*/\**

*Thông báo ngày 11/4/2018*

*Bài viết này đã cũ ,Thư viện hiện đã được chỉnh sửa để truyền tin an toàn hơn theo link nàyi :*

*<link bài mới> http://arduino.vn/tutorial/5897-thu-vien-truyen-du-lieu-bat-ky-byte-long-float-double-theo-goi-tin-cho-arduino*

*Bài viết này được giữ lại dưới dạng tài liệu tham khảo.*

*\*/*

*< Tác giả PhùngThái Sơn, Hà NAM, Việt Nam. Trang web: arduino.vn/NickChung>*

*<Mình xin giữ bản quyền tài liệu này, bạn có thể sử dụng tài cho mọi mục đích nhưng cần trích thông tin tác giả, cảm ơn bạn đã đón đọc, hi vọng bài viết thực sự có ích cho bạn!>*

https://github.com/NickChungVietNam/UART\_ARDUINO

http://arduino.vn/tutorial/1471-truyen-cac-so-kieu-long-int-float-trong-giao-tiep-serial-uart

Bạn đang dùng bản nâng cấp 13/6/2017

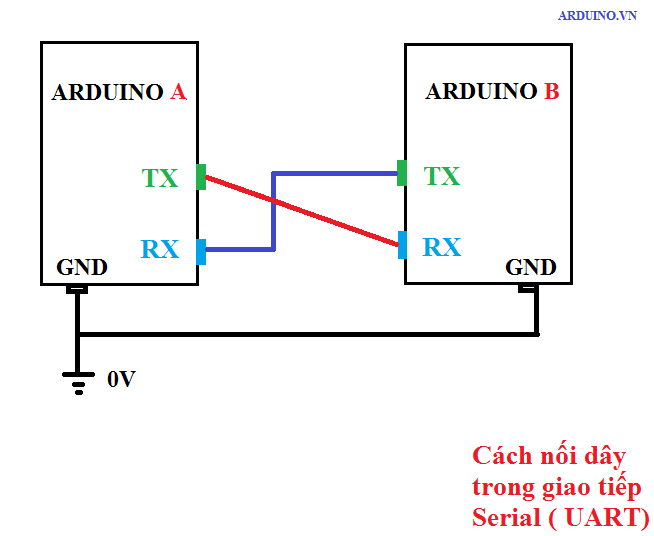
// hỗ trợ cho tất cả các phiên bản arduino, ESP8266, ESP32

Lời dẫn.

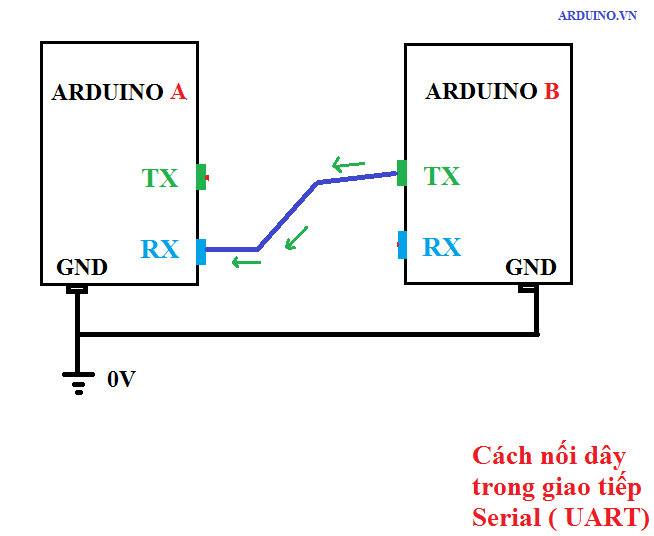
Rất dễ dàng để gửi một số hoặc chuỗi kí tự để hiển thị lên màn hình qua cổng Serial . Nhưng mọi chuyện không đơn giản như vậy khi ta muốn truyền thông số giữa 2 board arduino với nhau bằng Serial. Đừng lo, sau đây mình sẽ giúp bạn giải quyết vấn đề nan giải đó.

Giao tiếp Serial giữa 2 arduino .

Ai cũng biết giao tiếp Serial có tên gọi chính thức là giao tiếp UART, sử dụng 2 đường dây là TX ( đường xuất dữ liệu) và RX (đường nhận dữ liệu). Để kết nối ta phải nối theo quy tắc TX của board phát nối với RX còn lại của board nhận.



Vì giao tiếp này không sử dụng xung Clock,nên để truyền nhận chính xác thì 2 module phải thống nhất về tốc độ xung nhịp (như một dạng cộng hưởng xung nhịp). Điều này cho phép ta chỉ cần nối 1 dây duy nhất trong trường hợp một bên chỉ phát – bên kia chỉ thu.



Nan giải …??

Đã có những bài giới thiệu về giao tiếp Serial giữa 2 board arduino, nhưng đó chỉ là giới thiệu truyền dẫn 1 byte để demo (khi đó bạn chỉ có thể truyền đi một số có giá trị từ 0 -> 255).

Ví dụ như trong 1 dự án trạm khí tượng, bạn sẽ lên kế hoạch sử dụng 2 board arduino (1 board chính – 1 board phụ). Board chính đặt ở phòng để giám sát và điều khiển, board phụ đặt trên tròi kí tượng đọc các trị cảm biến rồi gửi về qua Serial. Hoặc pro hơn là các dự án về robot thám hiểm.

Một vấn đề mà bạn vấp phải là bạn chỉ biết chỉ biết truyền một số có giá trị trong khoảng từ 0 -> 255 (mặt cảm xúc), trong khi thực tế thì ta luôn cần nhiều hơn vậy.

Đó có thể là giá trị của biến (nhiệt , độ ẩm=90%,áp suất=2000atm, tốc độ=4500 rpm, ..v.v )một kiểu int, long, unsigned long hay kiểu kiểu số thực như float , double (ví dụ thường thấy ở các module GPS trả về như kinh độ 50.45656 độ, vĩ độ 102.55433).

Và bạn biết không, chính mình cũng đang bị chậm dự án bởi vấn đề này.

Thư viện hỗ trợ.

Ơn giời cậu đây rồi !

Đây là thư viện dành cho giao tiếp Serial giữa 2 board arduino dùng để truyền nhận tất cả các kiểu số và kí tự.

Công việc của nó cũng khá đơn giản :

Để truyền : Ví dụ để truyền số 12345 kiểu int , như ta đã biết trên arduino kiểu int có cỡ là 2 byte, ta sẽ tách 2 byte lưu trữ của biến này thành từng byte rồi xuất ra đường truyền.

Để nhận : Ta sẽ làm ngược lại là ghép 2 byte đã nhận về một số kiểu int.

Kĩ thuật này đã được mình ứng dụng trong việc lưu trữ các số (lớn hơn 255) vào EEPROM tại bài viết này:

Để hiểu hơn về phương pháp tách/ghép, bạn hãy xem mã nguồn của thư viện này nhé.

Giới thiệu thư viện.

Hiện tại thư viện có các chức truyền nhận các số kiểu:

Kiểu double : từ 2.2250738585072014 E – 308 đến 1.7976931348623158 E + 308.

void write\_double(double value);

double read\_double();

Kiểu float : từ 1.175494351 E – 38 đến 3.402823466 E + 38

void write\_float(float value);

float read\_float();

Kiểu int8\_t : từ -128 đến 127

void write\_int8\_t(int8\_t value);

int8\_t read\_int8\_t();

Kiểu uint8\_t (giống kiểu byte) : từ 0 đến 255

void write\_uint8\_t(uint8\_t value);

uint8\_t read\_uint8\_t();

Kiểu int16\_t( giống kiểu int): từ -32768 đến 32767

void write\_int16\_t(int16\_t value);

int16\_t read\_int16\_t();

Kiểu uint16\_t (giống kiểu unsigned int): từ 0 -> 65536

void write\_uint16\_t(uint16\_t value);

uint16\_t read\_uint16\_t();

Kiểu uint32\_t (giống kiểu unsigned long) :từ 0 ->4,294,967,295

void write\_uint32\_t(uint32\_t value);

uint32\_t read\_uint32\_t();

Kiểu int32\_t (giống kiểu long):từ -2,147,483,648 -> 2,147,483,647

void write\_int32\_t(int32\_t value);

int32\_t read\_int32\_t();

Kiểu unsigned char: 0->255 , tương ứng 256 kí tự trong bảng mã ASCII, mỗi lần chỉ gửi được 1 kí tự

void write\_char(unsigned char value);

unsigned char read\_char();

Gửi một chuỗi kí tự: (với n là số lượng kí tự muốn gửi (hoặc nhận))

void write\_string(unsigned char value[], uint32\_t n );

void read\_string(unsigned char \*value, uint32\_t n);

Gửi một mã dạng kí tự:

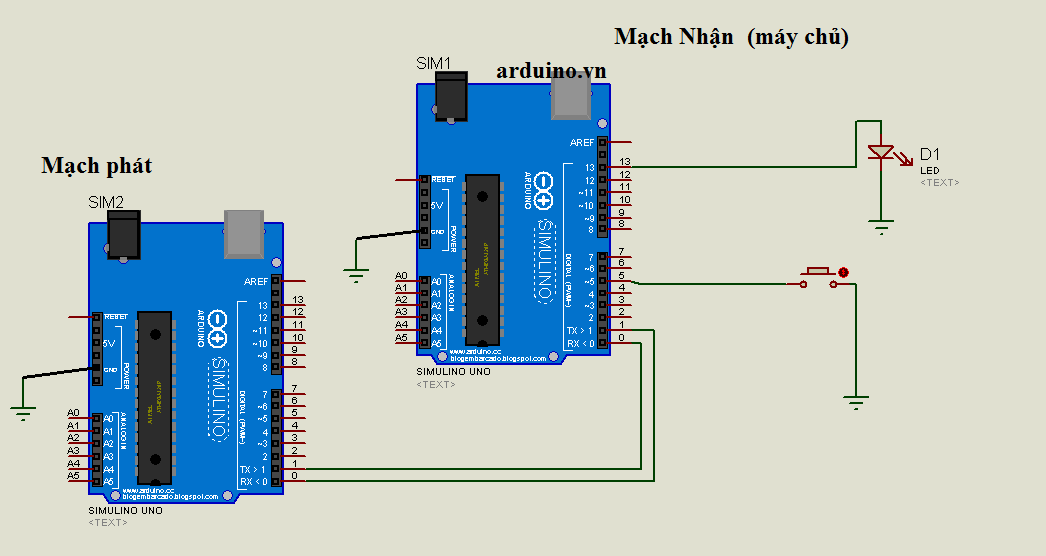
void send\_pass(unsigned char value[], uint32\_t n );

Kiểm tra xem mã có khớp không, nếu đúng thì trả về 1, sai trả về 0.

bool check\_pass(unsigned char value[], uint32\_t n );

void clear\_buffer();

Ví dụ 1: truyền số kiểu uint32\_t (unsigned long).



|  |  |
| --- | --- |
| MẠCH PHÁT | MẠCH Nhận |
| <  #include "UART\_ARDUINO.h"  UART Gui;  void setup() {  Gui.begin(9600); // baud: 200 -> 250,000.  }  uint32\_t Num=123456789;  void loop() {  if(Gui.check\_pass("abc",3)==true){  // kiểm tra có yêu cầu từ máy chủ  // Gui.clear\_buffer();// reset lại bộ nhớ đệm  Gui.write\_uint32\_t(Num);  }  }  > | <  #include "UART\_ARDUINO.h"  UART Nhan;// nhận  byte button=5;//  void setup() {  Nhan.begin(9600); // baud: 200 -> 250,000.  pinMode(13,HIGH);// PIN 13 OUT CHO LED  pinMode(button ,INPUT\_PULLUP);  }  uint32\_t Num;  void loop() {  if(digitalRead(button)==0){  delay(300);//chống nhiễu  Nhan.send\_pass("abc",3);  }  if(Serial.available()>=sizeof(Num)){  Num=Nhan.read\_uint32\_t();  if(Num==123456789){  digitalWrite(13,1);// bật đèn, báo hiệu test thành công.  }  }  }//loop  > |

Nhấn nút để yêu cầu mạch phát trả về dữ liệu

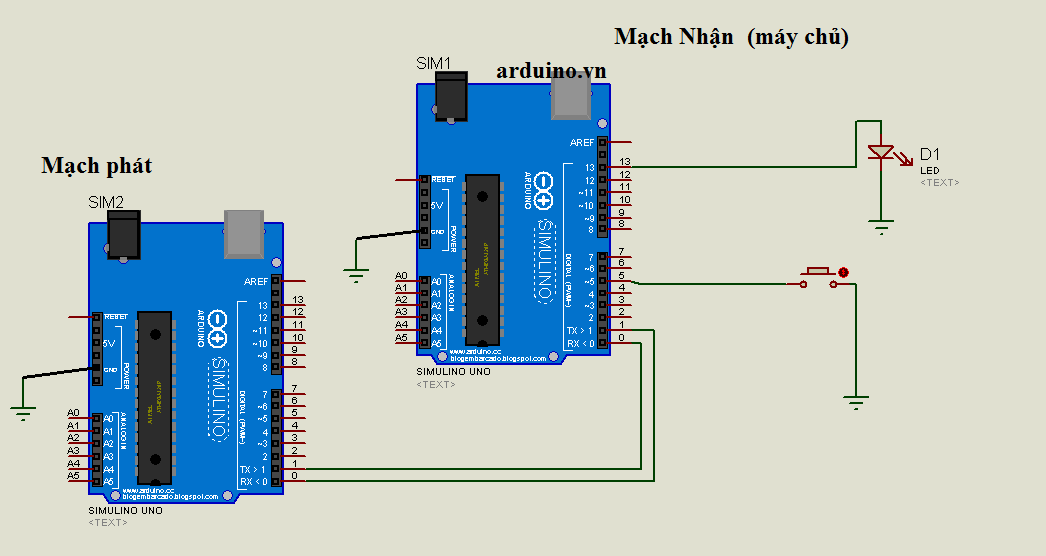
***Nhan.send\_pass("abc",3)***

Máy chủ gửi một mật khẩu yêu cầu là : abc

3 : là số lượng chữ cái

(do đó mật khẩu càng ngắn càng tốc độ nhanh hơn nhưng cũng dễ bị sai hơn(do bị trùng với số của dữ liệu))

Ví dụ 2: truyền số kiểu float



|  |  |
| --- | --- |
| MẠCH PHÁT | MẠCH Nhận |
| <  #include "UART\_ARDUINO.h"  UART Gui;  void setup() {  Gui.begin(9600); // baud: 200 -> 250,000.  }  float Num=12.345678;  void loop() {  if(Gui.check\_pass("abc",3)==true){  // kiểm tra có yêu cầu từ máy chủ  // Gui.clear\_buffer();// reset lại bộ nhớ đệm  Gui.write\_float(Num);  }  }  > | <  #include "UART\_ARDUINO.h"  UART Nhan;// nhận  byte button=5;//  void setup() {  Nhan.begin(9600); // baud: 200 -> 250,000.  pinMode(13,HIGH);// PIN 13 OUT CHO LED  pinMode(button ,INPUT\_PULLUP);  }  float Num;  void loop() {  if(digitalRead(button)==0){  // nhấn nút để yêu cầu mạch phát trả về dữ liệu  delay(300);//chống nhiễu  Nhan.send\_pass("abc",3);  }  if(Serial.available()>=sizeof(Num)){  Num=Nhan.read\_float();  if(Num==12.345678){  digitalWrite(13,1);// bật đèn, báo hiệu test thành công.  }  }  }//loop  > |

Nhấn nút để yêu cầu mạch phát trả về dữ liệu, nếu đèn sáng thì test truyền nhận thành công.

Truyền số khác:

Làm tương tự,

***void write\_double()*** từ 2.2250738585072014 E – 308 đến 1.7976931348623158 E + 308.

***void write\_float();*** từ 1.175494351 E – 38 đến 3.402823466 E + 38

***void write\_int8\_t();*** từ -128 đến 127

***void write\_uint8\_t();*** từ 0 đến 255

***void write\_int16\_t();*** từ -32768 đến 32767

***void write\_uint16\_t();*** từ 0 -> 65536

***void write\_uint32\_t();***từ 0 ->4,294,967,295

***void write\_int32\_t();*** từ -2,147,483,648 -> 2,147,483,647

Chú ý: trên arduino uno R3, kiểu float và double có cùng cỡ là 4 byte,( giá trịtừ 1.175494351 E – 38 đến 3.402823466 E + 38);

Trên arduino Due, kiểu double mới có cỡ 8 byte.

Tốc độ truyền các số có kích thước lớn (byte) sẽ chậm hơn các số có kích thước nhỏ hơn (điều này là dĩ nhiên).!

Ví dụ 3 truyền 1 kí tự trong bảng ASCII.

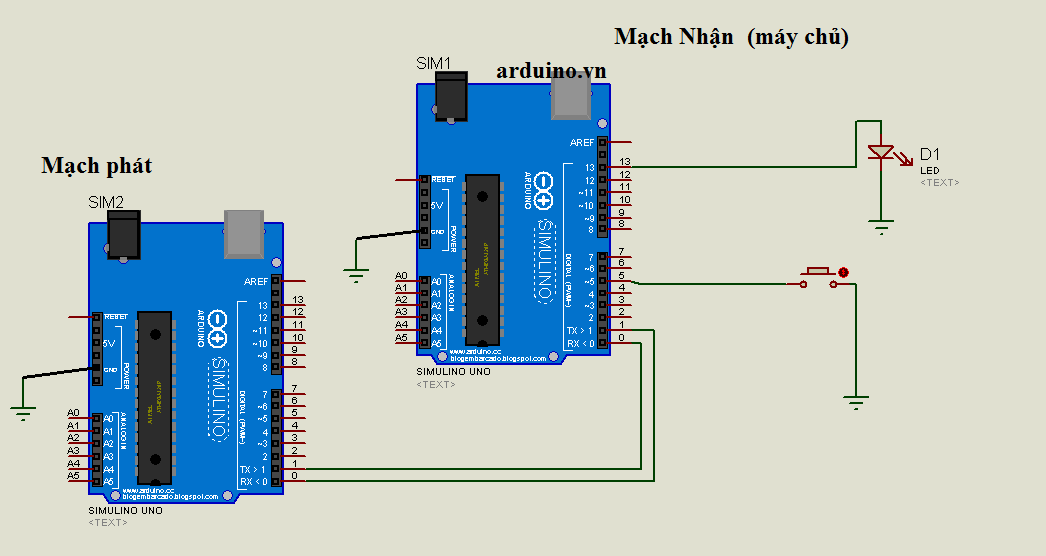
******

***Code : mở phần Examples trong file tải xuống để xem code ví dụ.!***

Ví dụ 4 :Truyền một chuỗi kí tự để nhắn tin.

***Code : mở phần Examples trong file tải xuống để xem code ví dụ.!***

Ví dụ 5 : Truyền cùng lúc nhiều số (đóng gói tập tin).

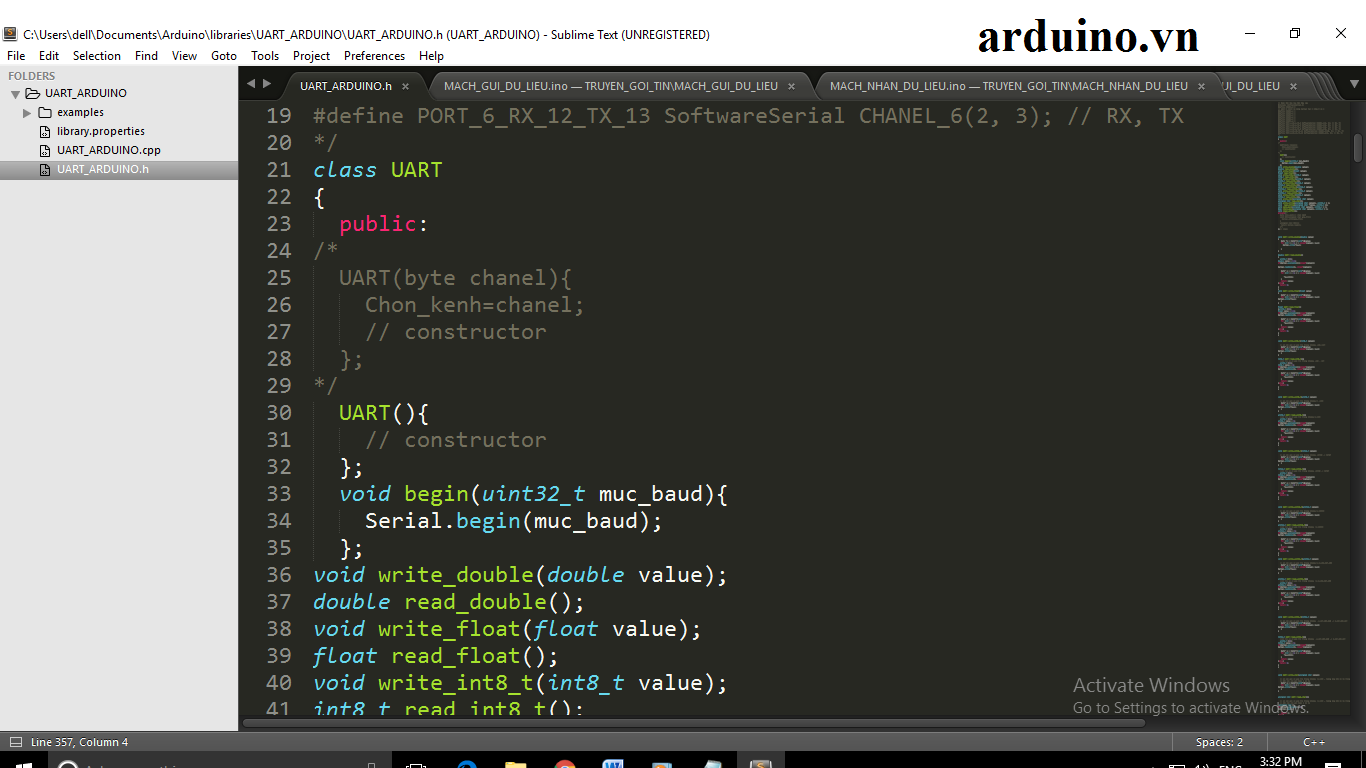


|  |  |
| --- | --- |
| Mạch phát (slave) | Mạch nhận (master). |
| <  #include "UART\_ARDUINO.h"  UART Gui;  void setup() {  Gui.begin(9600); // baud: 200 -> 250,000.  }  uint8\_t nhiet\_do;  uint16\_t do\_am;  int32\_t toc\_do;  uint32\_t do\_cao;  float toa\_do;  void loop() {    if(Gui.check\_pass("abc",3)==true){  // kiểm tra có yêu cầu từ máy chủ  gui\_du\_lieu(25,60,2700,3600,12.1234);  }  }  void gui\_du\_lieu(uint8\_t a, uint16\_t b, int32\_t c, uint32\_t d,float e){  // lưu ý : gói tin truyền đi có kích cỡ lớn nhất là 64 byte  Gui.write\_uint8\_t(a); // cần 1 byte  Gui.write\_uint16\_t(b); // cần 2 byte  Gui.write\_int32\_t(c); // cần 4 byte  Gui.write\_uint32\_t(d); // cần 4 byte  Gui.write\_float(e); // cần 4 byte  // tổng cộng cần 15 byte < (max =64)    }  > | <  #include "UART\_ARDUINO.h"  UART Nhan;// nhận  byte button=5;//  void setup() {  Nhan.begin(9600); // baud: 200 -> 250,000.  pinMode(13,HIGH);// PIN 13 OUT CHO LED  pinMode(button ,INPUT\_PULLUP);  //sử dụng một nút ấn (button) nối với pin 5 và nguồn âm  // nhấn nút để yêu cầu mạch phát trả về dữ liệu  }  uint8\_t nhiet\_do;  uint16\_t do\_am;  int32\_t toc\_do;  uint32\_t do\_cao;  float toa\_do;  void loop() {  if(digitalRead(button)==0){  // nhấn nút để yêu cầu mạch phát trả về dữ liệu  delay(300);//chống nhiễu  Nhan.send\_pass("abc",3);  //máy chủ gửi một mật khẩu yêu cầu là : abc  // 3 : là số lượng chữ cái  //(do đó mật khẩu càng ngắn càng tốc độ nhanh hơn nhưng cũng dễ bị sai hơn(do bị trùng với số của dữ liệu))  }    if(nhan\_du\_lieu(&nhiet\_do, &do\_am, &toc\_do,&do\_cao, &toa\_do)==true){    if(nhiet\_do==25){  if(do\_am==60){  if(toc\_do==2700){  if(do\_cao==3600){  if(toa\_do==12.1234){  digitalWrite(13,1);//bật đèn để báo test thử nghiệm thành công (số truyền qua là chính xác)  }  }  }  }  }  }//if lớn  }//loop  bool nhan\_du\_lieu(uint8\_t \*a, uint16\_t \*b, int32\_t \*c, uint32\_t \*d,float \*e){  //b1: tìm kích thước để kiểm tra gói tin truyền về đã đủ chưa  uint32\_t kich\_co\_goi\_tin=sizeof(\*a)+sizeof(\*b)+sizeof(\*c)+sizeof(\*d)+sizeof(\*e);  if(Serial.available()>=kich\_co\_goi\_tin){  //đã nhận đủ gói tin  /////////////  (\*a)=Nhan.read\_uint8\_t();  (\*b)=Nhan.read\_uint16\_t();  (\*c)=Nhan.read\_int32\_t();  (\*d)=Nhan.read\_uint32\_t();  (\*e)=Nhan.read\_float();  /////////////  // Nhan.clear\_buffer();// reset lại bộ nhớ đệm  return true;// cuối cùng trả về true báo đã nhận đủ và hoàn tất  }else{  return false;// báo chưa đủ  }    }  > |

Tất cả các ví dụ trên đều được mình test thành công.!

Kết :

Mình viết ra thư viện chỉ đơn thuần phục vụ nhu cầu về truyền dẫn số liệu. Tuy không theo quy tắc chuẩn của đóng gói và truyền dẫn (nó cần có thêm các khả năng như yêu cầu gửi lại, hết hạn, ghép các tập tin…). Arduino cũng hỗ trợ có thể có nhiều hơn 1 cổng Serial bằng thư viện SerialSoftWare. Nhưng thôi, tới đâu hay tới đấy, hi vọng trong tương lai mình và các bạn có thời gian để nâng cấp thêm cho nó.



<tác giả Thái Sơn>

Ngày hoàn thành : 14/2/2017 tại Hà Nam (đúng ngày lễ tình nhân Valentin các bạn ạ !)