title=V%C3%B2ng_th%E1%BB%B1c_thi&action=edit&redlink=1) (cyclic executive) có thể chạy được:

* setup(): hàm này chạy mỗi khi khởi động một chương trình, dùng để thiết lập các cài đặt
* loop(): hàm này được gọi lặp lại cho đến khi tắt nguồn board mạch

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

[](https://vi.wikipedia.org/wiki/T%E1%BA%ADp_tin:Arduino_led-5.jpg)

Đèn LED tích hợp với chân 13

#define LED\_PIN 13

voidsetup(){

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

}

voidloop(){

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

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

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

delay(1000);*// chờ trong 1s*

}

Một đặc điểm của hầu hết các board Arduino là chúng có một đèn LED và điện trở nối giữa chân 13 với đất; một đặc điểm thuận tiện cho nhiều ứng dụng đơn giản. Đoạn code ở trên không thể đọc được bởi một [compiler](https://vi.wikipedia.org/wiki/Compiler) C++ chuẩn như là một chương trình đúng, vì vậy khi ta click vào nút "Upload to I/O board" trong IDE này, một bản copy của đoạn code này sẽ được ghi vào một file tạm với một extra include header ở phía trên cùng và một hàm main () đơn giản nằm ở phía đáy, để làm cho thàn một chương trình C++ khả dụng.

Arduino IDE này sử dụng [GNU toolchain](https://vi.wikipedia.org/w/index.php?title=GNU_toolchain&action=edit&redlink=1) và AVR Libc để biên dịch chương trình, và sử dụng avrdude để upload chương trình lên board.

Vì nền tảng của Arduino là các vi điều khiển của Atmel, cho nên môi trường phát triển của Atmel, AVR Studio hoặc các phiên bản Atmel Studio mới hơn, cũng có thể được sử dụng để làm phần mềm phát triển cho Arduino.

## Phát triển

Arduino là một nền tảng phần cứng mã nguồn mở: Các thiết kế phần cứng tham khảo của Arduino được phân phối dưới dạng Creative CommonsAttribution Share-Alike 2.5 license và có sẵn trên website của Arduino. Một vài phiên bản phần cứng của Arduino còn đưa lên cả file Layout và thành phẩm. Mã nguồn cho IDE này cũng khả dụng và được xuất bản dưới dạng GNU General Public License, version 2.

Mặc dù các thiết kế phần cứng và phần mềm là miễn phí dưới dạng copyleft license, các nhà phát triển cũng được yêu cầu cái tên "Arduino" chỉ được dành riêng cho các sản phẩm chính thức và không được sử dụng cho các sản phẩm phái sinh mà được sự cho phép. Các văn bản chính sách chính thức về việc sử dụng tên Arduino nhấn mạnh rằng dự án này là mở đối với những người khác trong việc cộng tác để tạo ra sản phẩm chính thức. Nhiều sản phẩm tương thích với Arduino phát hành thương mại đã tránh cái tên "Arduino" bằng cách sử dụng từ phái sinh "-duino".

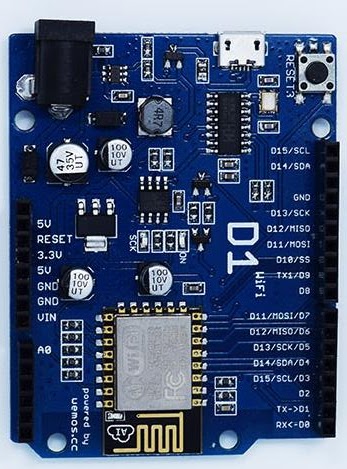
## Các ứng dụng

Xem thêm: Danh sách các dự án phần cứng mã nguồn mở

* Xoscillo: oscilloscope mã nguồn mở
* Các thiết bị khoa học
* Arduinome: một thiết bị điều khiển MIDI bắt chước Monome
* OBDuino: một máy tính hành trình sử dụng giao diện chẩn đoán on-board được tìm thấy trong hầu hết các loại xe hơi hiện đại
* Thiết bị đọc sách cho con người: thiết bị điện tử giá rẻ với đầu ra TV có thể chứa một thư viện năm ngàn cuốn sách (ví dụ như các biên soạn offline Wikipedia) trên một thẻ nhớ microSD
* Ardupilot: software / hardware máy bay không người lái
* ArduinoPhone

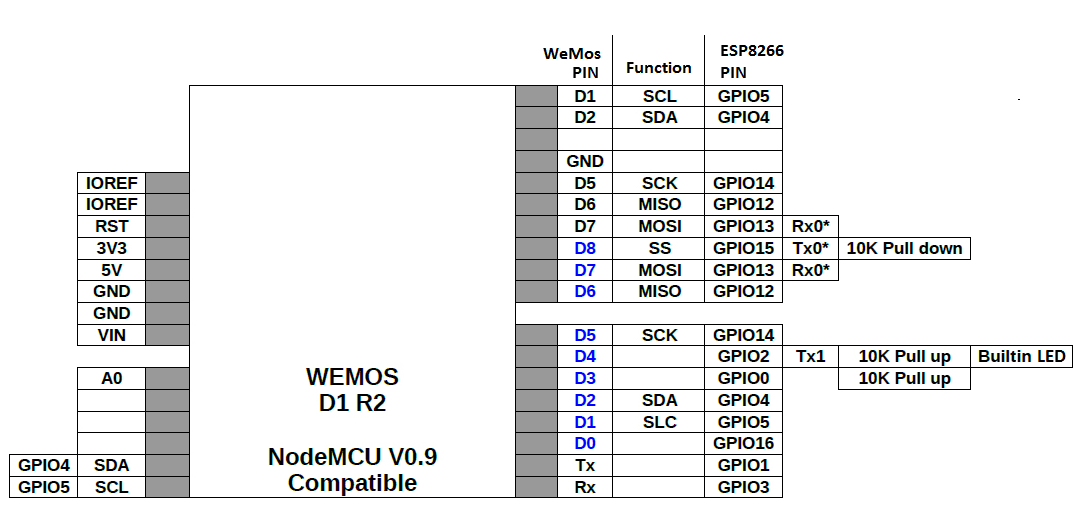
1. **Giới thiệu về Internet of Things**
2. **Làm quen với KIT ESP8266 Wemos D1 R2**

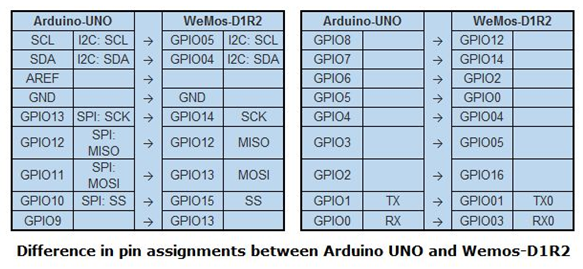
WEMOS D1 R2 là kit phát triển phiên bản mới nhất từ WeMos, kit được thiết kế với hình dáng tương tự Arduino Uno nhưng trung tâm lại là module wifi Soc ESP8266EX được build lại firmware để có thể chạy với chương trình Arduino.  Kit thích hợp và dễ dàng thực hiện các ứng dụng thu thập dữ liệu và điều khiển qua Wifi.

[](http://k3.arduino.vn/img/2017/09/28/0/3877_812450-1506569643-0-d1-2.jpg)

|  |  |
| --- | --- |
| **Vi điềukhiển** | **ESP8266EX** |
| Điệnáphoạtđộng | 3V3 |
| I/O Digital Pin | 11 |
| Analog Pin | 1 (Max input=3V2) |
| Xung clock | 80MHz/160MHz |
| Flash | 4Mb |
| Khốilượng | 25g |
| Kíchthước | 68.6mmX53.4mm |

## Sơ đồ pin digital





|  |  |  |
| --- | --- | --- |
| **Wemos D1 R2 PINs** | **ESP8266 PINs** | **Grove Arduino Base Shield PINs** |
| D0 | GPIO16 | D2 |
| D1 | GPIO5 (SLC) | D3 |
| D2 | GPIO4 (SDA) | D4 |
| D3 | GPIO0 | D5 |
| D4 | GPIO2 (Builtin LED) | D6 |
| D5 | GPIO14 (SCK) | D7 |
| D6 | GPIO12 | D8 |
| D7 |  |  |
| D8 |  |  |

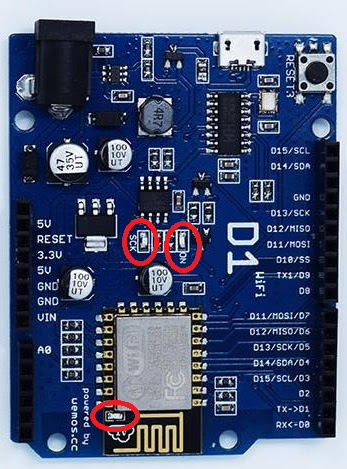
Lí do cần có bảng này là để dễ dàng trong việc viết code. Bởi khi viết code nếu muốn xuất tín hiệu HIGH cho pin số 3 trên kit thì ta ko thể viết "digitalWrite(3,HIGH);" mà phải viết là "digitalWrite(0,HIGH);" (Theo trên bảng thì chân D3 trên kit là chân 0 của ESP). Hay nói cách khác là ta phải điều khiển chân trên ESP8266. Và khi ta code xuất chân D3 mức HIGH thì chân D15 cũng được xuất HIGH, lí do là vì: các chân từ D11=>D15 là các chân "giả" để kit trông giống arduino hơn (ESP chỉ có 11 pin digital), vì vậy các pin D11=>D15 được nối lần lượt với D7=>D3.

**Tài liệu và liên kết tham khảo**

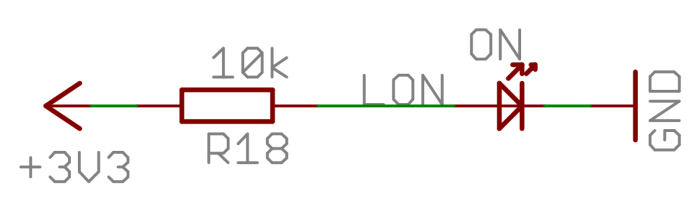
1. <https://vi.wikipedia.org/wiki/Arduino>
2. <http://arduino.vn/bai-viet/40-ban-co-biet-arduino-la-gi-khong-tim-hieu-them>
3. <http://arduino.vn/bai-viet/1769-gioi-thieu-ve-board-uno-wifi-wemos-d1-kha-tien-loi-cho-iot>
4. <http://www.instructables.com/id/Portable-installation-guide-of-Arduino-IDE-v165-fo/>

***Bài 2:Lập trình nhấp nháy đèn LED với Wemos D1 R2 (ESP8266)***

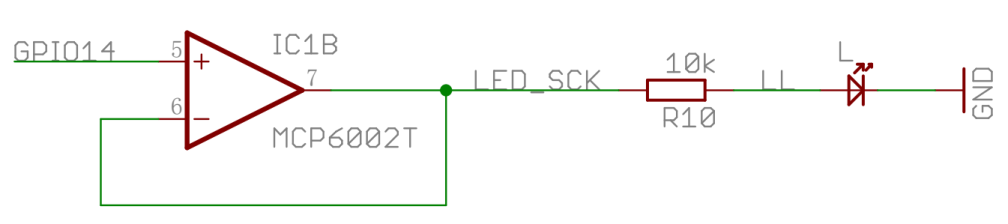
Trên KIT Wemos D1 R2 có 3 đèn LED đơn được đánh dấu như trên hình dưới đây:

******

Đèn LED ON được nối trực tiếp với nguồn thông qua điện trở 10k, do vậy khi cấp nguồi cho KIT thì đèn LED này luôn sáng.



Đèn LED SCK được nối với chân GPIO14 (General Purpose Input/Output PIN14) của chip ESP8266, do vậy có thể lập trình điều khiển đèn LED này thông qua chân 14.



LED BUILTIN được kết nối với chân GPIO2 và cũng có thể lập trình được, nhưng cần lưu ý: Khi ghi HIGH thì đèn tắt, ghi LOW thì đèn sáng (ngược với các LED thông thường).

**Ví dụ 1: Nhấp nháy đèn LED BULTIN**

Ở ví dụ đầu tiên chúng ta sẽ thực hiện nhấp nháy đèn LED BUILTIN nối với chân GPIO2. Ví dụ này đã được xây dựng sẵn trên Arduino IDE.

Trên Arduino IDE, mở **File \ Example**, di chuyển xuống *Example for WeMos D1 R2 & mini,* chọn **ESP8266 \ Blink.** Arduino IDE sẽ tự động thêm code sau vào Sketch:

*/\**

*ESP8266 Blink by Simon Peter*

*Blink the blue LED on the ESP-01 module*

*This example code is in the public domain*

*The blue LED on the ESP-01 module is connected to GPIO1*

*(which is also the TXD pin; so we cannot use Serial.print() at the same time)*

*Note that this sketch uses LED\_BUILTIN to find the pin with the internal LED*

*\*/*

void setup() {

*// Initialize the LED\_BUILTIN pin as an output*

pinMode(LED\_BUILTIN, OUTPUT);

}

// the loop function runs over and over again forever

void loop() {

*// Turn the LED on (Note that LOW is the voltage level*

*// but actually the LED is on; this is because*

*// it is active low on the ESP-01)*

digitalWrite(LED\_BUILTIN, LOW);

*// Wait for a second*

delay(1000);

*// Turn the LED off by making the voltage HIGH*

digitalWrite(LED\_BUILTIN, HIGH);

// Wait for two seconds (to demonstrate the active low LED)

delay(2000);

}

Để biên dịch ta chọn **Sketch \ Verify/Compile** (hoặc nhấn tổ hợp phím Ctrl+R).

Để nạp chương trình vào KIT và chạy ta chọn **Sketch \ Upload** (hoặc nhấn tổ hợp phím Ctrl+U).

Độc giả tự thực hiện nạp chương trình và quan sát kết quả trên KIT.

Yêu cầu:

1. Thay đổi giá trị thời gian delay và quan sát kết quả

2. Thay LED\_BUILDIN bằng số 2 để thấy kết quả không thay đổi

3. Thử thay LOW bằng *false* và thay HIGH bằng *true* và xem kết quả

**Ví dụ 2: Nhấp nháy đèn LED SCK**

Vẫn với ví dụ trên, ta thực hiện thay LED\_BUILTIN bằng số 14 và quan sát sự thay đổi của đèn LED SCK trên KIT.

Yêu cầu: Viết chương trình nhấp nháy 2 đèn BUILDIN và SCK theo kiểu đuổi nhau và đồng thời.

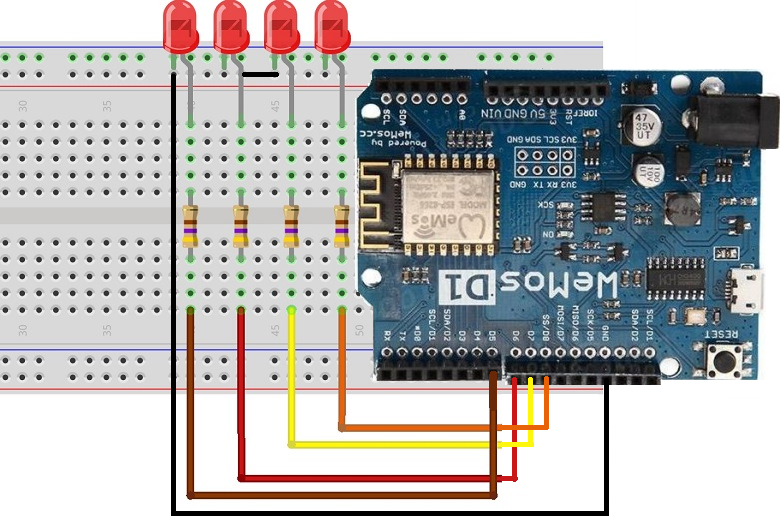
**Ví dụ 3: Nhấp nháy đèn LED cắm ngoài**

Sử dụng Bread Board (Test Board) để kết nối 4 đèn LED ngoài vào các chân D5, D6, D7, D8 của KIT WeMos (tương ứng với các chân GPIO14, GPIO12, GPIO13, GPIO15 của chip ESP8266) như hình vẽ dưới đây, sau đó lập trình tạo ra các hiệu ứng khác nhau trên các LED đã kết nối.

1. Nhấp nháy đồng thời 4 đèn

2. Nhấp nháy tuần tự 4 đèn

3. Lần lượt bật đèn ở từng chân sau đó tắt cả 4 đèn và lặp lại.



***Bài 3:Debug qua cổng Serial***

Arduino IDE cho phép người dùng xuất dữ liệu từ vi điều khiển ra màn hình máy tính qua cổng serial trong quá trình thực thi. Trước khi xuất dữ liệu ra cổng Serial, người dùng cần khởi tạo bằng lệnh:

Serial.begin(BAUD);

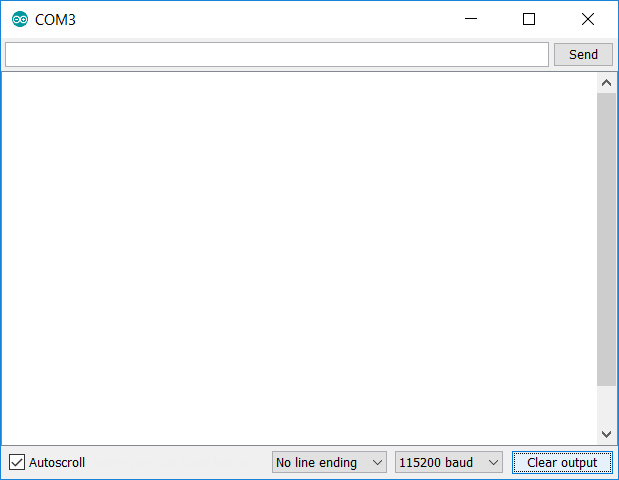
Trong đó BAUD là tốc độ truyển dữ liệu qua cổng Serial.

Để xuất dữ liệu ta dùng câu lệnh:

Serial.println(“Text”);

Trong đó “Text” là chuỗi ký tự muốn xuất ra màn hình máy tính.

Để xem dữ liệu xuất ra từ vi điều khiển, trên Arduino IDE mở **Tool \ Serial Monitor**hoặc nhấn tổ hợp phím Ctrl+Shift+M. Cửa số Serial Monitor sẽ hiện ra như hình dưới đây:



Để hiển thị đúng dữ liệu, cần chọn tốc độ nhận dữ liệu khớp với tốc độ truyền dữ liệu.

Hãy chạy ví dụ sau và bật chức năng Serial Monitor trên Arduino IDE để xem kết quả.

/\* Get Chip ID

\* wemos.cc

\*

\*

\*/

void setup() {

Serial.begin(115200);

}

void loop() {

Serial.println("");

Serial.println("");

Serial.println("Check ID in:");

Serial.println("https://www.wemos.cc/verify\_products");

Serial.printf("Chip ID = %08X\n", ESP.getChipId());

Serial.println("");

Serial.println("");

delay(5000);

}

***Bài 4: Sử dụng Grove ArduinoBase Shield***

Grove Arduino Base Shield V2 là một bo mạch mở rộng, tương thích hoàn toàn với KIT Arduino Uno Rev3. KIT Wemos D1 R2 cũng được thiết kế tương thích với Arduino Uno, do vậy có thể sử dụng Grove Arduino Base Shield để cắm vào KIT Wemos D1 R2. Tuy vậy cần lưu ý sự khác biệt về các chân như trên bảng dưới đây.

|  |  |  |
| --- | --- | --- |
| **Wemos D1 R2 PINs** | **ESP8266 PINs** | **Grove Arduino Base Shield PINs** |
| D0 | GPIO16 | D2 |
| D1 | GPIO5 (SLC) | D3 |
| D2 | GPIO4 (SDA) | D4 |
| D3 | GPIO0 | D5 |
| D4 | GPIO2 (Builtin LED) | D6 |
| D5 | GPIO14 (SCK) | D7 |
| D6 | GPIO12 | D8 |
| D7 |  |  |
| D8 |  |  |

**Ví dụ 1:Hiển thị dữ liệu với 4-digit display**

Trong nhiều trường hợp, người dùng có nhu cầu sử dụng LED 7 thanh để hiển thị thông tin. Có nhiều mạch điều khiển LED 7 thanh. Một trong số các mạch điều khiển LED 7 thanh phổ biến là TM1637. Datasheet và thư viện lập trình với TM1637 có thể tìm thấy ở các links dưới đây:

<https://github.com/avishorp/TM1637/blob/master/docs/TM1637_V2.4_EN.pdf>

<https://github.com/avishorp/TM1637>

**Ví dụ 2:Kết nối với màn hình Grove-LCD RGB Backlight**

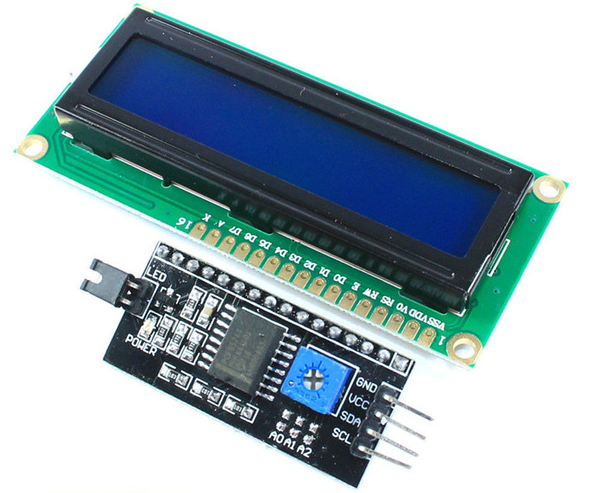
*Bước 1:*Cài đặt thư viện **DHT library for ESP**

*Bước 2:*  Cắm Grove Arduino Base Shield vào KIT Wemos D1 R2, sau đó cắm Grove LCD RGB Backlight vào chân I2C bất kỳ của Shield

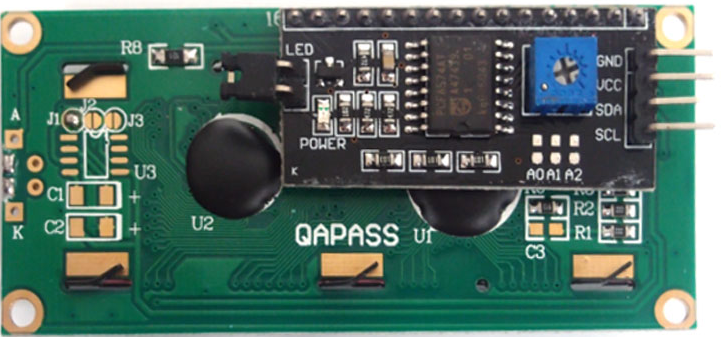
*Bước 3:* Bật Arduino IDE, Chạy các ví dụ trong File \ Examples \ Grove LCD RGB Backlight và quan sát kết quả, phân tích code chương trình.

**Ví dụ 3: Kết nối với màn hình LCD 1602 qua module I2C for LCD, không dùng Arduino Base Shield**

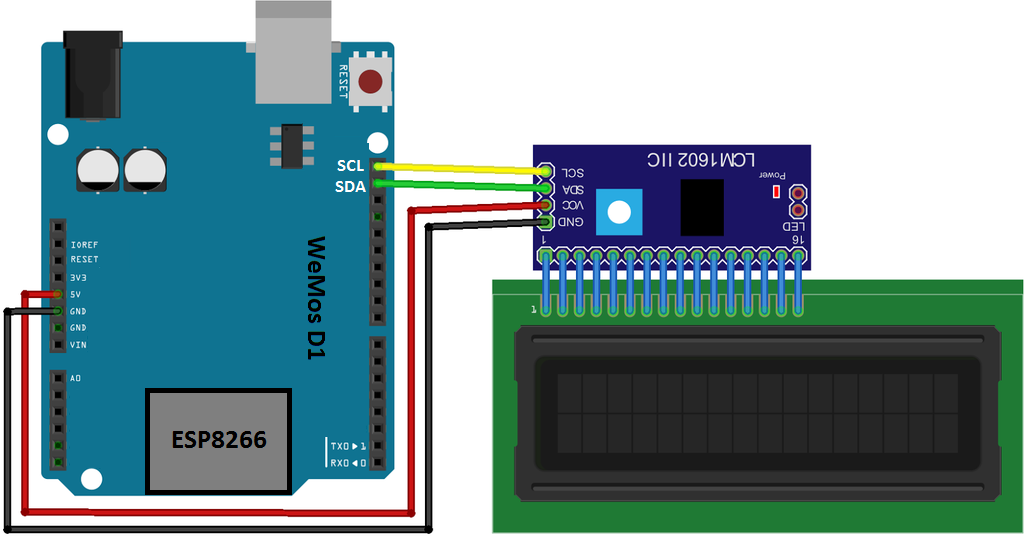
Trong trường hợp không có bo mạch mở rộng Arduino Base Shield và module Grove LCD RGB Backlight, ta có thể sử dụng màn hình LCD 1602 thông thường và module I2C for LCD như hình dưới đây:



Tiến hành hàn LCD 1602 với module I2C for LCD, kết quả thu được như hình dưới đây:



Tiếp theo chúng ta sử dụng 4 sợi dây nối để kết nối KIT Wemos D1 R2 với module I2C for LCD như hình sau:



Trên Arduino IDE, cài thư viện mang tên LiquidCrystal I2C.

Tiến hành quét cổng I2C để tìm địa chỉ. Mã nguồn chương trình quét như sau:

/\*

\* i2c\_port\_address\_scanner

\* Scans ports D0 to D7 on an ESP8266 and searches for I2C device. based on the

\* original code available on Arduino.cc and later improved by user

\* Krodal and Nick Gammon (www.gammon.com.au/forum/?id=10896)

\*/

#include <Wire.h>

void setup() {

Serial.begin(115200);

while (!Serial); // Wait for serial monitor

Serial.println("\n\nI2C Scanner to scan for devices on each port pair D0 to D7");

scanPorts();

}

uint8\_t portArray[] = {16, 5, 4, 0, 2, 14, 12, 13};

String portMap[] = {"D0", "D1", "D2", "D3", "D4", "D5", "D6", "D7"};

void scanPorts() {

for (uint8\_t i = 0; i < sizeof(portArray); i++) {

for (uint8\_t j = 0; j < sizeof(portArray); j++) {

if (i != j){

Serial.print("Scanning (SDA : SCL) - " + portMap[i] + " : " + portMap[j] + " - ");

Wire.begin(portArray[i], portArray[j]);

check\_if\_exist\_I2C();

}

}

}

}

void check\_if\_exist\_I2C() {

byte error, address;

int nDevices;

nDevices = 0;

for (address = 1; address < 127; address++ ) {

// The i2c\_scanner uses the return value of

// the Write.endTransmisstion to see if

// a device did acknowledge to the address.

Wire.beginTransmission(address);

error = Wire.endTransmission();

if (error == 0){

Serial.print("I2C device found at address 0x");

if (address < 16)

Serial.print("0");

Serial.print(address, HEX);

Serial.println(" !");

nDevices++;

} else if (error == 4) {

Serial.print("Unknow error at address 0x");

if (address < 16)

Serial.print("0");

Serial.println(address, HEX);

}

} //for loop

if (nDevices == 0)

Serial.println("No I2C devices found");

else

Serial.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

delay(1000); // wait 1 seconds for next scan, did not find it necessary

}

void loop() {

}

Bật Tools \ Serial Monitor để xem kết quả, ta sẽ thấy địa chỉ I2C trả về là 0x3F, tương ứng với cặp chân SDA = D2, SCL = D1. Trên cơ sở đó ta có thể viết chương trình hiển thị text lên màn hình LCD như ví dụ sau:

#include <Wire.h>

#include <LiquidCrystal\_I2C.h>

// set the LCD address to 0x27 for a 16 chars and 2 line display+

// I2C address = 0x3F

LiquidCrystal\_I2C lcd(0x3F,16,2);

int sdaPIN = D2;

int sclPIN = D1;

void setup()

{

Wire.begin(sdaPIN,sclPIN); // Initialize the I2C

lcd.init(); // initialize the lcd

lcd.backlight();

lcd.setCursor(3,0);

lcd.print("Hello, world!");

lcd.setCursor(2,1);

lcd.print("Wemos D1 R2");

}

void loop()

{

}

**Ví dụ 4:Kết nối với Grove Button**

**Ví dụ5: Kết nối với cảm biến nhiệt độ và độ ẩm DHT11**

*Bước 1:*Cài đặt thư viện **DHT library for ESP**

*Bước 2:*  Cắm Grove Arduino Base Shield vào KIT Wemos D1 R2, sau đó cắm Grove Temperature & Humidity Sensor vào chân D8 của Shield

*Bước 3:* Chạy ví dụ sau và xem kết quả hiển thị trên Serial Monitor

*#include "DHTesp.h"*

*DHTesp dht;*

*void setup()*

*{*

*Serial.begin(115200);*

*Serial.println();*

*Serial.println("Status\tHumidity (%)\tTemperature (C)\t(F)\tHeatIndex (C)\t(F)");*

// Connect DHT sensor to GPIO 12 (D8 connector of Grove Arduino Base Shield)

*dht.setup(12);*

*}*

*void loop()*

*{*

*delay(dht.getMinimumSamplingPeriod()\*3);*

*float humidity = dht.getHumidity();*

*float temperature = dht.getTemperature();*

*Serial.print(dht.getStatusString());*

*Serial.print("\t");*

*Serial.print(humidity, 1);*

*Serial.print("\t\t");*

*Serial.print(temperature, 1);*

*Serial.print("\t\t");*

*Serial.print(dht.toFahrenheit(temperature), 1);*

*Serial.print("\t\t");*

*Serial.print(dht.computeHeatIndex(temperature, humidity, false), 1);*

*Serial.print("\t\t");*

*Serial.println(dht.computeHeatIndex(dht.toFahrenheit(temperature), humidity, true), 1);*

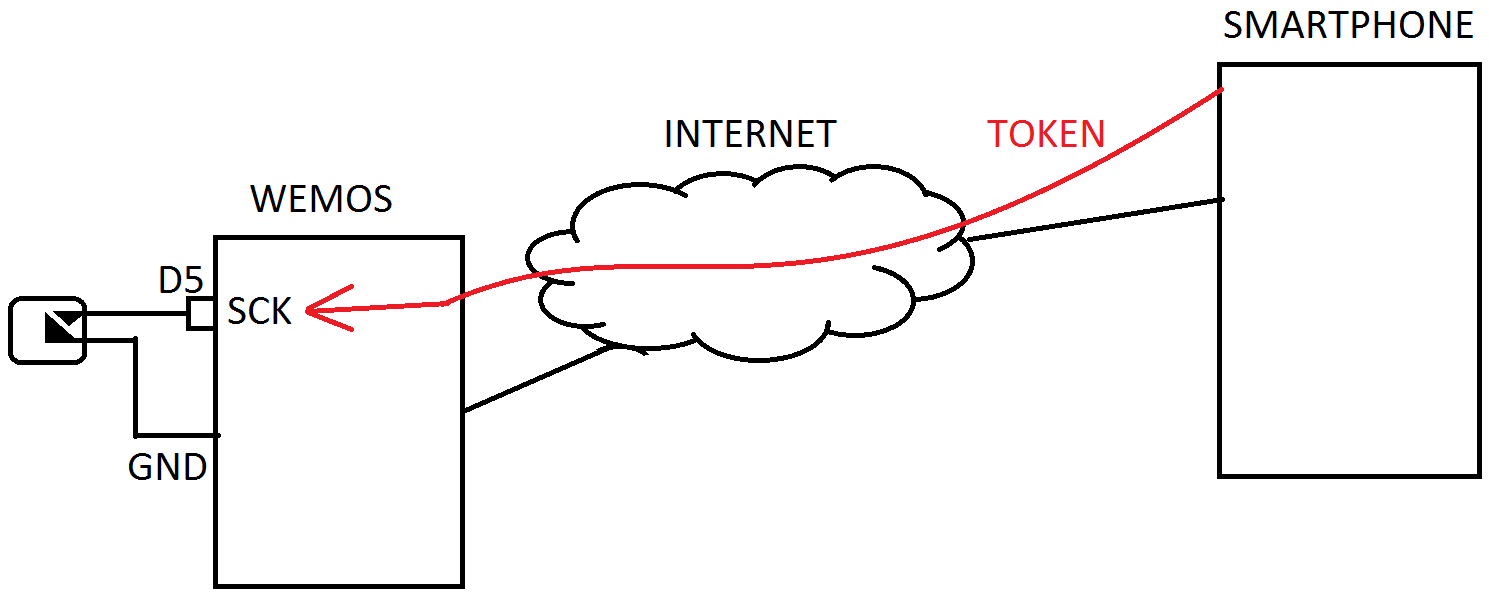
*}*

***Bài 5: Điều khiển thiết bị trên KIT Wemos D1 R2 bằng mobile app Blynk***

Ở bài thực hành này, chúng ta sẽ tiến hành điều khiển thiết bị ngoại vi trên KIT Wemos D1 R2 bằng Smartphone hoặc Tablet.

Trước hết, chúng ta cần cài đặt phần mềm trên thiết bị di động để thực hiện việc điều khiển thiết bị từ xa. Hiện nay, trên cả Android lẫn iOS đều cho phép cài đặt phần mềm Blynk, đây là phần mềm rất phù hợp cho việc điều khiển các thiết bị IoT. Blynk rất tiện lợi, vừa mạnh mẽ vừa linh hoạt.

**Ví dụ 1: Điều khiển đèn LED SCK trên KIT Wemos D1 R2 từ xa bằng Button trên chương trình Blynk**



1. Kết nối thiết bị di động với Internet qua wifi hoặc 3G

2. Cài đặt mobile app có tên là Blynk

3. Đăng ký account trên Blynk

4. Login vào Blynk

5. Tạo project, chọn KIT là Wemos D1 mini

thêm vào projectmột nút bấm (Button) với các tham số:

OUTPUT = D5 (tương ứng LED\_SCK trên KIT Wemos)

MODE = Switch

6. Nhấn vào nút hình tam giác trên giao diện của Blynk để chạy chương trình

Tiếp đến, chúng ta tiến hành thiết lập môi trường làm việc trên máy tính. Trên Arduino IDE chúng ta cần cài đặt thư viện để có thể giao tiếp với phần mềm Blynk trên Smartphone/Tablet.

1. Trên Arduino IDE, cài thư viện có tên Blynk

2. Trên menu File, mở ví dụ sau

File \ Examples\ Blynk \ Boards\_Wifi \ Esp8266\_Standalone

#define BLYNK\_PRINT Serial

#include <ESP8266WiFi.h>

#include <BlynkSimpleEsp8266.h>

// You should get Auth Token in the Blynk App.

// Go to the Project Settings (nut icon).

char auth[] = "YourAuthToken";

// Your WiFi credentials.

// Set password to "" for open networks.

char ssid[] = "YourNetworkName";

char pass[] = "YourPassword";

void setup()

{

// Debug console

Serial.begin(9600);

Blynk.begin(auth, ssid, pass);

}

void loop()

{

Blynk.run();

}

Sửa lại code chương trình, điền vào:

auth[] = Token - Lấy từ project trên Blynk

ssid[] = Wifi\_SSID: Tên mạng wifi để kết nối vào Internet

pass[] = Wifi\_Password: Mật khẩu kết nối wifi

3. Compile, Upload và chạy chương trình trên KIT Wemos

4. Bật Serial Monitor để theo dõi quá trình kết nối và trao đổi dữ liệu qua mạng Wifi

Nếu kết nối được thiết lập thành công và chương trình trên Blynk đang ở trạng thái running thì chúng ta có thể sử dụng nút bấm mềm trên Blynk để điều khiển đèn LED SCK trên KIT.

**Ví dụ 2: Điều khiển độ sáng tối của màn hình LCD nối với KIT Wemos D1 R2 từ xa bằng Slider trên chương trình Blynk**

1. Trên smartphone, cài chương trình Blynk, tạo project với Hardware Model là Wemos D1 mini

2. Thêm vào project đã tạo control có tên là Slider

3. Bấm vào Slider đã thêm, thiết lập OUTPUT là Virtual Pin với tên V1 và giá trị OUTPUT nằm trong khoảng từ 0 đến 255

4. Trên Arduino IDE, cài đặt các thư viện để làm việc với Blynk và màn hình LCD (Grove LCD RGB Backlight)

5. Tạo mới Sketch với mã nguồn sau:

#include <ESP8266WiFi.h>

#include <BlynkSimpleEsp8266.h>

#include <Wire.h>

#include "rgb\_lcd.h"

rgb\_lcd lcd;

char auth[] = "YourBlynkToken";

char ssid[] = "Wifi\_SSID";

char pass[] = "Wifi\_Password";

int pinValue = 0;

void setup()

{

// set up the LCD's number of columns and rows:

lcd.begin(16, 2);

lcd.setRGB(pinValue, pinValue, pinValue);

lcd.clear();

Serial.begin(115200);

Blynk.begin(auth, ssid, pass);

lcd.setCursor(2,0);

lcd.print("Hello world!");

lcd.setCursor(0,1);

lcd.print("From Wemos D1 R2");

}

BLYNK\_WRITE(V1)

{

pinValue = param.asInt();

Serial.print("Slider value: ");

Serial.println(pinValue);

lcd.setRGB(pinValue, pinValue, pinValue);

}

void loop()

{

Blynk.run();

}

6. Biên dịch, nạp chương trình lên KIT và chạy chương trình trên Blynk, thay đổi giá trị Slider trên Blynk để xem sự thay đổi ánh sáng nền của màn hình LCD.

***Bài 6:Kết nối và gửi thông tin từ Wemos D1 R2 lên cloud server***

Một trong những yêu cầu quan trọng trong ứng dụng IoT là dữ liệu phải được gửi và lưu trữ trên cloud. Ở bài này chúng ta sẽ thử tạo account trên một cloud phổ biến và đẩy dữ liệu thu thập được từ KIT Wemos D1 R2 lên trên cloud thông qua account đã tạo.

Có rất nhiều nền tảng IoT được các công ty lớn xây dựng sẵn. Hầu hết chúng cho phép dùng thử miễn phí hoặc miễn phí khi dữ liệu gửi lên rất nhỏ, một số miễn phí hoàn toàncho giáo dục. Các nền tảng IoT phổ biến có thể kể đến là AWS của Amazon, Watson của IBM, Azure của Microsoft, Google cloud của Alphabet, Thingspeak của Mathworks, Thingxyz của VNPT (Việt Nam)…

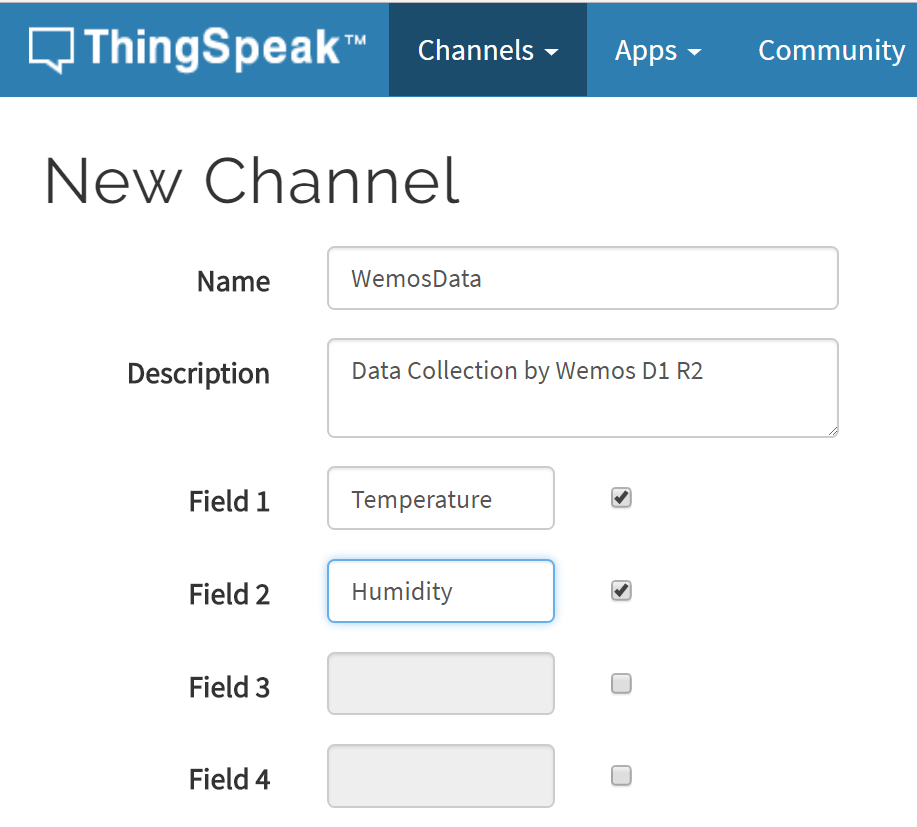
Trong số các nền tảng IoT kể trên, Thingspeak có nhiều ưu điểm và đặc biệt miễn phí hoàn toàn cho giáo dục. Do vậy chúng ta sẽ thực hiện việc tạo account và gửi dữ liệu lên cloud của nền tảng này. Dưới đây là bài viết hướng dẫn sử dụng bằng tiếng Anh:

<http://www.instructables.com/id/Send-sensor-data-DHT11-BMP180-to-ThingSpeak-with-a/>

*Bước 1:* Vào trang <https://thingspeak.com/>

Nếu đã có Account thì thực hiện login, ngược lại thực hiện Signup

*Bước 2:*New Channel với tên WemosData. Thêm vào Channel hai Fields là Temperature và Humidity như hình dưới:



*Bước 3:* Cắm cảm biến Nhiệt độ & Độ ẩm vào chân D8 của Shield (Chân D6 của ESP8266)

*Bước 4:* Trên Arduino IDE, cài *DHT sensor library for ESPx* và tạo một Sketch với mã nguồn sau:

#include "DHTesp.h"

#include "ESP8266WiFi.h"

//DHT config

#define DHTPIN D6 // what digital pin we're connected to

DHTesp dht;

// Wi-Fi Settings

const char\* ssid = "Your Wifi SSID"; // your wireless network name (SSID)

const char\* password = "Your Wifi Password"; // your Wi-Fi network password

WiFiClient client;

// ThingSpeak Settings

const int channelID = 461471;

// WRITE API KEY for your ThingSpeak Channel

String writeAPIKey = "Your ThingSpeak WRITE API KEY";

const char\* server = "api.thingspeak.com";

const int postingInterval = 2 \* 1000; // post data every 2 seconds

void setup()

{

Serial.begin(115200);

dht.setup(DHTPIN); // Connect DHT sensor to pin D6

Serial.print("Connecting");

WiFi.begin(ssid, password);

while (WiFi.status() != WL\_CONNECTED) {

Serial.print(".");

delay(100);

}

Serial.println("\r\nWiFi connected");

}

void loop()

{

// wait and then post again

delay(dht.getMinimumSamplingPeriod());

float humi = dht.getHumidity();

float temp = dht.getTemperature();

if (isnan(temp) || isnan(humi))

{

Serial.println("Failed to read from DHT sensor!");

return;

}

if (client.connect(server, 80))

{

// Construct API request body

String body = "field1=" + String(temp, 1) + "&field2=" + String(humi, 1);

client.print("POST /update HTTP/1.1\n");

client.print("Host: api.thingspeak.com\n");

client.print("Connection: close\n");

client.print("X-THINGSPEAKAPIKEY: " + writeAPIKey + "\n");

client.print("Content-Type: application/x-www-form-urlencoded\n");

client.print("Content-Length: ");

client.print(body.length());

client.print("\n\n");

client.print(body);

client.print("\n\n");

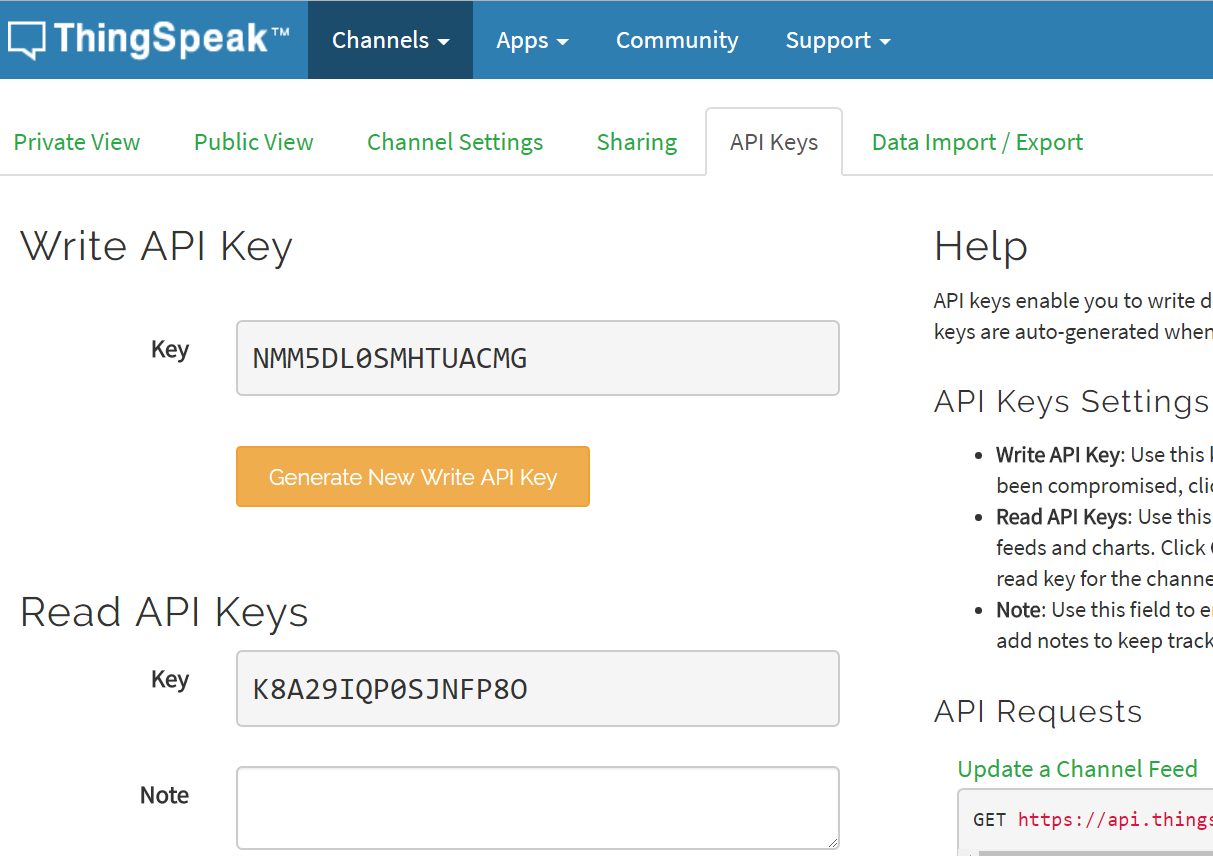
Serial.printf("Nhiet do %s - Do am %s\r\n", String(temp, 1).c\_str(), String(humi, 1).c\_str());

}

client.stop();

}

Lưu ý sửa tên mạng wifi, password wifi và ThingSpeak Write API Key (xem hình dưới) trong code.



*Bước 5:* Biên dịch và nạp chương trình lên KIT Wemos D1 R2, xem kết quả trên ThingSpeak:

