

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH  
TRƯỜNG ĐẠI HỌC BÁCH KHOA  
KHOA KỸ THUẬT GIAO THÔNG  
**BỘ MÔN CÔNG NGHỆ KĨ THUẬT Ô TÔ**

-----oo-----



LUẬN VĂN TỐT NGHIỆP ĐẠI HỌC

**MÔ HÌNH THỰC NGHIỆM ĐIỀU KHIỂN PHUN  
XĂNG ĐÁNH LỬA CHO ĐỘNG CƠ XE GĂN MÁY  
1 XYLANH**

SVTH: Võ Nhất Duy – 1710048

Lê Đức Huy – 1711487

GVHD: TS. Trần Đăng Long

TP. HỒ CHÍ MINH, tháng 8 năm 2021

## MỤC LỤC

Chương 1. TỔNG QUAN .....	12
1.1    Khái niệm và chức năng của sản phẩm trong đề tài .....	12
1.2    Lý do và mục đích của đề tài .....	12
1.3    Điều kiện làm việc .....	12
1.4    Yêu cầu kỹ thuật.....	12
Chương 2. BỐ TRÍ CHUNG CỦA MÔ HÌNH .....	14
2.1    Thông số đầu vào đề thiết kế bố trí chung .....	14
2.2    Sơ đồ bố trí chung mô hình.....	14
2.3    Nguyên lý hoạt động của mô hình.....	15
2.4    Nguyên lý tính toán thời gian phun và thời điểm đánh lửa.....	15
2.5    Thiết kế ngôn ngữ giao tiếp chung .....	17
Chương 3. CHƯƠNG TRÌNH ĐIỀU KHIỂN Ở VI XỬ LÝ .....	19
3.1    Các chức năng của chương trình điều khiển.....	19
3.2    Sự luân chuyển của dữ liệu trong bộ nhớ.....	19
3.3    Chương trình chính .....	20
3.4    Các chương trình ngắt thu thập tín hiệu cảm biến và điều khiển.....	22
3.4.1    Chương trình ngắt Đọc tín hiệu cảm biến ở các chân analog.....	22
3.4.2    Chương trình ngắt khi vi điều khiển nhận được tín hiệu CKP .....	22
3.4.3    Các chương trình ngắt để thao tác điều khiển .....	23
3.4.4    Giản đồ thời gian các chương trình ngắt.....	24
Chương 4. THIẾT KẾ CHƯƠNG TRÌNH GIAO TIẾP VỚI NGƯỜI DÙNG .	27
4.1    Điều khiển thông số vận hành.....	27
4.1.1    Chuẩn hóa dữ liệu.....	28

# *Luận văn tốt nghiệp đại học*

4.1.2 Xác định phần tử điều chỉnh.....	29
4.1.3 Chuyển đổi thành dữ liệu giao tiếp.....	30
4.1.4 Đóng gói dữ liệu giao tiếp .....	31
4.2 Hiển thị thông số vận hành .....	32
4.2.1 Lọc tách dữ liệu thành từng đơn vị giao tiếp hoàn chỉnh.....	33
4.2.2 Tách trường dữ liệu.....	33
4.2.3 Chuyển đổi thành dữ liệu .....	34
4.2.4 Thiết kế giao diện người dùng .....	35
Chương 5. CHẠY THỬ NGHIỆM VÀ ĐÁNH GIÁ KẾT QUẢ.....	38
5.1 Kiểm tra các chức năng của 2 chương trình .....	38
5.1.1 Kiểm tra chức năng đọc cảm biến và tính toán thông số trạng thái của chương trình ở vi xử lý .....	38
5.1.2 Kiểm tra chức năng gửi dữ liệu của chương trình ở vi xử lý và chức năng nhận dữ liệu, hiển thị thông số của chương trình LabVIEW .....	38
5.1.3 Kiểm tra chức năng tính toán thông số điều khiển của chương trình ở vi xử lý .....	39
5.1.4 Kiểm tra chức năng điều khiển của chương trình ở vi xử lý.....	39
5.1.5 Kiểm tra chức năng nhận dữ liệu, lưu tham số luật điều khiển vào EEPROM của chương trình ở vi xử lý và gửi dữ liệu, lưu tham số luật điều khiển của chương trình LabVIEW .....	39
5.1.6 Chạy thực nghiệm.....	43
5.2 Đánh giá kết quả .....	43
Chương 6. PHỤ LỤC CHƯƠNG TRÌNH GIAO TIẾP VỚI NGƯỜI DÙNG ..	44
6.1 Giới thiệu về LabVIEW .....	44

6.2 Tính năng điều chỉnh thông số làm việc .....	62
6.2.1 Khởi tạo chương trình điều khiển.....	62
6.2.2 Chuẩn hóa ma trận.....	63
6.2.3 Xác định phần tử mà người dùng điều chỉnh.....	65
6.3 Tính năng hiển thị thông số vận hành.....	67
6.3.1 Khởi tạo chương trình .....	67
6.3.2 Lọc tách dữ liệu thành từng đơn vị giao tiếp hoàn chỉnh.....	69
6.3.3 Giữ giá trị khi không có dữ liệu gửi lên và tách lấy đoạn chứa giá trị .....	70
6.3.4 Chuyển đổi chuỗi kí tự thành số liệu hiển thị.....	70
6.3.5 Tính năng hoàn tác thao tác người dùng.....	71
6.3.6 Tính năng lưu dữ liệu vào ổ cứng .....	73

### **Danh mục hình**

Hình 2-1 Sơ đồ bố trí chung mô hình .....	14
Hình 2-2 Các thành phần của module điều khiển .....	15
Hình 2-3. Nguyên lý tính toán thời gian phun.....	16
Hình 2-4. Nguyên lý tính toán góc đánh lửa .....	17
Hình 2-5 Cấu trúc dữ liệu giao tiếp hoàn chỉnh.....	17
Hình 3-1. Sự luân chuyển các buffer trong bộ nhớ.....	19
Hình 3-2. Giải thuật của chương trình chính điều khiển ở vi xử lý.....	20
Hình 3-3. Giản đồ thời gian cho chương trình chính .....	22
Hình 3-4. Giải thuật Chương trình ngắn Đọc tín hiệu cảm biến ở các chân analog .....	22

## *Luận văn tốt nghiệp đại học*

Hình 3-5. Giải thuật chương trình ngắn khi có tín hiệu CKP .....	23
Hình 3-6 Giải thuật các chương trình ngắn để thao tác điều khiển .....	24
Hình 3-7. Giản đồ thời gian cho các chương trình ngắn .....	25
Hình 4-1 Sơ đồ tổng quát của chương trình .....	27
Hình 4-2 Sơ đồ đường đi của dữ liệu trong chức năng điều khiển thông số vận hành .....	28
Hình 4-3 Sơ đồ chuẩn hóa đối tượng.....	28
Hình 4-4 Cấu trúc đoạn dữ liệu giao tiếp.....	29
Hình 4-5 Sơ đồ giải thuật xác định phần tử thay đổi.....	29
Hình 4-6 Thông tin sau khi chuyển đổi .....	30
Hình 4-7 Sơ đồ giải thuật chuyển đổi thành dữ liệu giao tiếp.....	31
Hình 4-8 Giải thuật chuyển đổi thành dữ liệu giao tiếp.....	31
Hình 4-9 Sơ đồ giải thuật đóng gói dữ liệu giao tiếp .....	32
Hình 4-10 Sơ đồ đường đi của dữ liệu trong chức năng hiển thị thông số vận hành .....	32
Hình 4-11 Sơ đồ giải thuật tách từng đơn vị dữ liệu hoàn chỉnh.....	33
Hình 4-12 Thuật toán tách lấy trường dữ liệu .....	33
Hình 4-13 Thuật toán chuyển đổi dữ liệu .....	34
Hình 4-14 Giải thuật cắt giá trị .....	34
Hình 4-15 Cơ chế chuyển đổi dữ liệu.....	35
Hình 4-16 Giao diện các chức năng điều khiển.....	36
Hình 4-17 Giao diện tính toán Thời gian mở kim phun .....	36
Hình 4-18 Giao diện tính toán góc đánh lửa thực tế .....	37
Hình 4-19 Giao diện bảng thông số điều chỉnh .....	37

# *Luận văn tốt nghiệp đại học*

Hình 5-1 Giao diện của LabVIEW khi chương trình hoạt động .....	39
Hình 5-2 Đơn vị giao tiếp được gửi đi.....	40
Hình 5-3 Nhập số liệu và gửi lần 1.....	41
Hình 5-4 Nhập số liệu và gửi lần 2.....	41
Hình 5-5 Nhập số liệu và gửi lần 3.....	42
Hình 5-6 Thực hiện hoàn tác .....	42
Hình 5-7 Dữ liệu được lưu vào file text.....	43
Hình 6-1 Giao diện của Labview .....	44
Hình 6-2 Giao diện của một chương trình điều khiển trên LabVIEW .....	45
Hình 6-3 Chương trình mô phỏng trên LabVIEW .....	45
Hình 6-4 Chương trình thu thập dữ liệu.....	46
Hình 6-5 Dữ liệu thô vào labVIEW.....	47
Hình 6-6 Dữ liệu qua xử lí.....	47
Hình 6-7 Giao diện Front Panel .....	48
Hình 6-8 Các cụm chức năng của Front Panel trong LabVIEW .....	49
Hình 6-9 Các chức năng con trong LabVIEW .....	49
hình 6-10 bảng chức năng controls.....	50
Hình 6-11 bảng chức năng indicator .....	51
Hình 6-12 Một vài kiểu dữ liệu trong LabVIEW.....	52
Hình 6-13 Giao diện lập trình ở Block Diagram .....	52
Hình 6-14 Hướng di chuyển của dòng dữ liệu .....	53
Hình 6-15 Phân biệt các loại dây nối.....	54
Hình 6-16 Quá trình chuẩn bị trước khi truyền dữ liệu [1].....	55

## *Luận văn tốt nghiệp đại học*

Hình 6-17 Quá trình truyền dữ liệu bằng giao thức Serial [1] .....	56
Hình 6-18 Quá trình tiếp nhận dữ liệu [1].....	56
Hình 6-19 Cơ chế kiểm tra tránh tràn buffer trong quá trình giao tiếp [1].....	57
Hình 6-20 Quá trình tiếp nhận và xử lí dữ liệu [1] .....	58
Hình 6-21 Cấu trúc của đoạn dữ liệu trong quá trình giao tiếp [1].....	58
Hình 6-22 Giao diện chức năng Case Structure.....	59
Hình 6-23 Giao diện chức năng Event Structure .....	60
Hình 6-24 Case Structure có thể điều khiển bởi nhiều kiểu dữ liệu khác nhau ...	60
Hình 6-25 Chức năng Delay .....	61
Hình 6-26 Chức năng Chuyển đổi kí tự thành số.....	61
Hình 6-27 Giao diện các phép toán trong LabVIEW .....	61
Hình 6-28 Khối lập trình cập nhật dữ liệu từ ô cứng vào các bảng giao tiếp .....	62
Hình 6-29 Chức năng trích xuất dữ liệu từ file Excel .....	62
Hình 6-30 Chức năng xóa phần tử trong bảng.....	63
Hình 6-31 Chức năng tìm kích thước của bản.....	63
Hình 6-32 Chức năng tìm giá trị lớn nhất/nhỏ nhất của bảng .....	63
Hình 6-33 Thuật toán chuyển đổi giá trị hệ số chuẩn hóa thành dữ liệu giao tiếp .....	64
Hình 6-34 Chức năng thay đổi kiểu hiển thị .....	64
Hình 6-35 Chức năng lấy kí tự ra khỏi chuỗi .....	64
Hình 6-36 Chức năng chia lấy phần nguyên và phần dư.....	65
Hình 6-37 Chức năng ghép các kí tự .....	65
Hình 6-38 Khối chức năng chuyển đổi dữ liệu sang Unsigned Integer. ....	65

# *Luận văn tốt nghiệp đại học*

Hình 6-39 Khối chức năng tìm phần tử mà người dùng thay đổi.....	66
Hình 6-40 Chức năng so sánh.....	66
Hình 6-41 Chức năng Lựa chọn.....	67
Hình 6-42 Khối chức năng đóng gói đơn vị hoàn chỉnh.....	67
Hình 6-43 Chức năng đếm số kí tự của chuỗi. ....	67
Hình 6-44 Chương trình giao tiếp với Arduino .....	68
Hình 6-45 Khối chức năng thiết đặt thông số kết nối Serial.....	68
Hình 6-46 Khối chức năng đọc dữ liệu từ Arduino.....	68
Hình 6-47 Khối chức năng gửi dữ liệu xuống Arduino.....	69
Hình 6-48 Khối chức năng kết thúc giao tiếp.....	69
Hình 6-49 Khối chức năng lọc tách từng đơn vị giao tiếp hoàn chỉnh.....	69
Hình 6-50 Chức năng tìm kí tự và tách chuỗi.....	69
Hình 6-51 Nghịch đảo chuỗi.....	70
Hình 6-52 Khối chức năng giữ giá trị.....	70
Hình 6-53 Các tính năng hỗ trợ người dùng trong quá trình giao tiếp.....	71
Hình 6-54 Sơ đồ chức năng Hoàn tác thao tác của người dùng .....	71
Hình 6-55 Khối chức năng xác định vị trí.....	72
Hình 6-56 Khối chức năng cập nhật dữ liệu vào buffer Backup .....	73
Hình 6-57 Khối chức năng sao lưu dữ liệu vào buffer giao tiếp .....	73
Hình 6-58 Sơ đồ chức năng lưu trữ dữ liệu vào ổ cứng .....	74
Hình 6-59 Chức năng xác định vị trí lưu trữ dữ liệu.....	74
Hình 6-60 Chức năng tải dữ liệu từ ổ cứng lên chương trình .....	75
Hình 6-61 Chức năng chuyển đổi kiểu số liệu của bảng tính thành kiểu kí tự ....	75

**TRƯỜNG ĐẠI HỌC BÁCH KHOA  
KHOA KỸ THUẬT GIAO THÔNG  
BỘ MÔN Ô TÔ – MÁY ĐỘNG LỰC**

**CỘNG HÒA XÃ HỘI CHỦ NGHĨA  
VIỆT NAM  
Độc Lập – Tự Do – Hạnh Phúc**

## **NHIỆM VỤ LUẬN VĂN TỐT NGHIỆP**

Sinh viên thực hiện:      Lê Đức Huy      MSSV: 1712665  
                                    Võ Nhật Duy      MSSV: 1713086

Ngành: **Kỹ thuật Ô tô – Máy động lực**

### **1. Số liệu ban đầu:**

- Thông số kỹ thuật của động cơ Honda Wave 110.
- Thông số board mạch Arduino Leonardo.

### **2. Nhiệm vụ luận văn:**

- Thiết kế chương trình điều khiển phun nhiên liệu và đánh lửa điện tử cho động cơ Honda Wave 110.
- Thiết kế chương trình giao tiếp với người dùng trên LabVIEW.

### **3. Nội dung thuyết minh:**

- (1) Tổng quan.
- (2) Thiết kế bối cảnh chung.
- (3) Chương trình điều khiển ở vi xử lý.
- (4) Chương trình giao tiếp với người dùng ở LabView.
- (5) Thực nghiệm.
- (6) Kế hoạch nghiên cứu.
- (7) Phụ lục.

*Luận văn tốt nghiệp đại học*

(8) Tài liệu tham khảo.

**4. Ngày giao nhiệm vụ:** Ngày 21 tháng 09 năm 2020.

**5. Ngày hoàn thành:** Ngày 14 tháng 8 năm 2021.

Nội dung và yêu cầu đã được thông qua Bộ môn Ô tô – Máy động lực

*Ngày ..... tháng .... năm 2021*

*Ngày ..... tháng..... năm 2021*

**Chủ nhiệm Bộ môn**

**GV hướng dẫn**

**Phần dành cho Khoa, Bộ môn:**

- Người duyệt (chấm sơ bộ):.....
- Đơn vị:.....
- Ngày bảo vệ: .....
- Điểm tổng kết:.....
- Nơi lưu trữ luận văn:.....

## LỜI CẢM ƠN

Đi qua những năm tháng Bách Khoa, ta mới biết tuổi trẻ đáng trân trọng như thế nào. Trân trọng, không hẳn là vì có những lúc khó khăn tưởng chừng như gục ngã, cũng không hẳn là vì ta biết mình trưởng thành đến đâu mà đơn giản là vì ta đã làm tất cả những điều đó cùng ai.

Cảm ơn Bách Khoa! 4 năm, có lẽ chẳng đáng gì so với cuộc đời những có thể đã là tất cả của tuổi thanh xuân. Khoảng thời gian vừa qua đã có nhiều khó khăn, cực nhọc nhưng cũng kèm theo đó là sự trưởng thành, những kỉ niệm đẹp, những con người có thể sẽ giúp đỡ ta trong suốt sự nghiệp sau này. Thanh xuân của ta có lẽ đã kết thúc tại đây, tại chính Bách Khoa thân thương này.

Isaac Newton từng nói: “Nếu tôi nhìn được xa hơn người khác, ấy là bởi vì tôi đang đứng trên vai những người khổng lồ”, tốt nghiệp đại học Bách Khoa là một thành tựu, một cột mốc to lớn trong cuộc đời của ta, nhưng có lẽ một mình ta sẽ không làm được điều đó. Lời cảm ơn em xin chân thành gửi đến quý thầy cô trường Đại học Bách Khoa nói chung và các thầy cô khoa Kỹ Thuật Giao Thông nói riêng đã cho chúng em tri thức cũng như tình thương. Cảm ơn anh Hồ Nam Hoa đã tận tình giúp đỡ em mỗi khi gặp khó khăn, chỉ dạy em những kiến thức mới cần để thực hiện luận văn. Đặc biệt cảm ơn thầy Trần Đăng Long, người đã hi sinh rất nhiều thời gian cũng như tâm huyết để dù dắt chúng em trong suốt khoảng thời gian làm đồ án cho đến luận văn. Thầy đã khai sáng cho em hướng thực hiện luận văn cũng như giúp em nhìn thấy những thiếu sót. Em xin chúc thầy cô luôn mạnh khỏe, tâm huyết dạy bảo để dù dắt các thế hệ sinh viên tiếp theo thành thợ, thành người.

Cuối cùng là lời cảm ơn đến các bạn lớp K17OTO2. Cảm ơn vì đã đồng hành những năm tháng vừa qua, cảm ơn đã cùng nhau chia sẻ những thăng trầm trong cuộc sống. Sau này ai cũng có lối đi riêng, khoảng trời riêng, có cuộc sống riêng nhưng vẫn hi vọng vẫn có thể giúp đỡ nhau trong cuộc sống.

Tạm biệt Bách Khoa!

## **LỜI NÓI ĐẦU**

Lịch sử phát triển của động cơ đốt trong đã trải qua hơn 100 năm, từ khi động cơ ra đời cũng đánh dấu sự phát triển vượt bậc về công nghệ của thế giới. Hiện nay với cuộc cách mạng 4.0, các chức năng tự động được tích hợp trên ô tô cũng ngày càng nhiều hơn. Hệ thống phun nhiên liệu trên oto nhờ đó cũng ngày càng được cải tiến nhằm cải thiện các tính năng vận hành. Từ bộ chế hòa khí cho đến phun xăng đơn điểm rồi phun xăng đa điểm, hiện nay tiên tiến nhất là hệ thống phun xăng trực tiếp. Đất nước ta đang trong thời kì công nghiệp hóa, Vinfast là doanh nghiệp Việt Nam đầu tiên tham gia sản xuất xe ô tô, tuy nhiên các thành phần quan trọng của xe lại được mua bản quyền và lắp ráp theo công nghệ của các nước đối tác. Điều khiển phun xăng đánh lửa trên động cơ tuy không mới, nhưng là nền tảng cần thiết để các thế hệ sinh viên có thể tiếp nối, nghiên cứu và phát triển để có thể bắt kịp công nghệ của các nước dẫn đầu, trong tương lai có thể Việt Nam sẽ có khả năng chế tạo động cơ riêng cho các sản phẩm của mình. Ngoài ra đề tài còn phục vụ cho sinh viên có thể hiểu thêm về cách thực hoạt động của động cơ, phục vụ cho trường tham gia các cuộc thi về tiết kiệm năng lượng. Vì các lí do trên nên chúng em quyết định chọn đề tài: "Xây dựng mô hình thực nghiệm điều khiển phun xăng đánh lửa trên động cơ xe máy 1 xylanh" cho luận văn tốt nghiệp của mình.

## **Chương 1. TỔNG QUAN**

### **1.1 Khái niệm và chức năng của sản phẩm trong đề tài**

Mô hình thực nghiệm phun xăng và đánh lửa điện tử cho động cơ xe gắn máy 1 xy  
lanh là mô hình cho phép người dùng có thể tùy chỉnh các thông số điều khiển phun  
xăng và đánh lửa cho động cơ.

Mô hình này có chức năng điều khiển phun xăng đánh lửa điện tử động cơ, giao  
tiếp với giao diện người dùng để từ đó điều chỉnh thông số điều khiển theo ý muốn của  
người dùng.

### **1.2 Lý do và mục đích của đề tài**

Với mô hình này, người dùng cho thẻ thay đổi thông số vận hành, đặc tính của  
động cơ ở nhiều chế độ và cho ra chương trình điều khiển phun xăng đánh lửa tối ưu  
cho việc tăng công suất, tiết kiệm nhiên liệu hoặc giảm phát thải tùy theo nhu cầu của  
sử dụng.

Mô hình của đề tài có thể góp phần phục vụ giảng dạy ở các trường đại học như  
khảo sát các tính năng vận hành, nguyên lý hoạt động, phục vụ nghiên cứu tham gia cuộc  
thi tiết kiệm nhiên liệu.

### **1.3 Điều kiện làm việc**

- Hiệu suất khí nạp thay đổi liên tục theo tốc độ động cơ và độ mở bướm ga.
- Tốc độ động cơ xe gắn máy tối đa khoảng từ 8000 – 9000 vòng/phút.
- Độ mở bướm ga từ 0 – 100%.
- Chịu rung động của động cơ.
- Tia lửa điện tạo ra có thể gây nhiễu tín hiệu.

### **1.4 Yêu cầu kỹ thuật**

- Đọc chính xác tín hiệu các cảm biến.

*Luận văn tốt nghiệp đại học*

- Tính toán đúng thời gian phun và góc đánh lửa.
- Kim phun và IC đánh lửa chấp hành đúng tín hiệu được gửi.
- Hiển thị chính xác và nhuyễn thông số của động cơ trên giao diện người dùng.
- Chương trình điều khiển đọc đúng lệnh tùy chỉnh từ giao diện người dùng.
- Dữ liệu truyền gửi tức thì, liên tục, chính xác theo thời gian thực.
- Phải có bước kiểm tra dữ liệu trước khi tiến hành xử lí.
- Có khả năng lưu trữ thông tin giao tiếp nhằm đối chiếu với dữ liệu từ bộ nhớ Arduino kiểm nghiệm tính chính xác.
- Tần số lấy mẫu của LabVIEW và Arduino độc lập.

## Chương 2. BỐ TRÍ CHUNG CỦA MÔ HÌNH

### 2.1 Thông số đầu vào để thiết kế bố trí chung

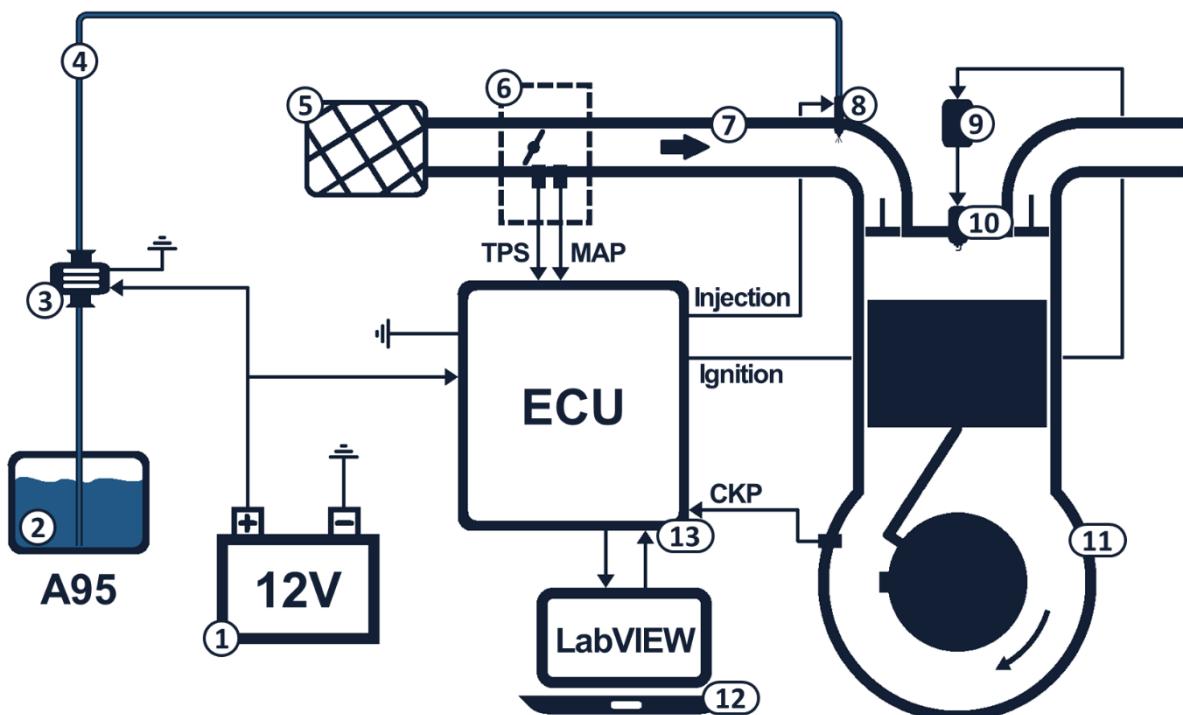
Hệ thống điều khiển đánh lửa trong đề tài là hệ thống đánh lửa điện dung CDI-AC.

Hệ thống điều khiển phun xăng trong đề tài là hệ thống phun xăng điện tử kiểu Djetronic.

Động cơ sử dụng các cảm biến vị trí trực khuỷu (CKP), cảm biến vị trí bướm ga (TPS), cảm biến áp suất chân không đường ống nạp (MAP) để xác định lượng phun nhiên liệu, nhiên liệu được phun trên đường ống nạp, sau bướm ga. Bướm ga, cảm biến MAP và cảm biến TPS được tích hợp vào cụm họng nạp.

Phương thức giao tiếp trong đề tài là loại Serial protocol.

### 2.2 Sơ đồ bố trí chung mô hình



Hình 2-1 Sơ đồ bố trí chung mô hình

- |              |                  |                                |
|--------------|------------------|--------------------------------|
| 1: Ắc quy    | 6: Cụm họng nạp  | 11: Khối block động cơ ban đầu |
| 2: Bình xăng | 7: Đường ống nạp | 12: Máy tính                   |

- 3: *Bơm xăng*                    8: *Kim phun*                    13: *Môđule điều khiển*  
4: *Đường ống xăng*            9: *Bobin đánh lửa*  
5: *Lọc gió*                        10: *Bugi*



*Hình 2-2 Các thành phần của module điều khiển*

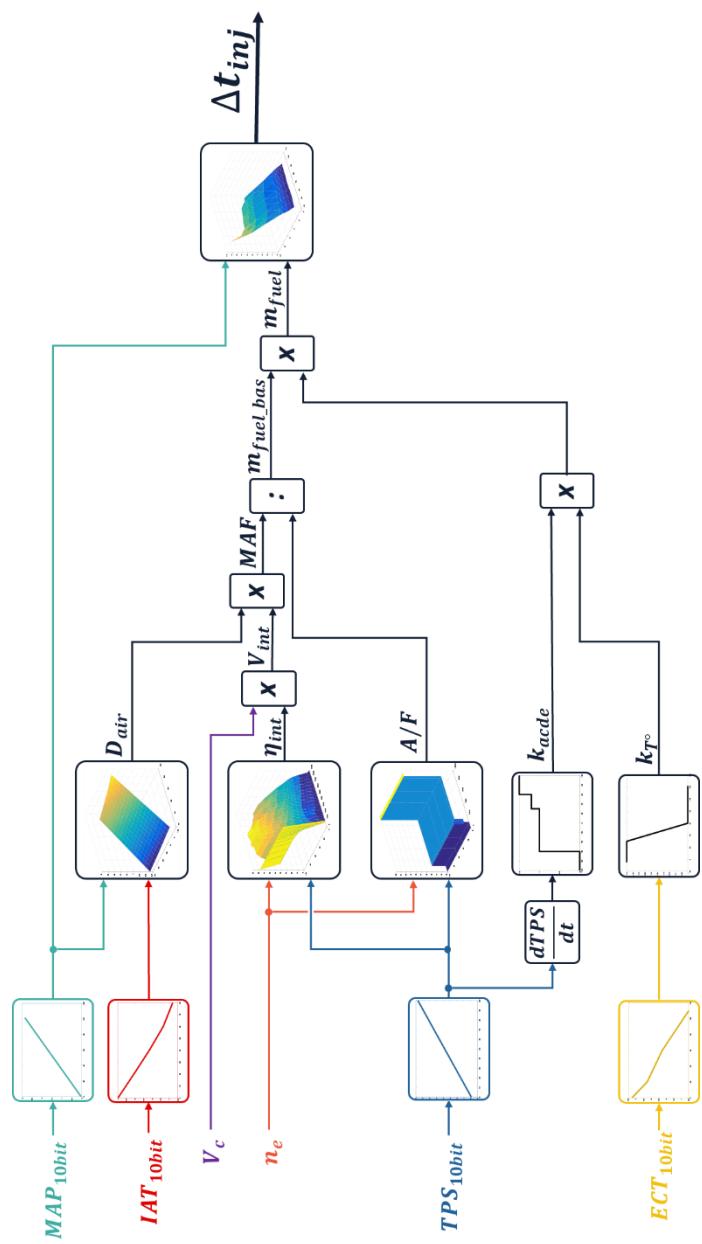
### **2.3 Nguyên lý hoạt động của mô hình**

Khi hoạt động, ác quy 12V (1) cấp nguồn cho bơm xăng (3) và ECU (13). ECU đọc và xử lý tín hiệu các cảm biến, tính toán và xuất tín hiệu cho kim phun (8) và bobin đánh lửa (9). Cùng lúc đó, ECU giao tiếp gửi nhận dữ liệu với máy tính (12) hiển thị giao diện người dùng.

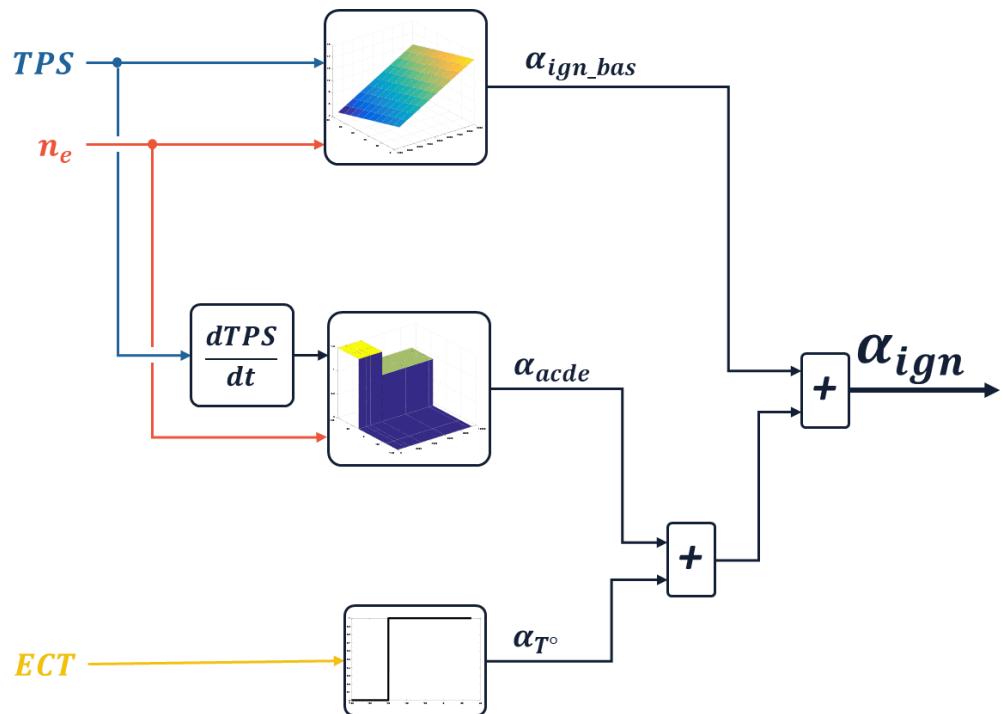
### **2.4 Nguyên lý tính toán thời gian phun và thời điểm đánh lửa**

Dựa vào các thông số từ cảm biến, thông qua các công thức, đồ thị và logic ta có thể tính toán ra được thời gian mở kim phun và góc đánh lửa sớm phù hợp với trạng thái vận hành của động cơ.

Các tín hiệu cảm biến (ngoài CKP) thông qua các khối chức năng calip sẽ thành giá trị thông số trạng thái đầu vào để tính toán. Các khối chức năng tính toán nhận các giá trị thông số đầu vào và cho ra giá trị thông số điều khiển, cụ thể là thời gian mở kim phun và góc đánh lửa. Tham số trong các khối chức năng có thể được người dùng tùy chỉnh bằng giao diện LabVIEW.

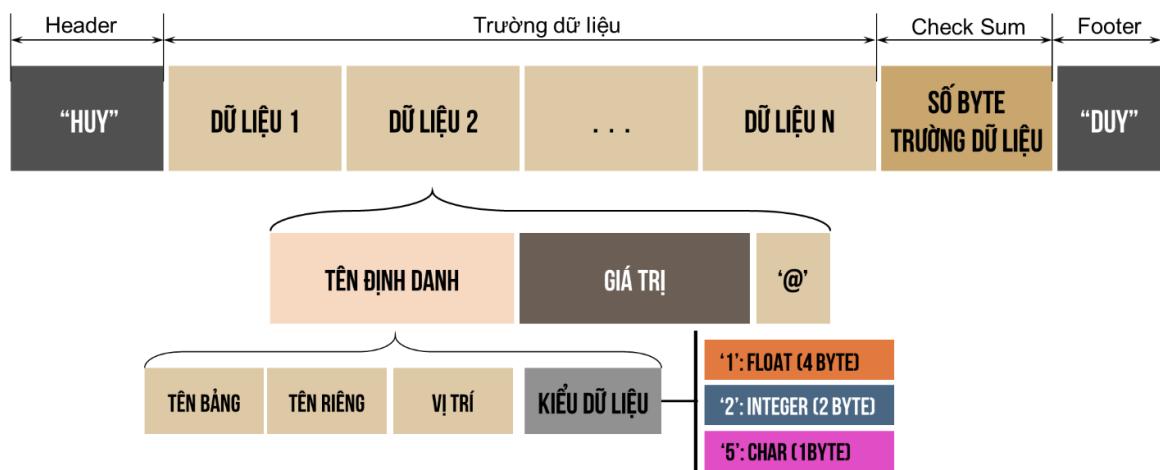


Hình 2-3. Nguyên lý tính toán thời gian phun



Hình 2-4. Nguyên lý tính toán góc đánh lửa

## 2.5 Thiết kế ngôn ngữ giao tiếp chung



Hình 2-5 Cấu trúc dữ liệu giao tiếp hoàn chỉnh

**Header:** gồm 3 ký tự 'HUY' nhằm xác định ký tự bắt đầu của chuỗi.

**Footer:** gồm 3 ký tự 'DUY'.

**Kí tự checksum:** Là kí tự có giá trị hệ thập phân bằng đúng với số kí tự có trong trường dữ liệu giúp phía bên nhận có thể kiểm tra tính chính xác của chuỗi trước khi xử lí.

**Trường dữ liệu:** Là chuỗi kí tự chứa nội dung giao tiếp, các thành phần bên trong bao gồm:

- Tên định danh: Chứa các giá trị giúp chương trình xác định vị trí từng phần tử điều chỉnh và phương pháp chuyển đổi dữ liệu
  - Tên bảng: Xác định phần tử thuộc bảng nào
  - Tên vị trí: Giá trị hàng, cột của phần tử trong bảng
  - Tên riêng: Trong 1 bảng điều chỉnh có thể có tối đa 3 hệ số điều chỉnh nên phải có tên riêng để phân biệt.
  - Kiểu dữ liệu: Giúp chương trình bên nhận thực hiện chuyển đổi thông tin chính xác.
- Giá trị: Là chuỗi kí tự chứa giá trị của các phần tử do người dùng điều chỉnh
- ‘@’: là kí tự kết thúc của mỗi chuỗi kí tự dữ liệu.

## Chương 3. XÂY DỰNG CHƯƠNG TRÌNH ĐIỀU KHIỂN Ở VI XỬ LÝ

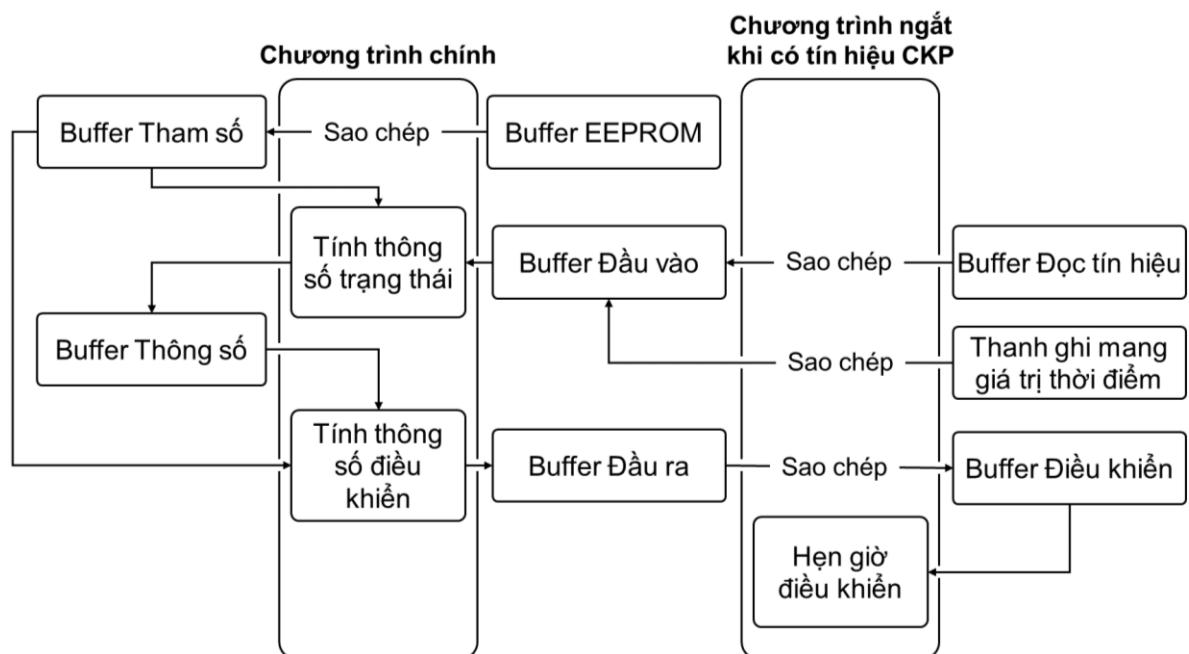
### 3.1 Các chức năng của chương trình điều khiển

Chương trình điều khiển thực hiện các chức năng thu thập tín hiệu cảm biến, xử lý chúng và tính toán các thông số, điều khiển phun xăng đánh lửa và giao tiếp với chương trình LabVIEW.

Các tín hiệu cảm biến analog được vi xử lý đọc từ bộ ADC và đưa vào tính toán như nguyên lý. Đầu ra của quá trình tính toán là thông số để điều khiển phun xăng đánh lửa. Song song với đó, vi xử lý giao tiếp với LabVIEW để hiển thị quá trình tính toán và nhận các lệnh của người dùng.

Chương trình bao gồm chương trình chính và các chương trình ngắn có độ ưu tiên khác nhau. Các chương trình ngắn được kích hoạt bởi một sự kiện hoặc hẹn giờ (kiểm soát bởi Timer).

### 3.2 Sự luân chuyển của dữ liệu trong bộ nhớ

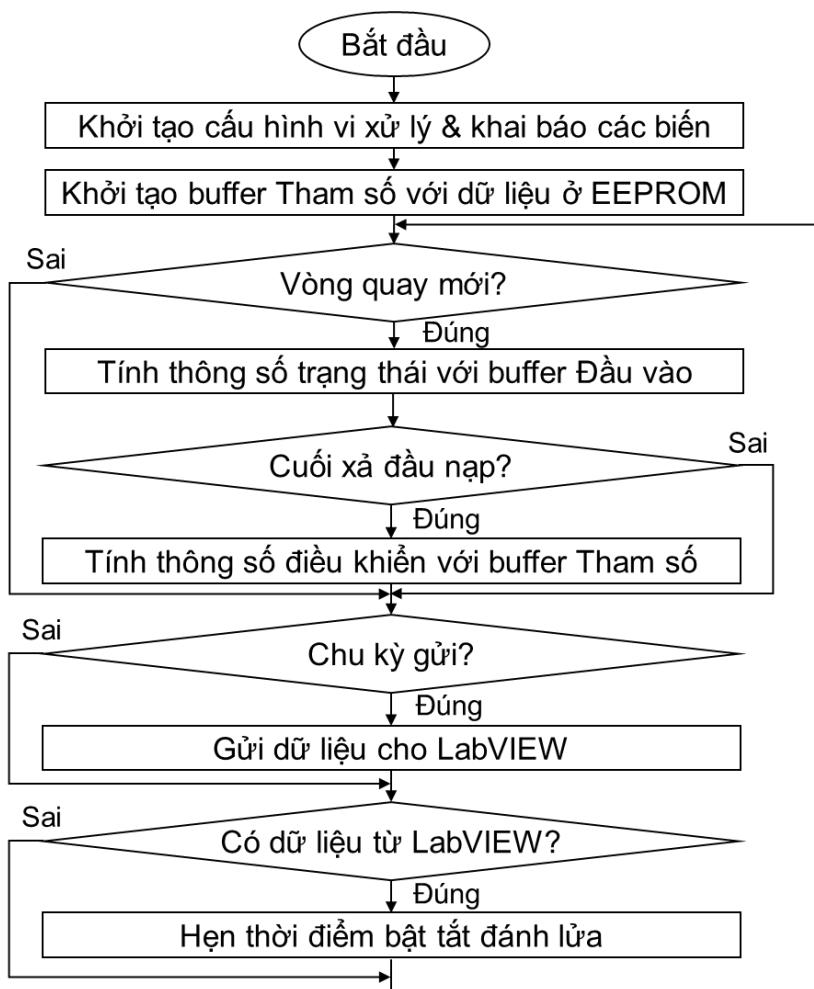


Hình 3-1. Sự luân chuyển các buffer trong bộ nhớ

Tại một thời điểm nhất định, nếu buffer X bị lấy đi làm dữ liệu đầu vào cho một công việc khi đang được cập nhật hoặc lại được cập nhật khi là dữ liệu đầu vào của một công việc đang được thực hiện, sẽ cho ra kết quả không theo mong muốn.

Để đảm bảo các chương trình không sử dụng các đầu vào thiếu chính xác, các giá trị cần thiết sẽ được sao chép ra một buffer khác để thực hiện công việc (như *Hình 5.*). Thêm vào đó, việc sao chép chỉ diễn ra khi các buffer được sao chép đang không được cập nhật hoặc đang là đầu vào của 1 công việc.

### 3.3 Chương trình chính



*Hình 3-2. Giải thuật của chương trình chính điều khiển ở vi xử lý.*

Khi bắt đầu chương trình, cấu hình của chương trình và các biến dùng trong chương trình được khởi tạo. Dữ liệu tham số của luật điều khiển lưu trong EEPROM

(Bộ nhớ chỉ đọc có thể lập trình được bằng điện tử) sẽ được sao chép vào buffer Tham số luật điều khiển.

Sau đó vòng lặp vô hạn được bắt đầu.

Nếu có cờ vòng quay mới, các thông số trạng thái được tính toán bằng buffer Đầu vào tính toán thông số trạng thái. Các thông số trạng thái bao gồm giá trị các cảm biến ECT, IAT, TPS, MAP, giá trị tốc độ động cơ trung bình trong vòng quay và pha làm việc của vòng quay mới là A (Hút - Nén) hay B (Nô - Xả). Việc xác định pha làm việc của động cơ sẽ diễn ra trong 8 vòng quay đầu khi động cơ bắt đầu quay dựa vào việc so sánh 2 giá trị MAP của 2 vòng quay liên tiếp nhau. Động cơ được coi là dừng nếu quá 2 giây chưa có cờ vòng quay mới. Buffer Đầu vào tính toán thông số trạng thái được cập nhật trong chương trình ngắt khi vi điều khiển nhận được tín hiệu CKP.

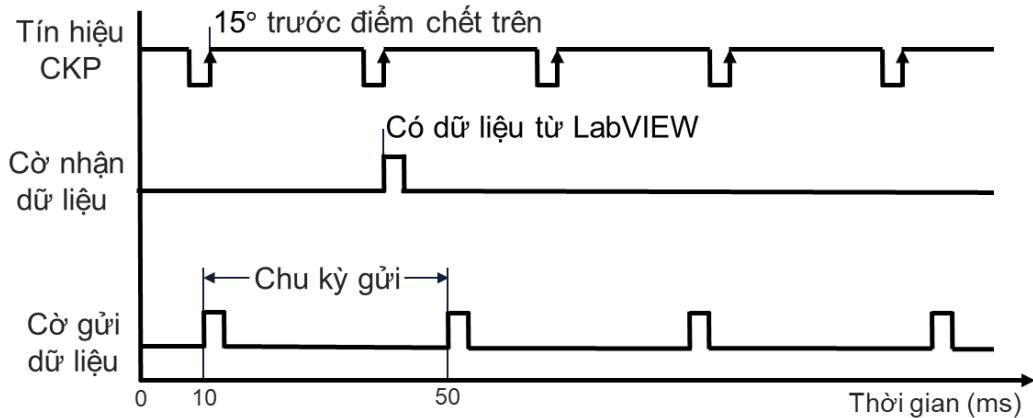
Nếu vòng quay mới rơi vào pha A, thông số điều khiển cũng sẽ được tính toán dựa trên thông số trạng thái và buffer Tham số (calip tín hiệu và luật điều khiển).

Sau khi hoàn thành thì xóa cờ vòng quay mới.

Nếu đến thời điểm gửi dữ liệu (chu kỳ gửi là 40ms), thông số trạng thái, điều khiển và xác nhận (nếu có) sẽ được đóng gói thành các chuỗi gửi theo cấu trúc dữ liệu giao tiếp. Các chuỗi gửi được gửi cho LabVIEW.

Nếu có dữ liệu đúng theo cấu trúc dữ liệu từ LabVIEW gửi đến, các lệnh đã nhận sẽ được xác định và thực thi. Các lệnh này bao gồm: Cập nhật tham số luật điều khiển; Bật tắt phun xăng đánh lửa; Lưu buffer Tham số luật điều khiển vào EEPROM. Sau khi thực thi xong lệnh từ LabVIEW, cờ gửi xác nhận được bật để gửi xác nhận đã thực thi lệnh cho LabVIEW trong chu kỳ

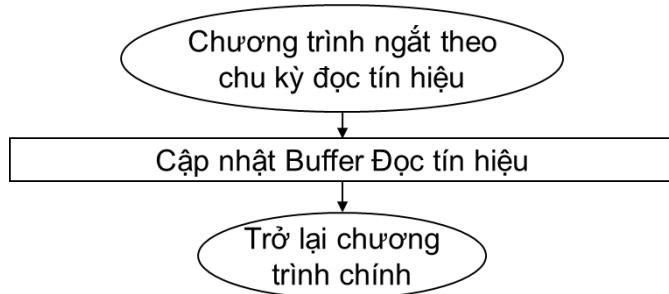
Lệnh Lưu buffer Tham số luật điều khiển vào EEPROM chỉ được thực thi khi phun xăng đánh lửa đang tắt. Và không thể bật lại phun xăng đánh lửa khi toàn bộ buffer chưa được lưu xong bào EEPROM.



Hình 3-3. Giản đồ thời gian cho chương trình chính

### 3.4 Các chương trình ngắt thu thập tín hiệu cảm biến và điều khiển

#### 3.4.1 Chương trình ngắt Đọc tín hiệu cảm biến ở các chân analog

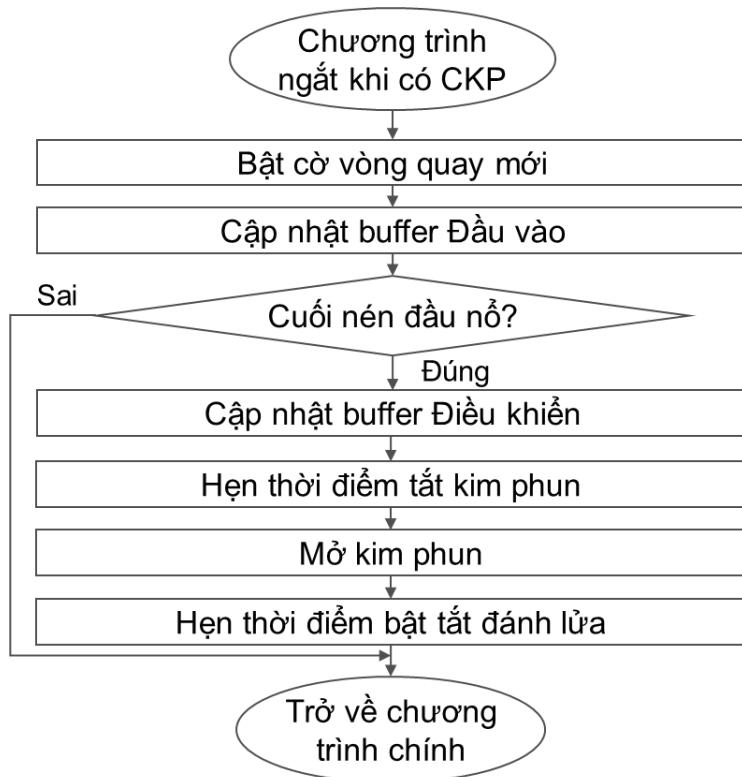


Hình 3-4. Giải thuật Chương trình ngắt Đọc tín hiệu cảm biến ở các chân analog

Khi tới thời điểm đọc tín hiệu điện áp các cảm biến ECT, IAT, MAP, TPS, số lần đọc trong vòng quay sẽ được lưu vào buffer Đọc tín hiệu. Các giá trị tín hiệu điện áp cũng được đọc từ các chân ADC tương ứng và lưu vào buffer Đọc tín hiệu.

Sau đó kết thúc chương trình ngắt.

#### 3.4.2 Chương trình ngắt khi vi điều khiển nhận được tín hiệu CKP



Hình 3-5. Giải thuật chương trình ngắt khi có tín hiệu CKP.

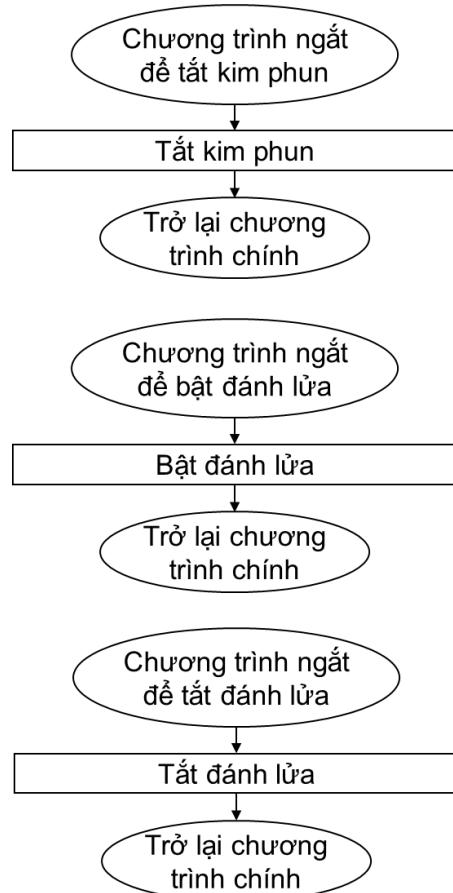
Khi có tín hiệu CKP cạnh lên ( $15^\circ$  trước điểm chết trên), cờ vòng quay mới được bật để tính toán trong chương trình chính.

Buffer Đầu vào tính toán thông số trạng thái được cập nhật từ buffer Đọc tín hiệu và các giá trị thời điểm (được lưu trong thanh ghi bộ đếm thời gian). Sau đó buffer Đọc tín hiệu sẽ được xóa để thu thập dữ liệu mới cho vòng quay mới.

Nếu vòng quay mới rơi vào pha B (Nô - Xả), vì xử lý sẽ hẹn thời điểm tắt kim phun, bật tắt đánh lửa và gửi tín hiệu mở kim phun tới chân điều khiển.

Sau đó kết thúc chương trình ngắt.

### 3.4.3 Các chương trình ngắt để thao tác điều khiển

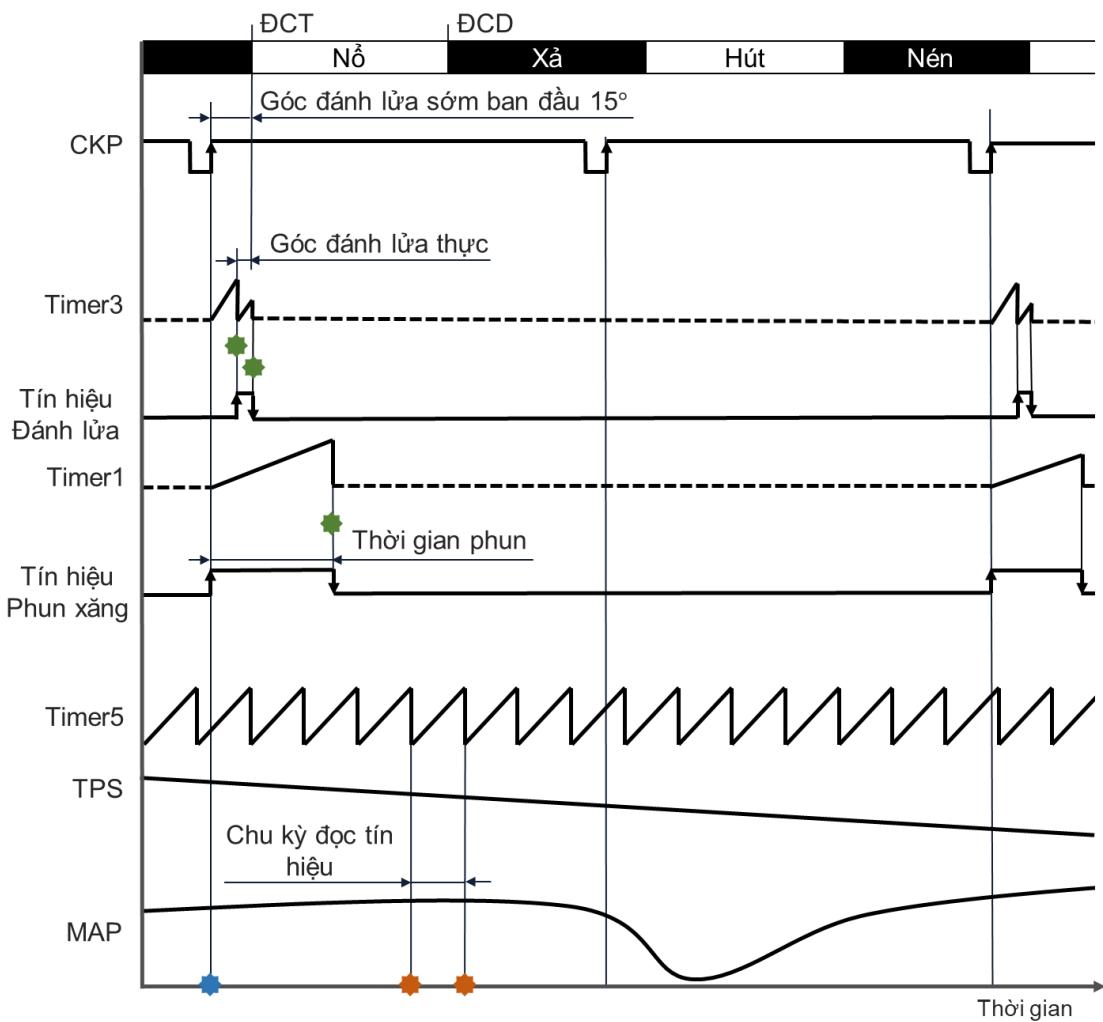


Hình 3-6 Giải thuật các chương trình ngắn để thao tác điều khiển

Khi tới thời điểm để thao tác điều khiển được hẹn ở chương trình ngắn khi vi điều khiển nhận được tín hiệu CKP, các chương trình ngắn được kích hoạt để gửi tín hiệu tương ứng tới chân điều khiển.

Sau khi hoàn thành thì kết thúc chương trình ngắn.

#### 3.4.4 Giản đồ thời gian các chương trình ngắn



Hình 3-7. Giản đồ thời gian cho các chương trình ngắt

Timer 5 được sử dụng để hẹn thời điểm đọc tín hiệu điện áp của các cảm biến. Cùng với đó giá trị thời điểm (thanh ghi Timer 5) cũng được lưu vào thanh ghi Input Capture 5 (ICR5) khi có tín hiệu CKP cạnh lên và được dùng để xác định tốc độ động cơ.

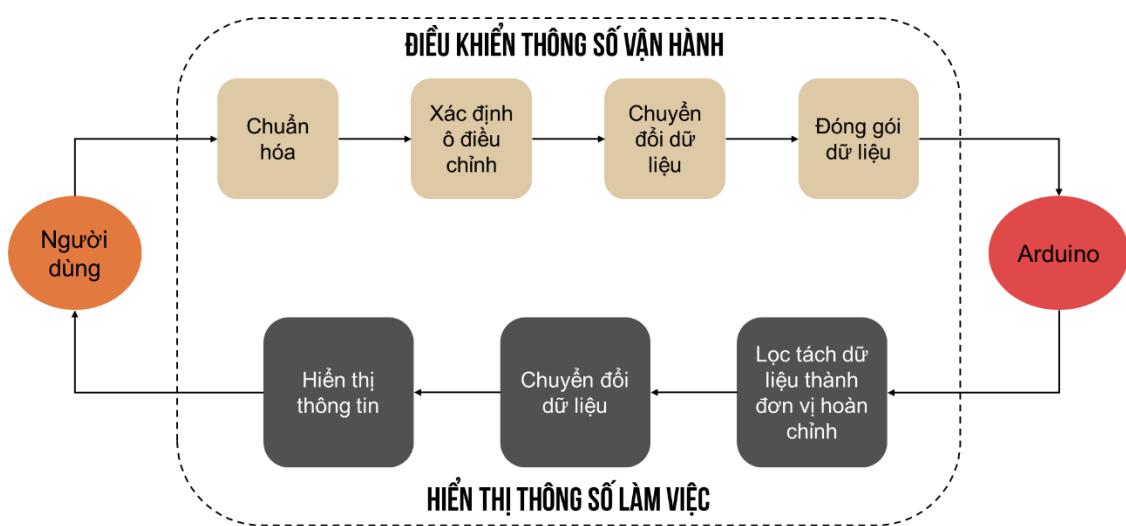
Timer 1 và Timer 3 được sử dụng lần lượt để điều khiển phun xăng và điều khiển đánh lửa. Khi có tín hiệu CKP cạnh lên và vòng quay mới rơi vào pha B, kim phun được mở, Timer 1 được hẹn thời điểm để đóng kim phun theo thời gian mở kim phun, Timer 3 được hẹn thời điểm bật và tắt đánh lửa theo thời gian hiệu chỉnh đánh lửa và thời gian

*Luận văn tốt nghiệp đại học*

đánh lửa ( $100\mu s$ ). Khi điều khiển xong thì Timer 1 và 3 tắt chức năng ngắt để tránh ảnh hưởng tới chương trình chính và các chương trình ngắt khác.

## Chương 4. THIẾT KẾ CHƯƠNG TRÌNH GIAO TIẾP VỚI NGƯỜI DÙNG

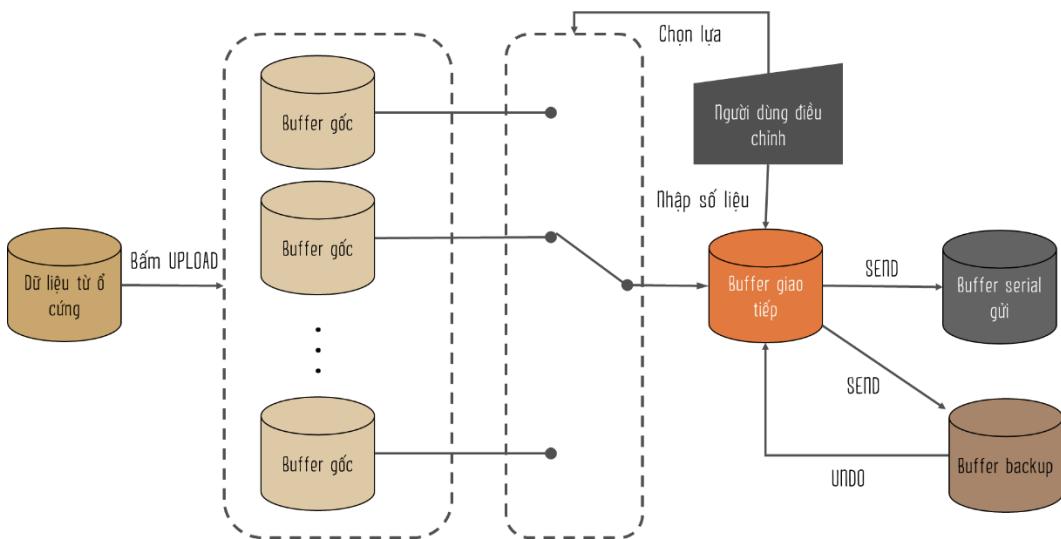
Để giao tiếp với người dùng, chương trình phải làm việc với 2 dòng dữ liệu: dòng dữ liệu gửi lên ở Arduino để hiển thị cho người dùng và dòng dữ liệu người dùng điều chỉnh để gửi xuống cho Arduino. Vì thế chương trình thực hiện 2 khối chức năng chính bao gồm: **Điều khiển thông số vận hành** và **Hiển thị thông số làm việc**.



Hình 4-1 Sơ đồ tổng quát của chương trình

### 4.1 Điều khiển thông số vận hành

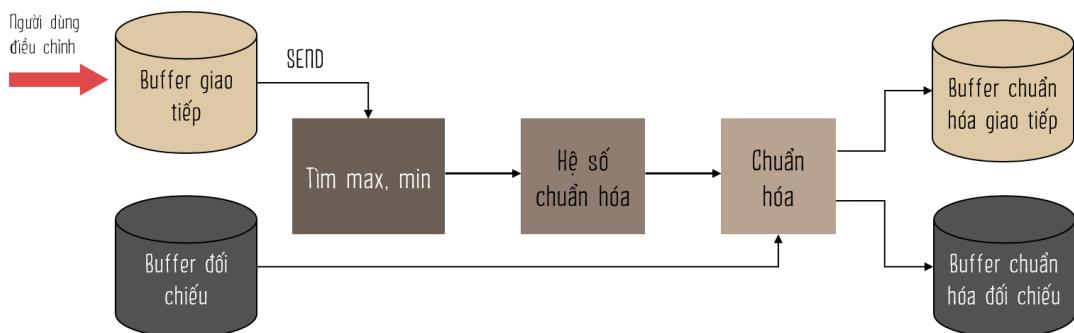
Chương trình chỉ cho phép người dùng điều chỉnh giá trị các ô trong 1 bảng mỗi lần gửi. Khi xác nhận gửi, chương trình sẽ xác định tên định danh, nội dung mà người dùng đã thay đổi. Sau đó sẽ tiến hành chuẩn hóa dữ liệu nhằm tối ưu dữ liệu, đảm bảo khả năng lưu trữ trong buffer của Arduino vì bộ nhớ Arduino có hạn. Tiếp theo sẽ tiến hành chuyển đổi thành dữ liệu theo cấu trúc ngôn ngữ giao tiếp, đóng gói dữ liệu thành đơn vị giao tiếp hoàn chỉnh và gửi xuống Arduino.



Hình 4-2 Sơ đồ đường đi của dữ liệu trong chức năng điều khiển thông số vận hành

#### 4.1.1 Chuẩn hóa dữ liệu

Vì RAM của Arduino dùng để lưu trữ dữ liệu gửi xuống từ LabVIEW là 8Kb (tức có giới hạn) nên để tránh tràn RAM dẫn đến mất thông tin. Vì thế chương trình phải thực hiện chuẩn hóa nhằm rút gọn dữ liệu giao tiếp nhưng vẫn đảm bảo độ chính xác theo yêu cầu.



Hình 4-3 Sơ đồ chuẩn hóa đối tượng

B1: Khi người dùng điều chỉnh thì buffer giao tiếp được cập nhật dữ liệu.

B2: Khi có tín hiệu SEND thì chương trình sẽ tiến hành tìm giá trị Max, Min trong bảng.

B3: Tính hệ số chuẩn hóa theo công thức đề ra.

B4: Lấy từng phần tử trong buffer giao tiếp và buffer đổi chiều chia cho hệ số chuẩn hóa, ta sẽ có được buffer chuẩn hóa giao tiếp và buffer chuẩn hóa đổi chiều.

#### 4.1.2 Xác định phần tử điều chỉnh

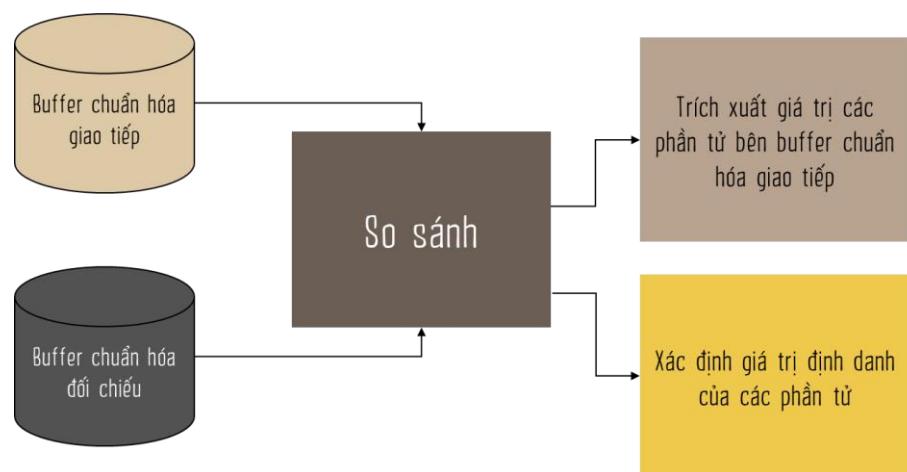
Khi người dùng điều chỉnh thì giá trị của buffer giao tiếp sẽ khác với buffer đổi chiều ở các phần tử điều chỉnh còn kích thước của buffer đổi chiều và buffer giao tiếp là như nhau. Để Arduino xác định được phần tử mà người dùng điều chỉnh đòi hỏi phải xác định được **Tên định danh** và **giá trị**

1. Tên định danh:

- Các giá trị ô điều chỉnh gồm Tên Bảng, Vị trí hàng/cột
- Các giá trị chuẩn hóa gồm Tên Bảng và Tên riêng của giá trị đó
- Kiểu dữ liệu: Mỗi bảng tùy theo độ chính xác và nội dung mà có các loại dữ liệu khác nhau.



Hình 4-4 Cấu trúc đoạn dữ liệu giao tiếp



Hình 4-5 Sơ đồ giải thuật xác định phần tử thay đổi

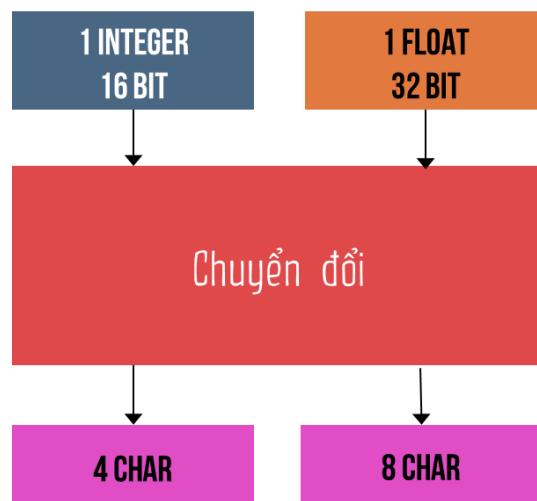
B1: Trích xuất đồng thời từng phần tử trong buffer đối chiếu và buffer giao tiếp và so sánh với nhau.

B2: Thời điểm hai phần tử khác nhau:

- Trích xuất giá trị của phần tử đó bên phía buffer giao tiếp
- Trích xuất giá trị định dạng của phần tử đó.

#### 4.1.3 Chuyển đổi thành dữ liệu giao tiếp

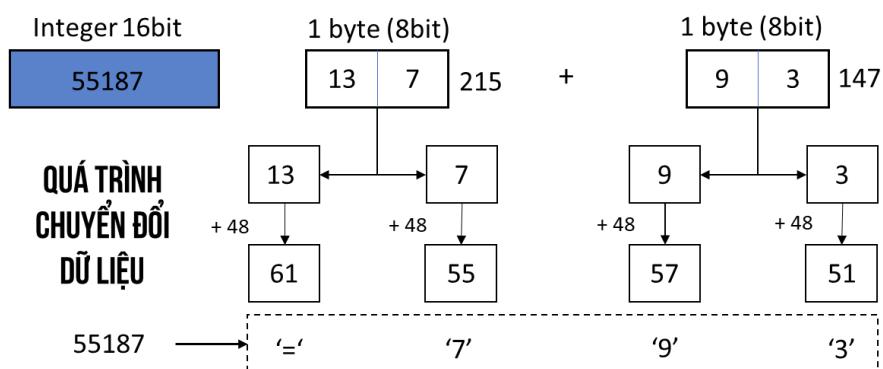
Kiểu dữ liệu đầu vào là dạng số học Integer (16-bit) hoặc Float (32-bit), nếu chuyển từng số liệu về dạng ký tự Char (8-bit) và gửi xuống Arduino thì chuỗi sẽ rất dài và kích thước chuỗi thay đổi liên tục (0.001 cần 5 byte, 1.21 cần 4 byte,...), còn nếu gửi thẳng những giá trị chuyển đổi ấy xuống Arduino thì với mỗi 1 giá trị Float sẽ cần 4 byte và 1 giá trị Integer cần 2 byte, tuy nhiên lại phát sinh vấn đề là ở những byte có giá trị là 0 (null) thì khi truyền gửi sẽ bị mất đi 1 ký tự với mỗi byte có giá trị 0, khiến dữ liệu bị sai lệch và quá trình chuyển đổi thông tin để hiển thị ở Arduino không thành công. Vì 2 lý do trên nên khi chuyển đổi sang dữ liệu giao tiếp phải tách 1 byte (4-bit cao và 4-bit thấp) thành 2 byte với giá trị mỗi byte nằm trong khoảng từ 48 ‘0’ đến 62 ( $48+2^4$ ), vừa đảm bảo số ký tự sau khi chuyển đổi của mỗi kiểu dữ liệu luôn có định, vừa có thể hiển thị để người dùng có thể giám sát quá trình chuyển đổi.



Hình 4-6 Thông tin sau khi chuyển đổi



Hình 4-7 Sơ đồ giải thuật chuyển đổi thành dữ liệu giao tiếp



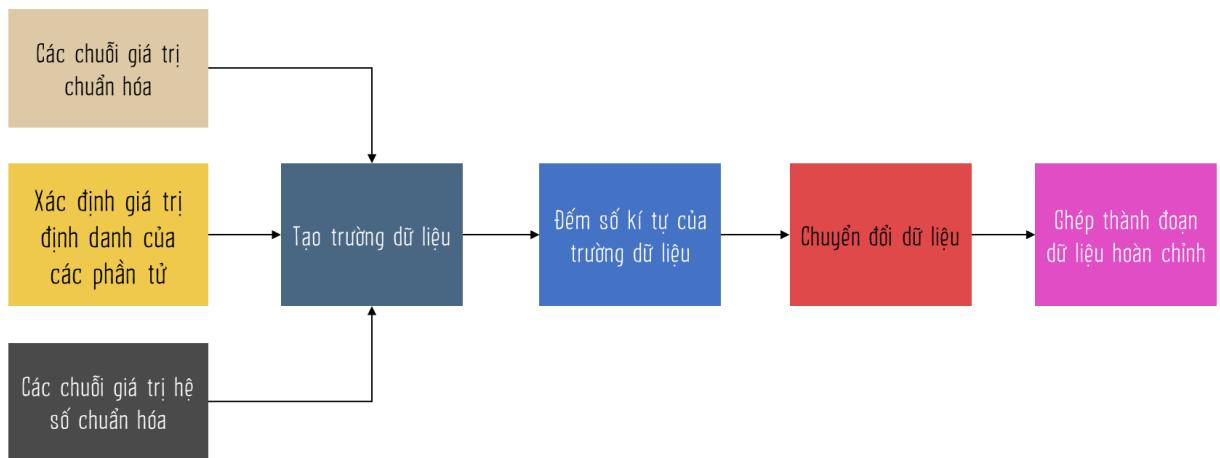
Hình 4-8 Giải thuật chuyển đổi thành dữ liệu giao tiếp

Cơ chế chuyển đổi dữ liệu.

- Giá trị đầu vào sẽ được thay đổi kiểu hiển thị thành dạng kí tự.
- Sau đó tách mỗi kí tự ra ta sẽ có được giá trị của từng byte.
- Ta lấy mỗi byte chia cho 16, phần nguyên sẽ là giá trị 4-bit cao, phần dư sẽ là giá trị 4-bit thấp.

#### 4.1.4 Đóng gói dữ liệu giao tiếp

Sau khi đã có tất cả phần tử cấu thành nên trường dữ liệu, cần phải tạo thêm 1 kí tự có giá trị hệ thập phân bằng tổng số kí tự của trường dữ liệu để khi gửi xuống, Arduino có thể kiểm tra xem chuỗi có chính xác hay không. Ngoài ra cần phải tạo kí tự bắt đầu và kí tự kết thúc để Arduino phân biệt những giá trị điều chỉnh trong các lần gửi để điều chỉnh theo ý muốn người dùng.



Hình 4-9 Sơ đồ giải thuật đóng gói dữ liệu giao tiếp

B1: Tạo trường dữ liệu.

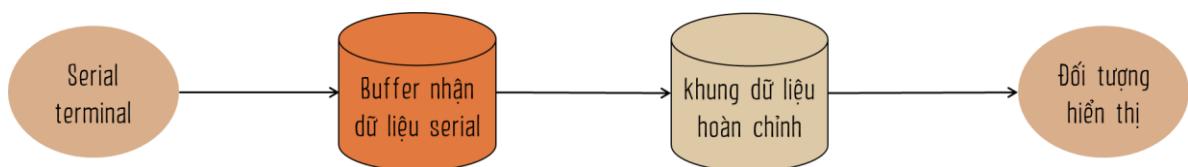
B2: Đếm số kí tự trường dữ liệu.

B3: Chuyển đổi giá trị thành kí tự.

B4: Ghép Header, Footer và kí tự kiểm tra vào với trường dữ liệu để tạo chuỗi giao tiếp hoàn chỉnh.

## 4.2 Hiển thị thông số vận hành

Đối tượng cần có để hiển thị chính là giá trị của các biến trong trường dữ liệu, vì thế phải tách lấy được từng đơn vị giao tiếp hoàn chỉnh, kiểm tra cấu trúc rồi tách lấy trường dữ liệu, sau đó tách các nội dung biến dữ liệu cùng với tên biến và thể hiện lên màn hình. Tuy nhiên từng đoạn dữ liệu Arduino gửi lên theo một tần số riêng và khác với tốc độ xử lí của LabVIEW nên để tránh tình trạng mất dữ liệu phải lưu toàn bộ dữ liệu nhận được từ Arduino vào 1 buffer rồi mới tiến hành tìm giá trị các biến.

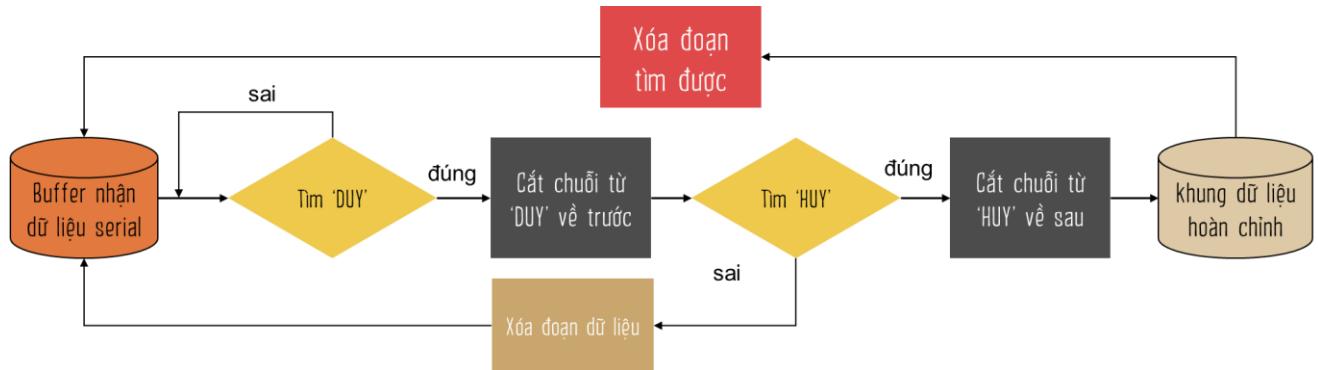


Hình 4-10 Sơ đồ đường đi của dữ liệu trong chức năng hiển thị thông số vận hành

**Buffer nhận dữ liệu Serial:** Lưu trữ tất cả dữ liệu mà Arduino gửi lên.

**Khung dữ liệu hoàn chỉnh:** Lưu giữ khung dữ liệu đã qua kiểm tra.

#### 4.2.1 Lọc tách dữ liệu thành từng đơn vị giao tiếp hoàn chỉnh



Hình 4-11 Sơ đồ giải thuật tách từng đơn vị dữ liệu hoàn chỉnh

B1: Tìm ký tự ‘DUY’.

B2: Cắt chuỗi bắt đầu từ ‘DUY’ trở về trước.

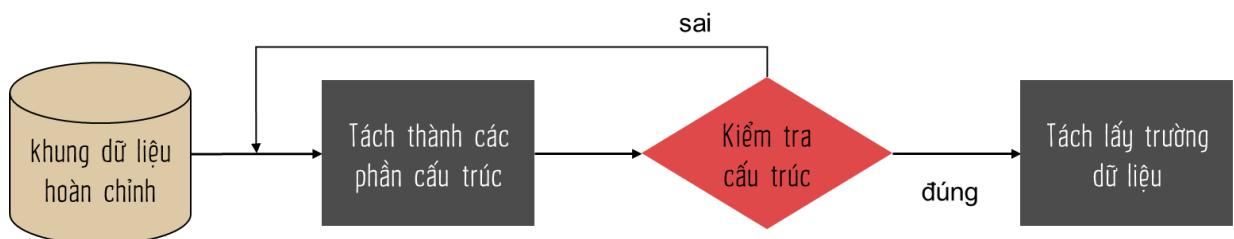
B3: Tìm ‘DUY’.

- Nếu không có ‘DUY’ thì xóa đoạn dữ liệu và trở về B1.
- Nếu có ‘DUY’ thì chuyển sang bước 4.

B4: Cắt chuỗi bắt đầu từ ‘HUY’ trở về sau.

B5 Lấy khung dữ liệu hoàn chỉnh vừa tìm được lưu vào buffer và xóa đoạn dữ liệu đó trong buffer.

#### 4.2.2 Tách trường dữ liệu



Hình 4-12 Thuật toán tách lấy trường dữ liệu

B1: Loại bỏ ‘HUY’ và ‘DUY’.

B2: Lấy ký tự Checksum.

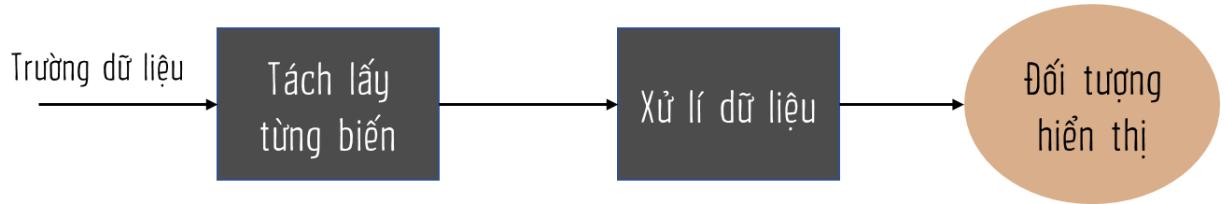
B3: Đếm số ký tự của đoạn dữ liệu.

B4: So sánh giá trị của Checksum và số kí tự đếm được của trường dữ liệu.

- Nếu không bằng nhau thì quay lại B1.
- Nếu bằng nhau thì thực hiện bước 5.

B5 Tách lấy trường dữ liệu chính xác.

#### 4.2.3 *Chuyển đổi thành dữ liệu*



*Hình 4-13 Thuật toán chuyển đổi dữ liệu*

B1: Tách lấy chuỗi giá trị của từng biến.

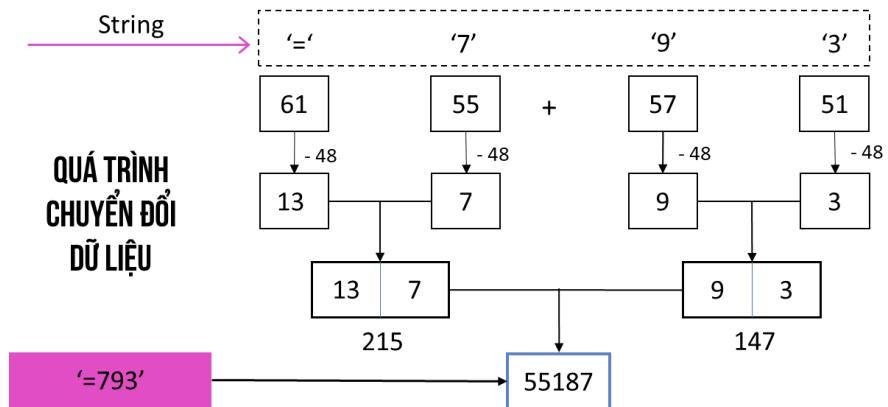
B2: Chuyển đổi dữ liệu thành thông tin rồi hiển thị trên giao diện.



*Hình 4-14 Giải thuật cắt giá trị*

B1: Tìm tên biến rồi lấy chuỗi phía sau đó, sau đó tìm kí tự '@' rồi lấy chuỗi phía trước.

B2: Cắt bỏ kí tự đầu, ta sẽ được chuỗi kí tự mang giá trị của biến.



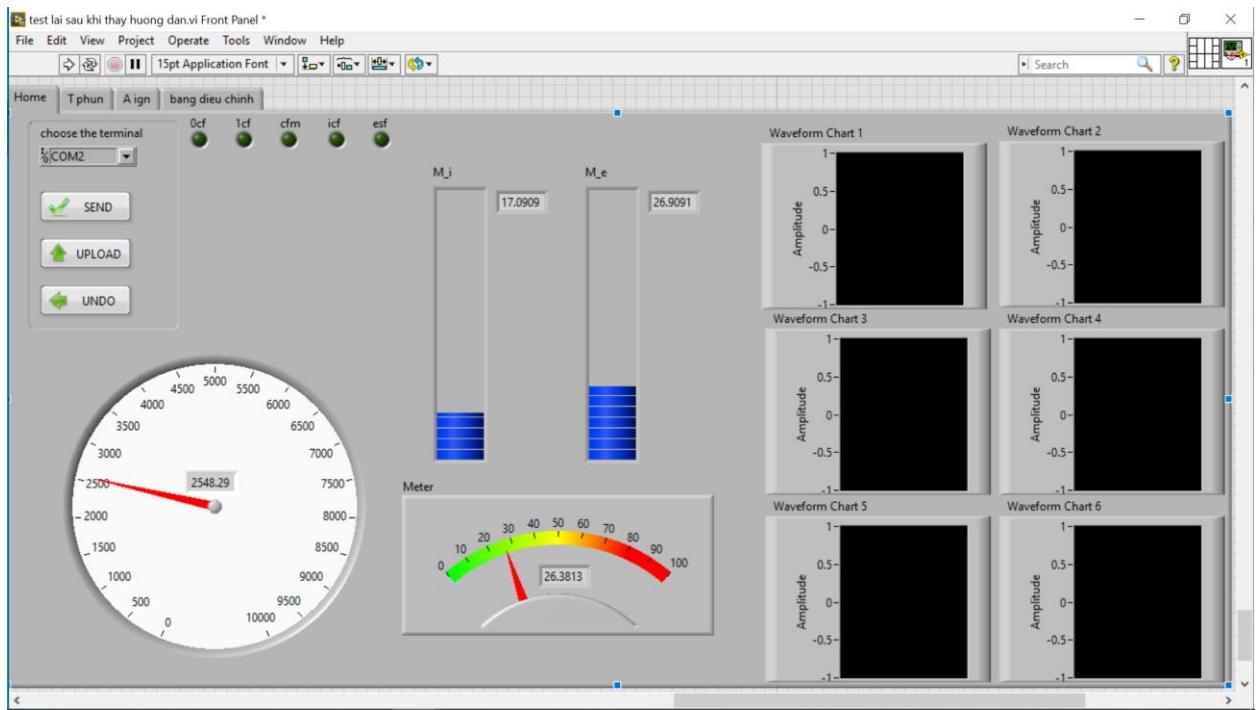
Hình 4-15 Cơ chế chuyển đổi dữ liệu

- B1: Chuyển đổi kiểu hiển thị của kí tự về hệ số thập phân.
- B2: Vì trong quá trình tạo ra chuỗi đã +48 vào giá trị các byte nên khi chuyển đổi lại thành số liệu phải -48.
- B3: Ghép 2 byte liền kề lại thành 1 byte.

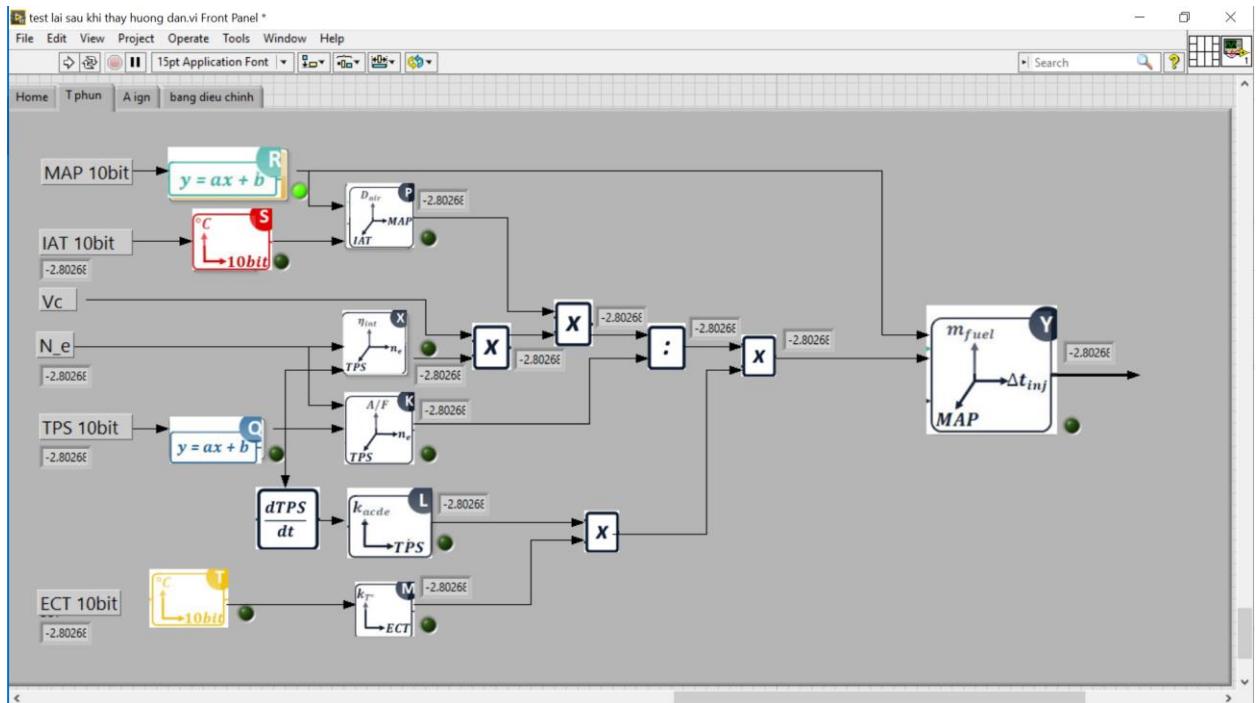
#### 4.2.4 Thiết kế giao diện người dùng

Giao diện người dùng bao gồm các sơ đồ nguyên lý tính toán, bảng giao tiếp chứa thông số để người dùng điều chỉnh, các đồng hồ biểu thị thông số vận hành và các đồ thị thể hiện sự thay đổi các thông số theo thời gian.

## Luận văn tốt nghiệp đại học

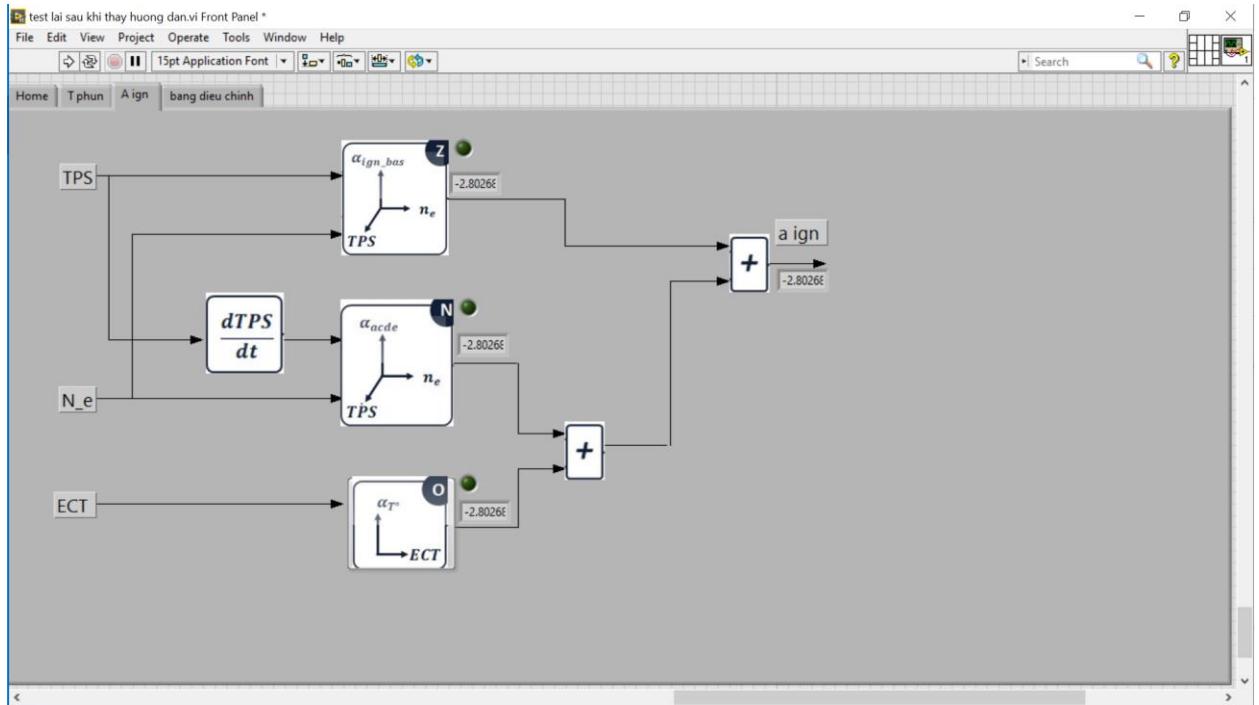


Hình 4-16 Giao diện các chức năng điều khiển



Hình 4-17 Giao diện tính toán Thời gian mở kim phun

## Luận văn tốt nghiệp đại học



Hình 4-18 Giao diện tính toán góc đánh lửa thực tế

The screenshot shows a control panel interface with a menu bar (File, Edit, View, Project, Operate, Tools, Window, Help) and a toolbar. The main area displays a data table titled "bang dieu chinh". The table has 10 columns and 10 rows. The columns are labeled with values: 47.657, 55.149, 64.21, 71.529, 78.499, 88.953, 97.492, 101.325, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0. The rows are labeled with values: 1, 1.5, 2, 2.5, 3, 5, 7, 8, 9, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0.

	47.657	55.149	64.21	71.529	78.499	88.953	97.492	101.325	0	0	0	0	0	0	0	0	0	0	0
1	0.6	0.5	0.4	0.5	0.5	0.4	0.4	0.3	0	0	0	0	0	0	0	0	0	0	
1.5	1	0.9	0.9	0.8	0.8	0.7	0.6	0.6	0	0	0	0	0	0	0	0	0	0	
2	1.4	1.3	1.3	1.2	1.2	1.2	1	0.9	0	0	0	0	0	0	0	0	0	0	
2.5	2.2	1.9	1.6	1.7	1.5	1.4	1.1	1.2	0	0	0	0	0	0	0	0	0	0	
3	2.7	2.3	2	2	1.7	1.9	1.5	1.4	0	0	0	0	0	0	0	0	0	0	
5	4.6	3.8	3.3	3.2	3.1	2.8	2.4	2.2	0	0	0	0	0	0	0	0	0	0	
7	6.3	5.2	4.6	4.5	4.4	4.1	3.3	3.2	0	0	0	0	0	0	0	0	0	0	
8	7.9	6.5	5.3	5.3	4.8	4.7	3.8	3.7	0	0	0	0	0	0	0	0	0	0	
9	9.5	7	5.9	5.8	5.5	5.3	4.2	4.1	0	0	0	0	0	0	0	0	0	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

Hình 4-19 Giao diện bảng thông số điều chỉnh

## **Chương 5. CHẠY THỬ NGHIỆM VÀ ĐÁNH GIÁ KẾT QUẢ**

### **5.1 Kiểm tra các chức năng của 2 chương trình**

#### **5.1.1 Kiểm tra chức năng đọc cảm biến và tính toán thông số trạng thái của chương trình ở vi xử lý**

Dùng một vi điều khiển khác xuất xung PWM (module điều chế độ rộng xung) để giả lập tín hiệu CKP sau gia công. Ta thiết lập cho xung PWM từ vi điều khiển này có thể thay đổi chu kỳ bằng tay ứng với tốc độ 500 đến 8000 vòng/phút của động cơ và có độ rộng xung không đổi.

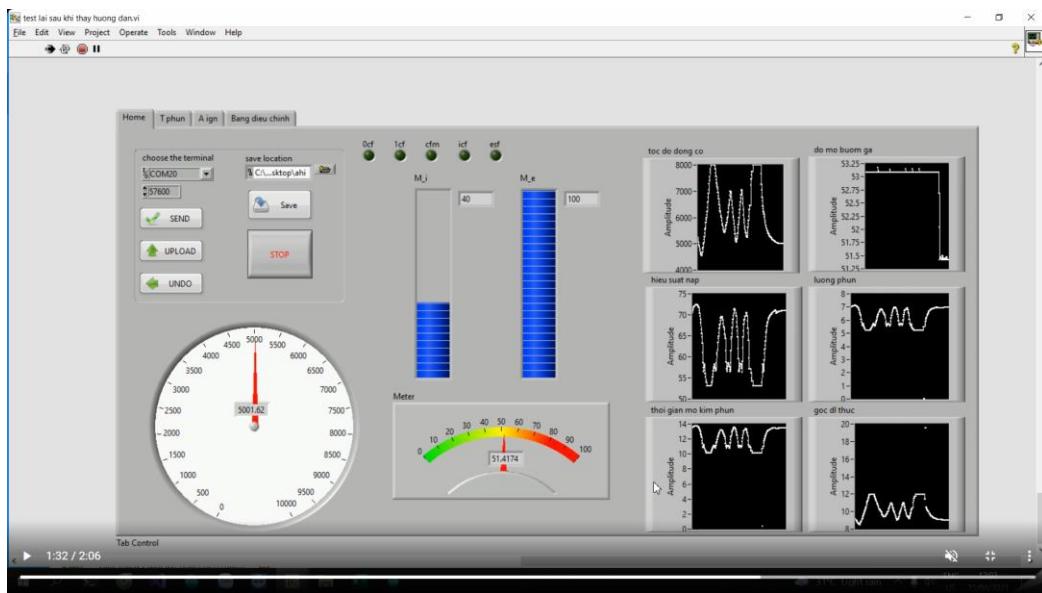
Dùng các biến trở xoay đã biết rõ thông số để giả lập các cảm biến ECT, IAT, TPS và MAP.

Một đoạn chương trình được thêm vào chương trình ở vi điều khiển của đề tài để gửi kết quả tính toán thông số trạng thái qua 1 cổng giao tiếp (khác với cổng giao tiếp với LabVIEW) để hiển thị lên màn hình chương trình Window Form.

Chạy chương trình để kiểm tra chương trình có đọc tín hiệu cảm biến và tính toán thông trạng thái đúng như mong muốn không.

#### **5.1.2 Kiểm tra chức năng gửi dữ liệu của chương trình ở vi xử lý và chức năng nhận dữ liệu, hiển thị thông số của chương trình LabVIEW**

Dùng các phần cứng như phần 5.1.1.. Chạy chương trình và kiểm tra xem các thông số trạng thái hiển thị trên màn hình LabVIEW có đúng mong muốn (như kết quả có được ở phần 5.1.1.) hay không.



Hình 5-1 Giao diện của LabVIEW khi chương trình hoạt động

### 5.1.3 Kiểm tra chức năng tính toán thông số điều khiển của chương trình ở vi xử lý

Chạy chương trình như phần 5.1.2.. Kiểm tra các giá trị kết quả tính toán do vi xử lý gửi lên có đúng như dự đoán không.

### 5.1.4 Kiểm tra chức năng điều khiển của chương trình ở vi xử lý

Dùng các phần cứng như phần 5.1.1.. Ở 2 chân điều khiển phun xăng và điều khiển đánh lửa, nối mạch qua điện trở qua điện trở rồi về mass. Dùng dao động ký để đọc và vẽ sơ đồ tín hiệu điện áp theo thời gian của chân nhận tín hiệu CKP và 2 chân điều khiển phun xăng đánh lửa.

Chạy chương trình và kiểm tra xem tín hiệu ở các chân có đúng như kết quả tính toán thông số điều khiển hay không.

### 5.1.5 Kiểm tra chức năng nhận dữ liệu, lưu tham số luật điều khiển vào EEPROM của chương trình ở vi xử lý và gửi dữ liệu, lưu tham số luật điều khiển của chương trình LabVIEW

Chạy chương trình, gửi các giá trị tham số luật điều khiển (ở nhiều bảng khác nhau) mới từ LabVIEW xuống vi xử lý. Gửi lệnh lưu buffer Tham số luật điều khiển vào EEPROM từ LabVIEW xuống cho vi xử lý và lưu buffer này ở LabVIEW vào file excel.

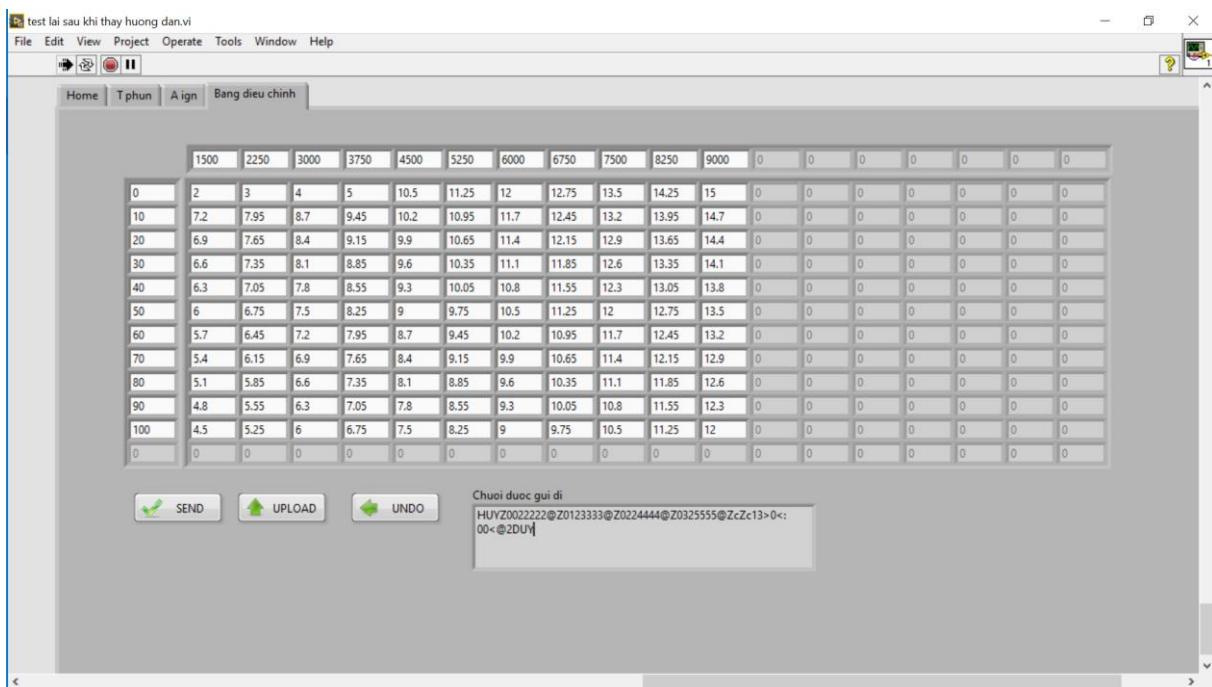
## Luận văn tốt nghiệp đại học

Nạp một chương trình mới vào vi điều khiển có chức năng giao tiếp với một chương trình Window Form để lưu dữ liệu từ EEPROM ra file excel theo format như file excel lưu từ LabVIEW.

So sánh 2 file excel, nếu 2 file giống nhau hoàn toàn thì các chức năng cần kiểm tra đạt yêu cầu.

### 5.1.6 Kiểm tra các tính năng điều chỉnh thông số người dùng của LabVIEW

**Kiểm tra tính năng xác định phần tử điều chỉnh:** Khi nhập dữ liệu và gửi xuống Arduino, kiểm tra buffer ‘chuỗi được gửi đi’ xem có giống cấu trúc đơn vị giao tiếp đã đề ra hay không.



Hình 5-2 Đơn vị giao tiếp được gửi đi

**Kiểm tra tính năng hoàn tác giao tác:** Nhập các số liệu và nhấn SEND để xác nhận gửi, sau 5 lần gửi, Bấm UNDO và kiểm tra xem giá trị của bảng có trùng khớp với giá trị qua từng lần gửi hay không.

## Luận văn tốt nghiệp đại học

The screenshot shows a software window titled "test lai sau khi thay huong dan.vi". The menu bar includes File, Edit, View, Project, Operate, Tools, Window, Help. The toolbar has icons for Home, Tphun, Align, and Bang dieu chinh. Below the toolbar is a 10x13 grid of numerical data. At the bottom are three buttons: SEND, UPLOAD, and UNDO. A text input field labeled "Chuoi duoc gui di" contains the text "HUVZ002116:@ DUY".

	1500	2250	3000	3750	4500	5250	6000	6750	7500	8250	9000	0	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	7.2	7.95	8.7	9.45	10.2	10.95	11.7	12.45	13.2	13.95	14.7	0	0	0	0	0	0	0	0	0	0	0
20	6.9	7.65	8.4	9.15	9.9	10.65	11.4	12.15	12.9	13.65	14.4	0	0	0	0	0	0	0	0	0	0	0
30	6.6	7.35	8.1	8.85	9.6	10.35	11.1	11.85	12.6	13.35	14.1	0	0	0	0	0	0	0	0	0	0	0
40	6.3	7.05	7.8	8.55	9.3	10.05	10.8	11.55	12.3	13.05	13.8	0	0	0	0	0	0	0	0	0	0	0
50	6	6.75	7.5	8.25	9	9.75	10.5	11.25	12	12.75	13.5	0	0	0	0	0	0	0	0	0	0	0
60	5.7	6.45	7.2	7.95	8.7	9.45	10.2	10.95	11.7	12.45	13.2	0	0	0	0	0	0	0	0	0	0	0
70	5.4	6.15	6.9	7.65	8.4	9.15	9.9	10.65	11.4	12.15	12.9	0	0	0	0	0	0	0	0	0	0	0
80	5.1	5.85	6.6	7.35	8.1	8.85	9.6	10.35	11.1	11.85	12.6	0	0	0	0	0	0	0	0	0	0	0
90	4.8	5.55	6.3	7.05	7.8	8.55	9.3	10.05	10.8	11.55	12.3	0	0	0	0	0	0	0	0	0	0	0
100	4.5	5.25	6	6.75	7.5	8.25	9	9.75	10.5	11.25	12	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

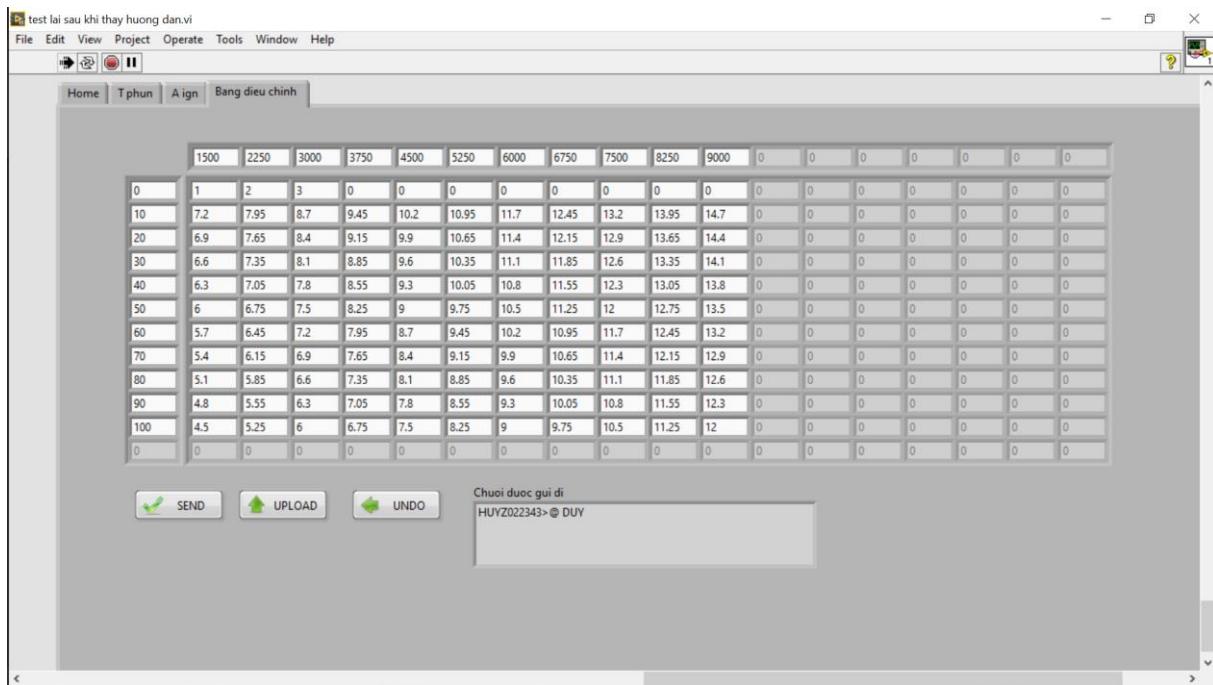
Hình 5-3 Nhập số liệu và gửi lần 1

The screenshot shows a software window titled "test lai sau khi thay huong dan.vi". The menu bar includes File, Edit, View, Project, Operate, Tools, Window, Help. The toolbar has icons for Home, Tphun, Align, and Bang dieu chinh. Below the toolbar is a 10x13 grid of numerical data. At the bottom are three buttons: SEND, UPLOAD, and UNDO. A text input field labeled "Chuoi duoc gui di" contains the text "HUVZ01222=4@ DUY".

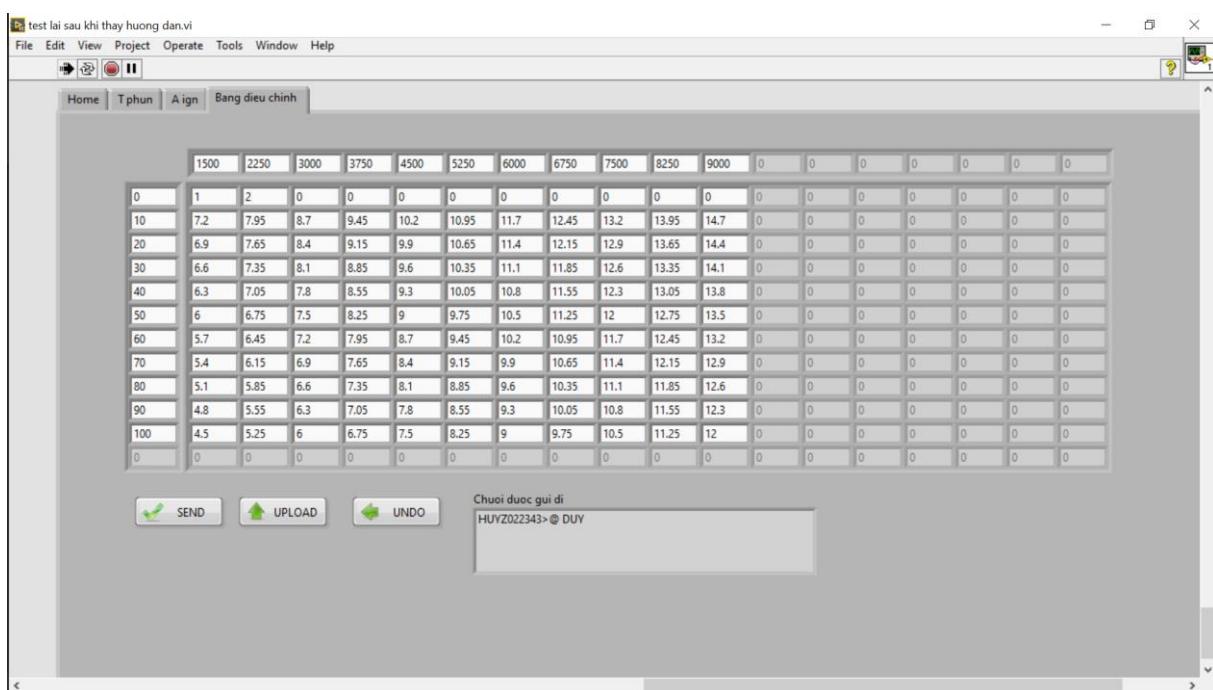
	1500	2250	3000	3750	4500	5250	6000	6750	7500	8250	9000	0	0	0	0	0	0	0	0	0	0	0
0	1	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	7.2	7.95	8.7	9.45	10.2	10.95	11.7	12.45	13.2	13.95	14.7	0	0	0	0	0	0	0	0	0	0	0
20	6.9	7.65	8.4	9.15	9.9	10.65	11.4	12.15	12.9	13.65	14.4	0	0	0	0	0	0	0	0	0	0	0
30	6.6	7.35	8.1	8.85	9.6	10.35	11.1	11.85	12.6	13.35	14.1	0	0	0	0	0	0	0	0	0	0	0
40	6.3	7.05	7.8	8.55	9.3	10.05	10.8	11.55	12.3	13.05	13.8	0	0	0	0	0	0	0	0	0	0	0
50	6	6.75	7.5	8.25	9	9.75	10.5	11.25	12	12.75	13.5	0	0	0	0	0	0	0	0	0	0	0
60	5.7	6.45	7.2	7.95	8.7	9.45	10.2	10.95	11.7	12.45	13.2	0	0	0	0	0	0	0	0	0	0	0
70	5.4	6.15	6.9	7.65	8.4	9.15	9.9	10.65	11.4	12.15	12.9	0	0	0	0	0	0	0	0	0	0	0
80	5.1	5.85	6.6	7.35	8.1	8.85	9.6	10.35	11.1	11.85	12.6	0	0	0	0	0	0	0	0	0	0	0
90	4.8	5.55	6.3	7.05	7.8	8.55	9.3	10.05	10.8	11.55	12.3	0	0	0	0	0	0	0	0	0	0	0
100	4.5	5.25	6	6.75	7.5	8.25	9	9.75	10.5	11.25	12	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Hình 5-4 Nhập số liệu và gửi lần 2

## Luận văn tốt nghiệp đại học



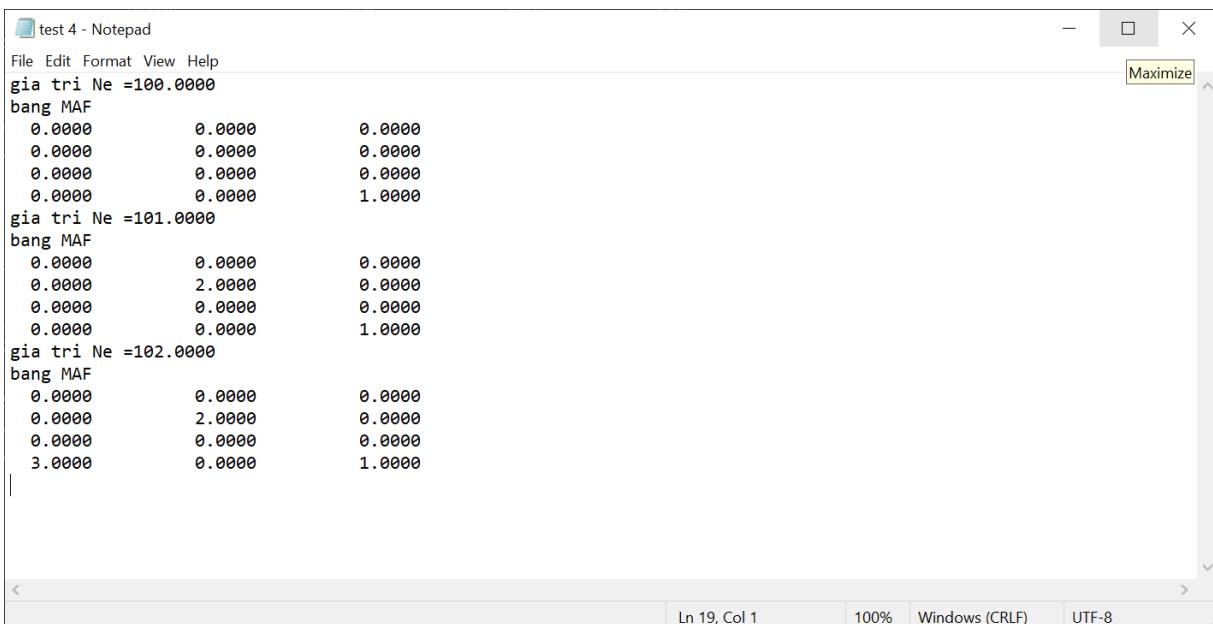
Hình 5-5 Nhập số liệu và gửi lần 3



Hình 5-6 Thực hiện hoàn tác

**Kiểm tra chức năng lưu trữ thông tin:** Sau mỗi lần bấm SAVE, dữ liệu sẽ được lưu vào file text 1 lần.

## Luận văn tốt nghiệp đại học



The screenshot shows a Notepad window titled "test 4 - Notepad". The content of the file contains experimental data for three different values of Ne (100.0000, 101.0000, and 102.0000). Each section starts with "gia tri Ne = [value]" followed by "bang MAF" and a 4x3 matrix of values. The matrices are:

gia tri Ne = 100.0000	0.0000	0.0000
	0.0000	0.0000
	0.0000	0.0000
	0.0000	0.0000
gia tri Ne = 101.0000	0.0000	0.0000
	0.0000	2.0000
	0.0000	0.0000
	0.0000	0.0000
gia tri Ne = 102.0000	0.0000	0.0000
	0.0000	2.0000
	0.0000	0.0000
	3.0000	0.0000

At the bottom of the window, there are status bars showing "Ln 19, Col 1", "100%", "Windows (CRLF)", and "UTF-8".

Hình 5-7 Dữ liệu được lưu vào file text

### 5.1.7 Chạy thực nghiệm

Tích hợp vi điều khiển với mạch đánh lửa, khuếch đại công suất kim phun, giá công tín hiệu CKP, và nguồn ổn áp 8V và 5V để tạo thành module điều khiển.

Xây dựng mô hình động cơ xe gắn máy Wave 110 gồm cụm họng nạp tích hợp 3 cảm biến và bướm ga, và hệ thống nhiên liệu (bơm xăng, kim phun).

Thiết lập mô hình hoàn thiện gồm mô hình động cơ, module điều khiển và máy tính chạy chương trình LabVIEW như *Hình 1*

Chạy chương trình, khởi động động cơ và gửi các lệnh từ giao diện người dùng. Quan sát và đánh giá kết quả thực nghiệm.

## 5.2 Đánh giá kết quả

Chạy mô phỏng các tín hiệu cảm biến và tính toán cho kết quả:

- Hiển thị khá mượt và chính xác tín hiệu.
- Các kết quả đều ra tính toán đạt yêu cầu.

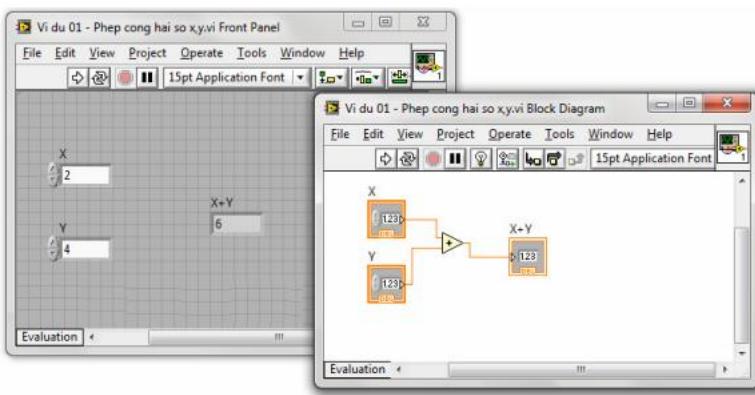
## Chương 6. PHỤ LỤC A: CHƯƠNG TRÌNH GIAO TIẾP VỚI NGƯỜI DÙNG

Chương trình giao tiếp sẽ bao gồm 2 khối chức năng chính: cho phép người dùng điều chỉnh thông số làm việc và thu thập dữ liệu từ Arduino gửi lên và chuyển đổi thành thông tin hiển thị lên màn hình giao diện. Các chức năng trên được xây dựng thông qua phần mềm Lập trình độ họa LabVIEW.

### 6.1 Giới thiệu về LabVIEW

LabVIEW (viết tắt của Laboratory Virtual Instrumentation Engineering Workbench) là môi trường ngôn ngữ đồ họa hiệu quả trong việc giao tiếp đa kênh giữa con người, thuật toán và các thiết bị. Gọi LabVIEW là ngôn ngữ đồ họa hiệu quả vì về cách thức lập trình, LabVIEW khác với các ngôn ngữ C (hay Python, Basic, vv.) ở điểm thay vì sử dụng các từ vựng (từ khóa) cố định thì LabVIEW sử dụng các khối hình ảnh sinh động và các dây nối để tạo ra các lệnh và các hàm như trong hình.

Về ý nghĩa kỹ thuật, LabVIEW cũng được dùng để lập trình ra các chương trình (source code: mã nguồn) trên máy tính tương tự các ngôn ngữ lập trình dựa trên chữ (text-based language) như C, Python, Java, Basic, vv.



Hình 6-1 Giao diện của Labview

**Ứng dụng:** LabVIEW giúp kỹ sư kết nối bất kỳ cảm biến, và bất kỳ cơ cấu chấp hành nào với máy tính. LabVIEW có thể được sử dụng để xử lý các kiểu dữ liệu như tín hiệu tương tự (analog), tín hiệu số (digital) hình ảnh (vision), âm thanh (audio), vv.

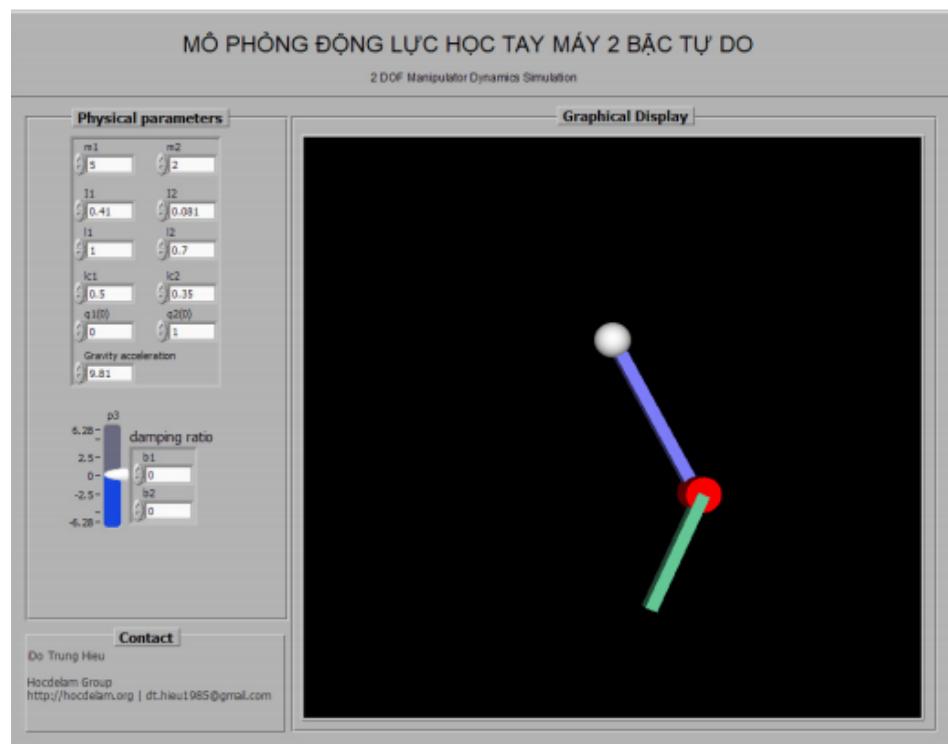
## Luận văn tốt nghiệp đại học

Trong thực tế ứng dụng Labview dùng để:

LabView có thể điều khiển các cơ cấu chấp hành một cách thủ công hay tự động.

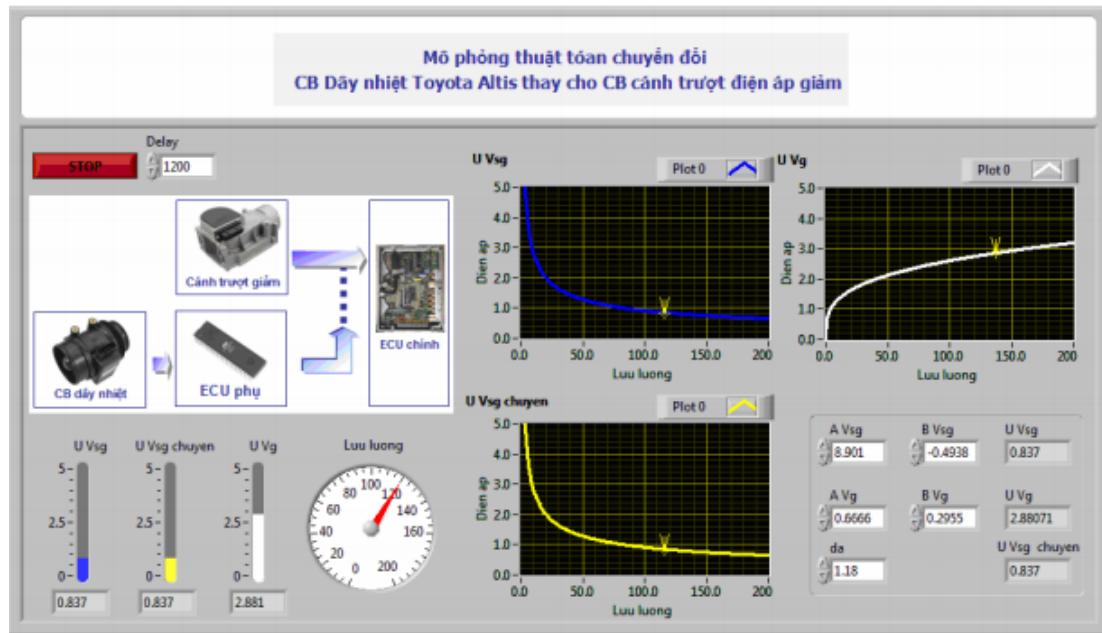


Hình 6-2 Giao diện của một chương trình điều khiển trên LabVIEW  
LabView dùng để mô phỏng.



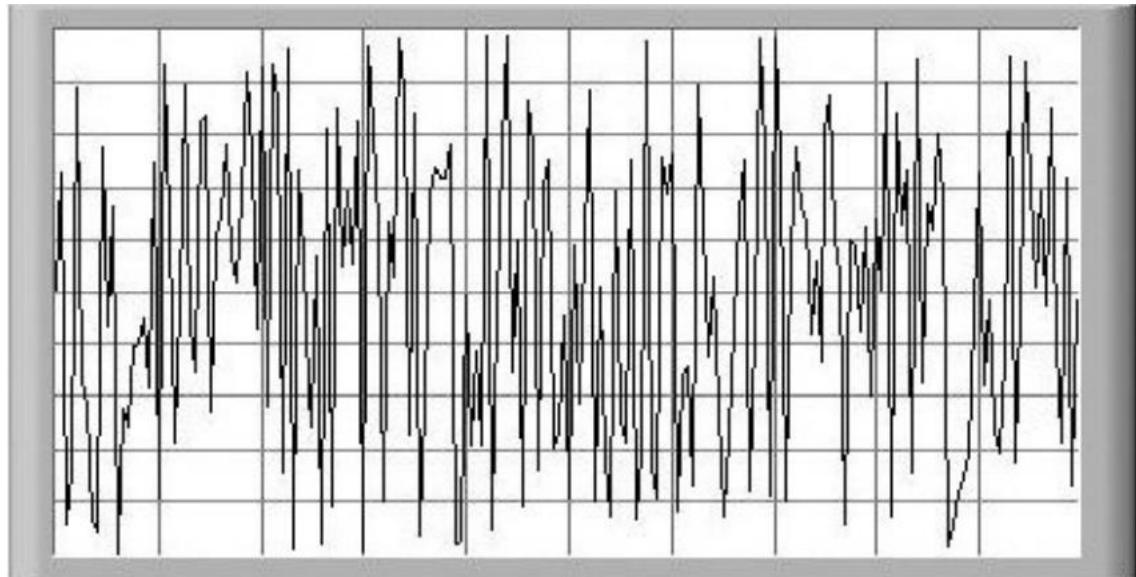
Hình 6-3 Chương trình mô phỏng trên LabVIEW  
Thu thập dữ liệu.

## Luận văn tốt nghiệp đại học

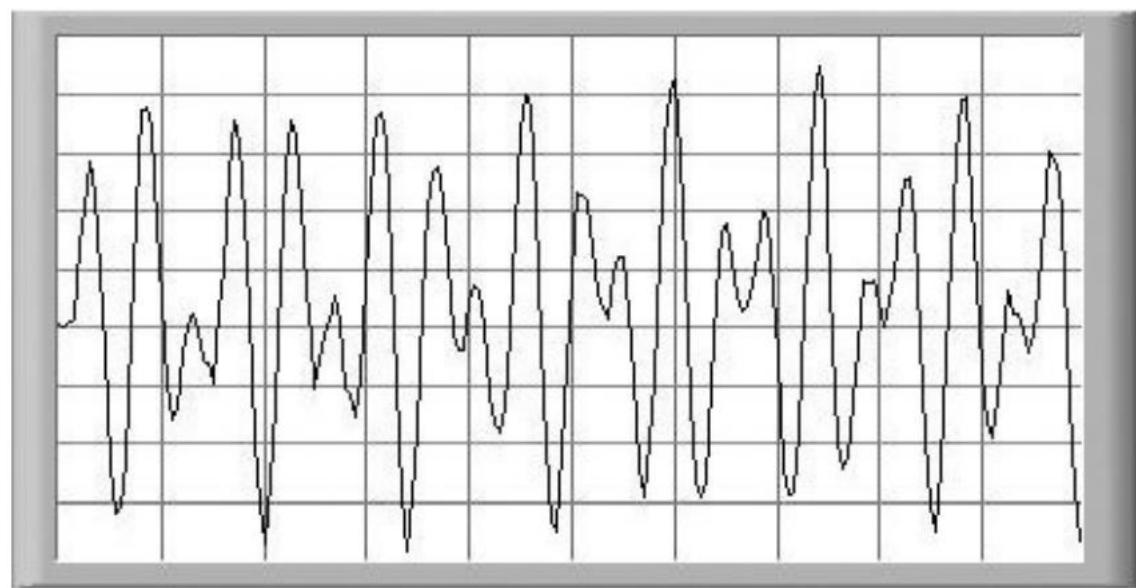


Hình 6-4 Chương trình thu thập dữ liệu

Xử lí dữ liệu vào.



Hình 6-5 Dữ liệu thô vào labVIEW

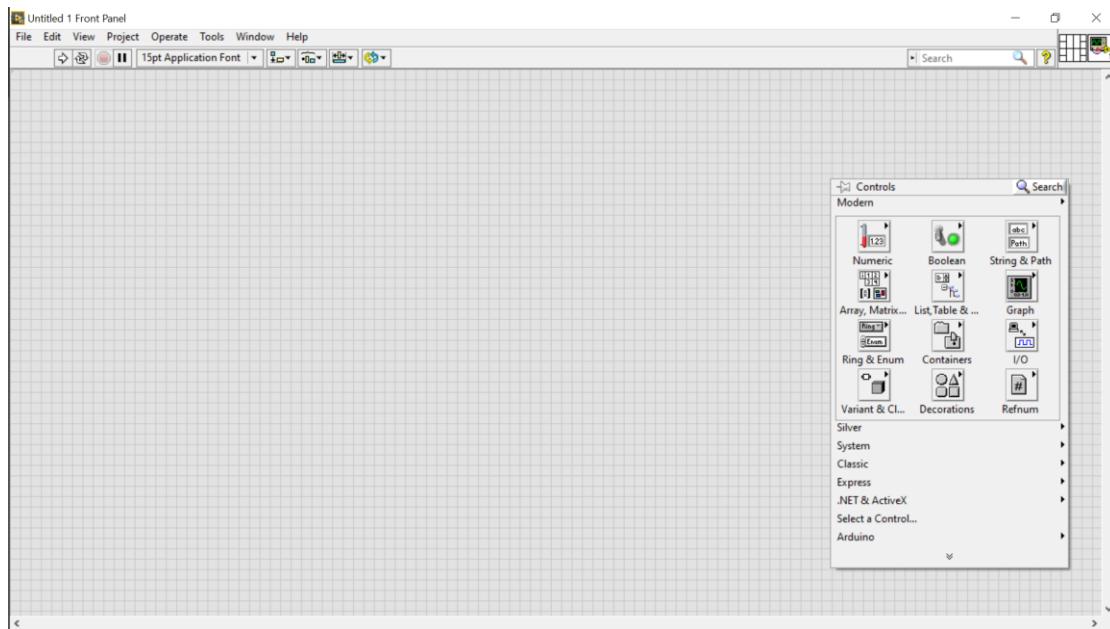


Hình 6-6 Dữ liệu qua xử lý  
**CÁC CỤM THÀNH PHẦN TRONG LABVIEW**

### 1. Front pannel

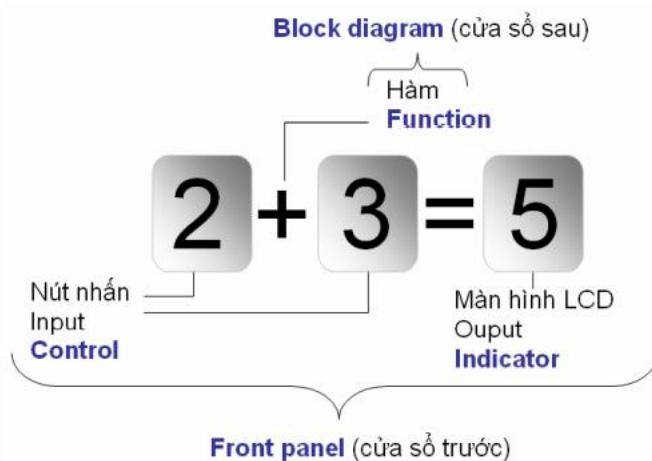
Hiểu 1 cách đơn giản thì front pannel (tấm nền trên) là cửa sổ có chức năng tương tác với người dùng. Front pannel cho phép người dùng nhập thông tin vào và xuất thông tin để giao tiếp với người dùng.

# Luận văn tốt nghiệp đại học



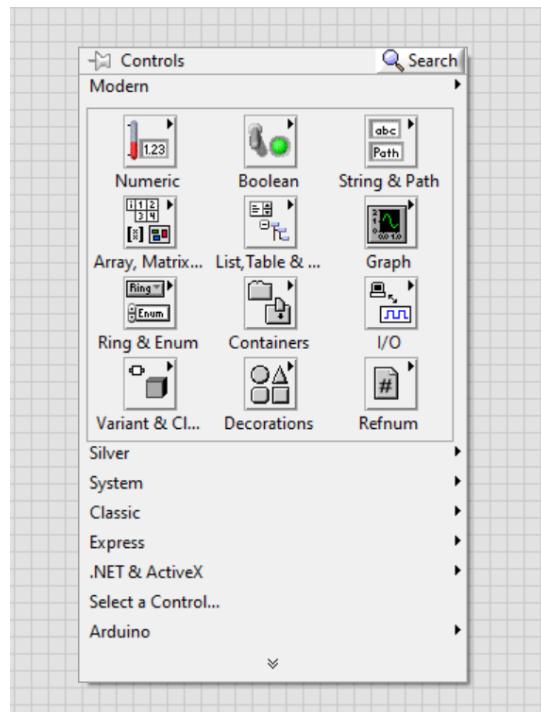
Hình 6-7 Giao diện Front Panel

## 1.1 Controls and Indicators (cụm chức năng điều khiển và hiển thị)



**Controls** có chức năng mô phỏng các công cụ vật lí thường dùng như là các nút vặn và công tắc chuyển đổi. Controls sử dụng kiểu giá trị đầu vào (Input) và các giá trị này cung cấp cho khối sơ đồ (block diagram).

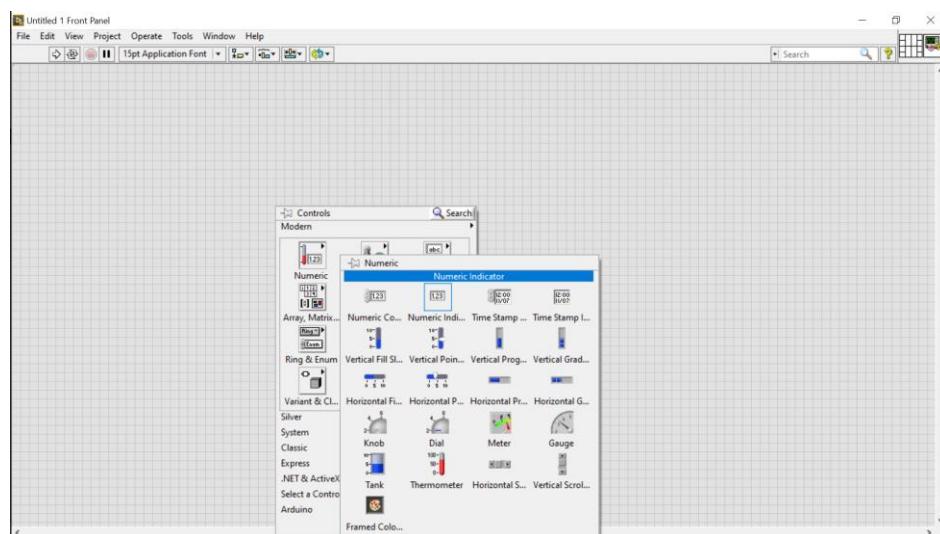
Controls = Thông tin nhập vào của người dùng = Dữ liệu nguồn.



Hình 6-8 Các cụm chức năng của Front Panel trong LabVIEW

**Indicators** có chức năng là hiển thị thông tin sau khi dữ liệu được xử lý bởi chương trình.

Indicators = thông tin xuất cho người dùng = đích đến hoặc ý nghĩa của dữ liệu.



Hình 6-9 Các chức năng con trong LabVIEW

## Luận văn tốt nghiệp đại học

	<b>Numeric control:</b> là control dạng số
	<b>Fill slide:</b> Control có dạng một thanh trượt
	<b>Pointer slide:</b> Control dạng thanh trượt có nút kéo
	<b>Knob:</b> Nút vặn
	<b>Dial:</b> Đĩa xoay
	<b>Constant:</b> hằng số, hay có thể xem là một dạng numeric control nhưng giá trị không thay đổi trong suốt quá trình chạy chương trình. Để tạo Constant ta lấy một Numeric control, chuột phải lên Numeric control chọn <b>Change to constant (Quy tắc vàng 1)</b> .
	<b>String control:</b> là 1 dòng chữ hay còn gọi là text control. Sử dụng text Control này để nhập các chữ hoặc chuỗi ký tự, hoặc một câu văn, xem thêm bài 4 – case structure.
	<b>Simulated signal:</b> một tín hiệu được mô phỏng sẵn trong LabVIEW có thể dùng như một Indicator trong một số trường hợp. Lấy Simulated signal bằng cách vào: BD> Express> Input> Simulated Signal. Xác lập các thông số khi bảng thông số hiện ra.  Nối đầu ra của khối vừa lấy với một Graph bằng cách chọn Right Click lên đầu ra, Create> Indicator.

*hình 6-10 bảng chức năng controls*

	<b>Numeric indicator:</b> là Indicator dạng số
	<b>Meter:</b> Indicator có dạng đồng hồ vuông
	<b>Gauge:</b> Indicator dạng đồng hồ vuông
	<b>Thermometer:</b> Cột nhiệt độ
	<b>Graduated Bar:</b> Thanh hiển thị quá trình
	<b>String:</b> là 1 dòng chữ hay còn gọi là text Control, dùng để xuất các chữ hoặc chuỗi ký tự, hoặc một câu văn, xem thêm bài 4 – Case structure. Lấy string indicator tại FP> Modern> String & Path> String Indicator.
	<b>Chart:</b> là biểu đồ hiển thị các giá trị theo trục thời gian
	<b>Graph:</b> là đồ thị thường được dùng để hiển thị các tín hiệu dạng sóng (waveform).
	<b>XY Graph:</b> đồ thị hiển thị quan hệ giữa hai tín hiệu X và Y hoặc dùng trong bài về đồ thị hàm số $y=f(x)$ – được trình bày trong bài 4.

Hình 6-11 bảng chức năng indicator

### 1.2 kiểu dữ liệu của Controls và Indicators

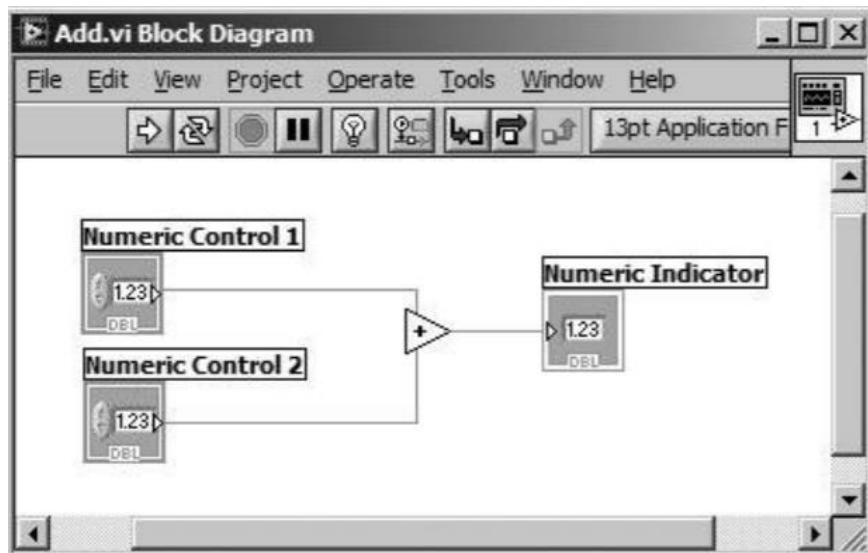
Là công cụ dùng để gán (ép) một numeric hay indicator vào một dãy giá trị nào đó.

Ký hiệu	Kiểu dữ liệu	Số bit	Khoảng giá trị
I8	Byte signed integer	8	-128 tới 127
I16	Word signed integer	16	-32,768 tới 32,767
I32	Long signed integer	32	-2,147,483,648 tới 2,147,483,647
I64	Quad signed integer	64	-1e19 tới 1e19
U8	Byte unsigned integer	8	0 tới 255
U16	Word unsigned integer	16	0 tới 65,535
U32	Long unsigned integer	32	0 tới 4,294,967,295

Hình 6-12 Một vài kiểu dữ liệu trong LabVIEW

## 2. Block diagram (sơ đồ khối)

Block diagram chứa các mã nguồn (dạng đồ họa). Block diagram cũng giống như những dòng ký tự lập trình trong các ngôn ngữ lập trình thông dụng như C+, Python,... Điểm khác biệt là ta xây dựng mã nguồn bằng việc nối các khối chứa từng chức năng riêng biệt lại với nhau bằng các “sợi dây”.



Hình 6-13 Giao diện lập trình ở Block Diagram

Lưu ý: khi chúng ta tạo 1 control hoặc indicator ở front pannel thì ở block diagram sẽ tự động tạo ra 1 phần tử tương ứng, và ta không thể xóa phần tử đó.

Phần tử tạo ra bởi Controls thì thông tin xuất phát phía bên phải (dây nối từ bên phải) và phần tử tạo bởi indicators thì thông tin đến từ phía bên trái (dây nối đến bên trái).



*Hình 6-14 Hướng di chuyển của dòng dữ liệu*

Các thư viện chức năng trong Block diagram bao gồm:

- Programming là nơi chứa hỗ trợ công cụ, hàm lập trình nói chung (giống các ngôn ngữ khác như C, Matlab, vv...).
- SignalExpress hỗ trợ thu thập dữ liệu, hiển thị tín hiệu trên máy tính.
- Advanced Signal Processing Toolkit hỗ trợ xử lý tín hiệu nâng cao.
- Control Design and Simulation hỗ trợ xây dựng các mô hình động lực học.
- Hệ thống và thiết kế bộ điều khiển (giống Matlab Simulink).
- Digital Filter Design Toolkit hỗ trợ thiết kế bộ lọc số.
- PID and Fuzzy Logic Toolkit hỗ trợ thiết kế bộ điều khiển.
- PID hoặc Fuzzy Logic FPGA hỗ trợ lập trình FPGA.
- Real-Time hỗ trợ lập trình ứng dụng thời gian thực.
- Internet Toolkit hỗ trợ giao tiếp qua mạng Internet.
- Database Connectivity Toolkit hỗ trợ kết nối cơ sở dữ liệu.
- Vision Development Module hỗ trợ công cụ phát triển hệ thống thu thập và xử lý ảnh.
- Simulation Interface Toolkit cho phép kết nối LabVIEW và Matlab Simulink.
- Vision and Motion hỗ trợ lập trình chuyển động nhiều bậc.

- LabVIEW Embedded Modudle là modun lập trình nhúng, dùng để lập trình vi điều khiển
- Express: Là nơi chứa các hàm thường dùng. Các hàm thường dùng là tập con của thư viện LabVIEW Programming.

### Wire (dây nối)

	Scalar	<a href="#">[View full size image]</a>	1D Array	2D Array	Color
Floating-Point Number	—	—	—	—	Orange
Integer Number	—	—	—	—	Blue
Boolean	—	—	—	—	Green
String	~~~~~	~~~~~	~~~~~	~~~~~	Pink
Cluster	~~~~~	~~~~~	~~~~~	~~~~~	Pink or Brown

*Hình 6-15 Phân biệt các loại dây nối*

### **3. Giao tiếp nối tiếp (serial communicate)**

Kết nối bằng công COM theo chuẩn RS232.

Mục đích là truyền và nhận dữ liệu.

Gồm 2 dây: 1 dây truyền (hoặc nhận) và 1 dây nối đất.

RS232 sử dụng phương thức truyền không đối xứng, tức là sử dụng tín hiệu điện áp chênh lệch giữa dây dẫn và dây mass.

Các cổng RS 232 có ngưỡng điện áp ước là -15V tới -3V và 3V tới 15V.

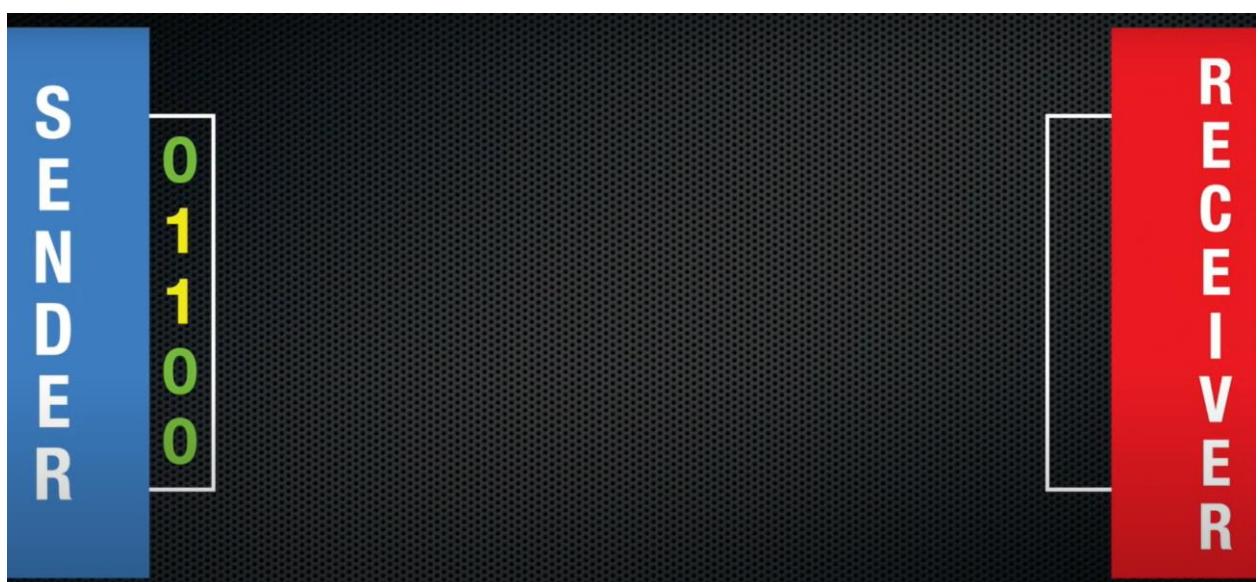
Tín hiệu điện áp lớn hơn 3V thì được coi là có logic 0 hoặc có giá trị cao (H).

Tín hiệu điện áp nhỏ hơn 3V thì được coi là có logic 1 hoặc có giá trị thấp (L).

Điện áp từ -3V đến 3V được coi là không có nghĩa.

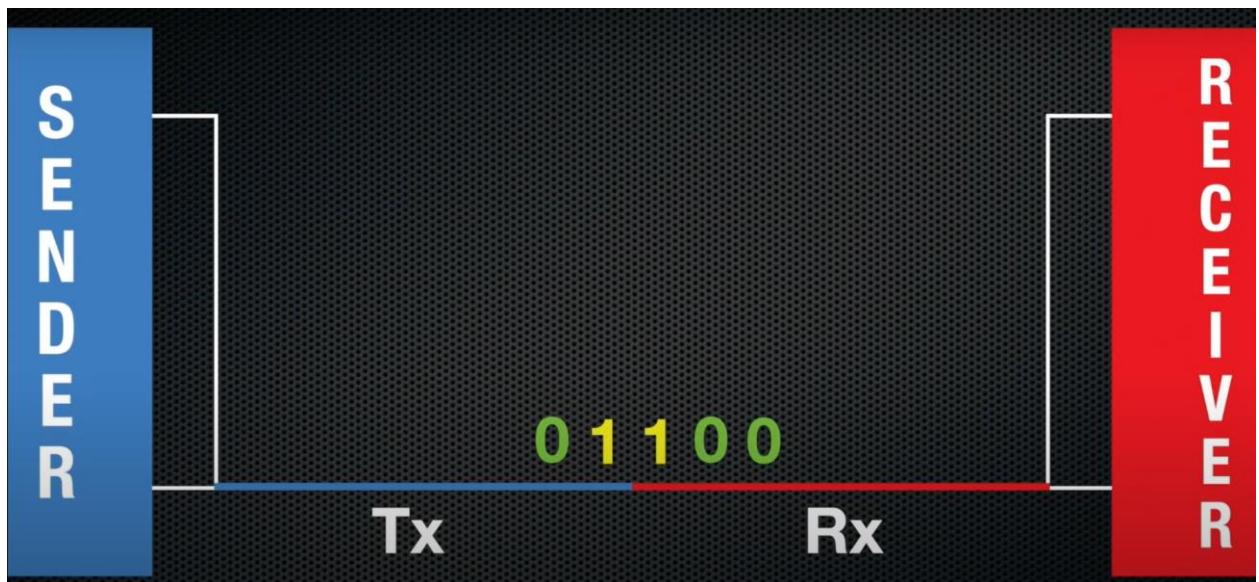
Để máy tính và labview hiểu nhau và phối hợp trơn tru thì ta phải có môi trường chung (hay còn gọi là phải có driver).

### **4. Cách truyền và nhận dữ liệu giữa arduino và labview**

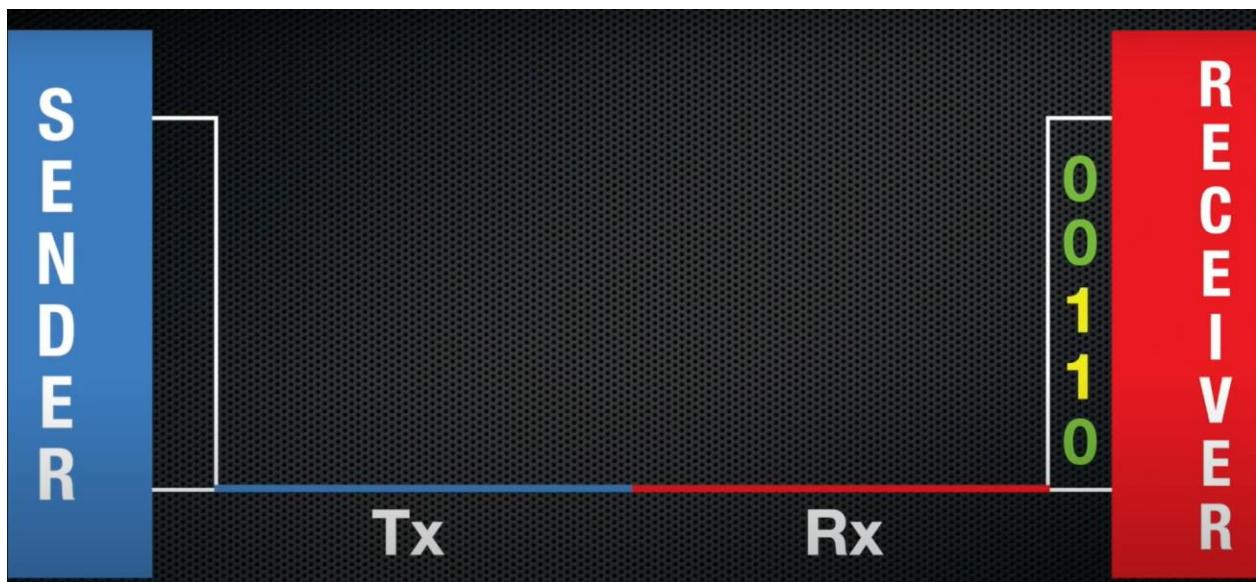


*Hình 6-16 Quá trình chuẩn bị trước khi truyền dữ liệu [1]*

Đầu tiên dữ liệu được tạo ra từ phía thiết bị truyền (send object), sau đó chúng được chứa trong buffer, ở đầu thiết bị nhận cũng có buffer. Thông tin được truyền qua lại giữa 2 thiết bị bởi đường dây. Dây ở phía thiết bị truyền gọi là the transmit data (TX wire) và ở thiết bị nhận là the received data (RX wire)



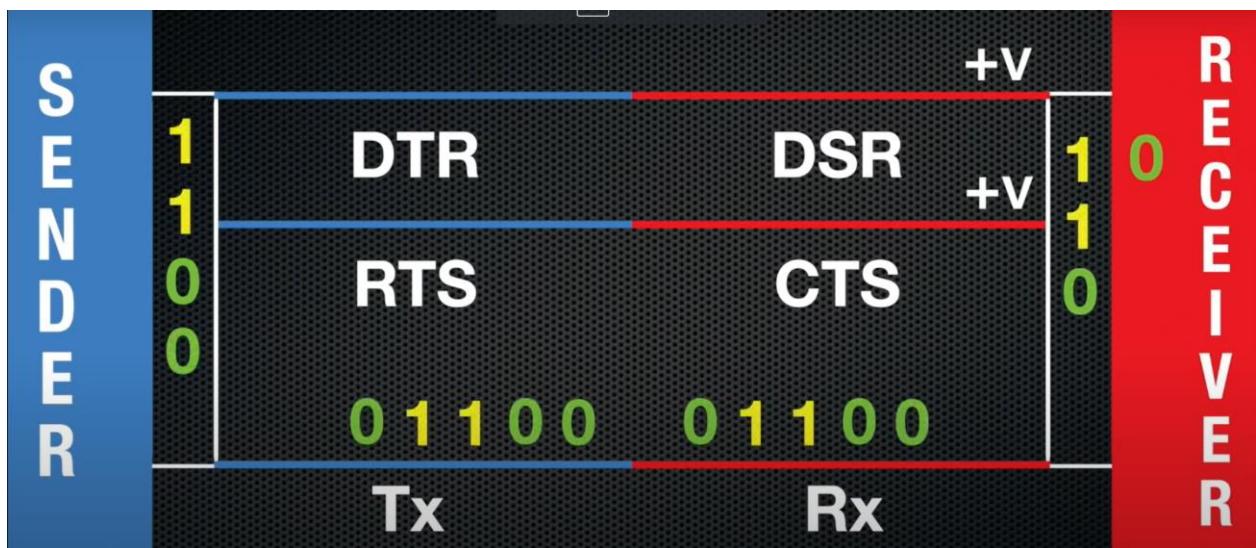
Hình 6-17 Quá trình truyền dữ liệu bằng giao thức Serial [1]



Hình 6-18 Quá trình tiếp nhận dữ liệu [1]

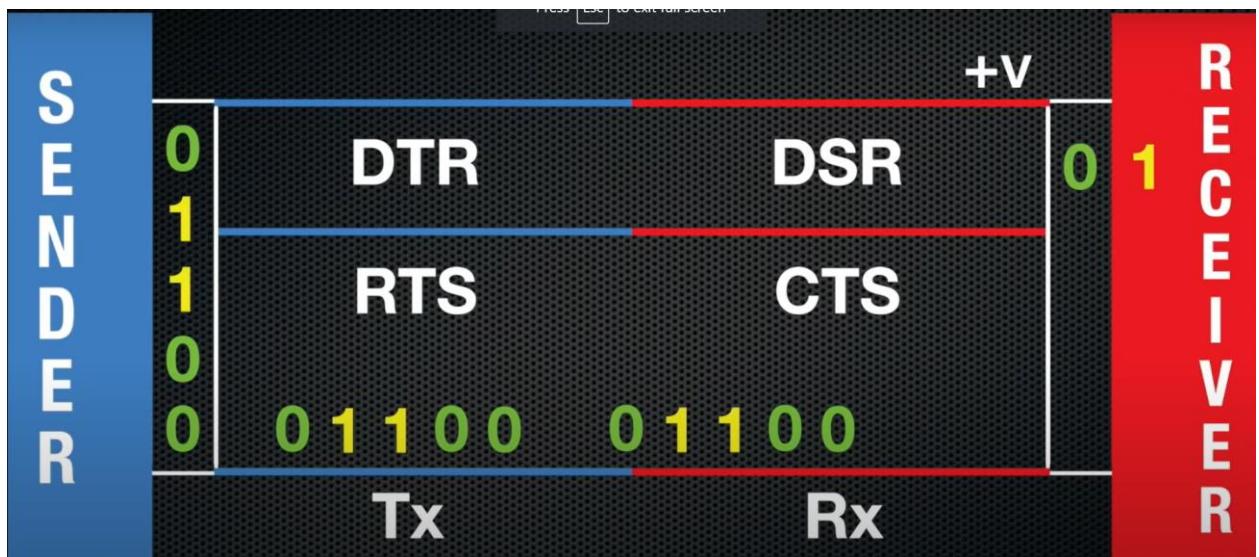
Để 2 thiết bị xác định được đầu bên kia đã sẵn sàng truyền (nhận) dữ liệu hay chưa thì giữa 2 thiết bị cần có thêm 1 đường dây nữa gọi là dây điều khiển. Phía thiết bị truyền thì đường dây được gọi là data terminal ready (DTR) và phía thiết bị nhận thì có tên là

data set ready (DSR). Khi cả 2 được kết nối thì 1 điện áp thích hợp được tạo ra bởi thiết bị nhận. Lúc này nhờ có sự chênh lệch điện áp mà thiết bị truyền nhận ra đầu kia và từ đó dữ liệu được truyền đi một cách an toàn. Tuy nhiên khi dữ liệu truyền quá nhanh và thiết bị nhận không kịp lấy dữ liệu ra từ buffer, buffer bị tràn (overflow) thì thông tin còn lại sẽ bị mất đi. Vì vậy giữa 2 thiết bị cần có thêm 1 đường dây nữa gọi là dây điều khiển để từ đó thiết bị nhận có thể đọc kịp khi thông tin được gửi từ phía gửi. Phía gửi gọi là ready to send (RTS) và phía nhận gọi là clear to send (CTS). Với đường dây này, phía nhận có thể báo cho phía gửi rằng khi nào nên tạm dừng gửi thông tin, từ đó việc buffer bị tràn không còn xảy ra.

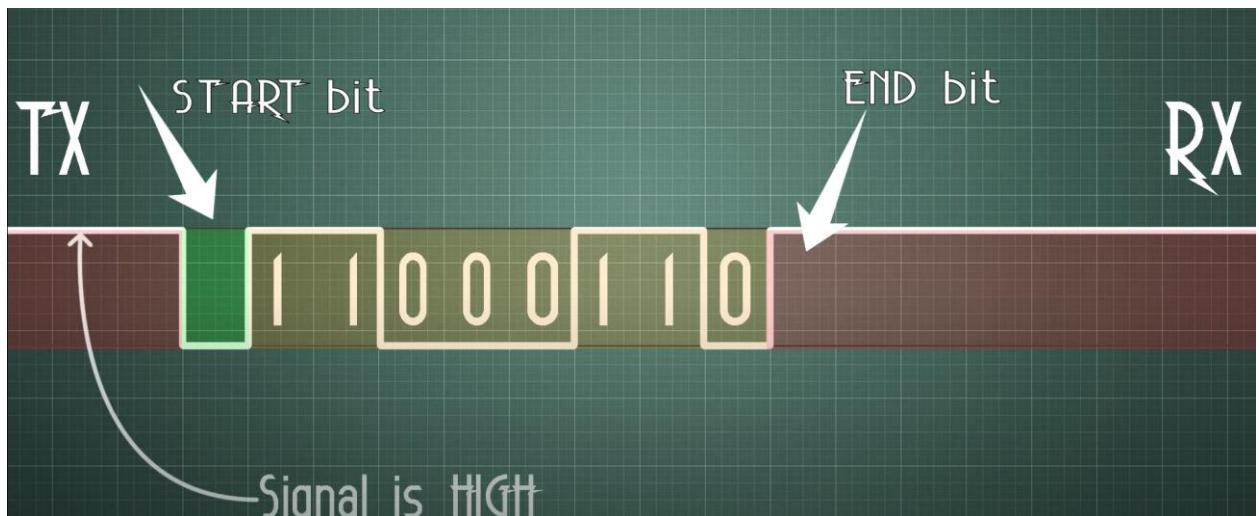


Hình 6-19 Cơ chế kiểm tra tránh tràn buffer trong quá trình giao tiếp [1]

Lúc đó phía nhận sẽ ngắn điện áp ở dây điều khiển để báo cho phía gửi tạm dừng việc gửi dữ liệu lại.



Hình 6-20 Quá trình tiếp nhận và xử lý dữ liệu [1]



Hình 6-21 Cấu trúc của đoạn dữ liệu trong quá trình giao tiếp [1]

Tuy nhiên lại xuất hiện 1 vấn đề nữa là thiết bị nhận sẽ không thể xác định được khi nào 1-bit dữ liệu kết thúc và bit tiếp theo sẽ bắt đầu, từ đó thì sẽ xác định sai gói dữ liệu. Vì vậy ta cần xác định tốc độ truyền dữ liệu.

Bit rate (tốc độ truyền 1-bit dữ liệu trong 1s).

Baud rate (tốc độ truyền 1 gói tín hiệu trong 1s).

Giả sử baud rate được thiết lập là 9600 => thời gian truyền 1-bit là  $1/9600 = 104$  microsecond. Đầu tiên ta đọc bit bắt đầu là 104pms. Tại đây ta đến điểm bắt đầu của gói dữ liệu. Sau đó ta mất 52pms để đến bit đầu tiên, tại đây bit đầu được lưu vào trong buffer và sao đó mất 104pms đến bit thứ 2, lưu bit vào trong buffer và cứ tiếp tục như vậy cho đến bit cuối cùng. Sau đó khi gặp bit kết thúc gói dữ liệu thì phía nhận sẽ tự động reset buffer và bắt đầu nhận tín hiệu mới.

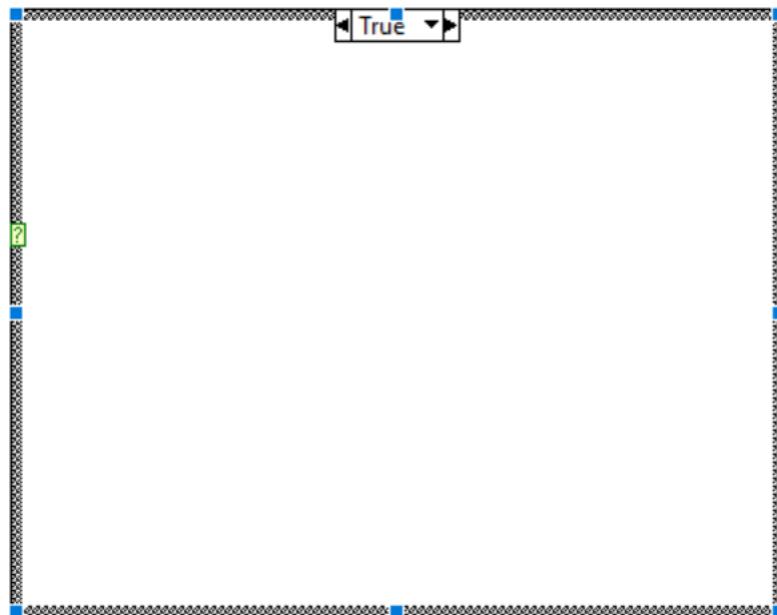
### Các hàm chức năng trong labview phục vụ đồ án

**Vòng lặp while loop:** thu thập thông tin liên tục theo thời gian thực nên chương trình phải thực hiện vòng lặp cho đến khi người dùng dừng chương trình thì thôi.



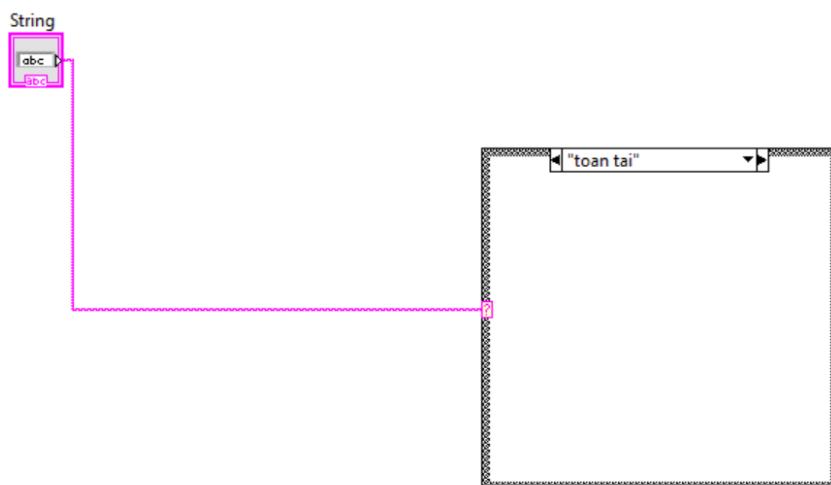
Hình 6-22 Giao diện chức năng Case Structure

**Case structure:** dùng để chọn chế độ vận hành. Ở mỗi chế độ vận hành sẽ thực hiện 1 chương trình với các hệ số điều khiển sao cho phù hợp



Hình 6-23 Giao diện chức năng Event Structure

**String control:** dùng để chọn chế độ (điều kiện để kích hoạt casestructure)



Hình 6-24 Case Structure có thể điều khiển bởi nhiều kiểu dữ liệu khác nhau

**Delay:** điều chỉnh tốc độ truyền và nhận giữa arduino và labview sao cho cả 2 không bị tràn dữ liệu (overflow) và để bên thiết bị nhận đọc kịp (không bị mất dữ liệu)

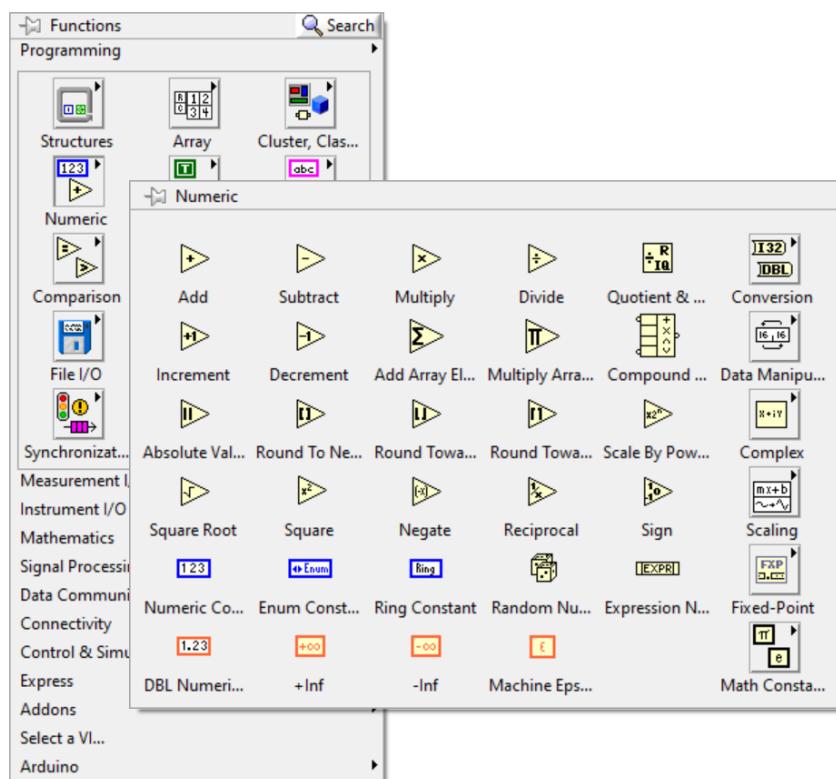


Hình 6-25 Chức năng Delay

**Decimal string to number:** Thay đổi dữ liệu từ Char thành số đếm (Number) mà vẫn giữ nguyên nội dung hiển thị.



Hình 6-26 Chức năng Chuyển đổi kí tự thành số



Hình 6-27 Giao diện các phép toán trong LabVIEW

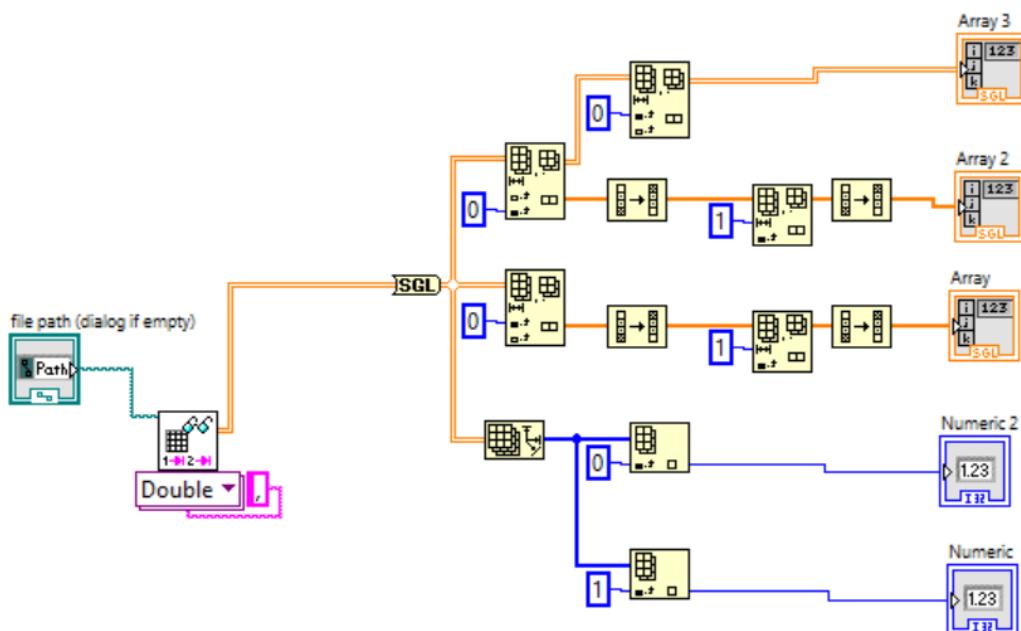
**Numeric control:** điều chỉnh hệ số của các giá trị thông số nhằm điều khiển động

cơ

## 6.2 Tính năng điều chỉnh thông số làm việc

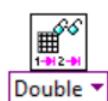
### 6.2.1 Khởi tạo chương trình điều khiển

Thông tin của các bảng điều khiển được lưu trữ trong ổ cứng, khi khởi động chương trình, các dữ liệu ấy được tải lên giao diện, Tùy theo lệnh điều khiển mà chương trình sẽ quyết định xem bảng điều khiển nào được phép hiển thị lên màn hình để người dùng điều chỉnh.



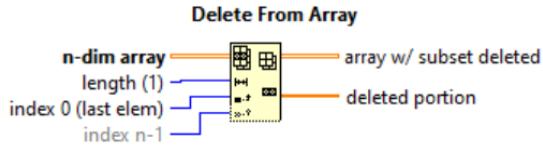
Hình 6-28 Khởi lập trình cập nhật dữ liệu từ ổ cứng vào các bảng giao tiếp

**Read Delimited SpreadSheet:** đọc dữ liệu bảng tính dưới dạng (.csv) với kiểu dữ liệu đầu ra là Array (64-bit Double Precision Float).



Hình 6-29 Chức năng trích xuất dữ liệu từ file Excel

**Delete From Array:** Cắt Array để tạo thành một Array mới có kích thước nhỏ hơn.



Hình 6-30 Chức năng xóa phần tử trong bảng

Bảng trích xuất từ ô cứng là bảng 2 chiều gồm 3 phần: Bảng 1 chiều của biến 1;

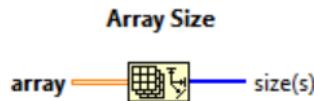
Bảng 1 chiều của biến 2.

Bảng 2 chiều chứa nội dung phụ thuộc vào 2 biến trên.

Các kiểu dữ liệu và phương pháp chuẩn hóa mỗi thành phần là khác nhau.

Vì vậy khi cập nhật lên giao diện phải tiến hành tách bảng thành 3 bảng khác nhau.

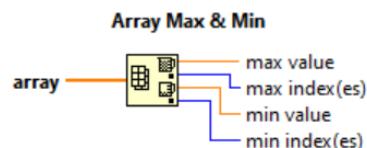
**Size Array:** Xác định kích thước của mảng đầu vào.



Hình 6-31 Chức năng tìm kích thước của bảng

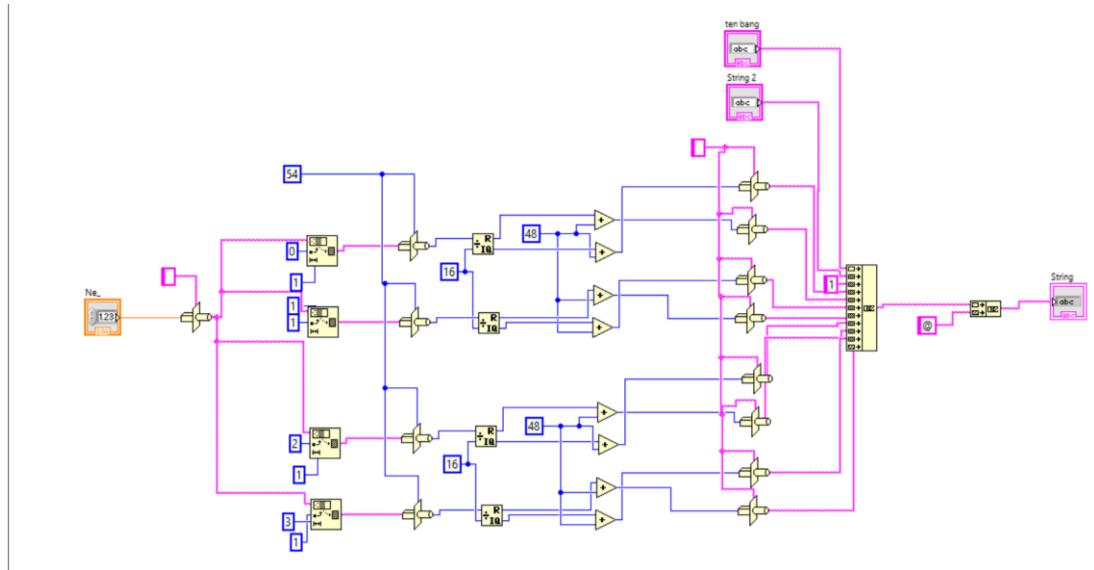
### 6.2.2 Chuẩn hóa ma trận

**Array Max & Min:** Trích xuất giá trị max, min của bảng.



Hình 6-32 Chức năng tìm giá trị lớn nhất/nhỏ nhất của bảng

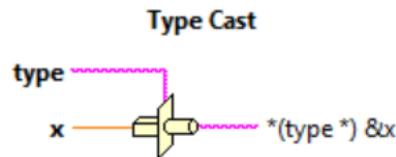
Sau khi có được hệ số chuẩn hóa, tiến hành chuyển đổi sang dữ liệu giao tiếp



Hình 6-33 Thuật toán chuyển đổi giá trị hệ số chuẩn hóa thành dữ liệu giao tiếp

**Type Cast:** Thay đổi kiểu hiển thị của biến mà không làm thay đổi giá trị.

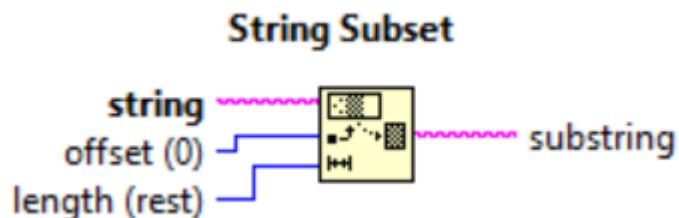
VĐ: 1 số có kiểu dữ liệu Float (32-bit) thì sau khi qua khối TypeCast sẽ hiển thị 4 Char (8-bit) có giá trị từ 0-127.



Hình 6-34 Chức năng thay đổi kiểu hiển thi

Sau khi có được dãy kí tự, tiến hành tách từng kí tự ra khỏi chuỗi.

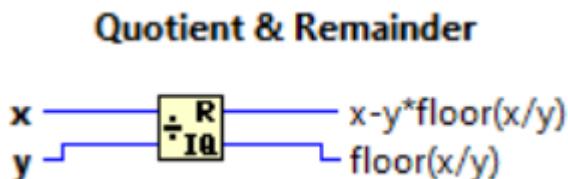
**String Subset:** Tách kí tự/chuỗi con ra khỏi chuỗi ban đầu



Hình 6-35 Chức năng lấy kí tự ra khỏi chuỗi

Để tách 1 Byte (4-bit cao và 4-bit thấp) thành 2 byte, trước tiên ta sẽ tách thành 4-bit cao và 4-bit thấp, sau đó tạo 1 byte có giá trị bằng 4-bit cao và 1 byte có giá trị bằng 4-bit thấp. Vậy là ta đã chuyển đổi được giá trị từ 1 byte thành 2 byte.

**Quotient & Remainder:** Chia lấy phần nguyên và phần dư

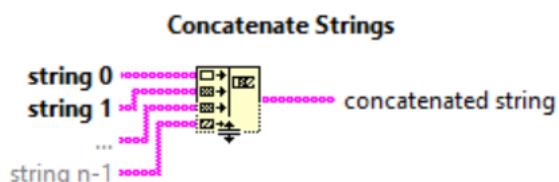


Hình 6-36 Chức năng chia lấy phần nguyên và phần dư

Phần nguyên ( $x/y$ ) là giá trị 4-bit cao và phần dư là giá trị 4-bit thấp.

Sau khi đã tách được đủ các kí tự đã tính, ta sẽ tiến hành ghép các kí tự lại theo thứ tự.

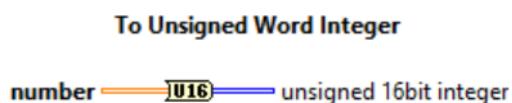
**Concatenate Strings:** Ghép các kí tự lại với nhau.



Hình 6-37 Chức năng ghép các kí tự

Sau khi có được hệ số chuẩn hóa, ta tìm được bảng chuẩn hóa, tuy nhiên bảng chuẩn hóa có giá trị là các số nguyên trong khi kiểu dữ liệu của bảng lại là số Float (32-bit), tức là giá trị hiển thị có kiểu dữ liệu nhỏ hơn so với giá trị của bảng, cần phải tối ưu.

To Unsigned Word Integer: Chuyển đổi kiểu dữ liệu

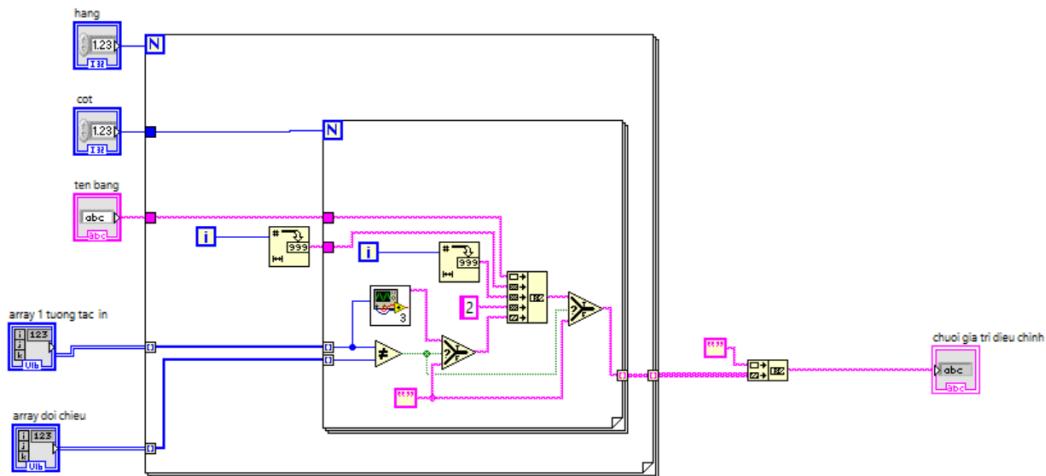


Hình 6-38 Khối chức năng chuyển đổi dữ liệu sang Unsigned Integer.

### 6.2.3 Xác định phần tử mà người dùng điều chỉnh

## Luận văn tốt nghiệp đại học

Sau khi chuẩn hóa xong, tiến hành so sánh 2 bảng chuẩn hóa để tìm giá trị khác nhau.



Hình 6-39 Khối chức năng tìm phần tử mà người dùng thay đổi

Kích thước của 2 bảng chuẩn hóa luôn bằng nhau, trích xuất lần đồng thời lượt từng phần tử trong cả 2 bảng rồi so sánh với nhau. Khi phát hiện 2 số liệu khác nhau sẽ cập nhật giá trị của bảng chuẩn hóa giao tiếp và chuyển đổi về chuỗi kí tự.

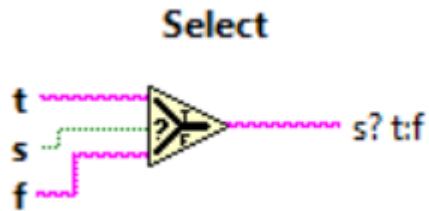
**Not Equal:** So sánh hai phần tử đầu vào có khác nhau không. Nếu khác nhau sẽ xuất ra kết quả True và ngược lại.

**Not Equal?**



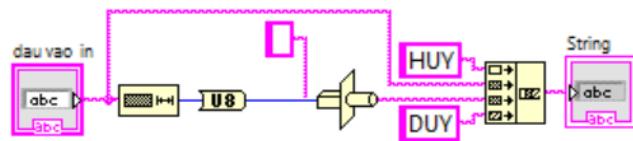
Hình 6-40 Chức năng so sánh

Select: Lựa chọn trường hợp để thực hiện, nếu tín hiệu là True thì thi hành phần tử đầu vào **t**, nếu tín hiệu là False thì thi hành phần tử đầu vào là **f**.



Hình 6-41 Chức năng Lựa chọn

Sau khi có được giá trị chuyển đổi của các phần tử cần tìm, tiến hành ghép các phần tử lại thành trường dữ liệu. Sau đó tạo kí tự Checksum để chương trình nhận có thể kiểm tra tính chính xác. Sau đó tạo thành chuỗi kí tự hoàn chỉnh.



Hình 6-42 Khởi chạy chức năng đóng gói đơn vị hoàn chỉnh

**String Length:** Đếm số kí tự trong 1 chuỗi. Kết quả đầu ra là số nguyên.

### String Length

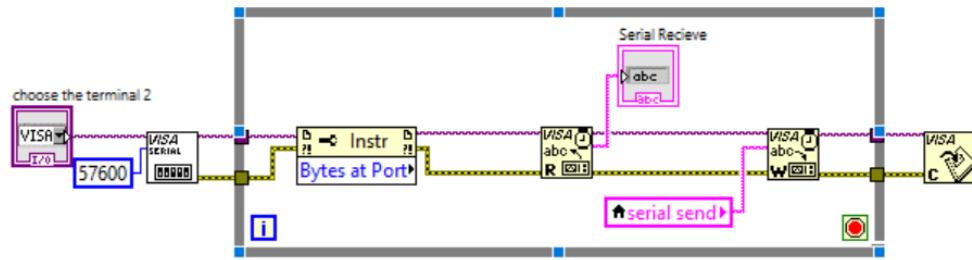


Hình 6-43 Chức năng đếm số kí tự của chuỗi.

## 6.3 Tính năng hiển thị thông số vận hành.

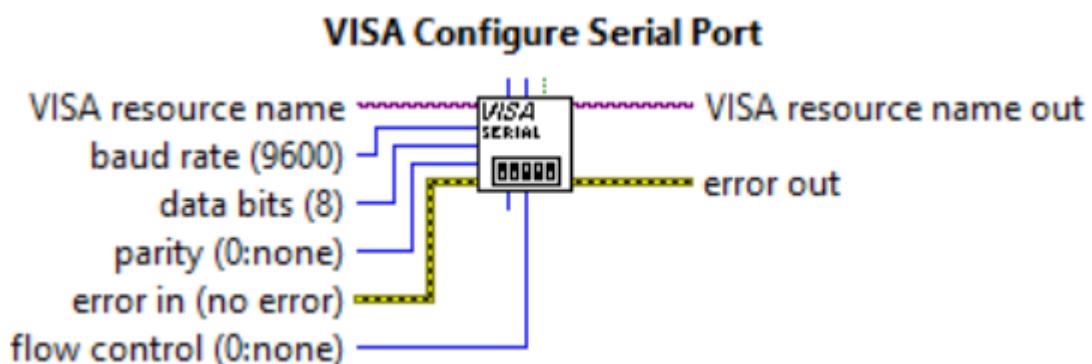
### 6.3.1 Khởi tạo chương trình

Thiết lập trạng thái kết nối với LabVIEW.



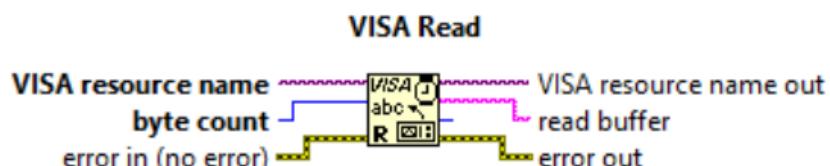
Hình 6-44 Chương trình giao tiếp với Arduino

**VISA Configure Serial Port:** thiết đặt thông số kết nối Serial.



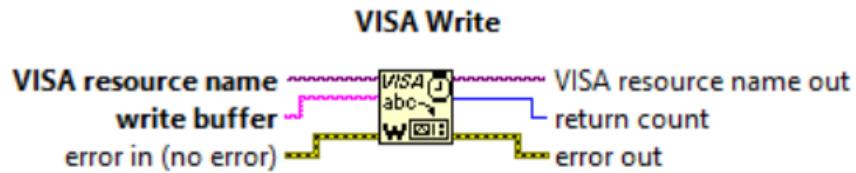
Hình 6-45 Khối chức năng thiết đặt thông số kết nối Serial

**VISA Read:** Đọc dữ liệu từ Arduino.



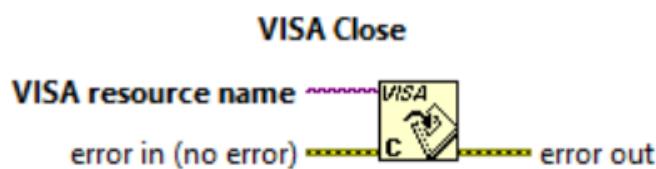
Hình 6-46 Khối chức năng đọc dữ liệu từ Arduino

**VISA Write:** Gửi dữ liệu xuống Arduino.



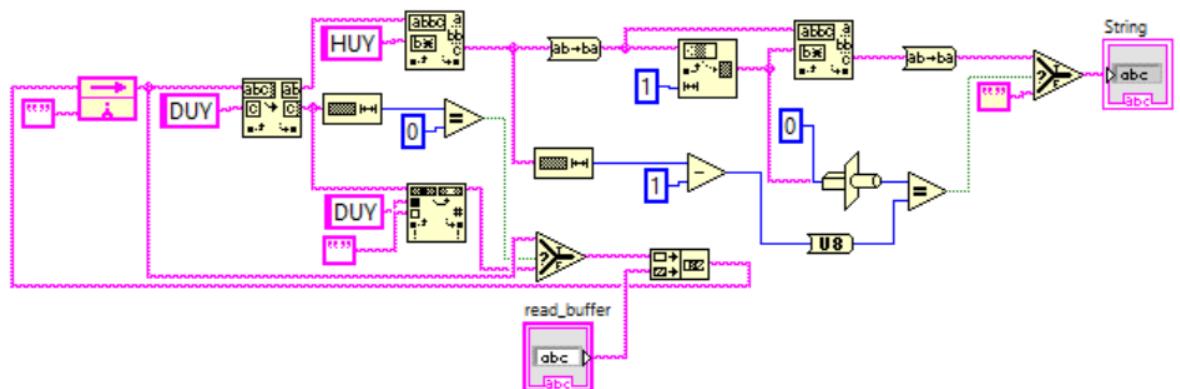
Hình 6-47 Khối chức năng gửi dữ liệu xuống Arduino

**VISA Close:** Kết thúc chức năng giao tiếp.



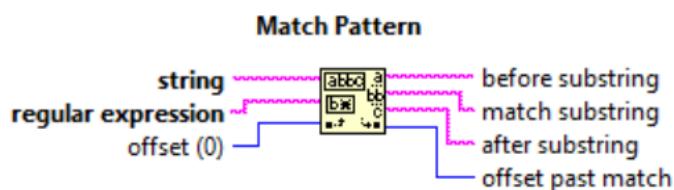
Hình 6-48 Khối chức năng kết thúc giao tiếp

### 6.3.2 Lọc tách dữ liệu thành từng đơn vị giao tiếp hoàn chỉnh



Hình 6-49 Khối chức năng lọc tách từng đơn vị giao tiếp hoàn chỉnh

**Match Pattern:** tìm kí tự và tách chuỗi thành 3 đoạn thành phần.



Hình 6-50 Chức năng tìm kí tự và tách chuỗi

**Reverse String:** Nghịch đảo chuỗi.

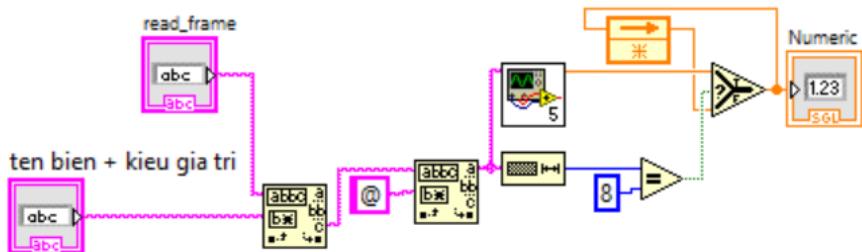
### Reverse String

string  reversed

Hình 6-51 Nghịch đảo chuỗi

#### 6.3.3 Giữ giá trị khi không có dữ liệu gửi lên và tách lấy đoạn chứa giá trị

Vì tàn số gửi dữ liệu và tốc độ xử lí của LabVIEW là khác nhau nên khi trường hợp LabVIEW nhận và xử lí thông tin hoàn thành nhưng dữ liệu Arduino chưa gửi lên, nếu LabVIEW tiếp tục cập nhật thì giá trị của đối tượng sẽ không ổn định và dịch chuyển liên tục. Vì vậy cần phải có chức năng giữ giá trị cũ khi không có dữ liệu.

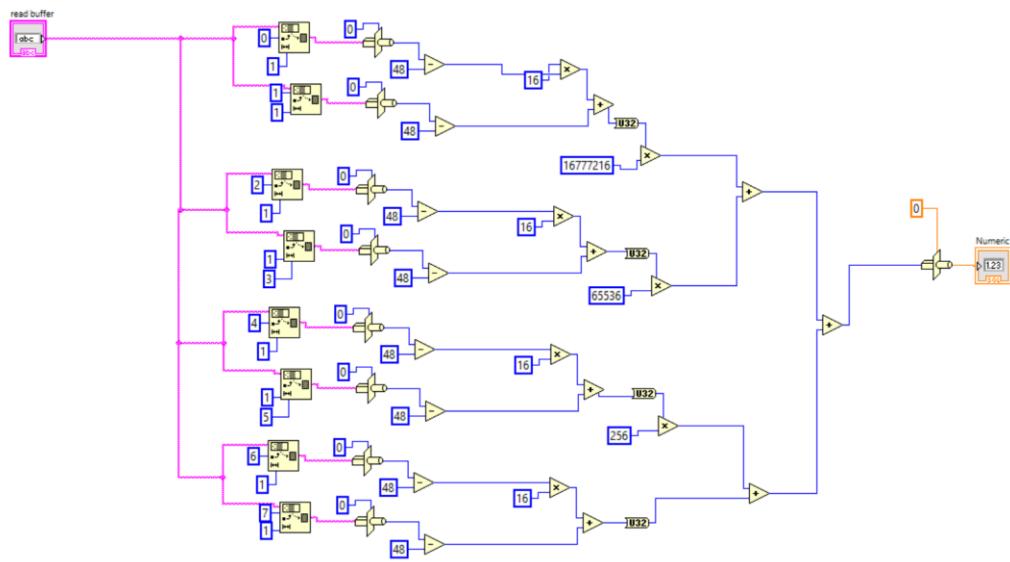


Hình 6-52 Khối chức năng giữ giá trị

Sau khi tách lấy phần kí tự chứa giá trị của biến, vì đều từ float tách thành nên luôn chứa 8 kí tự. Kiểm tra độ dài của đoạn, nếu số kí tự là 8 thỏa điều kiện thì cho phép đoạn xuất lên màn hình. Còn nếu không đúng thì lấy giá trị cũ của trường hợp đúng trước đó để hiển thị.

#### 6.3.4 Chuyển đổi chuỗi kí tự thành số liệu hiển thị

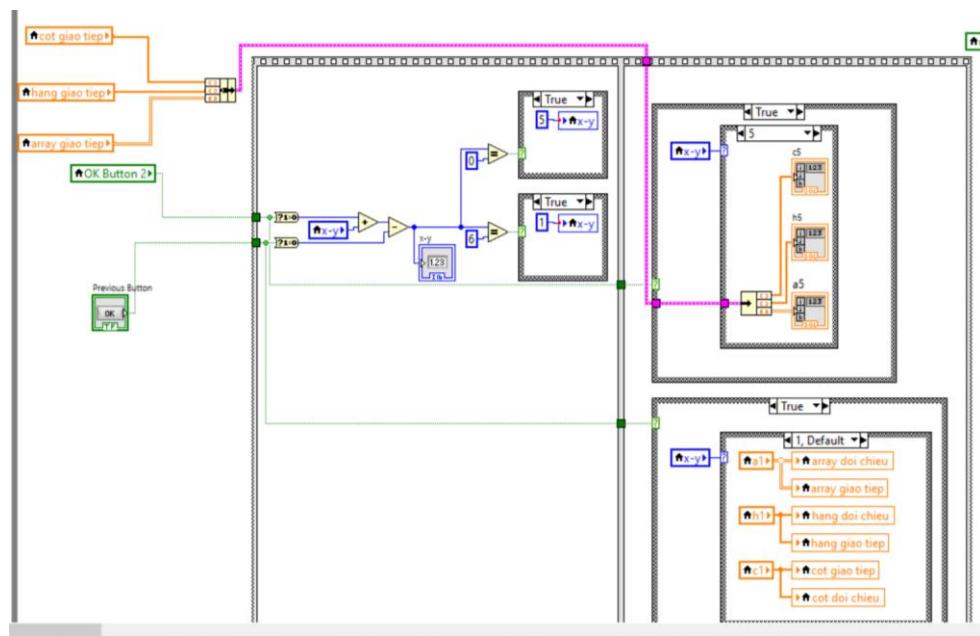
Chuỗi sẽ được tách ra thành từng kí tự, sau đó chuyển đổi kiểu hiển thị sang dạng số nguyên. Vì giá trị 0 là kí tự Null là kí tự kết thúc nên sẽ bị mất dữ liệu ghi gửi nên chương trình gửi đã +48 vào, vì thế cần phải -48 vào các giá trị rồi ghép 2 số liệu liền kề thành 1 byte, ghép các kí tự lại với nhau ta sẽ được số liệu cần hiển thị.



Hình 6-53 Các tính năng hỗ trợ người dùng trong quá trình giao tiếp

### 6.3.5 Tính năng hoàn tác thao tác người dùng

Khi người dùng nhập sai và gửi dữ liệu đi, khi muốn hoàn tác lại số liệu trước đó chương trình sẽ lấy giá trị cũ lưu trong buffer cập nhật vào buffer giao tiếp. Khi có tín hiệu SEND khởi động năng lực sẽ tiến hành lưu thông số vào các buffer backup.



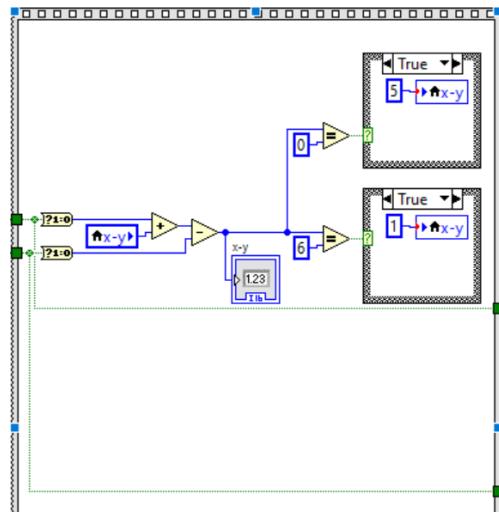
Hình 6-54 Sơ đồ chức năng Hoàn tác thao tác của người dùng

Để thực hiện chức năng trên, chương trình bao gồm 3 khối chức năng bên trong bao gồm:

Khối chức năng Xác định vị trí nhằm lựa chọn chính xác buffer backup để cập nhật sao cho theo ý người dùng.

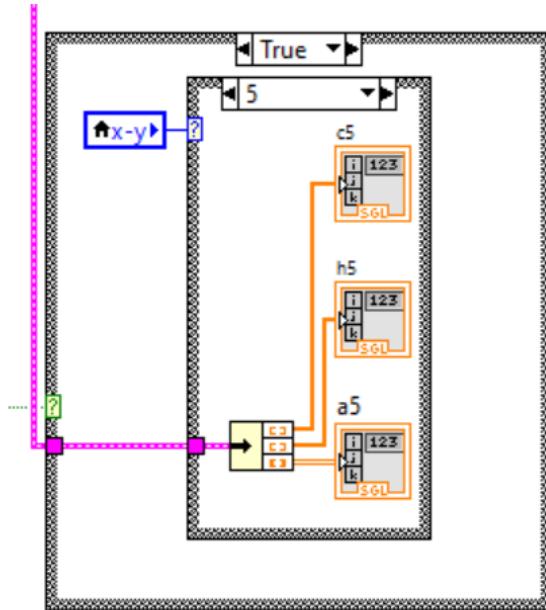
Khối chức năng gán giá trị của Buffer giao tiếp vào Buffer backup.

Khối chức năng cập nhật Buffer backup vào Buffer giao tiếp.



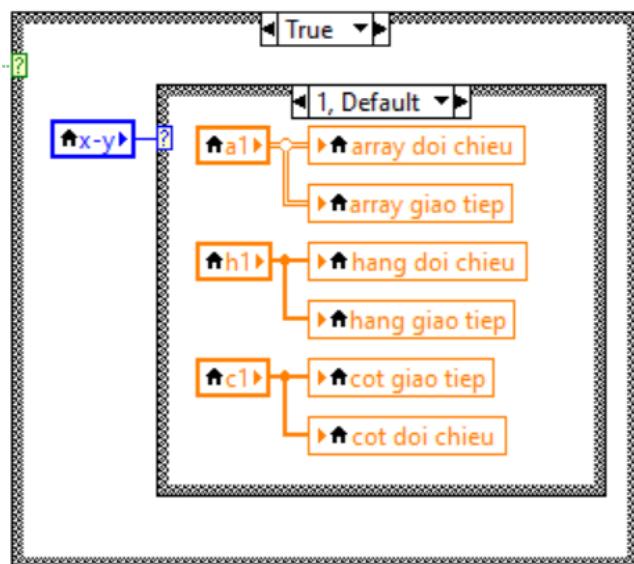
Hình 6-55 Khối chức năng xác định vị trí

Cứ mỗi tín hiệu điều khiển sẽ được chuyển đổi về giá trị 0-1, nếu là tín hiệu SEND thì vị trí của chương trình +1, nếu là tín hiệu UNDO thì tín hiệu chương trình -1. Khi chương trình đạt giá trị Max, khi có tín hiệu SEND thì sẽ gán giá trị vị trí về 1, còn khi giá trị vị trí là 1, nếu có tín hiệu UNDO thì gán giá trị vị trí về Max.



Hình 6-56 Khối chức năng cập nhật dữ liệu vào buffer Backup

Khi có tín hiệu SEND, chương trình sẽ xác nhận giá trị vị trí rồi cập nhật dữ liệu của Buffer giao tiếp vào Buffer Backup theo trường hợp cụ thể.



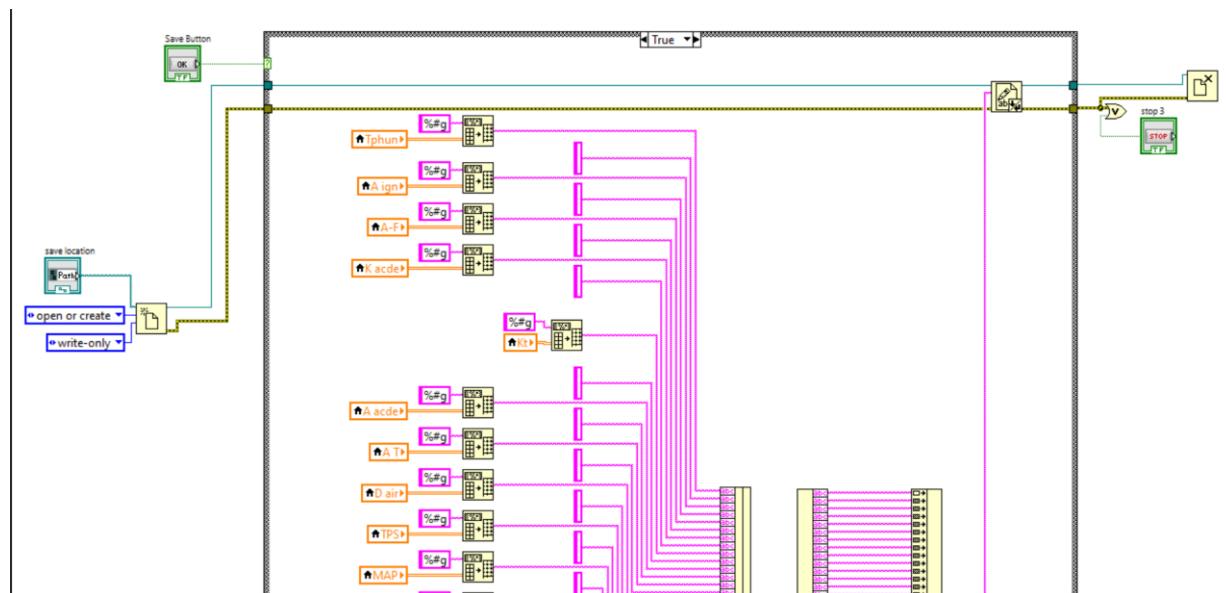
Hình 6-57 Khối chức năng sao lưu dữ liệu vào buffer giao tiếp

Khi có tín hiệu UNDO, chương trình sẽ xác nhận giá trị vị trí, sau đó xác định đúng buffer Backup và cập nhật giá trị vào buffer giao tiếp.

### 6.3.6 Tính năng lưu dữ liệu vào ổ cứng

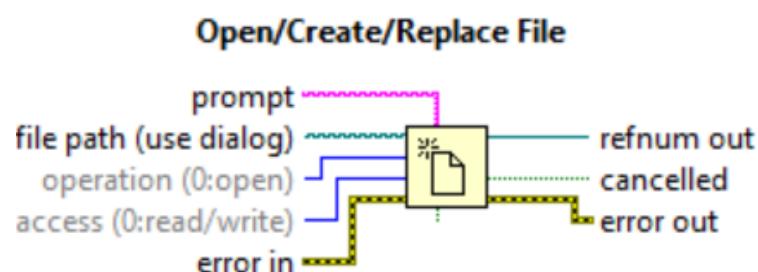
## Luận văn tốt nghiệp đại học

Để có thể kiểm tra xem việc giao tiếp có chính xác hay không, ta cần phải đổi chiều dữ liệu gửi xuống của LabVIEW với dữ liệu đọc được của Arduino nên cần phải có tính năng lưu trữ dữ liệu.



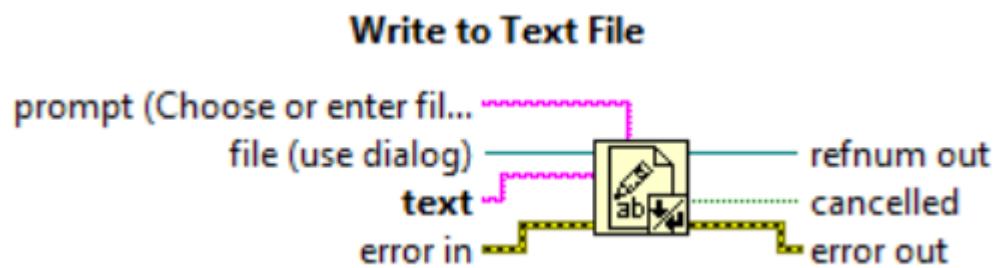
Hình 6-58 Sơ đồ chức năng lưu trữ dữ liệu vào ổ cứng

**Open/Create/Replace File:** khôi phục chức năng xác định vị trí lưu trữ dữ liệu



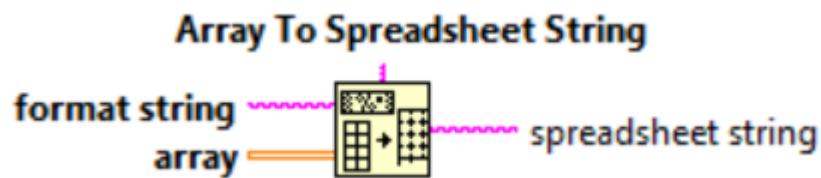
Hình 6-59 Chức năng xác định vị trí lưu trữ dữ liệu.

Write to Text File: Tải dữ liệu từ ổ cứng lên chương trình.



Hình 6-60 Chức năng tải dữ liệu từ ổ cứng lên chương trình

Array to Spreadsheet String: chuyển đổi kiểu số liệu của bảng tính thành kí tự.



Hình 6-61 Chức năng chuyển đổi kiểu số liệu của bảng tính thành kiểu kí tự

## **TÀI LIỆU THAM KHẢO**

- [1] vi.wikipedia.org.
- [2] Jeffrey Travis, Jim Kring (2006). *LabVIEW for Everyone: Graphical Programming Made Easy and Fun, Third Edition*. Prentice Hall.
- [3] Bài giảng kỹ thuật điện điện tử. Đại học Bách Khoa Tp Hồ Chí Minh – Khoa Điện Điện Tử – Phòng Thí Nghiệm Máy Điện và Thực Tập Điện – 2009.
- [4] Văn Thị Bông – Huỳnh Thanh Công (2005). Lý thuyết động cơ đót trong. Nhà Xuất Bản Đại học Quốc gia Tp Hồ Chí Minh.
- [5] LM7805 datasheet [7] PC817 datasheet.
- [6] BT151 datasheet.
- [7] IR2184 datasheet.
- [8] IRF540 datasheet.
- [9] FT232R USB UART IC Datasheet.
- [10] STM8S Reference manual.
- [11] STM8S207xx STM8S208xx datasheet.