Arduino Simulator:

1. Truy Cập?

* <https://wokwi.com/>
* Đăng nhập = tài khoản Google

1. Tạo 1 Dự Án?

* Click Avatar góc phải trên + chọn “My projects” + vào Tab “YOUR PROJECTS” + Click “NEW PROJECT” + chọn loại Board + chỉnh sửa dự án + Click nút “SAVE” + điền tên dự án + Click “SAVE”

1. Thêm 1 File?

* Click nút mũi tên bên phải “Library Manager” + chọn “New file…” + đặt tên + nhấn “CREATE”

1. Đổi tên 1 File?

* Chọn File đó + Click nút mũi tên bên phải “Library Manager” + chọn “Delete” + chọn “OK”

Arduino IDE – Giao Diện Lập Trình Arduino:

1. Sketch?

* Vào Tab “SKETCHBOOK” ở bên trái + ở đây sẽ hiện tất cả thư mục bên trong thư mục “C:\Users\pc\Documents\Arduino”, không hiện File, nói chung là vô dụng
* Khi bạn mở 1 File Code Arduino là nó tự mở tất cả các File Code khác và các File C++ bên cạnh vào các Tab khác nhau, đồng thời, trong các File đó, phải có 1 File trùng tên với thư mục chứa tất cả các File này, đây là File chính

1. Phần Mở Rộng File Code Arduino?

* “.ino”

1. Boards Manager?

* Chứa các công cụ thích hợp để tải biên dịch Code cho các Board khác nhau
* Mỗi Boards Manager sẽ chứa các loại Board khác nhau
* Để cài đặt các Boards Manager
* Vào Tab “BOARDS MANAGER” bên trái + ở phần bên trái cuộn xuống tới cái Boards Manager muốn cài đặt + chọn phiên bản + Click “INSTALL” + chờ nó cài đặt
* Để gỡ cài đặt 1 Boards Manager thì làm tương tự như trên, nhưng Click “REMOVE” + chờ nó gỡ cài đặt
* Để chọn loại Board để Arduino IDE có phương án thích hợp để biên dịch
* Vào Vào Tab “Tools” + chọn “Board” + chọn Boards Manager + chọn loại Board mong muốn
* Để chọn cái cổng mà con Arduino đang cắm vào để Arduino IDE biết đường mà tải Code đúng nơi
* Vào Tab “Tools” + chọn “Port” + chọn cái cổng mới hiện khi cắm Arduino vào
* Để biên dịch tất cả các File trong thư mục hiện tại
* Click nút “Verify” góc trái trên, khi này tất cả các File trong thư mục hiện tại sẽ được Concat với nhau với File chính làm đầu, rồi tới các File sau theo thứ tự bảng chữ cái, sau đó biên dịch toàn bộ, bắt buộc khi biên dịch phải gặp duy nhất 1 hàm setup và duy nhất 1 hàm loop, File biên dịch sẽ lưu ở đâu đó
* Để tải File biên dịch vào Arduino
* Click nút “Upload” góc trái trên
* Code trong hàm setup sẽ được chạy lần đầu tiên khi Arduino khởi động lại, và Code trong hàm loop sẽ được chạy đi chạy lại sau đó
* Cú pháp y chang C++

1. Web Lập Trình Trên Arduino Giả Lập?

* <https://www.tinkercad.com/>
* Đăng nhập bằng tài khoản Google + vào phần Circuit + Click Tinker This

1. Tài Liệu Arduino?

* <https://www.arduino.cc/reference/en/>

1. Lưu Code?

* Vào Tab “File” + chọn “Save As…” + chọn chỗ lưu + đặt tên + Click nút “Save”
* Khi lưu, nó sẽ tạo ra 1 thư mục, File Code ở bên trong, cả thư mục cả File Code đều cùng tên

1. Mở 1 Code Viết Sẵn?

* Vào Tab “File” + chọn “Examples” + chọn chủ đề + chọn bài Code
* Code viết sẵn sẽ mở ở cửa sổ mới

Issues – Các Vấn Đề:

1. Mắc Điện Trở Đúng Cách?

* Bẻ 2 chân sát vuông góc với thân, cắm thẳng sâu vào lỗ

1. Đèn LED Không Lên?

* Dây nối yếu

1. Cách Dễ Nhất Để Không Gặp Vấn Đề?

* Làm từng bước 1 rồi kiểm tra thay vì làm nguyên dự án rồi cắm điện
* Bắt đầu từ mạch đơn giản LED + điện trở

1. Cảm Biến Siêu Âm Đo Được Khoảng Cách = 0?

* Reset Arduino, dùng Pin khác

Arduino Board – Bảng Vi Mạch Arduino:

1. Sát Chữ ON Là Gì?

* Đèn LED thông báo Arduino đang được cắm điện

1. Sát Chữ L Là Gì?

* Đèn LED nối với Pin 13, độ sáng dựa vào điện thế của Pin 13

1. Pin GND Là Gì?

* Pin GND là Pin luôn có điện thế = 0
* Cơ thể con người có điện thế thấp hơn GND, khoảng –3V, nên nếu nối GND với tay ta sẽ tạo thành mạch điện

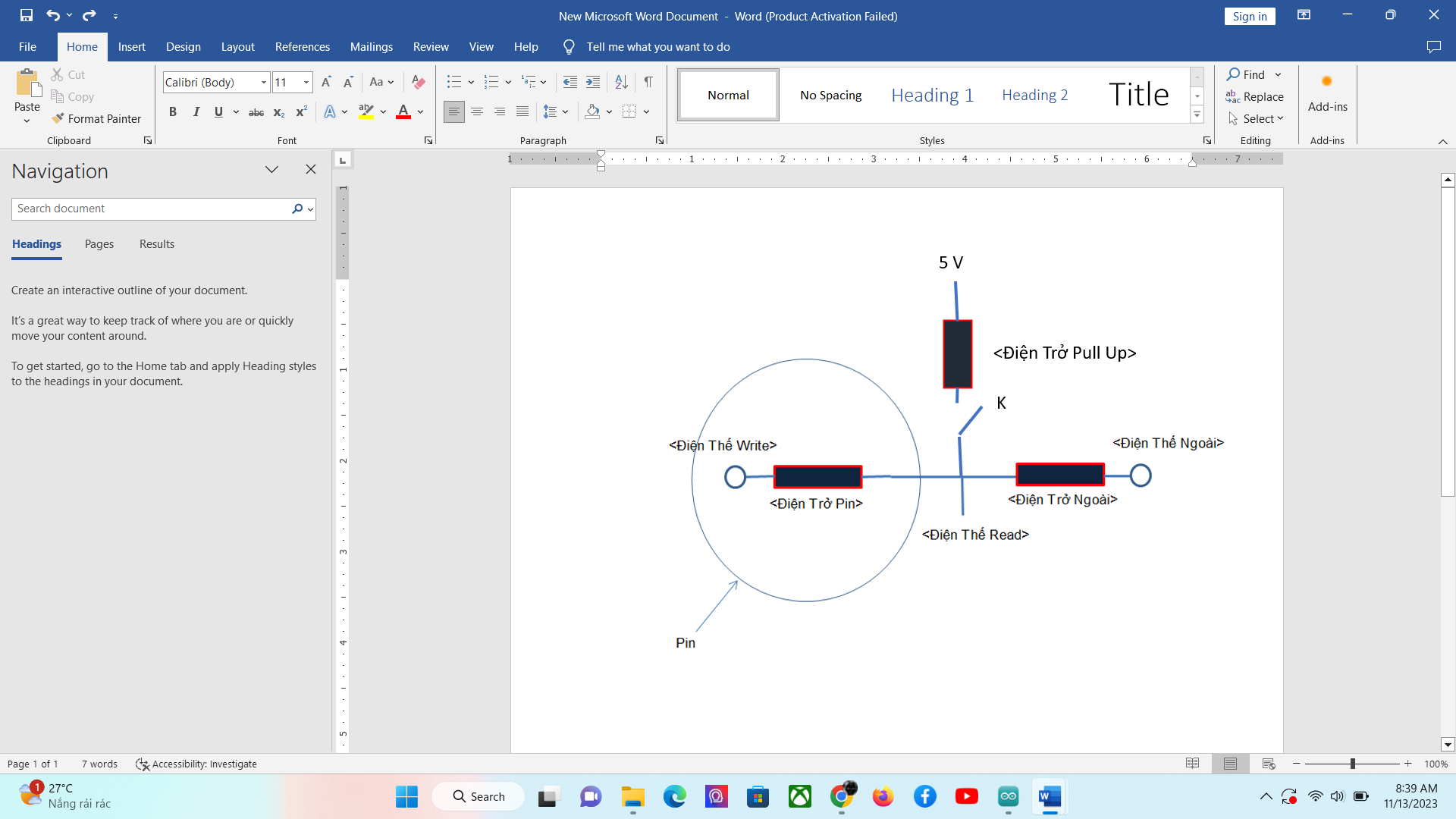
Global Constants – Hằng Toàn Cục:

1. HIGH = 5
2. LOW = INPUT = 0
3. CHANGE = OUTPUT = 1
4. RISING = INPUT\_PULLUP = 2
5. FALLING = 3
6. A0 = 14
7. A1 = 15
8. A2 = 16
9. A3 = 17
10. A4 = 18
11. A5 = 19
12. LED\_BUILTIN = 13

Pin:

1. Tổng Quan?

* Trong Arduino, tổng cộng có 20 Pin ta có thể điều khiển, trong đó, có 14 Pin được đánh số từ 0 đến 13, các Pin có dấu “~” bên cạnh là Analog Write Pin, còn lại là Digital Pin, thêm 6 Pin được đánh số từ A0 đến A5 là Analog Read Pin
* Sơ lược cấu tạo 1 Pin bất kì



* <Điện Trở Ngoài> đã tính luôn điện trở dây
* Khi Pin ở chế độ Input hoặc Input Pull Up, <Điện Trở Pin> sẽ cực lớn, nếu ở chế độ Output, thì sẽ cực nhỏ
* Khi Pin ở chế độ Input Pull Up, khóa K đóng, còn lại thì hở, <Điện Trở Pull Up> sẽ ở tầm vừa phải và bé hơn nhiều so với <Điện Trở Pin>, để nếu cắt dây ở chỗ <Điện Trở Ngoài> thì <Điện Thế Read> sẽ gần = 5 V, còn không thì nó chả khác mẹ gì Input
* Mặc định khi mới mua về Arduino, các Pin sẽ ở chế độ Input và

<Điện Thế Write> = 0

1. Chỉ Định Chế Độ Của Pin?

* Để chuyển sang Output

pinMode(<Số Của Pin>, 1);

* Để chuyển sang Input

pinMode(<Số Của Pin>, 0);

* Để chuyển sang Input Pull Up

pinMode(<Số Của Pin>, 2);

* Bản chất thay đổi trạng thái <Điện Trở Pin> và khóa K

1. Áp Điện Thế Nhị Phân Vào 1 Pin?

digitalWrite(<Pin Index>, <Điện Thế>);

* Nếu <Điện Thế> = 0, hoặc là số âm không nguyên, hoặc là số dương < 1, thì sẽ quy đổi về 0, các trường hợp còn lại quy đổi về 5
* <Pin Index> sẽ được loại bỏ phần thập phân
* Bản chất thay đổi <Điện Thế Write> thành 0 V hoặc 5 V
* Ví dụ

digitalWrite(A3, HIGH);

1. Đọc Trạng Thái Của 1 Pin?

<Trạng Thái> = digitalRead(<Pin Index>);

* <Trạng Thái> = 0 nếu <Điện Thế Read> dưới 2.5 V, ngược lại = 1
* Ví dụ

foo = digitalRead(13);

* Ta có

|  |  |
| --- | --- |
| foo | 1 |

1. Áp Điện Thế Liên Tục Vào 1 Pin?

analogWrite(<Số Của Pin>, <Điện Thế>);

* Nếu <Điện Thế> có giá trị nguyên từ 0 đến 255 thì sẽ quy đổi thành từ 0 đến 5, vượt ngoài khoảng này và nguyên thì lấy 8 Bit cuối
* Bản chất gán điện thế từ 0 V đến 5 V cho <Điện Thế Write>
* Chỉ có Analog Write Pin mới dùng được lệnh này, các loại Pin khác sẽ miễn nhiễm lệnh này
* Ví dụ

analogWrite(11, 120);

1. Đọc Giá Trị Điện Thế Tại 1 Pin?

<Điện Thế Chưa Xử Lí> = analogRead(<Số Của Pin>);

* <Điện Thế Chưa Xử Lí> có giá trị nguyên từ 0 đến 1023
* Ta có công thức

<Điện Thế Read> = <Điện Thế Chưa Xử Lí> / 1023 \* 5

* Chỉ có Analog Read Pin mới dùng được lệnh này, các loại Pin khác sẽ trả về giá trị nào đó không xác định
* Ví dụ

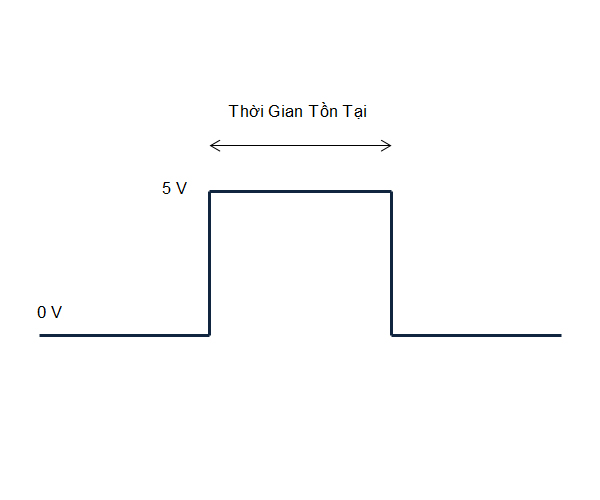
foo = analogRead(A0);

* Ta có

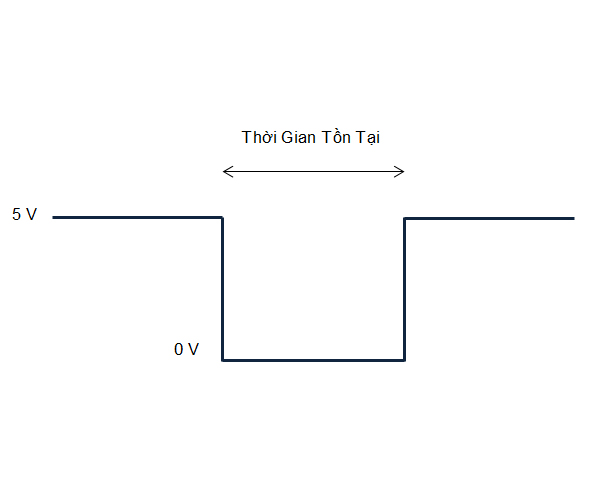
|  |  |
| --- | --- |
| foo | 857 |

1. Đo Thời Gian Tồn Tại Của Xung Điện?

* 1 xung điện hợp lệ trong Arduino có các dạng sau
* Xung lên



* Xung xuống



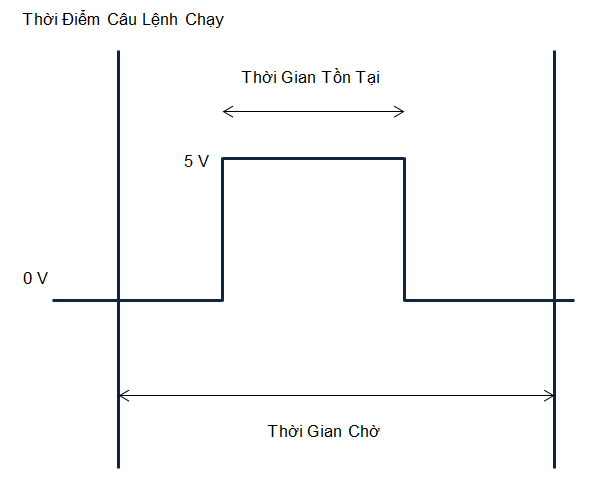
* Để đo <Thời Gian Tồn Tại> của 1 xung điện xuất hiện tại 1 Pin

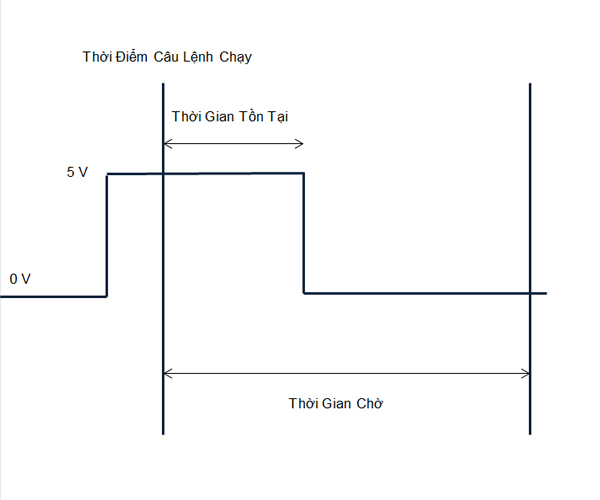
<Thời Gian Tồn Tại> = pulseIn(<Pin Index>, <Điện Thế>, <Thời Gian Chờ>);

* Mặc định

|  |  |
| --- | --- |
| <Thời Gian Chờ> | 1000000 |

* <Điện Thế> = 5 tương ứng với việc đo <Thời Gian Tồn Tại> của xung lên
* <Điện Thế> = 0 tương ứng với việc đo <Thời Gian Tồn Tại> của xung xuống
* Lệnh trên sẽ tạm dừng chương trình cho đến khi đo được kết quả hoặc quá <Thời Gian Chờ> mà vẫn chưa bắt gặp được xung nào thì trả về 0
* <Thời Gian Chờ> tính theo Micro giây
* Lưu ý xung điện là của <Điện Thế Read>
* Minh họa cho trường hợp xung lên





Serial:

1. Cách Mở Cửa Sổ Serial?

* Click biểu tượng góc phải trên trong IDE

1. Cách In Giá Trị Ra Màn Hình Serial?

* Tại cửa sổ Serial, ở góc phải dưới, chọn “9600 baud”
* Chạy lệnh sau 1 lần

Serial.begin(9600);

* Để in không xuống dòng

Serial.print(<Giá Trị>);

* Để in rồi xuống dòng

Serial.println(<Giá Trị>);

* Để xóa tất cả trên màn hình, Click “Clear output”, hoặc đóng rồi mở lại cửa sổ Serial
* Lưu ý các lệnh trên ban đầu sẽ xem xem cửa sổ Serial có được mở không, nếu có thì nó in ra cửa sổ luôn, còn không thì không làm gì cả

1. Serial Input?

* Khi bạn nhập 1 chuỗi kí tự vào ô Input trong cửa sổ Serial rồi nhấn Enter, thì chuỗi đó sẽ được + vào Buffer hiện tại
* Ví dụ
* Buffer hiện tại = "abc", nhập "123” rồi nhấn Enter, Buffer thành "abc123"
* Đóng cửa sổ Serial sẽ làm rỗng Buffer
* Trả về số kí tự của Buffer hiện tại

Serial.available();

* Ví dụ

foo = Serial.available();

* Ta có

|  |  |
| --- | --- |
| Buffer | "abc1234" |
| foo | 7 |

* Trả về Buffer hiện tại dưới dạng String và làm rỗng nó

Serial.readString();

* Ví dụ

foo = Serial.readString();

* Ta có

|  |  |
| --- | --- |
| foo | "abc1234" |

* Trả về Buffer hiện tại dưới dạng Integer và làm rỗng nó

Serial.parseInt();

* Lưu ý nếu Buffer không thể chuyển sang Integer thì nó sẽ kiếm dãy kí tự liên tục đầu tiên chỉ toàn số, rồi trả về dãy này dưới dạng Integer, nếu không tìm được dãy nào thì trả về 0
* Ví dụ

foo = Serial.parseInt();

* Ví dụ

|  |  |
| --- | --- |
| Buffer | "abc1234ab123" |
| foo | 1234 |

1. UART (Universal Asynchronous Receiver Transmitter) Pin?

* Pin 0 là Rx (Receiver)
* Pin 1 là Tx (Transmitter)
* 2 Pin này đéo dùng được khi sử dụng Serial và khi tải Code từ máy tính vào Arduino

Time – Thời Gian:

1. Tạm Dừng Chương Trình trong vài giây?

delay(<Số Mili Giây>);

* Ví dụ

delay(2000);

* Đây là 1 hàm Interrupt

1. Tạm Dừng Chương Trình Trong Tích Tắc?

delayMicroseconds(<Số Micro Giây>);

* Thời gian tạm dừng tối đa là 16383 Micro giây, nếu <Số Micro Giây> > 16383, thì thời gian dừng sẽ quy đổi về dưới 16383 Micro giây một cách không xác định
* Ví dụ

delayMicroseconds(10);

* Đây không phải hàm Interrupt

1. Trả Về Số Mili Giây Đã Trôi Qua Kể Từ Lúc Arduino Bắt Đầu Chạy?

<Số Mili Giây> = millis();

* Đây là 1 hàm Interrupt
* Khi Arduino chạy gần 50 ngày, số Mili giây được Reset, <Số Mili Giây> là số nguyên
* Để trả về số Micro giây đã trôi qua

<Số Micro Giây> = micros();

* Đây không phải hàm Interrupt, nó dựa trực tiếp vào Clock gốc Arduino
* Khi Arduino chạy gần 70 phút, số Micro giây được Reset, <Số Micro Giây> là số nguyên

C++:

1. Kiểu Dữ Liệu Thay Thế?

* byte = unsigned char

1. Clip Dữ Liệu?

<Giá Trị Clip> = constrain(<Giá Trị Gốc>, <Cận Dưới>, <Cận Trên>);

* Ví dụ

foo = constrain(4, 5, 9);

* Ta có

|  |  |
| --- | --- |
| foo | 5 |

Interrupt:

1. Bản Chất?

* Arduino có Clock gốc độc lập, tần số 16MHz, được đi qua 5 Prescaler là chia 1, chia 8, chia 64, chia 256, chia 1024
* Khi đi qua Prescaler, tần số Clock gốc sẽ bị chia bao nhiêu đó tùy theo bạn muốn chia bao nhiêu, mục đích là để bạn đếm được lâu hơn nhưng bù lại độ chính xác thấp hơn
* Interrupt là 1 hàm mà khi có 1 sự kiện nào đó xảy ra, CPU sẽ tạm dừng công việc hiện tại để chạy hàm này, sau đó quay lại thực hiện tiếp công việc
* Nếu khi đang chạy 1 hàm Interrupt mà gặp 1 hàm Interrupt khác, hàm Interrupt khác bị bỏ qua
* Sơ đồ kí hiệu Pin Arduino UNO

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Chữ ghi trên Pin | Chân trên CPU | Kí hiệu Digital | Kí hiệu Port | Kí hiệu Pin Change |
| 0 | 2 | D0 | PD0 | PCINT16 |
| 1 | 3 | D1 | PD1 | PCINT17 |
| 2 | 4 | D2 | PD2 | PCINT18 |
| 3 | 5 | D3 | PD3 | PCINT19 |
| 4 | 6 | D4 | PD4 | PCINT20 |
| 5 | 11 | D5 | PD5 | PCINT21 |
| 6 | 12 | D6 | PD6 | PCINT22 |
| 7 | 13 | D7 | PD7 | PCINT23 |
|  |  |  |  |  |
| 8 | 14 | D8 | PB0 | PCINT0 |
| 9 | 15 | D9 | PB1 | PCINT1 |
| 10 | 16 | D10 | PB2 | PCINT2 |
| 11 | 17 | D11 | PB3 | PCINT3 |
| 12 | 18 | D12 | PB4 | PCINT4 |
| 13 | 19 | D13 | PB5 | PCINT5 |
|  |  |  |  |  |
| A0 | 23 | D14 | PC0 | PCINT8 |
| A1 | 24 | D15 | PC1 | PCINT9 |
| A2 | 25 | D16 | PC2 | PCINT10 |
| A3 | 26 | D17 | PC3 | PCINT11 |
| A4 | 27 | D18 | PC4 | PCINT12 |
| A5 | 28 | D19 | PC5 | PCINT13 |
|  |  |  |  |  |

* Các bộ Counter trong Arduino UNO

|  |  |  |
| --- | --- | --- |
| Các thanh ghi của bộ 1 | Các thanh ghi của bộ 2 | Các thanh ghi của bộ 3 |
| TCNT0 (8 Bit) | TCNT1 (16 Bit) | TCNT2 (8 Bit) |
|  | TCCR1A |  |
|  | TCCR1B |  |
|  | TIMSK1 |  |
|  | OCR1A |  |
|  |  |  |
|  |  |  |

* Ở đây ta xét bộ 2
* OCR1A gồm 16 Bit, mặc định = 0
* TCNT1 là Counter 16 Bit, nó được gắn với Prescaler do bạn chọn, có thể gán trực tiếp 1 giá trị nào đó cho nó và tiếp tục đếm từ giá trị này
* Để chọn Prescaler chia 1, hay chia 8, …, chỉnh các Bit của TCCR1B, cấu trúc của nó như sau, giá trị mặc định các Bit = 00000011

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|  |  |  | WGM13 | WGM12 | CS12 | CS11 | CS10 |

* Bảng thực trị

|  |  |  |  |
| --- | --- | --- | --- |
| CS12 | CS11 | CS10 | Prescaler |
| 0 | 0 | 1 | Chia 1 |
| 0 | 1 | 0 | Chia 8 |
| 0 | 1 | 1 | Chia 64 |
| 1 | 0 | 0 | Chia 256 |
| 1 | 0 | 1 | Chia 1024 |

* Để chỉnh chế độ đếm, tức là đếm tới 65535 rồi về 0 ngay lập tức hay là đếm tới 65535 rồi đếm ngược về 0, chỉnh các Bit của TCCR1A và TCCR1B, mặc định TCCR1A = 00000001, cấu trúc TCCR1A như sau

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|  |  |  |  |  |  | WGM11 | WGM10 |

* Bảng thực trị

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| WGM13 | WGM12 | WGM11 | WGM10 | Chế độ |
| 0 | 0 | 0 | 0 | Đếm từ 0 tới 65535 rồi về ngay lập tức về 0 |
| 0 | 1 | 0 | 0 | Đếm từ 0 tới giá trị của OCR1A rồi ngay lập tức về 0 |

* Để cho phép Interrupt khi TCNT1 tràn hoặc chạm tới giá trị OCR1A, …, chỉnh các Bit của TIMSK1, giá trị mặc định = 00000000, cấu trúc như sau

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|  |  |  |  |  |  | OCIE1A | TOIE1 |

* Khi TOIE1 = 0, giá trị của Vector TIMER1\_OVF\_vect luôn = False
* Khi TOIE1 = 1, giá trị của TIMER1\_OVF\_vect sẽ = True khi TCNT1 từ 65535 ngay lập tức về 0, và các thời điểm khác = False
* Khi OCIE1A = 0, giá trị của Vector TIMER1\_COMPA\_vect luôn = False
* Khi OCIE1A = 1, giá trị của TIMER1\_COMPA\_vect sẽ = True khi TCNT1 từ OCR1A ngay lập tức về 0, và các thời điểm khác = False
* Các thanh ghi Pin Change Interrupt

|  |
| --- |
| PCICR |
| PCMSK0 |
| PCMSK1 |
| PCMSK2 |
|  |
|  |
|  |
|  |

* Cả PCMSK0, PCMSK1, PCMSK2 mặc định = 00000000
* Cấu trúc PCMSK0

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| PCINT7 | PCINT6 | PCINT5 | PCINT4 | PCINT3 | PCINT2 | PCINT1 | PCINT0 |

* Cấu trúc PCMSK1

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|  | PCINT14 | PCINT13 | PCINT12 | PCINT11 | PCINT10 | PCINT9 | PCINT8 |

* Cấu trúc PCMSK2

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| PCINT23 | PCINT22 | PCINT21 | PCINT20 | PCINT19 | PCINT18 | PCINT17 | PCINT16 |

* PCICR mặc định = 00000000, cấu trúc như sau

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|  |  |  |  |  | PCIE2 | PCIE1 | PCIE0 |

* Như vậy 1 Pin sẽ thuộc 1 trong 3 nhóm là 0, 1 và 2, gọi Index của nhóm là <n>
* Vector PCINT<n>\_vect sẽ luôn = False, nó chỉ ở trạng thái True ở thời khắc 1 Pin thuộc nhóm <n> thay đổi từ 0 lên 5 hoặc từ 5 xuống 0 V và Bit tương ứng với Pin này trong PCMSK<n> = 1

1. Trả Về Mã External Interrupt Của 1 Pin?

<Mã External Interrupt> = digitalPinToInterrupt(<Số Của Pin>);

* Mã External Interrupt của mỗi Pin sẽ khác nhau ở nhưng Board khác nhau
* Đối Với UNO

|  |  |
| --- | --- |
| <Số Của Pin> | <Mã External Interrupt> |
| 2 | 0 |
| 3 | 1 |
| Giá trị khác 2 và 3 | –1 |

1. Tạo Hàm External Interrupt?

attachInterrupt(<Mã Interrupt>, <Hàm>, <Chế Độ>);

* Khi lệnh trên được chạy, sẽ có 1 cái Event Listener được tạo ra, mỗi khi Pin ứng với <Mã Interrupt> được kích, <Hàm> sẽ được chạy ngay lập tức

|  |  |
| --- | --- |
| <Chế Độ> | Kích khi nào |
| 1 | Khi Pin chuyển từ 0 lên 5 hoặc 5 về 0 V |
| 2 | Khi Pin chuyển từ 0 lên 5 V |
| 3 | Khi Pin chuyển từ 5 về 0 V |

1. Tạo Hàm Internal Interrupt?

ISR (<Interrupt Vector>){  
 <Làm Gì Đó>

}

* <Interrupt Vector> là 1 Vector, nếu = True thì <Làm Gì Đó> được thực hiện còn không thì thôi, ví dụ TIMER1\_OVF\_vect

1. Ngăn Không Cho Interrupt?

* Interrupt chỉ có hiệu lực nếu cờ Interrupt trong CPU là True, nếu False, thì bạn có kích kiểu gì thì CPU vẫn đéo quan tâm
* Để chỉnh cờ Interrupt về False

cli();

* Để chỉnh cờ Interrupt về True

sei();

Memory – Bộ Nhớ:

1. EEPROM?

* Là bộ nhớ giữ nguyên khi tắt hay Reset máy
* Để Import thư viện làm việc với EEPROM

#include <EEPROM.h>

* Để trả về số Byte của EEPROM

EEPROM.length()

* Arduino UNO có dung lượng EEPROM = 1024 Byte, mặc định khi mới mua về giá trị mỗi Byte = 11111111

1. Đọc Giá Trị 1 Ô Byte Trong EEPROM?

<Giá Trị Nguyên> = EEPROM.read(<Index>);

* <Index> phải là số nguyên từ 0 đến số Byte của EEPROM – 1
* <Giá Trị Nguyên> = quy đổi nhị phân thành nguyên không dấu

1. Ghi Giá Trị Vào 1 Ô Byte Trong EEPROM?

EEPROM.write(<Index>, <Giá Trị>);

* <Giá Trị> sẽ được quy đổi về nhị phân rồi mới ghi vào ô Byte thứ <Index>

Rules

Set Input Digital Pin thành HIGH để tạo Pull Up Resistor, khi mở thì 1 khi đóng thì 0

Serial Graph plot kết quả in trên Terminal, dùng dấu phẩy ngăn cách giữa các giá trị

giữa các Channel

Library

Servo.h

Servo

Usage:

Khởi tạo Servo Object

Ex:

Servo foo

Object:

attach(Pin Code)

Usage:

Kết nối với Pin điều khiển Servo

Ex:

foo.attach(13)

write(Angle)

Usage:

Quay Servo

Quay hơi chậm nên có thể bị ghi đè

Ex:

foo.write(165)

Stepper.h

Stepper Object(Steps, Pin Code \* 4)

Usage:

Khởi tạo Stepper Object

Ex:

Stepper foo(2048, 8, 10, 9, 11)

Object:

setSpeed(RPM)

Usage:

Chỉnh tốc độ Stepper Motor

Ex:

foo.setSpeed(3)

step(Steps)

Usage:

Quay Stepper Motor với số bước nào đó

Ex:

foo.step(1024)

LiquidCrystal.h

LiquidCrystal Object(RS Pin Code, E Pin Code, 4 \* Data Pin Code)

Usage:

Khởi tạo LCD Object

Ex:

LiquidCrystal lcd(2, 3, 4, 5, 6, 7)

Object:

begin(Columns, Rows)

Usage:

Bắt đầu làm việc với màn hình LCD với

kích thước để hiển thị kí tự là Columns \* Rows

Ex:

lcd.begin(16, 2)

print(Text)

Usage:

In chữ ra màn hình LCD

Ex:

lcd.write(“Hello”)

setCursor(Column, Row)

Usage:

Thay đổi vị trí con trỏ

Ex:

lcd.setCursor(2, 1)

clear()

Usage:

Xóa màn hình và đưa con trỏ trở về vị trí đầu

Ex:

lcd.clear()

DHT.h

DHT Object(Pin Code, Type)

Usage:

Khởi tạo DHT Object

Ex:

DHT dht(2, DHT11)

Object:

begin()

Usage:

Bắt đầu cảm biến

Ex:

dht.begin()

readHumidity()

Usage:

Trả về phần trăm độ ẩm không khí

Ex:

dht.readHumidity() 55.0

readTemperature(Is Fahrenheit)

Usage:

Trả về nhiệt độ theo đơn vị độ C hoặc F

Ex:

dht.readTemperature(False) 26.0

IRremote.h

IrReceiver

begin(Pin Code)

Usage:

Bắt làm việc với IR Sensor

Ex:

IrReceiver.begin(13)

decode()

Usage:

Trả về giá trị Decode

Decode là 0 nếu đang trong trạng thái hứng sóng

Decode là 1 nếu đã hứng được sóng

Ex:

IrReceiver.decode() 1

resume()

Usage:

Đưa giá trị Decode về 0

Ex:

IrReceiver.resume()

IrReceiver.decode() 0

decodedIRData

decodedRawData

Usage:

Trả về mã Code của sóng đã nhận được hiện

tại

Trả về 0 nếu không có mã Code sóng trong bộ

nhớ

Ex:

IrReceiver.decodedIRData.decodedRawData

4010852096

shiftOut(Data Pin Code, Clock Pin Code, LSBFIRST | MSBFIRST, Byte):

Usage:

Truyền dữ liệu Byte theo kiểu liên tục mỗi Bit mỗi Clock

LSBFIRST, MSBFIRST là ưu tiên truyền Bit ở bên phải hay trái trước

Ex:

shiftOut(13, 12, LSBFIRST, 0b10111001)

pulseIn(Pin Code, Signal)

Usage:

Tính thời gian Pulse ở Input Pin

Nếu Signal là HIGH, chờ Input Pin từ LOW lên HIGH, sau đó bắt đầu tính

thời gian cho đến khi Input Pin xuống LOW

Trả về thời gian với đơn vị

Ex:

pulseIn(13, HIGH) 308