Khung truyền thông bao gồm các thao tác:

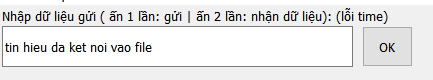
+Nhấp “auto check connect”: Khi được nhấp, tại comboBox dưới dòng “Cổng hiện có” sẽ hiện số cổng COM cho phép kết nối truyền dữ liệu. Thông thường sẽ là COM 3.

+ Tại khung bên phải là chọn thông số cho giao thức UART bao gồm: cổng kết nối, số bit dữ liệu trong một khung truyền (Databit), số bít dừng (BitStop), tốc độ truyền b/s (Baud), cờ định hướng được chọn cùng stick (DTR), bit điều khiển (FlowControl), tín hiệu có bắt tay (Parity).

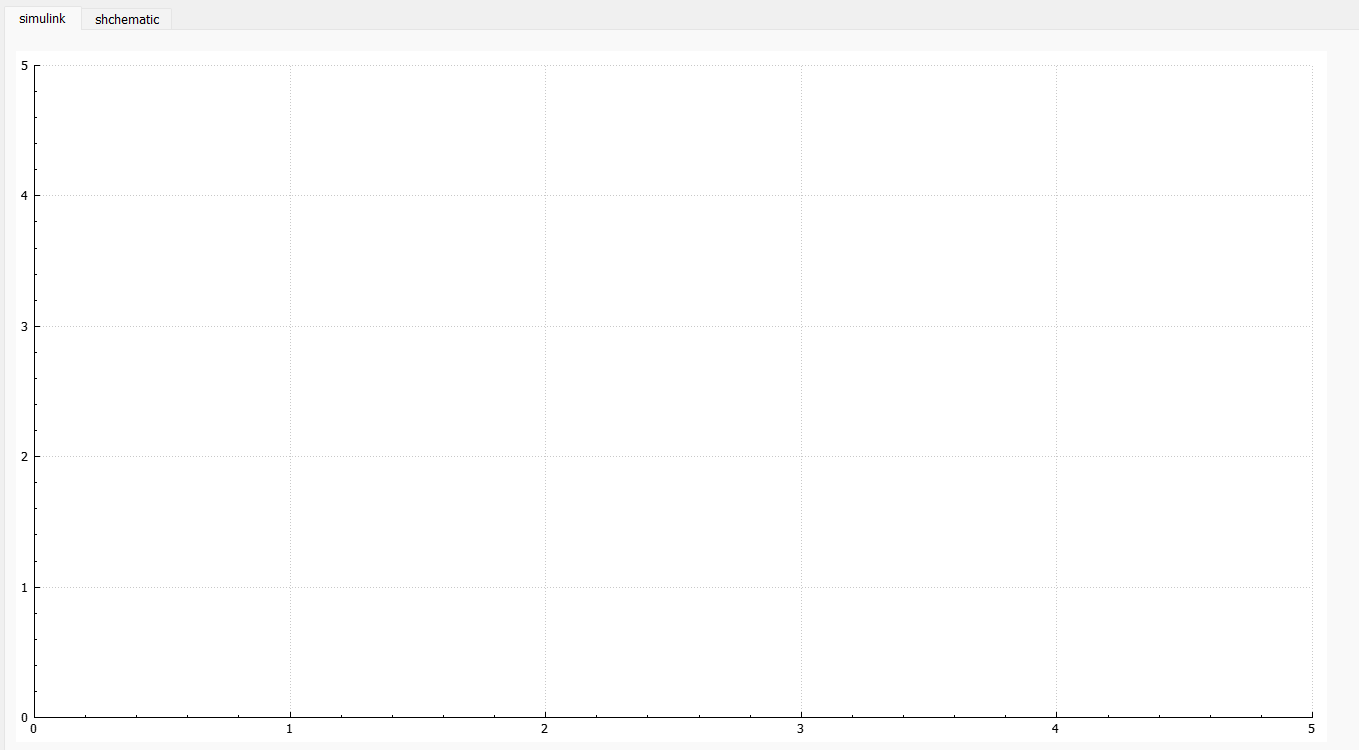
+ Sau khi chọn xong, nhấp vào button “CONNECT” cổng tương ứng sẽ xác định giao thức có các thông số truyền, nhận và hiển thị ở dòng “ket noi”.

+ Trong quá trình truyền có thể điều chỉnh độ trễ giao thức qua Spin Box dưới Combox hiện cổng có thể kết nối.

Khung Messenge: nơi thông báo các hoạt động của App bao gồm: truyền hay nhận, lưu file, kiểm tra, tiến trình. Ngoài ra cũng sẽ thông báo các vấn đề sau khi so sánh tín hiệu mẫu và thu được.



Được tạo ở phần kiểm tra App có chức năng truyền tín hiệu đi. Tại đây khi phát triển cũng là nơi thay thế và dùng để hiển thị các thao tác để truyền, nhận tín hiệu mẫu theo ý tưởng thực hiện đề tài.



Khung này bao gồm việc hiển thị tín hiệu, kiểm tra giữa hai tín hiệu “Mẫu” chuẩn và tín hiệu thu về từ bảng mạch (simulink) và hiện thị bảng mạch chỉ thị vị trí sai (nếu được) của bảng mạch. Trên trình hiển thị của simulink cho phép chức năng phóng to và thu nhỏ bằng chuột, điều này cho phép người dùng thuận tiện trong việc xem tín hiệu.

**Code:**

App được xây dựng từ QT một môi trường lập trình cung cấp các phương tiện, lớp thư viện của C++ và Python để xây dựng app cho Mobile hay PC.

Đề tài được lấy ý tưởng xây dựng ban đầu từ opensource của pulseview một Application dùng để hiển thị các tín hiệu digital và analog thu được thông qua kết nối USB. Nó giúp máy tính giống một máy hiện sóng thông thường. Nhờ tính năng này, ban đầu đã lấy nó làm cơ sở để phát triển ứng dụng nhưng do tính phức tạp được xây dựng từ Cmake không có hỗ trợ lớn từ qml. Vì thế nhóm đã định hướng sang các Example của Qt với UI có sẵn và các ví dụ về SerialPort để xây dựng app từ các thư viện hỗ trợ đồ họa của Qt lấy từ thư viện mẹ “QWidget” và kết nối “QserialPort”.

UI hỗ trợ rất lớn các công cụ xây dựng giao diện như LabView để xây dựng App. Với sự hỗ trợ mạnh mẽ từ các lớp với thư viện tương ứng cùng các thành phần hàm, kiểu xây dựng sẵn có thể đọc trên qt.doc. Giúp nhóm nhanh chóng tìm hiểu và liên kết các giao diện con và tạo chức năng tương ứng cho giao diện đó.

1.Kết nối.

Nhóm sử dụng lớp “QSerialPort” của Qt với các cung cấp kết nối với giao thức qua cổng COM trên máy tính, cho phép người dùng chọn cổng, baud truyền, số bít dữ liệu trong khung truyền, số bít dừng một khung, số bít điều khiển, cờ định hướng, tín hiệu bắt tay. Hiện QserialPort có thể hỗ trợ tốc độ truyền tương ứng của RS232 kiểu cũ cao nhất là 115200 bms. Tuy hạn chế nhưng cũng đảm bảo yêu cầu làm việc cho kiểm tra với phần cứng.

Phần code kết nối dùng cho làm việc với serialport được dựng trong lớp “serialportcnt.h” và “serialportcnt.cpp”. Bao gồm các hàm định nghĩa lại và tự định nghĩa như:

+Kiểu boolean begin(…), end(), send(…), recevei() có nhiệm vụ báo kết nối đã được set chưa, gửi tín hiệu kết thúc truyền nối tiếp, gửi – nhận tín hiệu.

+ Kiểu String setSerialInfo(), readAll() dùng để lên danh sách cổng đang kết nối và gửi thông tin của cổng được chọn để thiết lập kết nối truyền nối tiếp.

2.Hiện thị

Hiển thị nhóm dùng lớp “QMainWindow” được QT hỗ trợ cho window là đối tượng chính để kết nối. Ngoài ra có thể chuyển đổi để sử dụng môi trường tương ứng cho IOS, Linux sẽ được cấu hình tại file .pro (Hiện tại chỉ dừng ở window).

QMainWindow cung cấp một GUI mạnh cho window với khả năng làm như một trình giao diện kế thừa từ lớp mẹ QOBJECT là tiền thân cho mọi lớp trong QT. Với sự hỗ trợ mạnh mẽ về giao diện người dùng bằng UI. Người thiết kế có thể xây dựng nhanh chóng một giao diện có nhiều chức năng ví dụ như ComboBox, Button, Graphic, ProcessBar,…

Nhóm sử dụng thiết kế trên giao diện từ mainwindow.ui để thuận tiện quan sát và thao thác nhanh các chức năng của phần mềm.

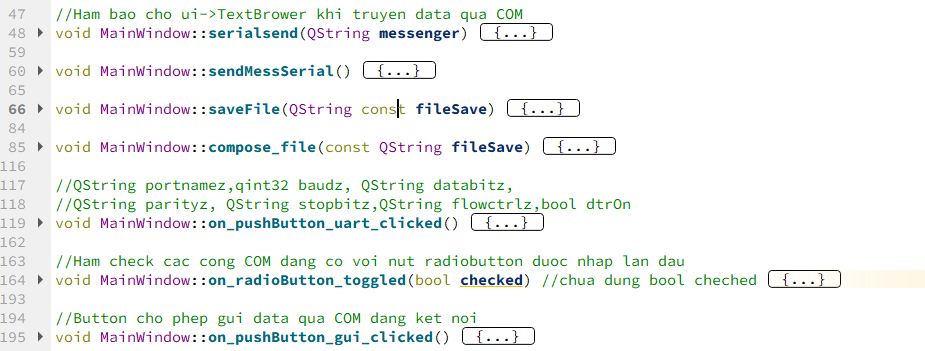
a.Khung dành cho kết nối:

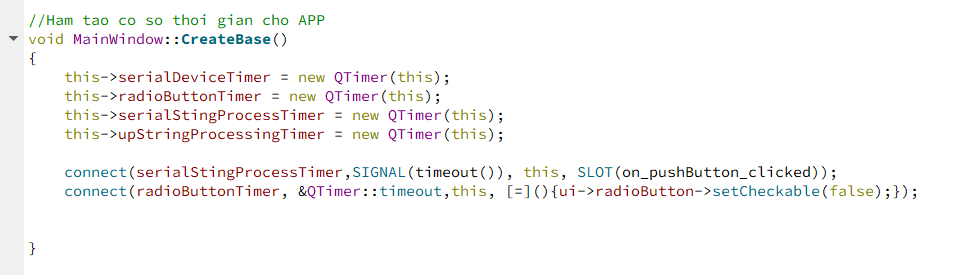
Bao gồm các ComboBox để chọn thuộc tính kết nối:

* ComboBox(danh sách cổng hiện có),
* comboBox\_port(chọn cổng),
* comboBox\_baud(chọn tốc độ truyền),
* comboBox\_bitstop(chọn số bít dừng),
* comboBox\_databit(chọn số bit dữ liệu trong một khung,
* comboBox\_flowcontrol(chọn số bit điều khiển),
* comboBox\_partity(tín hiệu bắt tay);

Ngoài ra còn có :

* radionButton(dùng để tự kiểm tra cổng kết nối sẽ hiển thị danh sách cho comboBox với thời gian lặp là 300 ms);
* spinBoxProcessingDelay(chọn độ trễ cho kết nối);
* checkBoxDTR( dùng để chọn có cờ định hướng) ;
* pushButton\_uart: khi ấn sẽ là xác nhận thông tin cổng từ các comboxBox và mở cổng truyền. Lúc này nút sẽ hiện chữ “DISCONNECT”, khi ấn lần nữa sẽ đóng cổng, reset thuộc tính kết nối và hiện chữ “CONNECT”.

 Cơ sở thời gian của toàn bộ app được xây dựng từ hàm CreateBase():



b. Đồ thị và bảng mạch mô phỏng chức năng.

QT có hỗ trợ một wigdet được xây dựng tối ưu hóa cho vẽ biểu đồ và đồ thị một cách trực quan và đẹp nhất bằng thư viện “qcustomplot”. Thư viện được nhóm dùng để xây dựng đồ thị tín hiệu trên widgetsimulink nằm trong khung của tabWidget trên UI. Vì gọi ‘ui->widgetsimulink->…’ không ảnh hưởng gì đến tabWidget. Tại đây, tabWidget chỉ có nhiệm vụ chuyển đổi widget trong nó giữa hai phần hiển thị là simulink( ứng với tín hiệu) và schematic( ứng với bảng mạch được mô phỏng).

Trên simulink, vì tín hiệu bảng nhớ là các tín hiệu nhị phân nên việc hiển thị nhóm hiển thị dạng xung vuông có tín hiệu mức 1 và 0.

Trên schmatic, nhóm dùng các label để hiện thị phần tử IC trên bảng mạch với lớp Qpixmap cho phép hiển thị ảnh và hàm với label.move(x,y) để hiện thị vị trí của nó trên schmatic. Khi so sánh tín hiệu báo sai thì sẽ thay label là một ảnh IC có màu đỏ để báo nó sai.

c.Thông báo.

Tiện cho việc kiểm tra quá trình thực hiện chức năng, nhóm xây dựng một textBrowser giúp thông báo như một ‘applicate out’ để báo cáo quá trình như: kết nối, lưu file tín hiệu, báo cáo lỗi sai,…

Hàm addMes(QString text, bool agrT) sẽ viết tiếp vào textBrowser ứng với hiện thị là khung dưới Messenger chuỗi ‘text’ khi biểu thức boolean agrT là đúng còn sai thì không báo.

