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

Chương này sẽ giới thiệu về đường truyền CAN và 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.

**Tổng quan về CAN BUS**

Mạng điều khiển cục bộ CAN – Controller Area Network (hay còn gọi là CAN BUS) là mạng lưới đường truyền thông tin giữa các bộ điều khiển độc lập dưới dạng đường truyền BUS. Trong đó các thành phần kết nối trong mạng có quyền ngang nhau trong việc truyền và nhận thông tin (multi master). Vì vậy, đây là một giao thức truyền thông nối tiếp hổ trợ hiệu quả phân phối thời gian thực và được kiểm soát với mức độ bảo mật rất cao.

**Một số các tính năng giao thức CAN là:**

* CAN bus là đa chủ. Khi bus dữ liệu rỗi, bất kỳ thiết bị nào gắn vào bus đều có thể bắt đầu gửi một thông điệp.
* Giao thức CAN bus rất linh hoạt. Các thiết bị kết nối với bus không có địa chỉ, có nghĩa là thông điệp không được truyền đi từ một nút này đến một nút khác dựa trên các địa chỉ. Tốc độ giao tiếp bus CAN là không cố định. Bất kỳ tốc độ truyền thông có thể được thiết lập cho các thiết bị gắn vào một bus.
* Tất cả các thiết bị trên bus có thể phát hiện ra lỗi. Các thiết bị đã phát hiện lỗi ngay lập tức thông báo cho tất cả các thiết bị khác.
* Nhiều thiết bị có thể được kết nối với bus tại cùng một thời điểm, và về mặt lí thuyết không có một giới hạn nào về số lượng các thiết bị có thể được kết nối. Trong thực tế, số thiết bị có thể được gắn vào một bus bị hạn chế bởi thời gian trễ của bus và tải điện.

Trong chương này, chúng tôi trình bày tổng quan về hệ thống mô phỏng CAN Bus, cách thức hoạt động cũng như các thành phần chính của hệ thống. Chương tiếp theo, chúng tôi sẽ trình bày cụ thể hơn về phần thực hiện Phần mềm cơ sở dữ liệu CAN của chúng tôi.

**Chương 2: CƠ SỞ LÝ THUYẾT VÀ MÔI TRƯỜNG THỰC HIỆN**

Chương này sẽ trình bày cơ sở lí thuyết, những kiến thức cơ bản về cơ sở dữ liệu, kỹ thuật lập trình cũng như môi trường phát triển của phần mền cơ sở dữ liệu CAN, bao gồm lựa chọn hệ điều hành và môi trường phát triển tích hợp IDE.

**QT software framework**

***Giới thiệu chung về QT Framework***

* Định nghĩa framework: Là một thư viên các lớp được xây dựng hoàn chỉnh, là nền tảng để phát triển các phần mềm ứng dụng. Thay vì tự tay viết tất cả mã cho các lớp , hàm cho toàn bộ dự án phần mềm của mình, các lập trình viên dùng framework để tiết kiệm thời gian và công sức nhưng vẫn đạt hiệu quả mong muốn bằng cách kế thừa các lớp có sẵn từ các thư viện trong framework
* Qt hổ trợ các nền tảng sau: Windows, Linux, OS X, iOS, Android, WinRT…
* Qt hổ trợ ngôn ngữ C/C++/Java/Python/…
* Qt hổ trợ lập trình giao diện chuẩn theo mô hình model/view và hổ trợ các lớp lập trình giao diện động trong Qt Quick
* Với QT chúng ta không cần thiết phải viết giao diện từ những dòng code phức tạp, mà hoàn toàn có thể vẽ giao diện trong tệp có phần mở rộng .ui

***Mô hình Model/View trong QT Framework***

* Mô hình MVC (Model-View-Controller) là một kiến trúc phần mềm, nó giúp cho các nhà phát triển phần mềm tách các ứng dụng của họ ra thành 3 phần có nhiệm vụ riêng biệt và độc lập với các thành phần khác, trong đó:
* Model: chứa các phương thức xử lí và truy xuất dữ liệu.
* View: Đảm bảo việc hiển thị thông tin và tương tác với người dùng.
* Controller: là trung gian giữa Model và View, nhận các yêu cầu từ tương tác người dùng từ view và tác động làm thay đổi dữ liệu trong model.
* QT framework sử dụng mô hình model/view thay cho mô hình MVC, trong đó lớp Controller sẽ được tích hợp trong View thông qua Delegate.
* Lí do lựa chọn mô hình model/view: mô hình này chia phần mền thành các module riêng biệt, nên sẽ dễ dàng hơn cho việc bảo trì, nâng cấp, phát triển sau này.
* Hoạt động của model/view:
* Ban đầu model sẽ sử dụng phương thức setModelData () của mình để lấy dữ liệu từ tập dữ liệu gốc (dataset), và View sẽ tự cập nhật dữ liệu được lấy từ model thông qua phương thức setModel ().
* Mỗi tương tác chỉnh sửa của người dùng trên giao diện sẽ gọi một đối tượng của 1 lớp Delegate tương ứng, thông qua delegate sẽ làm thay đổi dữ liệu tương ứng

trong model, khi đó model sẽ cập nhật hiển thị mới trên View cũng như phát đi một signal để làm thay đổi dữ liệu ở dữ liệu gốc, signal này được kết nối với một slot tương ứng ở lớp quản lí tập dữ liệu gốc, slot này sẽ làm thay đổi dữ liệu trong dataset.

***Cơ chế Signal và Slot trong QT Framework***

* Signal: khi một sự kiện nào đó xảy ra, một signal sẽ được phát đi, thực ra nó chỉ là một phương thức của một lớp nhưng không có phần thân hàm {}. Các lớp Widget có sẵn trong Qt có rất nhiều signal được định nghĩa sẵn, và chúng ta cũng có thể viết các signal riêng cho các lớp của tự định nghĩa. Signal không có kiểu trả về, kiểu trả về của signal luôn luôn là void.
* Slot:  là một phương thức bình thường của một lớp, các phương thức này sẽ được gọi khi có một signal nào đó được phát đi. Cũng giống như signal, các lớp Widget trong Qt cũng có sẵn rất nhiều slot và chúng ta cũng có thể viết slot cho lớp của riêng chúng ta.
* Connect: Signal và slot được kết nối qua từng đối tượng thông qua phương thức connect.

**Connect (Đối tượng 1, SIGNAL (signal thuộc đối tượng 1), Đối tượng 2, SLOT (slot thuộc đối tượng 2));**

Trên đây là toàn bộ cơ sở lí thuyết cơ bản phục vụ cho quá trình thiết kế phần mềm cơ sở dữ liệu CAN

# **Chương 3: KẾT QUẢ THỰC HIỆN VÀ ĐÁNH GIÁ**

Chương này trình bày tiến trình thực hiện các chức năng của phần mềm, nêu rõ từng bước thực hiện các nhiệm vụ, cùng với kết quả đạt được ở mỗi phần .Cuối cùng là phần tự đánh giá kết quả đã đạt được trong thời gian thực hiện đồ án của nhóm.

**Thiết kế và thi công**

***Phần giao diện***

Giao diện CAN database gồm có giao diện chính và các cửa sổ chỉnh sửa các thông điệp hoặc signal.

***Giao diện chính***

Giao diện màn hình chính là một cửa sổ chính sử dụng QMainWindow, trong đó có Menu các hành động như: tạo mới database, mở database có sẵn, lưu database đã mở, lưu lại database dưới dạng tên khác, thoát khỏi chương trình, chức năng hổ trợ Help,…những hành động này sẽ được mô tả bởi các QAction trong Qt và toàn bộ những QAction sẽ được gói trong QMenu tại thanh MenuBar.

***Kết quả***

Giao diện chính hoàn thành ngoài phần hiển thị dữ liệu, còn có thêm các context menu hiển thị các chức năng thêm/sửa/xóa khi click chuột phải vào một vị trí thuộc QTreeView hoặc QTableView nào đó.

***Giao diện của sổ chỉnh sửa***

Phần này gồm 3 cửa sổ:

- Cửa sổ chỉnh sửa dùng cho message

- Cửa sổ chỉnh sửa dùng cho signal

- Cửa sổ chỉnh sửa dùng cho signal trong message

***Phần xử lí tệp***

***Thực hiện các chức năng trên thanh menu màn hình chính***

* Tạo database
* Load database
* Lưu database

***Phần chỉnh sửa cơ sở dữ liệu***

* Thêm message
* Thêm signal vào message
* Chỉnh sửa signal
* Chỉnh sửa dùng delegate
* Xóa message

**Đánh giá**

* ***Giao diện***

- Giao diện đã thể hiện được đầy đủ các thông tin message/signal

-Thông tin hiển thị trong các Table với các mục Header giới thiệu dễ dàng quan sát và chỉnh sửa.

- Giao diện vẫn chưa hoàn thiện và đang chỉnh sửa để có giao diện đầy đủ như các khối Network, ECUs, Environment variables và có Icon dễ nhìn hơn.

* ***Phần xử lí tệp***

Thực hiện được các chức năng như mở file, tạo file và lưu file

Khi load database, giữ liệu đã đọc được chính xác với file .dbc có sẵn, các thông tin về message, signal đã được lưu trữ đúng vị trí

Chức năng save as đã hoạt động đúng khi lưu được file dưới dạng .dbc và .txt. So sánh file đã lưu với thông tin của Message của Signal vẫn đúng theo format ban đầu.

* ***Phần chỉnh sửa cơ sở dữ liệu***

Đã tạo được giao diện tương đối dễ tương tác cho người dùng theo mô hình chuẩn model/view

Tạo được database mới và chỉnh sửa được database có sẵn theo đúng yêu cầu dùng delegate và dùng hộp thoại.

Lưu được database sau khi chỉnh sửa.

Đồ án đã đạt được mục tiêu đề ra và các chức năng chính có thể hoạt động tốt. Mặc dù có nhiều lỗi xuất hiện trong quá trình kiểm thử nhưng sau khi chỉnh sửa chúng tôi đã hoàn thành được đề tài.

# **KẾT LUẬN**

Sau hơn 3 tháng thực hiện, chúng tôi đã cơ bản hoàn thành được đề tài với đúng các chức năng và yêu cầu đề ra từ đầu.

* Phần mềm đã đáp ứng được nhiệm vụ xây dựng và chỉnh sửa cơ sở dữ liệu CAN dựa trên datasheet của hệ thống CAN thực tế cần kiểm thử. Trong đó, hoàn thành các chức năng thêm/sửa/xóa các message/signal như mục tiêu ban đầu.
* Tuy chưa có giao diện thực sự nổi bật, phần mềm đã có giao diện dễ nhìn, dễ sử dụng đối với người dùng

Phần mềm “Cơ sở dữ liệu” đóng vai trò quan trọng trong hệ thống “Mô phỏng CAN BUS”. Nhờ có cơ sở dữ liệu CAN, phần mềm mô phỏng CAN (CAN simulator software) mới có thể giửi đi được những message đúng định dạng, và xác nhận được những message phản hồi từ thiết bị.

Tuy vẫn còn nhiều chức năng chưa được hoàn thiện nhưng đề tài đã hoàn thành được các chức năng cơ bản để có thể sử dụng độc lập. Trong tương lai gần hệ thống sẽ được chỉnh sửa để có thể mô tả các hệ thống hoàn thiện, phức tạp hơn và giao diện mang tính thẩm mỹ hơn và có nhiều tiện ích hơn.