**Chương 1: Tổng quan về CAN simulator**

* 1. **Giới thiệu chung:**

Trong cuộc sống hiện nay, Oto là một phương tiện đi lại phổ biến trên thế giới và chủ yếu tại các nước phát triển . Chính vì vậy ngành công nghiệp oto không ngừng được phát triển trong những năm gần đây, và để hỗ trợ trong sản xuất ,vận hành và sữa chữa luôn có một hệ thống thiết bị, phần mềm phục vụ việc mô phỏng, kiểm tra, phân tích phát hiện lỗi, hoạt động của các ECU, cảm biến… trong oto , để luôn đảm bảo được mức độ an toàn cần thiết cho người sử dụng. Chính vì lý do đó tại FPT đã có một dự án mang tên “ CAN SIMULATOR ” . Và dưới đây là sơ đồ khối mô hình kết nối khi sử dụng , sơ đồ khối mô tả chung của đề tài.



*Hình 1.1 : Sơ đồ khối mô hình kết nối khi sử dụng*



*Hình 1.2 : Sơ đồ khối mô tả chung hệ thống CAN SIMULATOR*.

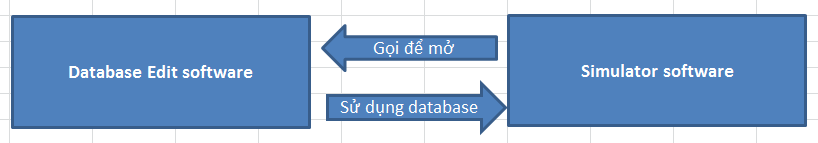
Dự án được thực hiện chủ yếu dựa trên việc lập trình ngôn ngữ C/C++, sử dụng kit TM4C123G của hang Texas Instruments và một số phần mềm hỗ trợ khác như CCS, Qt Creator,....

* 1. **Các thành phần của hệ thống và giải pháp thực hiện:**

Hệ thống có 3 phần chính:

* + 1. **CAN software PC**

Gồm hai phần mềm chính : Database Edit Software và Simulator Sofware. Sử dụng Qt framework để viết UI trên ngôn ngữ C++.



* **Database Edit software**
* Là một chương trình độc lập
* Hỗ trợ được các loại data format: Intel, Motorola…
* Hỗ trợ chỉnh sửa message, signal, thêm/ xóa signal vào message/ signal

=> Xây dựng các hàm thao tác với message/signal, mỗi message được lưu thành danh sách liên kết với bao gồm các signal thuộc message đó,đọc dữ liệu từ database nhận diện được các format của mỗi hãng.

* **Simulator software**

**Nhiệm vụ chính:**

* Cấu hình được những cổng CAN nào sử dụng, tốc độ nào
* Load được database vào trong test model
* Thêm/xóa được sơ đồ kết nối các khối (trace, interactive generator)
* Thêm/xóa được các message vào trong IG(interactive generator)
* Trong IG có thể edit được các waveform của từng signal. Waveform: sine, toggle, square
* Trong IG có thể setting cycle của từng message.
* Trong lúc đang simulate, có thể chỉnh sửa nội dung message ở dạng raw data hoặc physical value
* Trace window hiển thị các thuộc tính của message: ID, length, direction, data, signal value.
* Statistic window hiển thị

**1.2.2 PC driver**

Tạo ra file mô tả nhằm liên kết giữa phần cứng và phần mềm, phần mềm lấy dữ liệu phần cứng từ file mô tả đồng thời muốn gửi dữ liệu cho phần cứng thì có thể viết vào file mô tả đó.

Nhiệm vụ chính:

+) Nhận biết thiết bị cắm, rút

+) CAN software có thể đóng, mở thiết bị qua device node

+) CAN software có thể cấu hình baudrate thiết bị qua file descriptor

+) CAN software có thể gửi/nhận message CAN qua file descriptor

* + 1. **CAN simulatior device**

Cấu hình 2 giao tiếp CAN và USB để có thể nhận và gửi các message.

- Nhiệm vụ chính:

+) Giao tiếp USB để nhận điều khiển từ PC

+) Giao tiếp USB gửi message từ CAN (CAN nhận được)

+) Giao tiếp USB thông báo thiết bị CAN bị đứt kết nối.

* 1. **Kết luận chung:**

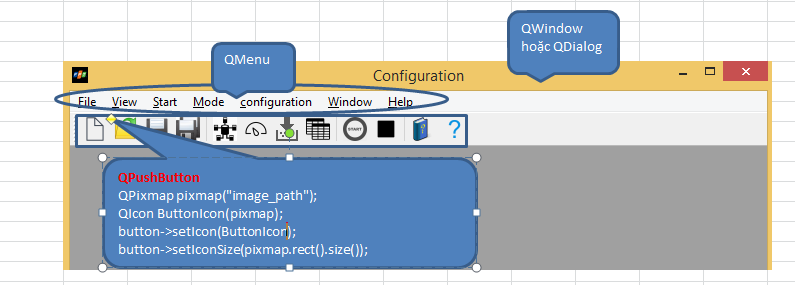
**Chương 2: Hên xui**

**2.1 Giới thiệu chương:**

**2.2 Các Thư viện được sử dụng:**

**2.2.1 QWindow và QDialog**

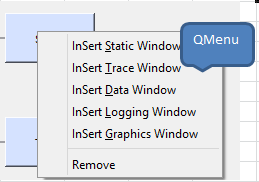
Tạo cửa sổ giao diện chính



2.2.2 QMenu

Tạo thanh Menu trên giao diện chính.

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



2.2.3 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.

2.2.4 QLine

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

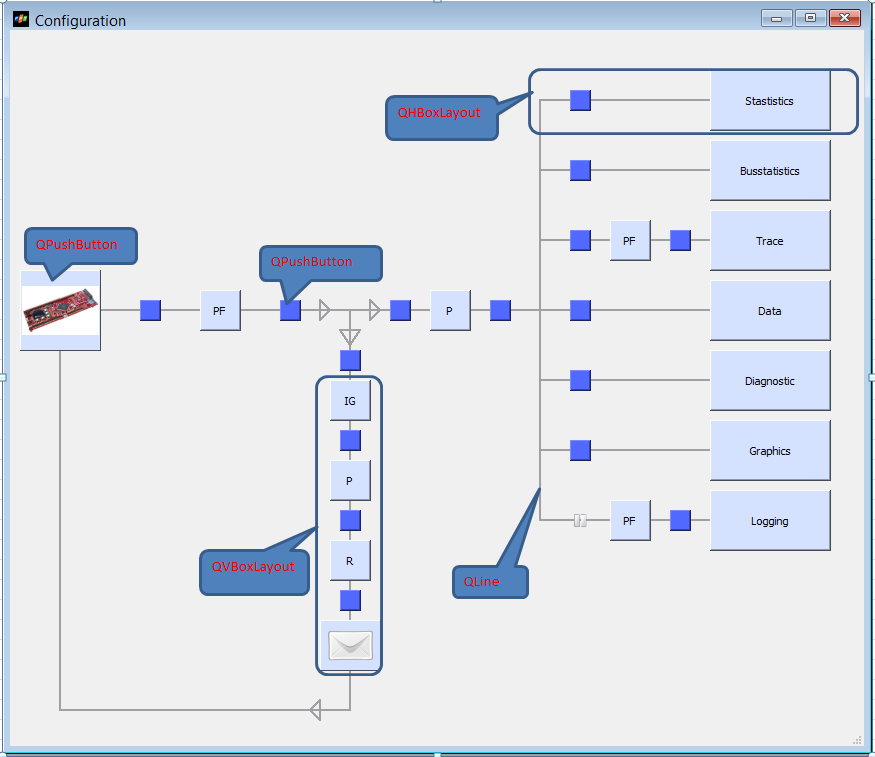
2.2.5 QPoint

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.

2.2.6 QPainter

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



2.2.7 QVBoxlayout, QHBoxLayout, QGridLayout

2.2.8 QMdiSubWindow

2.2.9 QPixmap

2.2.10 QTableWidgetItem

2.2.11 QLabel

2.3 Kết luận chương:

**Chương 3: Công**