{Trang trắng này dùng để dán bản Nhận xét của người hướng dẫn, hoặc thay trang  
này bằng Nhận xét của người hướng dẫn}

{Trang trắng này dùng để dán bản Nhận xét của người phản biện, hoặc thay trang này bằng Nhận xét của người phản biện}

# **TÓM TẮT**

# **NHIỆM VỤ ĐỒ ÁN TỐT NGHIỆP**

# **LỜI NÓI ĐẦU**

Trong thời gian hơn 4 tháng thực hiện Capstone Project tại FPT Software, em đã được tiếp xúc, làm quen với môi trường làm việc tại công ty, cũng như học hỏi được nhiều kiến thức bổ ích, tích lũy thêm kinh nghiệm để chuẩn bị những hành trang cho bản thân nhằm đáp ứng nhu cầu tuyển dụng của các doanh nghiệp sau khi ra trường.

Em xin chân thành cảm ơn FPT Software và lãnh đạo Khoa Điện tử - Viễn thông đã tạo điều kiện để em được tham gia và thực hiện đồ án tốt nghiệp dưới hình thức Capstone Project. Em xin chân thành cảm ơn giảng viên hướng dẫn - cô Trần Thị Hương và hơn hết là anh Nguyễn Văn Thành – FPT Software đã quan tâm, hướng dẫn nhiệt tình để em có thể hoàn thành được đồ án này.

*+ Giới thiệu đề tài*

*+ Tính cấp thiết của đề tài.*

*+ Kết quả mong muốn đạt được. (phạm vị hướng tới)*

*+ Phạm vi đề tài (mình làm)*

*+ Tóm tắt chương*

*+ Lời cảm ơn.*

# **LỜI CAM ĐOAN**

MỤC LỤC

[**TÓM TẮT** 3](#_Toc482584640)

[**NHIỆM VỤ ĐỒ ÁN TỐT NGHIỆP** 4](#_Toc482584641)

[**LỜI NÓI ĐẦU** 5](#_Toc482584642)

[**LỜI CAM ĐOAN** 6](#_Toc482584643)

[MỤC LỤC 6](#_Toc482584644)

[**DANH MỤC HÌNH VẼ** 9](#_Toc482584645)

[**DANH MỤC CÁC TỪ VIẾT TẮT** 10](#_Toc482584646)

[**MỞ ĐẦU** 11](#_Toc482584647)

[**CHƯƠNG 1: TỔNG QUAN HỆ THỐNG MÔ PHỎNG CAN BUS** 12](#_Toc482584648)

[1.1. Giới thiệu chương. 12](#_Toc482584649)

[1.2. Tổng quan về CAN BUS 12](#_Toc482584650)

[1.3. Hệ thống mô phỏng CAN BUS 13](#_Toc482584651)

[1.4. Các thành phần của hệ thống. 15](#_Toc482584652)

[1.4.1. CAN Software 15](#_Toc482584653)

[1.4.1.1. Phần mềm mô phỏng CAN. 16](#_Toc482584654)

[1.4.1.2. Phần mềm cơ sở dữ liệu CAN. 16](#_Toc482584655)

[1.4.2. CAN Device 17](#_Toc482584656)

[1.4.3. PC Driver và Protocol 18](#_Toc482584657)

[1.5. Kết luận chương (nhắc lại phạm vi đề tài lần nữa). 19](#_Toc482584658)

[CHƯƠNG 2: CƠ SỞ LÝ THUYẾT VÀ MÔI TRƯỜNG THỰC HIỆN (Trang 19) 20](#_Toc482584659)

[2.1. Giới thiệu chương. 20](#_Toc482584660)

[2.2. Công cụ thực hiện. 20](#_Toc482584661)

[2.2.1. Môi trường hệ điều hành Linux. 20](#_Toc482584662)

[2.2.2. Giới thiệu chung về Qt Creater Framework. 21](#_Toc482584663)

[2.2.3. Các ưu điểm của QT Framework. 22](#_Toc482584664)

[2.3. Phần mềm mô phỏng CAN 22](#_Toc482584665)

[2.3.1. Tổng quan phần mềm. 22](#_Toc482584666)

[2.3.2. Tiến trình thực hiện. 23](#_Toc482584667)

[2.3.3. Thiết kế giao diện. 23](#_Toc482584668)

[2.3.4. Giao tiếp các khối(Parent – Child). 27](#_Toc482584669)

[2.3.5. Các nhánh trên giao diện phần mềm. 27](#_Toc482584670)

[2.3.5.1. Nhánh Generator. 27](#_Toc482584671)

[2.3.5.2. Nhánh Hardware/Main. 28](#_Toc482584672)

[2.3.5.3. Nhánh View. 28](#_Toc482584673)

[2.3.6. Quản lý dữ liệu. 29](#_Toc482584674)

[2.3.6.1. Luồng dữ liệu hệ thống. 29](#_Toc482584675)

[2.3.6.2. Luồng dữ liệu Interative Generator Block. 30](#_Toc482584676)

[2.3.6.3. Luồng dữ liệu Filter Block. 30](#_Toc482584677)

[2.3.6.4. Luồng dữ liệu Hardware Block. 31](#_Toc482584678)

[2.3.6.5. Luồng dữ liệu Trace Block 32](#_Toc482584679)

[2.4. Cấu hình dữ liệu. 32](#_Toc482584680)

[2.4.1. Giới thiệu về Json. 32](#_Toc482584681)

[2.4.2. QJson trong QT framework. 32](#_Toc482584682)

[2.5. Kết luận chương. 34](#_Toc482584683)

[CHƯƠNG 3: KẾT QUẢ THỰC HIỆN VÀ ĐÁNH GIÁ (Trang 33) 35](#_Toc482584684)

[3.1. Giới thiệu chương. 35](#_Toc482584685)

[3.2. Bảng phân công công việc. 35](#_Toc482584686)

[3.3. Giao diện Simulator software. 36](#_Toc482584687)

[3.3.1. Giao diện các hộp thoại. 36](#_Toc482584688)

[3.3.2. Giao diện khối IG 36](#_Toc482584689)

[3.3.3. Giao diện khối Filter 37](#_Toc482584690)

[3.3.4. Giao diện khối Hardware 39](#_Toc482584691)

[3.3.5. Giao diện khối Trace 40](#_Toc482584692)

[3.4. Edit fie Cấu hình theo định dạng json 41](#_Toc482584693)

[3.5. Hoạt động phần mềm mô phỏng CAN. (trang 41 - 45) 42](#_Toc482584694)

[3.6. Kết quả nghiệm thu.(trang 44- 50) 46](#_Toc482584695)

[3.7. Kết luận chương. 50](#_Toc482584696)

[KẾT LUẬN (Trang 50**)** 51](#_Toc482584697)

[HƯỚNG PHÁT TRIỂN ĐỀ TÀI (Trang 51) 52](#_Toc482584698)

[TÀI LIỆU THAM KHẢO (Trang 52) 53](#_Toc482584699)

[PHỤ LỤC (Code) (Trang 53...) 54](#_Toc482584700)

# **DANH MỤC HÌNH VẼ**

# **DANH MỤC CÁC TỪ VIẾT TẮT**

# **MỞ ĐẦU**

Cùng với sự phát triển không ngừng của khoa học công nghệ, công nghiệp ôtô cũng đã và đang được hoàn thiện nhằm mục đích đem đến cho người dùng những trải nghiệm ngày càng thoải mái và tiện nghi hơn. Sự ra đời của các công nghệ, thiết bị hiện đại như cảnh báo chệch làn đường, cảnh báo tiền va chạm, hệ thống hỗ trợ phanh và điều khiển động cơ bánh lái…đã mang lại cho người dùng những chuyến hành trình ngày càng thú vị. Tuy nhiên, để áp dụng những công nghệ mới này lên trên một sản phẩm xe ô tô thực tế để đưa ra ngoài thị trường thì từ khâu kiểm thử, vận hành cho đến phát triển các ứng dụng về sau thì đây cũng là một khó khăn đối với các nhà nghiên cứu, cũng như sản xuất ô tô công nghiệp. Việc sử dụng cả hệ thống thực tế chỉ để kiểm thử với một ECU (Electronic Control Unit) mới nhằm hướng tới sự tương thích giữa các ECU với nhau có thể gây ra sự phức tạp cho toàn bộ hệ thống CAN BUS (Controller Area Network).

Chính vì vậy, việc xây dựng nên một phần mềm có thể mô phỏng hoạt động như một hệ thống hoàn chỉnh nhằm thay thế các ECU thực tế là thật sự cần thiết. Nó sẽ giúp cho việc phát triển thêm ứng dụng đơn giản hơn, với độ tin cậy và hiệu quả cao.

Hiểu được tầm quan trọng của nó, nhóm chúng em đã đăng ký tham gia Đồ án tốt nghiệp dưới hình thức Capstone Project cùng FPT Software với đề tài “THIẾT KẾ VÀ THI CÔNG HỆ THỐNG MÔ PHỎNG CAN BUS”. Phạm vi của em thực hiện trong đề tài là “THIẾT KẾ PHẦN MÊM MÔ PHỎNG CAN”.

Nội dung báo cáo gồm các chương:

Chương 1: Tổng quan hệ thống mô phỏng CAN Bus

Chương 2: Cơ sở lý thuyết và môi trường thực hiện

Chương 3: Kết quả thực hiện và đánh giá

# **CHƯƠNG 1: TỔNG QUAN HỆ THỐNG MÔ PHỎNG CAN BUS**

# Giới thiệu chương.

.

.

.

.

# Tổng quan về CAN BUS



*Hình 1.1 : Sơ đồ khối mô hình kết nối CAN BUS*

.

.

.

.

# Hệ thống mô phỏng CAN BUS

Dưới đây là sơ đồ khối mô hình kết nối khi sử dụng , sơ đồ khối mô tả chung của đề tài. Hệ thống mô phỏng CAN Bus giao tiếp các ECU trong ô tô sử dụng đường truyền CAN bus, bao gồm CAN Device thực hiện hoạt động truyền nhận thông điệp tương tự như một ECU thực tế,PC Driver có nhiệm vụ giao tiếp giữa hardware và software và CAN Simulator software là phần mềm mô phỏng, hiện thị định dạng của thông điệp, tín hiệu, đồng thời có thể cấu hình chọn lựa cổng CAN, tốc độ baud…với cửa sổ thống kê hiện thị tương tự như trên phần mềm CANoe Vector đã và đang được các nhà phát triển, sản xuất ô tô trên thế giới sử dụng hiện nay.



*Hình 1.2 : Sơ đồ khối mô tả giao tiếp hệ thống CAN BUS*

Để có thể kiểm tra được ECU có hoạt động đúng yêu cầu thiết kế hay không, ta sẽ xây dựng mô hình kiểm tra như trên , trong đó ý tưởng chính dự trên việc giả lập một hệ thống ô tô hoàn chỉnh trên máy tính và kết nối nó với ECU thực tế để kiểm tra bằng cách truyền các thông điệp yêu cầu ECU thực hiện và nhận lại các thông điệp phản hồi từ ECU, sau đó xử lý nội dung của phản hồi và hiển thị lên màn hình thống kê một cách trực quan với độ chính xác cao.

ECU cần kiểm tra sẽ được kết nối với CAN Device thông qua cổng CAN. CAN Device kết nối với Software, nơi chứa hệ thống giả lập gồm các ECU (Electronic Control Unit) mô phỏng hệ thống thực tế thông qua cổng USB nhờ PC driver .PC driver đóng vai trò cầu nối giao tiếp giữa hardward và software thông qua cơ chế truyền (ghi) và nhận (đọc) tin từ các file descriptor.



**Hình 1.3  Sơ đồ khối hệ thống CAN BUS**

Sau khi kết nối giữa ECU và Software , mô hình hệ thống sẽ gồm nhiều ECU được kết nối với nhau thông qua CAN BUS như là một hệ thống CAN thực tế hoàn chỉnh.

# Các thành phần của hệ thống.

Hệ thống mô phỏng CAN Bus bao gồm: Phần mềm mô phỏng CAN, phần mềm cơ sở dữ liệu CAN, CAN Device và PC Driver

# CAN Software

.

*Hình 1.3 CAN Software bao gồm: Database Edit Software và Simulator software.*

# Phần mềm mô phỏng CAN.

Là một phần mềm độc lập, có nhiệm vụ xử lí việc truyền nhận dữ liệu cũng như phân tích và hiển thị nội dung của các thông điệp, mà định dạng của các thông điệp đó được lấy từ D atabase edit software. Bao gồm các chức năng

* Cấu hình hệ thống, cổng CAN nào đang hoạt động , tốc độ Baud…
* Load được database vào test model.
* Add/remove được sơ đồ khối kết nối đồng thời cập nhật cấu hình và lưu lại.
* Add/remove được các message vào trong IG. (interactive generator)
* Trong IG có thể setting cycle của từng message.
* Trace window hiển thị các thuộc tính của message: ID, length, direction, data, signal value.

# Phần mềm cơ sở dữ liệu CAN.

Database Edit Software là phần mềm độc lập quản lí cơ sở dữ liệu của hệ thống, lưu giữ các thông điệp (message) và các tín hiệu (signal), hỗ trợ các loại format Little Endian và Big Endian , MSB và LSB. Các chức năng chính của Database Software là: Tạo mới database và mở 1 database có sẵn.

Hổ trợ các chức năng:

- Thêm/sửa/xóa các thông điệp và tín hiệu trong các thông điệp.

- Thêm/sửa/xóa các nốt mạng (node network) và các biến môi trường EV (environment variable).

Vai trò chính của Database edit software là tạo dựng một cơ sở dữ liệu gồm các node network và các thông điệp gửi đi cũng như nhận về từ chính node đó để cung cấp dữ liệu cho việc mô phỏng ở simulator software.

# CAN Device

**CAN Device:** Chức năng chính của CAN Device là tạo một node CAN có thể tùy chỉnh được tốc độ của node để có thể kết nối được với một CAN BUS bất kỳ để lấy các CAN\_Frame trong CAN BUS và gởi lên PC.

Để thực hiện các chức năng đó chúng ta sử dụng giao thức USB và CAN. Board DK-TM4C123G hỗ trợ

# PC Driver và Protocol

**PC driver**: PC Driver có nhiệm vụ giao tiếp giữa hardware và software .Khi một device được kết nối với PC, driver nhận nhiệm vụ tạo ra vùng nhớ đệm thông qua các file descriptor. Khi ta muốn truyền một thông điệp từ PC xuống Hardware thì Driver sẽ nhận biết được thông điệp, sau đó ghi vào file descriptor tương ứng. Hardware sẽ đọc dữ liệu từ file descriptor và thực hiện nội dung thông điệp. Thông điệp phản hồi sẽ được truyền theo chiều ngược lại đến Software.

* Nhận biết được thiết bị connect, disconnect thông qua cổng USB, đưa ra thông tin, trạng thái của thiết bị.
* CAN software có thể đóng/ mở thiết bị thông qua device node.
* CAN software có thể cấu hình baurd rate thông qua file descriptor.
* CAN software có thể gửi/nhận message thông qua file descriptor

# Kết luận chương (nhắc lại phạm vi đề tài lần nữa).

# CHƯƠNG 2: CƠ SỞ LÝ THUYẾT VÀ MÔI TRƯỜNG THỰC HIỆN (Trang 19)

# Giới thiệu chương.

Thiết kế phần mềm mô phỏng CAN là nhiệm vụ chính của nhóm trong đề tài “THIẾT KẾ VÀ THI CÔNG HỆ THỐNG MÔ PHỎNG CAN BUS”.

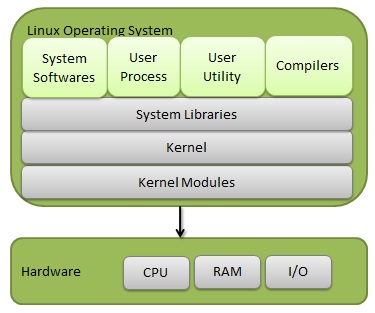
Trong chương này sẽ trích dẫn, trình bày những nội dung kiến thức liên quan trong đồ án được sử dụng để tham khảo , nghiên cứu trong quá trình thực hiện đồ án được cung cấp bởi những tài liệu và website chính thống , uy tín.

# Công cụ thực hiện.

# Môi trường hệ điều hành Linux.

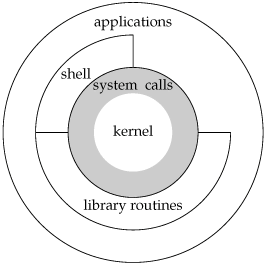
* Giới thiệu hệ điều hành Linux:

Tổng quan về hệ điều hành Linux được mô tả như hình dưới :



* Kernel: Là trung tâm điều khiển của hệ điều hành Linux, chứa các mã điều khiển hoạt động toàn bộ hệ thống. Kernel được phát triển không ngừng ,được phát triển theo dạng module do vậy kích thước thật của kernel rất nhỏ. Nó chịu trách nhiệm cho tất cả hoạt động chính của hệ điều hành, bao gồm nhiều module khác nhau và tương tác trực tiếp với phần cứng. Kernel cung cấp sự trừu tượng bắt buộc để ẩn chi tiết phần cứng mức thấp cho các chương trình hệ thống hoặc ứng dụng.
* System Libraries: Là các chương trình hoặc các hàm đặc biệt sử dụng các chương trình ứng dụng hoặc các tiện ích hệ thống truy cập các tính năng của kernel.
* Các tiện ích : Các tiên ích được người dùng thường xuyên sử dụng . Nó được dùng cho nhiều thứ như thao tác tập tin, đĩa nén, sao lưu tập tin… tiện ích trong Linux có thể là các lệnh thao tác hay chương trình giao diện đồ họa. Hầu hết các tiện ích trong Linux là sản phẩm của chương trình GNU. Linux có sẵn rất nhiều tiện ích như chương trình biên dịch, chương trình gỡ lỗi, soạn thảo văn bản,… tiện ích thể được sử dụng bởi người dùng hoặc hệ thống, một số tiện ích được xem là chuẩn trong hệ thống Linux như passwd, ls, ps, vi,..
* Kiến trúc Linux:

Kiến trúc Linux được thể hiện như hình dưới:



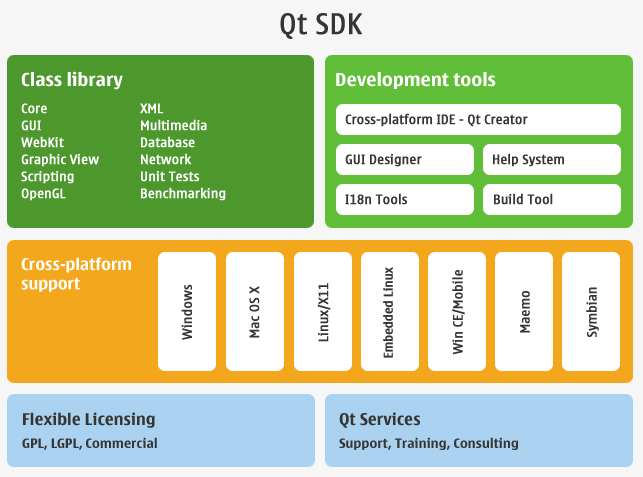
* System calls: là giao diện chính giữa kernel và không gian sử dụng, khoảng 300 system call cung cấp các dịch vụ kernel chính.
* Shell: Cung cấp các tập lệnh cho người dùng thao tác với kernel để thực hiện công việc. Shell đọc các lệnh từ người dùng và xử lý. Có nhiều loại Shell được sử dụng trong Linux .Shell sử dụng chính trong Linux là GNU Bourne Again Shell , phát triển từ Bourne Shell.
* Lợi ích của Linux:

-Sử dụng Linux bạn sẽ có nhiều lợi điểm. Trong số những hệ điều hành thông dụng nhất hiện nay , Linux là hệ điều hành mở miễn phí được nhiều người sử dụng nhất hiện nay.

-Bản thân Linux cũng có sẵn toàn bộ nghi thức mạng TCP/IP, giúp bạn dễ dàng kết nối Internet và gửi thư điện tử. ….

-Khả năng thích ứng của hệ điều hành giúp bạn chuyển nó từ một nền tảng này sang một nền tảng khác mà nó hoạt động vẫn tốt.

# Giới thiệu chung về Qt Creater Framework.



*Hình 2.1 Mô tả về QT FrameWork*

- Qt là một Application Framework , tạo ra một framework có khả năng thiết kế, tạo ra những phần mềm có thể chạy trên nhiều nền tảng khác ,hỗ trợ đa ngôn ngữ : C++ , Pascal, Python, Ruby….

**-** Hình ảnh trên mô tả đầy đủ về QT FrameWork :

* Cross Flatform: hỗ trợ chạy trên nhiều nền tảng như Windows, Mac , Linux,Maemo, Symbian,…
* Qt modular class library: Các thư viện thành phần cốt lõi cho Framework là Qt Core, QtGUI, Qt Widget, QT QML, Qt Quick, Qt Quick Controls , QT Quick Layouts, QT NetWork, ....
* Development tool : Qt Creator là công cụ dùng để thiết kế giao diện đồ họa GUI.

# 2.2.3. Các ưu điểm của QT Framework.

- Code 1 lần và chạy khắp mọi nơi: code trên Windows, chạy trên Linux, Mac, thiết bị di động, thiết bị nhúng...

- Các thư viện class trực quan, đơn giản và dễ sử dụng.

- Tạo ra code dễ đọc, dễ mở rộng và dễ tái sử dụng.

- Ứng dụng tạo ra có hiệu suất xử lý cao và chạy khá nhanh.

- OpenWebkit cho phép xây dựng và chạy các ứng dụng dựa trên nền web, ví dụ như game viết bằng Html5, Css, JavaScript.

* 1. Thiết kế chương trình theo hướng đối tượng:
* Có 5 giai đoạn phát triển hệ thống phần mềm theo hướng đối tượng:
* Phân tích yêu cầu.

Bằng việc tìm hiểu các trường hợp sử dụng (use case) để nắm bắt các yêu cầu của khách hàng, của vấn đề cần giải quyết. Qua trường hợp sử dụng này, các nhân tố bên ngoài có tham gia vào hệ thống cũng được mô hình hóa bằng các tác nhân. Mỗi trường hợp sử dụng được mô tả bằng văn bản, đặc tả yêu cầu của khách hàng.

* Phân tích.

Từ các đặc tả yêu cầu trên, hệ thống sẽ bước đầu được mô hình hóa bởi các khái niệm lớp, đối tượng và các cơ chế để diễn tả hoạt động của hệ thống. Trong giai đoạn phân tích chúng ta chỉ mô tả các lớp trong lĩnh vực của vấn đề cần giải quyết chứ chúng ta không đi sâu vào các chi tiết kỹ thuật.

* Thiết kế.

Trong giai đoạn thiết kế, các kết quả của quá trình phân tích được mở rộng thành một giải pháp kỹ thuật. Một số các lớp được thêm vào để cung cấp cơ sở hạ tầng kỹ thuật như lớp giao diện, lớp cơ sở dữ liệu, lớp chức năng, …

* Lập trình.

Đây còn gọi là bước xây dựng, giai đoạn này sẽ đặc tả chi tiết kết quả của giai đoạn thiết kế. Các lớp của bước thiết kế sẽ được chuyển thành mã nguồn theo một ngôn ngữ lập trình theo hướng đối tượng nào đó.

* Kiểm tra.

Trong giai đoạn kiểm tra, có bốn hình thức kiểm tra hệ thống:

-Kiểm tra từng đơn thể (unit testing) được dùng kiểm tra các lớp hoặc các nhóm đơn.

-Kiểm tra tính tích hợp (integration testing), được kết hợp với các thành phần và các lớp để kiểm tra xem chúng hoạt động với nhau có đúng không.

-Kiểm tra hệ thống (system testing) chỉ để kiểm tra xem hệ thống có đáp ứng được chức năng mà người dùng yêu cầu không.

-Kiểm tra tính chấp nhận được(acceptance testing), việc kiểm tra này được thực hiện bởi khách hàng, việc kiểm tra cũng thực hiện giống như kiểm tra hệ thống.

* Các bước cần thiết để thiết kế chương trình :

Để thiết kế một chương trình theo hướng đối tượng, chúng ta phải trải qua bốn bước sau, từ đó chúng ta xây dựng được một cây phả hệ mang tính kế thừa và các mối quan hệ giữa các đối tượng:

* Xác định các dạng đối tượng (lớp) của bài toán (định dang các đối tượng).
* Tìm kiếm các đặc tính chung (dữ liệu chung) trong các dạng đối tượng này, những gì chúng cùng nhau chia sẻ.
* Xác định được lớp cơ sở dựa trên cơ sở các đặc tính chung của các dạng đối tượng.
* Từ lớp cơ sở, sử dụng quan hệ tổng quát hóa để đặc tả trong việc đưa ra các lớp dẫn xuất chứa các thành phần, những đặc tính không chung còn lại của dạng đối tượng. Bên cạnh đó, chúng ta còn đưa ra các lớp có quan hệ với các lớp cơ sở và lớp dẫn xuất; các quan hệ này có thể là quan hệ kết hợp, quan hệ tập hợp lại, quan hệ phụ thuộc.
  1. Phần mềm mô phỏng CAN.

# Tổng quan phần mềm.

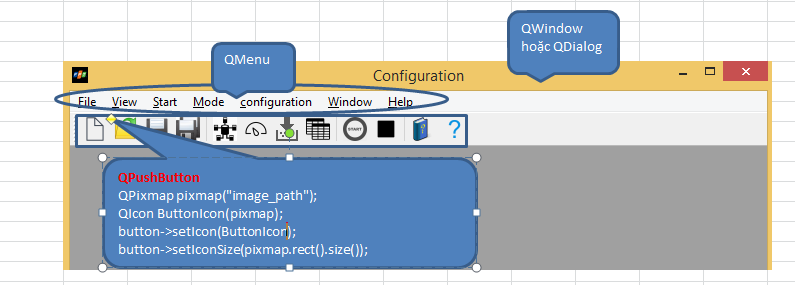
# Tiến trình thực hiện.

# Thiết kế giao diện.

Thiết kế giao diện CAN simulator sử dụng các thư viện:

**QWindow và QDialog**

Tạo cửa sổ giao diện chính

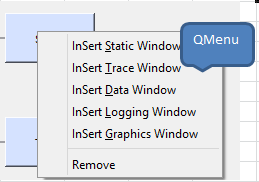


*Hình 2.2: Cửa sổ giao diện chính sử dụng QWindow và QDialog*

**QMenu**

Tạo thanh Menu trên giao diện chính.

Tạo Menu khi click vào một button trên giao diện Configuration.



*Hình 2.2: Right Click để insert các khối*

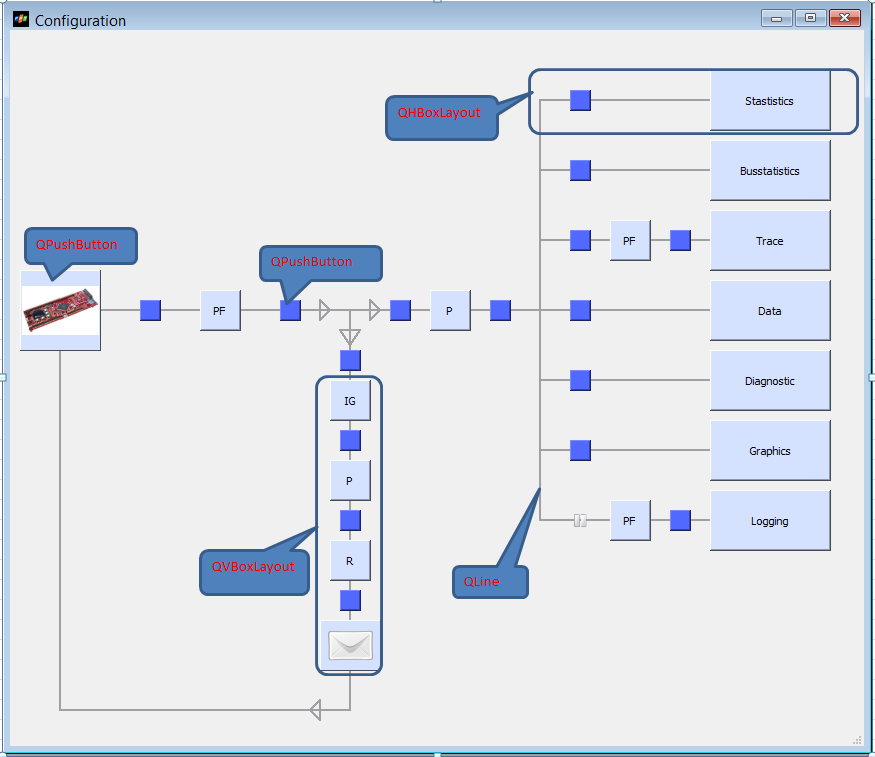
**QPushbutton**

Tạo các nút nhấn trên giao diện chính.

Mỗi nút nhấn đảm nhiệm một chức năng, một event khác nhau như : Tạo mới một hộp thoại khi được doubleclick, show Menu khi click right button. Thực hiện kết nối các nút khác nhau.

**QPainter**

Thực hiện vẽ Line trên giao diện chính, tạo màu hay kích thước của các Line.



*Hình 2.2: QPainter để điều chỉnh màu và kích thước các Line*

**QLine**

Thống qua QPainter tạo đường dẫn kết nối giữa các Button với nhau.

**QPoint**

Trả về tọa độ của bất kì điểm nào trên giao diện.

Trả về tạo độ của các Button để tính toán điểm giữa các Button để tạo các Line.

**QVBoxlayout, QHBoxLayout, QGridLayout**

* + Xắp xếp các Button,Text theo hàng ngan(QHboxLayout).
  + Xắp xếp các Button,Text theo hàng dọc (QVBoxLayout).
  + Căn chỉnh tổng thể (QGridLayout).

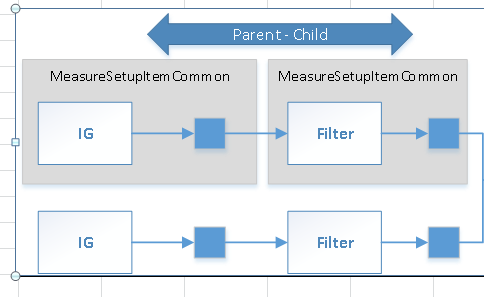
**QMdiSubWindow**

Tạo một của sổ nằm trong giao diện chính .

**QTableWidgetItem**

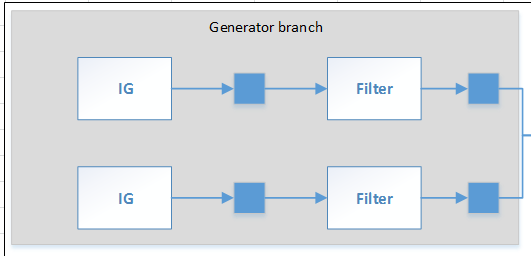
Tạo giao diện khối IG (Interative Generator)

# Giao tiếp các khối(Parent – Child).

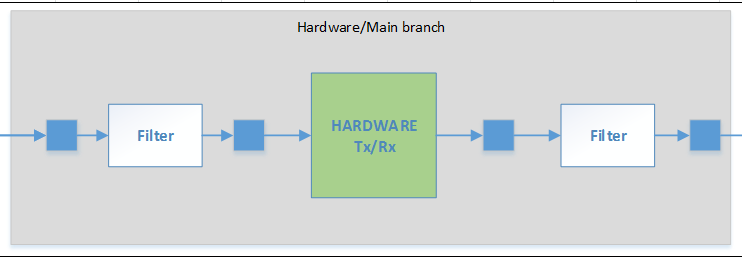


# Các nhánh trên giao diện phần mềm.

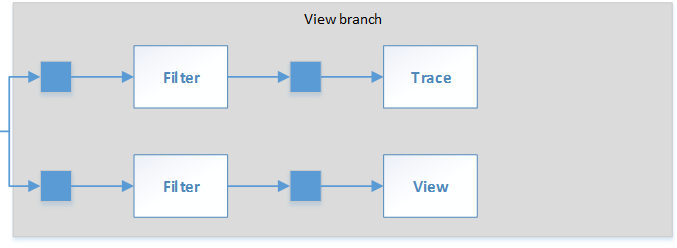
# Nhánh Generator.



# Nhánh Hardware/Main.

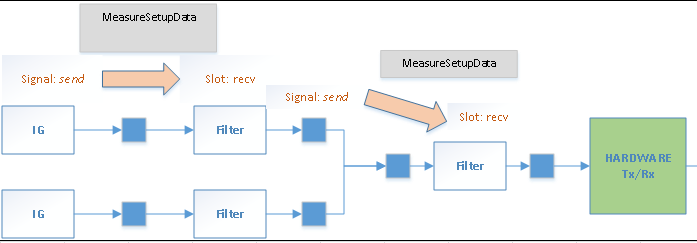


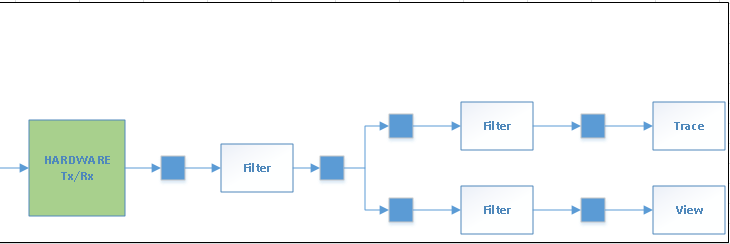
# Nhánh View.



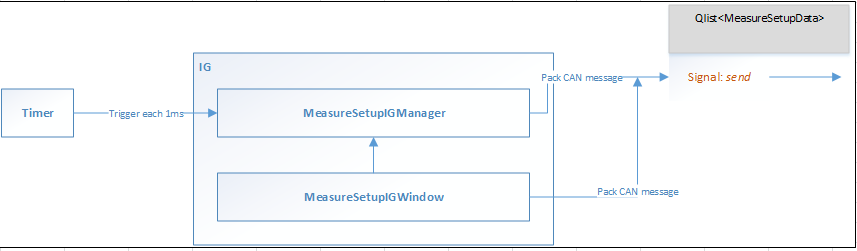
# Quản lý dữ liệu.

# Luồng dữ liệu hệ thống.

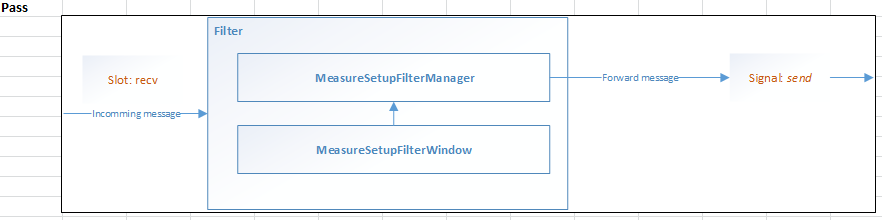


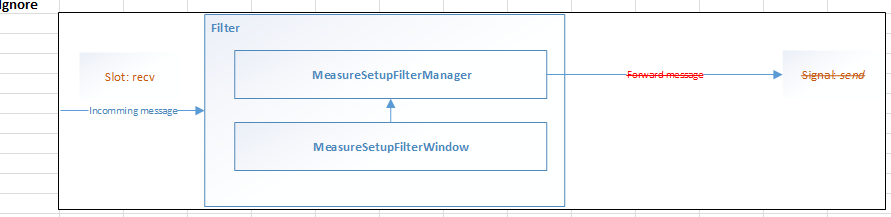


# Luồng dữ liệu Interative Generator Block.

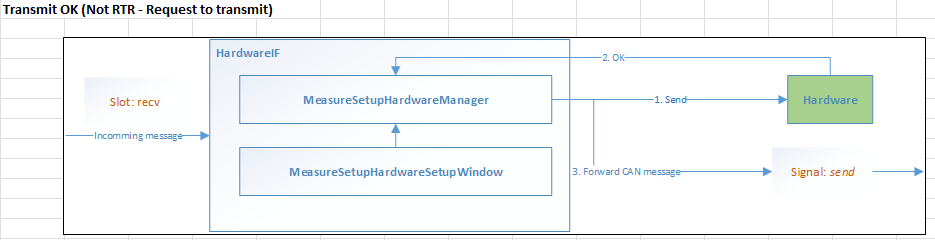


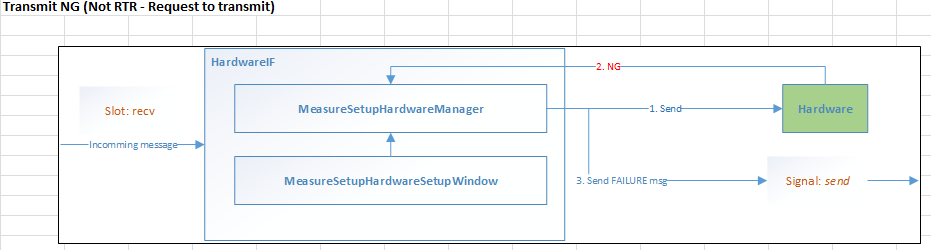
# Luồng dữ liệu Filter Block.

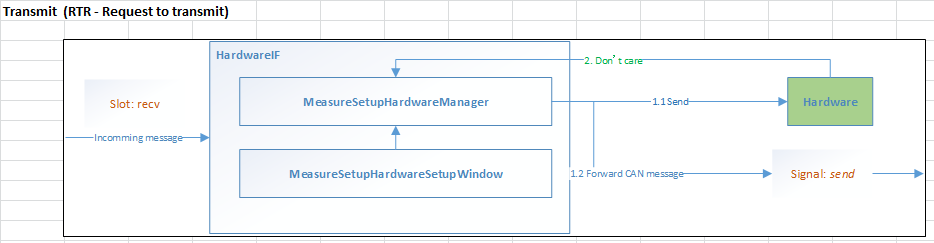




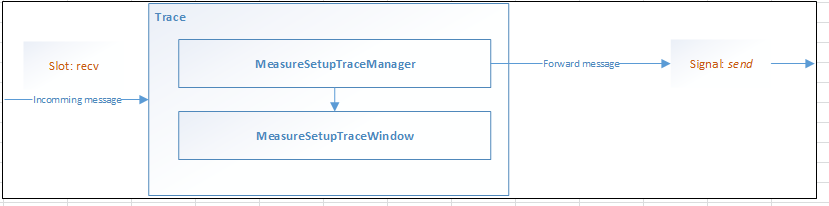
# Luồng dữ liệu Hardware Block.







# Luồng dữ liệu Trace Block



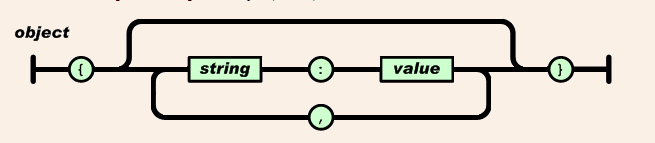
# Cấu hình dữ liệu.

# Giới thiệu về Json.

JSON là một định dạng hoán vị dữ liệu nhanh, chúng dễ dàng cho chúng ta đọc và viết , được xây dựng trên cấu trúc Key- Value, chúng có thể là một Object, Array, Value, String, Number để mô tả dữ liệu của UI .

# QJson trong QT framework.

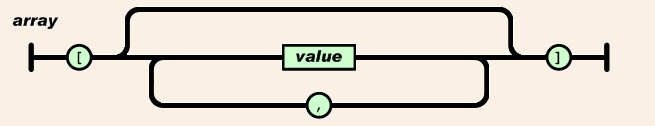
**QJsonObject**



Một đối tượng là một hỗn độn các cặp tên và giá trị, một đối tượng bắt đầu bởi dấu ngoặc nhọn trái { và kết thúc với dấu ngoặc nhọn phải }. Từng tên được theo sau bởi dấu hai chấm : và các cặp tên/giá trị được tách nhau bởi dấu phẩy.

Qt hỗ trợ lớp QJsonObject cho phép edit các đối tượng một cách dễ dàng.

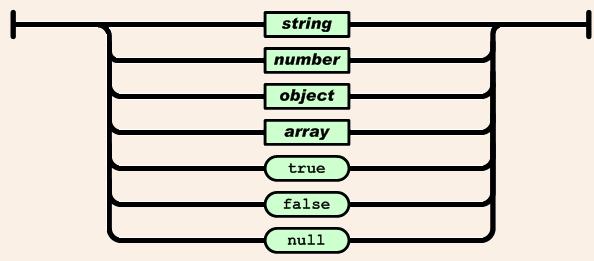
**QJsonArray:**



Một mảng là tập hợp các giá trị đã được sắp xếp. Một mảng bắt đầu bởi dấu mở ngoặc vuông trái [ và kết thúc với dấu ngoặc vuông phải ]. Các giá trị được cách nhau bởi dấu phẩy,.

Qt hỗ trợ lớp QJsonArray cho phép edit insert , remove các giá trị từ mảng.

**QJsonValue:**



Một giá trị có thể là một chuỗi string trong những trích dẫn kép hoặc là một số, hoặc true, hoặc false , hoặc Null, hoặc một đối tượng, hoặc một mảng. Những cấu trúc này có thể đã được lồng vào nhau.

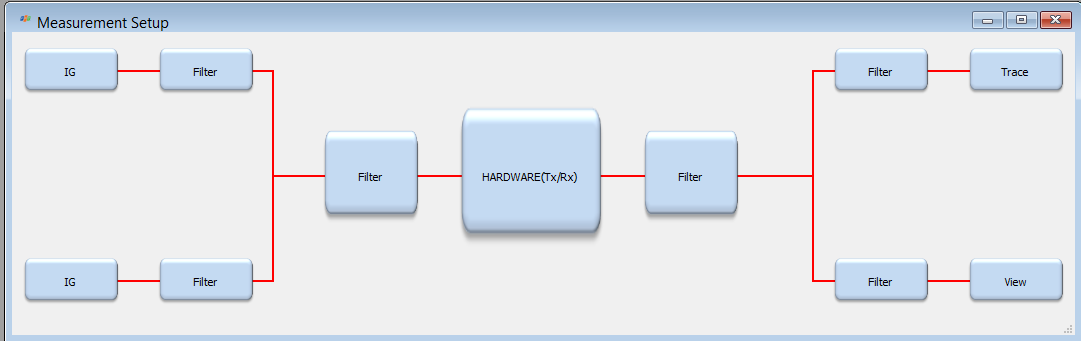
# Kết luận chương.

# CHƯƠNG 3: KẾT QUẢ THỰC HIỆN VÀ ĐÁNH GIÁ (Trang 33)

# 3.1. Giới thiệu chương.

# 3.2. Bảng phân công công việc.

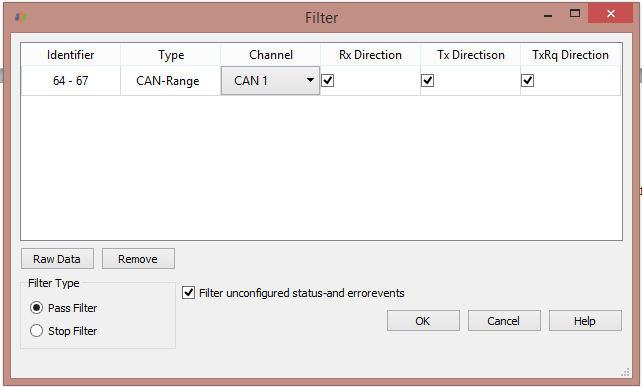
# 3.3. Giao diện Simulator software.

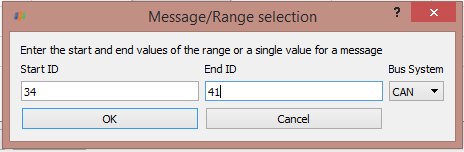


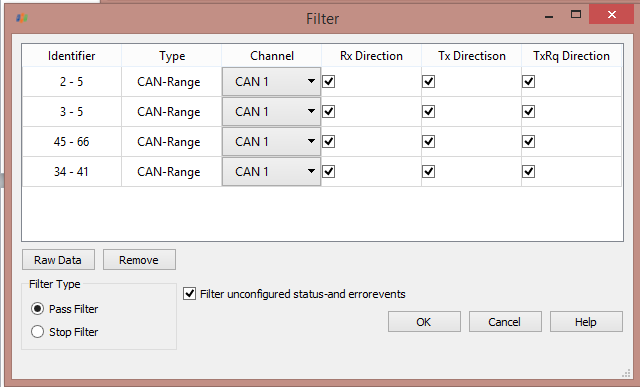
# Giao diện các hộp thoại.

# Giao diện khối IG

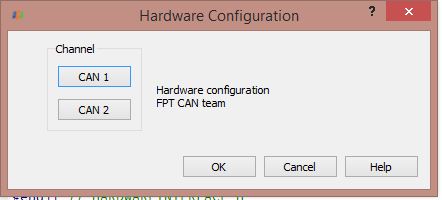
# Giao diện khối Filter

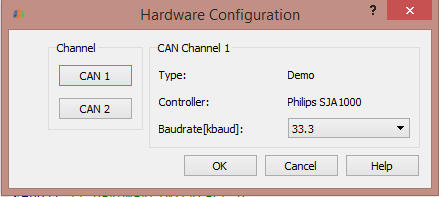




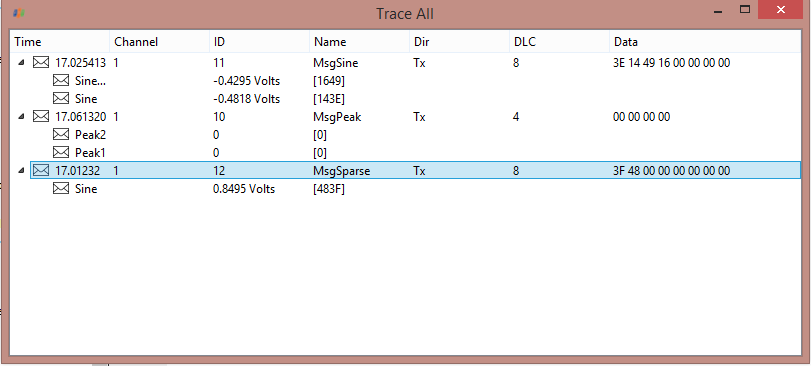


# Giao diện khối Hardware





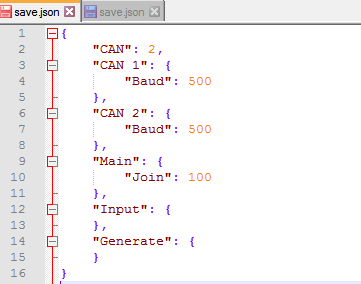
# Giao diện khối Trace



# Edit fie Cấu hình theo định dạng json

Tạo, edit và save được file config theo định dạng json.

**File Json khi mở file simulator ban đầu.**



**File Json được lưu lại sau khi chèn thêm các khối và node.**

****

*Hình 3.2. File Cấu hình theo định dạng json*

# Hoạt động phần mềm mô phỏng CAN. (trang 41 - 45)

# Kết quả nghiệm thu.(trang 44- 50)

# Kết luận chương.

# KẾT LUẬN (Trang 50**)**

# HƯỚNG PHÁT TRIỂN ĐỀ TÀI (Trang 51)

# TÀI LIỆU THAM KHẢO (Trang 52)

# PHỤ LỤC (Code) (Trang 53...)