HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

Phạm Khương Duy

XÂY DỰNG HỆ THỐNG ĐIỀU KHIỂN VÀ THU NHẬN DỮ LIỆU CHO ROBOT DỊCH VỤ

LUẬN VĂN THẠC SĨ KỸ THUẬT

(Theo định hướng ứng dụng)

TP.HỒ CHÍ MINH - NĂM 2022

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

Phạm Khương Duy

XÂY DỰNG HỆ THỐNG ĐIỀU KHIỂN VÀ THU NHẬN DỮ LIỆU CHO ROBOT DỊCH VỤ

CHUYÊN NGÀNH: HỆ THỐNG THÔNG TIN

MÃ SỐ: 8.48.01.04

LUẬN VĂN THẠC SĨ KỸ THUẬT

(Theo định hướng ứng dụng)

NGƯỜI HƯỚNG DẪN KHOA HỌC

TS. CHUNG TẤN LÂM

TP.HỒ CHÍ MINH - NĂM 2022

LỜI CAM ĐOAN

Tôi cam đoan rằng luận văn **"Xây dựng hệ thống điều khiển và thu nhận dữ liệu cho robot dịch vụ"** là công trình nghiên cứu của chính tôi.

Tôi cam đoan các số liệu, kết quả nêu trong luận văn là trung thực và chưa từng được ai công bố trong bất kỳ công trình nào khác.

Không có sản phẩm/nghiên cứu nào của người khác được sử dụng trong luận văn này mà không được trích dẫn theo đúng quy định.

TP. Hồ Chí Minh, ngày 04 tháng 5 năm 2022

Học viên thực hiện luận văn

Phạm Khương Duy

LỜI CẢM ƠN

Trong quá trình học tập và thực hiện luận văn, tôi đã nhận được sự quan tâm quý báu và hướng dẫn nhiệt tình của quý Thầy Cô, cùng với sự động viên và ủng hộ của gia đình, bạn bè và đồng nghiệp.

Với lòng kính trọng và biết ơn, tôi xin gửi lời cảm ơn chân thành đến: Ban Giám Đốc, Phòng Đào tạo Sau đại học của Học viện Công Nghệ Bưu Chính Viễn thông cơ sở TP. Hồ Chí Minh và quý Thầy Cô đã tạo mọi điều kiện thuận lợi giúp tôi hoàn thành luận văn.

Tôi xin bày tỏ lòng biết ơn chân thành và sâu sắc nhất tới người thầy kính yêu **Thầy TS. Chung Tấn Lâm** đã hết lòng giúp đỡ, trực tiếp hướng dẫn tận tình, động viên khích lệ, tạo điều kiện cho tôi trong suốt quá trình thực hiện luận văn.

Từ đáy lòng mình tôi xin bày tỏ sự biết ơn vô hạn đến gia đình thân yêu của tôi và xin chân thành cảm ơn bạn bè thân thiết, đồng nghiệp trong cơ quan đã động viên, hỗ trợ tôi trong lúc khó khăn để tôi có thể học tập và hoàn thành luận văn.

Mặc dù đã có nhiều cố gắng, nỗ lực tìm tòi nghiên cứu, nhưng do thời gian có hạn và kinh nghiệm nghiên cứu khoa học còn hạn chế nên không thể tránh khỏi những thiếu sót. Tôi rất mong nhận được sự góp ý thiết thực của quý Thầy Cô cùng bạn bè đồng nghiệp để kiến thức của tôi ngày một hoàn thiện hơn.

Xin chân thành cảm ơn!

TP. Hồ Chí Minh, ngày 04 tháng 5 năm 2022

Học viên thực hiện luận văn

Phạm Khương Duy

MỤC LỤC

DANH MỤC CÁC THUẬT NGỮ, CHỮ VIẾT TẮT

Viết tắt	Tiếng Anh	Tiếng Việt
F&B	Food and Beverage Service	Dịch vụ cung cấp đồ ăn và thức uống
SCADA	Supervisory Control and Data Aquisition	Thu nhận dữ liệu và điều khiển
Cobot	Collaborate robot	Robot cộng tác
OPC	OLE for Process Control	Kiến trúc liên kết các đối tượng phân tán trong tự động hóa công nghiệp
OLE	Object Linking and Embedding	Chuẩn giao tiếp dữ liệu giữa các phần mềm theo cơ chế client-server
PLC	Programmable Logic Controller	Bộ điều khiển logic khả trình

DANH SÁCH BẢNG

DANH SÁCH HÌNH VẼ

PHẦN MỞ ĐẦU

1. Tính cấp thiết của đề tài

Ngày nay, trong xu thế cuộc cách mạng công nghiệp lần thứ 4 (CMCN 4.0) đang diễn ra sôi động trên toàn thế giới, Việt Nam đã và đang tham gia tích cực trên nhiều lĩnh vực khác nhau của xã hội như chính phủ, truyền thông đại chúng, y học, khoa học kỹ thuật, …

Trí tuệ nhân tạo cùng với đô thị thông minh, nhà máy thông minh, căn hộ thông minh đang dần được xây dựng nền tảng định hình và hoàn thiện. Các công việc không cần thiết sẽ biến mất, con người lao động tại các khâu sản xuất sẽ dần được thay thế bởi robot. Và một loại robot đặc biệt được phát triển trong thời gian gần đây là robot cộng tác. Robot cộng tác có ưu điểm là cấu trúc nhỏ họn, độ tin cậy cao, tiêu thụ điện năng thấp, dễ xử lý và giá thành rẻ. Những tính năng này làm cho cobot rất hiệu quả để áp dụng rộng rãi không chỉ trong công nghiệp mà còn trong dịch vụ và cuộc sống hàng ngày.

Khi robot được ứng dụng trong lĩnh vực dịch vụ, vấn đề đặt ra là làm cách nào để xây dựng một nền tảng có thể quản lý hệ thống dịch vụ một cách hiệu quả mang lại giá trị cộng thêm từ hệ thống. Nền tảng đó cần có các tính năng như thu thập dữ liệu từ hệ thống, kết nối phần cứng, cảnh báo phần cứng, tổng hợp báo cáo, tổng hợp dữ liệu lịch sử, cơ sở dữ liệu và cài đặt các kết nối điều hành từ xa. Người sử dụng nền tảng đó cùng với robot dịch vụ có thể thống kê, đánh giá số liệu kinh doanh, thói quen khách hàng, … để định hướng kế hoạch kinh doanh phù hợp, đáp ứng nhu cầu khách hàng, phát triển kinh doanh trong tương lai.

Xuất phát từ những lý do trên cùng với sự đồng ý của thầy TS. Chung Tấn Lâm tôi chọn đề tài luận văn: **"XÂY DỰNG HỆ THỐNG ĐIỀU KHIỂN VÀ THU NHẬN DỮ LIỆU CHO ROBOT DỊCH VỤ"**, luận văn góp phần vào việc giải quyết các vấn đề hết sức cần thiết và ứng dụng trong các hệ sinh thái thông minh của quá

trình chuyển đổi số và cuộc CMCN 4.0. Đề tài có sự tham gia hỗ trợ về định hướng học thuật của Công ty TNHH chế tạo máy 3C (Công ty được hỗ trợ bởi Quỹ đổi mới sáng tạo Vingroup VINIF trong dự án mã số VINIF.2020.NCUD.DA059) để kết quả đề tài mang tính thực tiễn, có khả năng ứng dụng sau khi nghiên cứu.

2. Mục đích nghiên cứu

Trên thế giới, robot dịch vụ đang được nghiên cứu, phát triển và đưa vào sử dụng ngày càng trở nên phổ biến như robot bán hàng tự động, robot nấu ăn , robot chăm sóc sức khỏe cho người bệnh, robot hút bụi, robot giữ nhà…

Tuy nhiên các kết quả các robot dịch vụ trên xuất hiện dưới dạng là các sản phẩm thương mại với chi phí khá cao. Việc điều khiển từ xa và thu nhận các dữ liệu từ quá trình xử lý, sử dụng robot hầu như không mở để có thể truy cập thông tin. Hệ thống tích hợp như vậy cần được nghiên cứu giải mã để xây dựng nền tảng công nghệ nguồn phục vụ công cuộc chuyển đổi số trong các lĩnh vực dịch vụ trong một đô thị thông minh trong tương lai sắp tới.

Đề tài luận văn này xây dựng nền tảng hệ thống thu nhận dữ liệu và điều khiển từ xa cho hệ thống tích hợp robot dịch vụ thay thế con người bao gồm hệ thống tự động hóa thuộc lớp thao tác cố định non-realtime và lớp thao tác lập trình được realtime, để từ đó có thể triển khai các mô hình kinh doanh dịch vụ thông minh, đem lại hiệu quả kinh tế và tối ưu trong khâu quản lý và vận hành robot trong lĩnh vực dịch vụ. Các robot không nhất thiết dạng cánh tay mà có thể có các dạng cơ cấu chấp hành để thực hiện các dịch vụ khác nhau. Do điều kiện phần cứng nên đề tài tập trung vào loại robot có thao tác cố định non-realtime.

Xây dựng được một hệ thống điều khiển và thu nhận dữ liệu cho robot dịch vụ dạng thao tác cố định non-realtime để tối ưu chi phí để triển khai hiệu quả tại các cửa hàng kinh doanh lĩnh vực ẩm thực ăn uống như nhà hàng, khách sạn, căn-tin, các cửa hàng thức ăn nhanh, ….

3. Đối tượng và phạm vi nghiên cứu

Đối tượng nghiên cứu

Xây dựng nền tảng một hệ thống thu nhận dữ liệu từ các hoạt động của một trạm robot gồm các cơ cấu tác động thực hiện các thao tác nấu bếp với các cơ cấu chấp hành tương ứng ở mức độ cơ bản và là bước quan trọng nhất của hệ thống đó là thu thập dữ liệu cấp thấp, để từ đó có các bước phát triển tiếp theo. Các bước tiếp theo như phát triển các báo cáo, cảnh báo, vẽ đồ thị, kết đám mây và kết nối các giao diện người sử dụng sẽ được nghiên cứu dễ dàng hơn khi có được nền tảng kết nối giữa phần thiết bị điều khiển với phần mềm cấp cao hơn, và các vấn đề này có thể được nghiên cứu trong các đề tài khác.

Phạm vi nghiên cứu

Đề xuất một trạm robot dịch vụ với các tác vụ điều khiển trình tự thuộc lớp non-realtime. Xây dựng một hệ thống thu nhận dữ liệu trong mạng cục bộ LAN cho một trạm robot dịch vụ đó để theo dõi và cài đặt các tham số làm việc tương ứng. Lập trình điều khiển trạm robot này dùng bộ điều khiển logic khả trình công nghiệp PLC. Các biến I/O của PLC và các thanh ghi cấu hình hệ thống PLC được xây dựng thông qua các định nghĩa Tag trên một phần mềm OPC Server chạy trên Windows. Một chương trình sẽ truy xuất các Tag này để thực hiện tính năng giám sát hoạt động của hệ thống. Các dữ liệu thu nhận được sẽ được tiếp tục khai thác với các tính năng truy vấn cơ sở dữ liệu, tạo báo cáo, các cảnh báo, vẽ đồ thị; tuy nhiên, các tính năng này không nằm trong phạm vi nghiên cứu của đề tài này.

Phương pháp nghiên cứu

Đề tài sử dụng kết hợp nhiều phương pháp nghiên cứu khác nhau như: phương pháp thu thập tài liệu, phương pháp phân tích thiết kế hệ thống điều khiển trình tự, phương pháp tích hợp hệ thống, phương pháp lập trình và mô phỏng thử nghiệm.

Các thiết bị được sử dụng: máy tính Laptop, bộ điều khiển logic khả trình (PLC, Programmble Logic Controller), robot dịch vụ là dạng máy móc tự động được

chuyên môn hóa một tác vụ trình tự nào đó được điều khiển bằng PLC; tuy nhiên thiết bị này sẽ được giả định tối thiểu trên phần mềm và phần cứng phù hợp để minh họa quy trình hoạt động của hệ thống.

Các công cụ lập trình được sử dụng: công cụ lập trình Visual Studio C#, phần mềm OPC Server KepServerEX, phần mềm lập trình PLC GXWork2.

CHƯƠNG 1: NGHIÊN CỨU TỔNG QUAN

1.1 Tổng quan về robot dịch vụ

Ở nhiều quốc gia, các doanh nghiệp dịch vụ đang gặp không ít khó khăn trong việc thuê nhân công để khôi phục việc kinh doanh sau thời gian dịch bệnh. Bên cạnh đó, quay trở lại thị trường lao động sau thời gian dịch bệnh, nhiều nhân viên cũng không mấy mặn mà với công việc phục vụ vất vả với nhiều áp lực. Một nghiên cứu trên Science Robotics vào giữa tháng 4/2022 cho thấy những công việc liên quan đến chuẩn bị thức ăn và phục vụ có nguy cơ cao sẽ bị thay thế bởi robot hơn là các công việc khác liên quan đến giáo dục hoặc chăm sóc sức khỏe [1].

Khi xã hội hạn chế tiếp xúc giữa người với người, thì robot phục vụ đang được chú trọng trong ngành ẩm thực. Nhà hàng Roger trong khách sạn Ameswell mới khai trương năm ngoái tại Mountain View, California (Mỹ) hiện cũng đang có hai robot Servi, cao 104 cm làm việc [2]. Hai robot này do Hãng Bear Robotics tại Silicon Valley chế tạo, chịu trách nhiệm dọn dẹp chén đĩa dơ, mang vào trong bếp và đưa vào máy rửa chén. Ở Singapore, nhà bếp ảo của Grab tại Hillview sử dụng robot để đưa các món đã hoàn tất từ nhà bếp ra cho shipper. Ở Paris, tại một cửa hàng Pizza, toàn bộ quy trình làm bánh, từ nhào bột đến đóng gói vào hộp, đều do robot đảm nhiệm, trong căn bếp làm bằng kính trong suốt có những con robot màu bạc đa năng - tên là Pazzi – có khả năng hoàn thành khoảng 80 hộp bánh pizza mỗi giờ.

Tuy nhiên, theo Business Insider, chính những ông chủ nhà hàng ở Anh, Pháp hoặc Mỹ lại cho rằng tuy các robot về cơ bản khá tiện lợi nhưng chúng chưa thể thay thế hoàn toàn con người. Ở hình thái hiện tại, những robot này không phải là không có thiếu sót. Robot cũng không dễ dàng thích nghi với môi trường mới, những tình huống không dự đoán trước hoặc tình huống bất ngờ, là một phần trong hoạt động của các nhà hàng đông khách. Vì vậy, mặc dù robot rất hiệu quả để thay thế cho các hoạt động tẻ nhạt, không vệ sinh hoặc nguy hiểm, nhưng chúng lại thiếu một dạng trí

tuệ thông minh nhất định của con người khi cần hiểu các yêu cầu phức tạp hơn trong thực tế.

Ngoài ra, robot cũng không rẻ: robot Servi, của Bear Robotics, chi phí khoảng 999 USD/tháng bao gồm cả cài đặt và hỗ trợ. Việc chi số tiền không nhỏ để sở hữu robot phục vụ sẽ buộc các nhà đầu tư phải cân nhắc, và đặt biệt là các nhà nghiên cứu phát triển cân đối giữa chi phí và giá trị tăng thêm sau đầu tư.

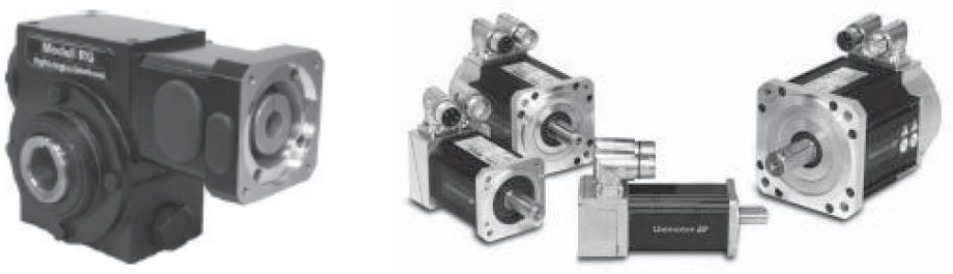
Ta có thế thấy là chi phí của việc ứng dụng robot hiện tại khá cao do thường tích hợp nhiều tính năng không cần thiết cho ứng dụng. Do đó đề tài sẽ tập trung vào loại robot dịch vụ có tính năng vừa đủ đáp ứng yêu cầu của công việc: thay vì phải sử dụng một robot 5,6 bậc tự do thừa chuyển động thì đề tài định hướng các robot khoảng 2, 3 bậc tự do để thay thế thao tác nào đó của con người giảm được chi phí đầu tư ban đầu.

Đề tài này xem xét hai loại robot được định hướng ứng dụng vào lĩnh vực dịch vụ, đó là, (1) Robot có thao tác cố định và đơn giản như các hệ tự động hóa hoạt động trình tự và (2) Robot có lớp thao tác chuyển động phức tạp và lập trình được như robot cộng tác (cobot).

1.1.1 Robot điều khiển trình tự

Robot điều khiển trình tự thường là sự kết hợp của các cơ cấu chấp hành cơ bản như cơ cấu chuyển động tịnh tiến *(Hình 1.1)* và và cơ cấu chấp hành chuyển động quay *(Hình 1.2)* để tạo ra chuyển động phức tạp hơn với 1-3 bậc tự do để giả lập các thao tác nào đó theo yêu cầu thực tế. Các chuyển động thành phần trên có thể là các điều khiển chuyển động vị trí không hồi tiếp để thực hiện các chuyển động điểm-điểm, Với điều khiển trình tự, các hệ thống robot dạng này chiếm đa số các thao tác dịch vụ trong thực tế ứng dụng. Một ví dụ điển hình về tích hợp hệ thống cho robot điều khiển trình tự *(Hình 1.3)*.

Hình 1.1: Một mẫu cơ cấu chấp hành tịnh tiến [8]



Hình 1.2: Một mẫu cơ cấu chấp hành quay động cơ AC với hộp giảm tốc [8]

Hình 1.3: Một mẫu robot dịch vụ dùng cơ cấu tịnh tiến điều khiển trình tự [9]

1.1.2 Robot cộng tác

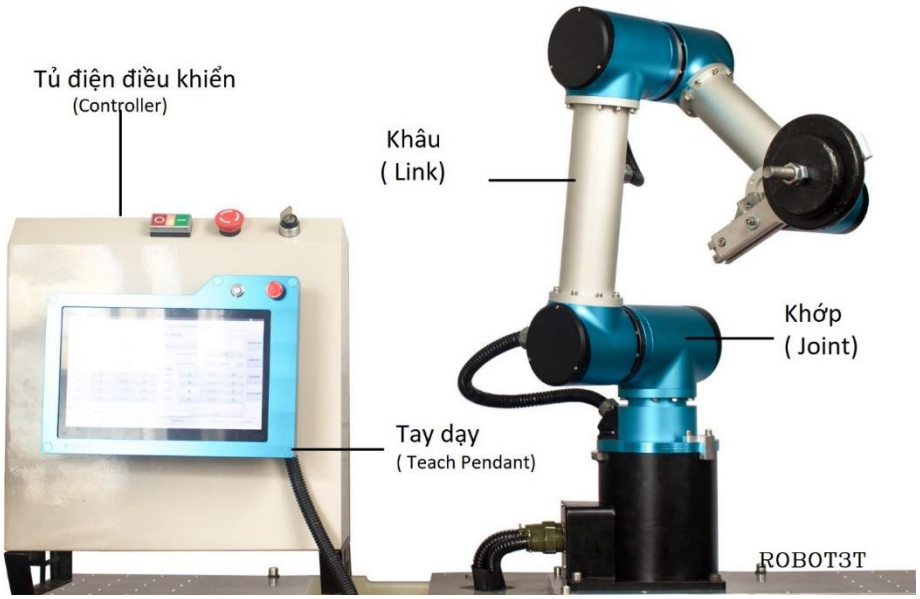
Robot cộng tác (cobot - collaborate robot) là một loại robot được thiết kế để làm việc chung với con người trong một không gian chung. Một trong những điểm khác biệt giữa robot cộng tác so với robot công nghiệp chính là robot công nghiệp thường hoạt động độc lập, cần có thiết bị bảo vệ và rào chắn để tránh gây nguy hiểm đến con người; còn robot cộng tác được đánh giá cao bởi tính an toàn, chính vì thế đây được coi là trợ thủ đáng tin cậy khi làm việc cùng con người trong các hoạt động sản xuất công nghiệp và dịch vụ. Một số ưu điểm của cobot liệt kê trong bảng 1.1

Robot cộng tác có ưu điểm là cấu trúc nhỏ họn, độ tin cậy cao, tiêu thụ điện năng thấp, dễ xử lý và giá thành rẻ. Những tính năng này làm cho robot cộng tác rất hiệu quả để áp dụng rộng rãi cho nhiều ngành công nghiệp và cuộc sống. Ngày nay, robot cộng tác không chỉ cách mạng hóa ngành công nghiệp sản xuất, chúng còn được sử dụng kết hợp với các công nghệ hiện tại để đổi mới các ngành dịch vụ nhằm tăng giá trị thực và cho cuộc sống của chúng ta.

Bảng 1.1: Một số ưu điểm cobot so với robot công nghiệp truyền thống

Stt	Mục đánh giá	Cobot	Robot công nghiệp
1	Chi phí đầu tư và vận hành	Thấp 10.000-150.000	Cao >150.000$
2	Khả năng di chuyển	Dễ dàng di chuyển đầu công tác tiếp cận đối tượng.	Quá trình di chuyển và cài đặt tốn nhiều thời gian
3	Giao tiếp người dùng	Dễ dàng sử dụng, cấu trúc lệnh đơn giản	Yêu cầu người vận hành có chuyên môn, cấu trúc lệnh tương đối phức tạp
4	Chuẩn bảo vệ	Tích hợp cảm biến dừng ngay khi có phát hiện ngoại lực, đảm bảo an toàn	Không tích hợp cảm biến, chỉ dừng khi có lệnh, có thể gây nguy hiểm cho

			người dùng. Cần không gian che chắn
5	Không gian chiếm chỗ, lắp đặt	Nhỏ gọn, nhẹ, không gian chiếm chỗ ít. Lắp đặt linh hoạt	Nặng, cồng kềnh, chiếm nhiều diện tích sử dụng. Lắp đặt tốn nhiều thời gian.



Hình 1.4: Cấu trúc cơ bản của một robot cộng tác [10].

Về cơ bản cấu tạo của một cobot gồm 3 phần chính *(Hình 1.4)*:

Tay máy (manupulator): Sự lắp ráp nối tiếp nhau của 2 thành phần chính: Khâu (Link) và Khớp (Joint).

Tủ điều khiển (controller cabinet): chứa các phần tử điện của hệ thống điều khiển

Tay dạy (teach pendant): công cụ giao tiếp giữa người dùng và cobot.

Một số lý do giải thích cho sự phát triển nhanh chóng của robot cộng tác:

- Robot công nghiệp trong ngành ô tô đã tương đối bão hòa và cần hướng tới các ngành sản xuất khác như: điện tử, dược phẩm, thực phẩm và các ngành công nghiệp khác, dịch vụ.

- Robot công nghiệp truyền thống được thiết kế cho ngành công nghiệp ô tô đã được tự động hóa sản xuất, nhưng kết quả cho thấy bắt đầu có dấu hiệu kém hiệu quả và không đáp ứng được nhu cầu ngày càng cao.

- Sự hợp tác giữa con người và robot hiệu quả hơn và an toàn hơn

- Robot cộng tác có thể mở rộng áp dụng cho những ngành như dịch vụ, đa phương tiện, phẫu thuật y tế, giáo dục…

- Giá thành robot cộng tác thấp góp phần đẩy mạnh sản xuất cho các doanh nghiệp vừa và nhỏ.

1.2 Các thành phần hệ thống SCADA

SCADA (Supervisory Control And Data Acquisition) là một hệ thống điều khiển giám sát và thu thập dữ liệu nhằm hỗ trợ con người điều hành và quản lý một hệ thống hiệu quả.

Cấu trúc một hệ SCADA có các thành phần cơ bản sau:

- Trạm điều khiển giám sát trung tâm: gồm một hay nhiều máy chủ trung tâm (central host computer server).
- Trạm thu thập dữ liệu trung gian: gồm các khối thiết bị vào ra đầu cuối từ xa RTU (Remote Terminal Units) hoặc là các khối điều khiển logic khả trình PLC (Programmable Logic Controllers) có chức năng giao tiếp với các I/O (gồm các cảm biến và các cơ cấu chấp hành lắp đặt trên máy).
- Hệ thống truyền thông: bao gồm các mạng truyền thông công nghiệp, các thiết bị viễn thông và các thiết bị chuyển đổi dồn kênh có chức năng truyền dữ liệu cấp trường đến các khối điều khiển và máy chủ.
- Giao diện người – máy HMI (Human - Machine Interface): gồm các thiết bị

hiển thị quá trình xử lý dữ liệu để người vận hành điều khiển các quá trình hoạt động của hệ thống.

Các thành phần tự động hóa và robot không hoạt động độc lập mà hoạt động với sự kết nối với nhau, giữa các bộ phận này với bộ phận khác, giữa máy móc với người sử dụng và người điều khiển điều hành bảo trì hệ thống, các bộ phận kinh doanh cũng như người chủ của hệ thống. Do đó hệ thống SCADA là cần thiết đối với mọi hoạt động kể cả các hoạt động dịch vụ trong bối cảnh chuyển đổi số

1.3 Bộ điều khiển logic khả trình

1.3.1 Giới thiệu chung

Bộ điều khiển logic khả trình (PLC - Programmable Logic Controller) là thiết bị cho phép thực hiện linh hoạt các giải pháp điều khiển logic thông qua một ngôn ngữ lập trình. Người sử dụng có thể lập trình để thực hiện một loạt trình tự các sự kiện. Các sự kiện này được kích hoạt bởi tác nhân kích thích (ngõ vào) tác động vào PLC hoặc qua các hoạt động có trễ như thời gian định thì hay các sự kiện được đếm.

Một khi sự kiện được kích hoạt thật sự, nó bật ON hay OFF thiết bị điều khiển bên ngoài được gọi là thiết bị vật lý. Một bộ điều khiển lập trình sẽ liên tục "lặp" trong chương trình do "người sử dụng lập ra" chờ tín hiệu ở ngõ vào và xuất tín hiệu ở ngõ ra tại các thời điểm đã lập trình.

Để khắc phục những nhược điểm của bộ điều khiển dùng dây nối (bộ điều khiển bằng rờ-le, relay) người ta đã chế tạo ra bộ PLC nhằm thỏa mãn các yêu cầu sau:

- Lập trình PLC đơn giản, ngôn ngữ lập trình dễ học
- Gọn nhẹ, dể dàng bảo quản, sửa chữa
- Dung lượng bộ nhớ lớn để có thể chứa được những chương trình phức tạp
- Hoàn toàn tin cậy trong môi trường công nghiệp
- Giao tiếp được với các thiết bị thông minh khác như : máy tính , nối mạng , các môi Module mở rộng
- Giá cả có thể cạnh tranh được

Một số dòng PLC của họ Mitsubishi như sau:

- FX1S, FX1N, FX2N, FX2NC: là loại PLC tầm trung
- FX3U, FX3UC, FX3S, FX3G: là loại PLC dòng cao cấp

Hình 1.5: Các dòng PLC Mitsubishi [11]

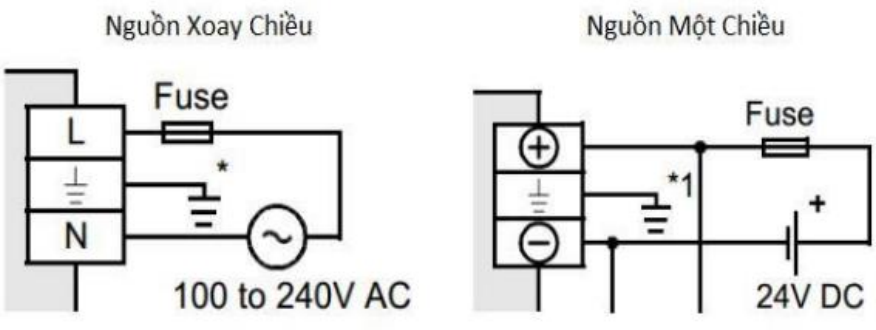
Dòng sản phẩm mới PLC FX3U *(Hình 1.5)* là thế hệ thứ ba trong gia đình họ FX-PLC, là một PLC dạng nhỏ gọn và thành công của hãng Mitsubishi Electric.

Sản phẩm được thiết kế đáp ứng cho thị trường quốc tế, tính năng đặc biệt mới là hệ thống "adapter bus" được bổ xung cho hệ thống bus hữu ích cho việc mở rộng thêm những tính năng đặc biệt và khối truyền thông mạng. Khả năng tối đa có thể mở rộng lên đến 10 khối trên bus mới này.

Hình 1.6: PLC Mitsubishi FX3U [11]

Các sơ đồ kết nối I/O cơ bản trên PLC:

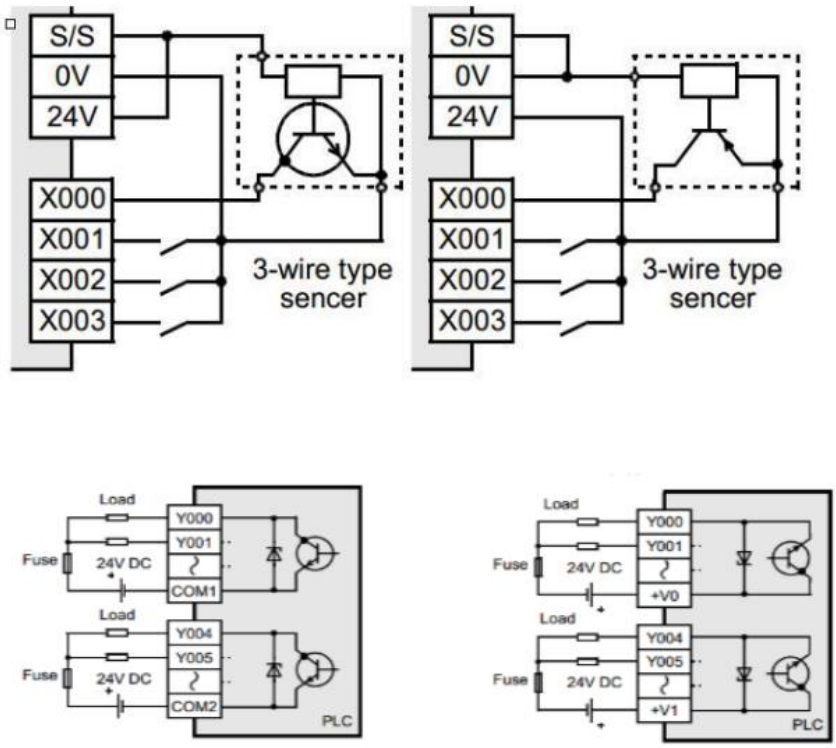
(1) Kết nối nguồn cho PLC: có 2 kiểu kết nối nguồn cho PLC bao gồm kết nối nguồn điện 220VAC và nguồn điện 24VDC *(Hình 1.4)*



Hình 1.7: Sơ đồ kết nối cấp nguồn cho PLC

(2) Kết nối các cổng vào và cổng ra I/O cho PLC:

Sơ đồ kết nối cổng vào và cổng ra cho PLC được cho trên sơ đồ *Hình 1.5*.



Hình 1.8: Sơ đồ kết nối cổng vào và cổng ra cho PLC

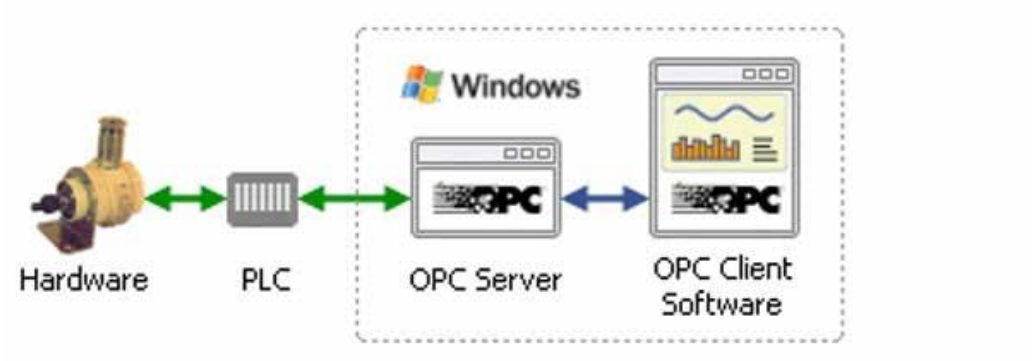
1.3.2 Phần mềm lập trình PLC GX Works 2

GX Works2 là phần mềm cấu hình và lập trình thế hệ kế tiếp cho điều khiển FX và Q Series. GX Works2 cho phép nhà phát triển có thể "trộn" và kết hợp từ 5 ngôn ngữ lập trình khác nhau, phù hợp với các phong cách lập trình khác nhau. Người lập trình có thể lựa chọn ngôn ngữ để phát triển cho phù hợp với công việc. Môi trường này tuân theo tiêu chuẩn IEC1131-3, cũng cho phép các bộ phận của dự án lưu trong thư viện để sử dụng trong các ứng dụng trong tương lai. Hoàn toàn tùy biến các cài đặt, có nghĩa là lựa chọn công cụ và các phím tắt để tối ưu hóa khả năng trực giác của riêng người dùng. Tích hợp mạng và các module chức năng đặc biệt giữ cho các tập tin dự án tổ chức và dễ dàng truy cập. Tính năng PLC ảo trên máy tính mô phỏng cho phép hệ thống hoàn chỉnh trước khi có phần cứng. Sau khi dự án được tải về hệ thống thực tế, GX Works2 bao gồm nhiều chế độ theo dõi , theo dõi chức năng, và khả năng gỡ lỗi trực tuyến cho phép kiểm soát được tình trạng của ứng dụng.

Hình 1.9: Phần mềm lập trình cho PLC Mitsubishi GX Works 2

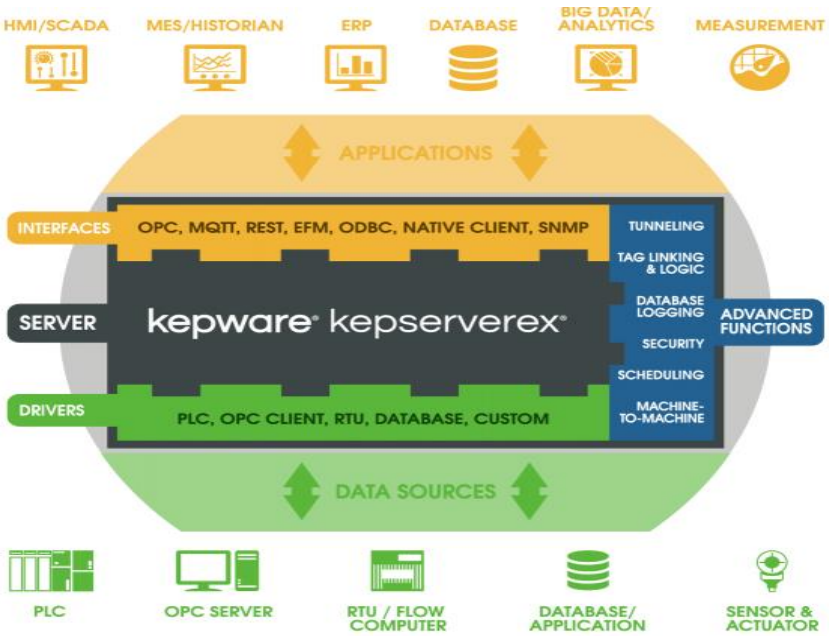
1.4 Hệ thống OPC server - KepserverEX

OPC (OLE for Process Control), với OLE là Object Linking and Embedding. OPC là 1 chuẩn giao tiếp dữ liệu giữa các phần mềm theo cơ chế client-server, được sử dụng rộng rãi trong công nghiệp, đảm bảo tính linh hoạt và tương thích giữa các thành phần từ nhiều hãng sản xuất khác nhau trong hệ thống. Sơ đồ tổng quan OPC Server được cho trên *Hình 1.10*.



Hình 1.10: Sơ đồ tổng quan OPC server

1.4.1 Giới thiệu phần mềm Kepware OPC



Hình 1.11: Sơ đồ tổng quan phần mềm KEPServerEX [12]

KepserverEX là một tiêu chuẩn tương tác giao diện phần mềm cho phép trao đổi dữ liệu an toàn và đáng tin cậy giữa các chương trình windows và các thiết bị phần cứng công nghiệp bao gồm các loại PLC, các bộ điều khiển truyền thông Modbus, Profibus, Profinet,…. KepserverEX độc lập với các nền tảng và đảm bảo truyền thông tin liên tục trên nhiều thiết bị của các nhà cung cấp khác nhau *(Hình 1.11)*.

Nguyên lý hoạt động của Kepware OPC

Thuộc tính OPC mô tả giao diện kết nối giữa máy khách và máy chủ, máy chủ và máy chủ, bao gồm quyền truy cập dữ liệu thời gian thực, giám sát các báo động và sự kiện, truy cập dữ liệu lịch sử và các ứng dụng khác.

Kịch bản kết nối OPC cổ điển là một kết nối máy chủ-máy khách trên một máy tính, ngoài ra cũng có các tuỳ chọn khác:

- Kết nối máy khách OPC với một số máy chủ OPC, đây được gọi là tập hợp OPC.
- Kết nối máy khách OPC với OPC server qua mạng. Điều này có thể được thực hiện với đường hầm OPC.
- Kết nối máy chủ OPC với máy chủ OPC khác để chia sẻ dữ liệu. Điều này được gọi là bắc cầu OPC.

Giải pháp SCADA dùng Visual Studio C#

KEPServerEX đóng vai trò trung gian thực hiện các kết nối dữ liệu từ PLC lên Visual studio và ngược lại, KEPServerEX sẽ được cài đặt trong máy tính chạy SCADA, thông qua các kết nối vật lý của máy tính với PLC, KEPServerEX lấy dữ liệu từ PLC và gửi dữ liệu qua Visual studio C# và ngược lại.

Phần mềm KEPServerEX OPC hỗ trợ tối đa hầu hết các dòng PLC trên thị trường, do đó có thể thực hiện SCADA trên Visual Studio cho đa số toàn bộ các loại PLC mà vẫn đảm bảo tốc độ truyền dữ liệu.

Dữ liệu sản xuất/dịch vụ thu thập từ các PLC hiện trường sẽ được lưu trữ thông qua phần mềm cơ sở dữ liệu SQL Server, từ đó xuất ra các loại báo cáo và lưu trữ dữ liệu sản xuất với các định dạng khác nhau.

Ưu điểm giải pháp

- Vì phần mềm KEPServerEX có thể kết nối với rất nhiều loại PLC, do đó đa số PLC đều có thể kết nối được với giao diện SCADA trên Visual Studio C#.
- Cấu hình và cài đặt KEPServerEX dễ dàng thực hiện.
- Có thể cài đặt trên các máy tính có cấu hình trung bình như PC có cấu hình RAM 4GB, bộ vi xử lý Core I3.
- Tốc độ truyền dẫn dữ liệu nhanh, đảm bảo tính liên tục khi kết nối với các dòng PLC.

1.4.2 Đăng ký OPC

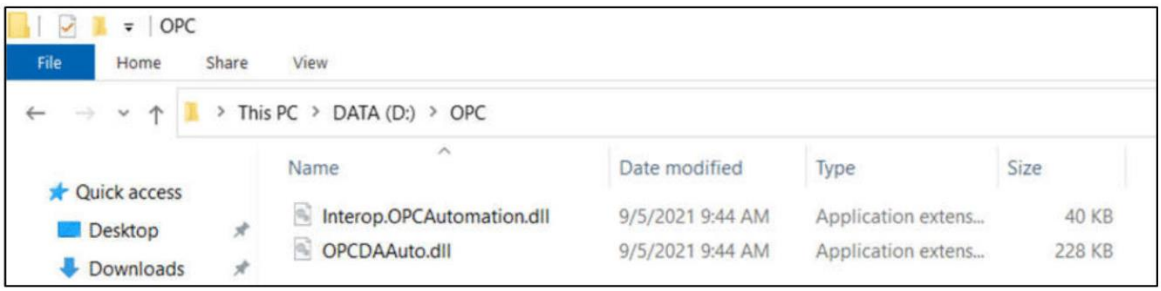
OPC được thực hiện trong các cặp client/server. Máy chủ OPC là một chương trình phần mềm chuyển đổi giữa giao thức truyền thông phần cứng được PLC sử dụng thành ra giao thức OPC. Phần mềm máy OPC client là bất kỳ chương trình nào mà cần kết nối với phần cứng, chẳng hạn như là HTML. Máy OPC client sử dụng máy chủ OPC để lấy được dữ liệu từ hay gửi các lệnh tới phần cứng.

OPC được sử dụng như là công cụ truyền thông dữ liệu từ các PLC tại hiện trường đến phần mềm Visual Studio thông qua KEPServerEX.

Để sử dụng OPCDA thực hiện việc kết nối Visual Studio (C#) với KEPServerEX, khi thao tác với phần mềm cần 2 file DLL (Dynamic Link Library) sau:

OPCDAAuto: OPC DA Connect sử dụng OPC DA Auto Service. Dịch vụ này phải được đăng ký trên máy tính để cho phép điều hướng trên máy chủ. OPC DA Connect phân phối phiên bản OPCDAAuto.dll và đăng ký dll này trong khi cài đặt.

Interop OPCAutomation: Bao gồm thư viện các hàm sử dụng cho OPC DA, thư viện này chứa các câu lệnh truy vấn hoặc đọc ghi dữ liệu khi thực hiện kết nối Visual studio với KEPServerEX.



Hình 1.12: Thư viện dll kết nối KEPServerEX với Visual Studio C#

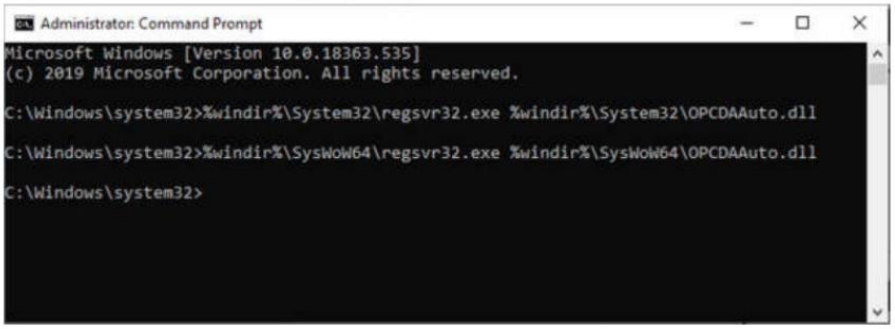
Đăng ký OPCDA Auto Service

Bước 1: Copy file OPCDAAuto.dll vào 2 thư mục C:\windows\System32 và C:\windows\SysWOW64.

Bước 2: Chạy CMD (Command Prompt) bằng quyền Admin: Click start, Nhập "CMD" (D ở tab tìm kiếm của window, Click chuột phải vào Command Prompt (2), sau đó nhấn chọn Run as administrator (3).

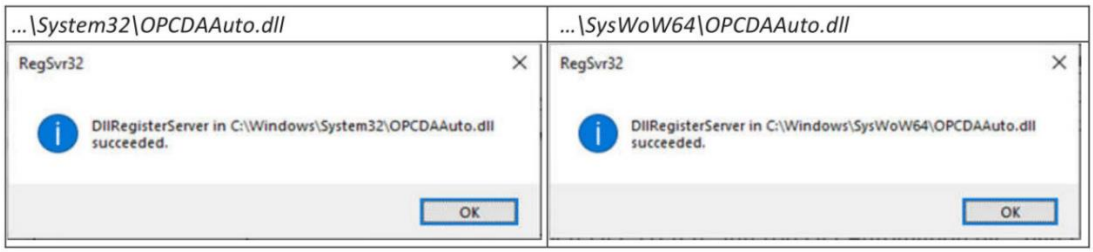
Bước 3: Nhập 2 câu lệnh ở bảng sau để đăng ký OPC

TT	Câu lệnh đăng ký OPC Server
1	*%windir%\System32\regsvr32.exe %windir%\System32\OPCDAAuto.dll*
2	*%windir%\SysWoW64\regsvr32.exe %windir%\SysWoW64\OPCDAAuto.dll*



Hình 1.13: Đăng ký OPC Server trên máy tính win10

Nếu đăng ký thành công thì Windows sẽ hiện 2 thông báo sau:



1.4.3 Cấu hình PLC Mitsubishi trên KEPServerEX

(1). Thêm mới Channel

Xem **PHỤ LỤC P1.1**.

(2). Thêm mới Device

Xem **PHỤ LỤC P1.2.**

(3). Khai báo địa chỉ PLC Mitsubishi trong KEPServerEX

PLC Mitsubishi có các vùng nhớ được cấp phát bảo đảm cho PLC hoạt động. Các vùng nhớ này được đặt tên chuyên dùng để việc lập trình PLC được dễ dàng. Tương tự như vậy, các vùng nhớ này được ánh xạ vào các địa chỉ khai báo trên KepserverEX để bảo đảm kết nối chính xác với các vùng nhớ đã cấp phát trên PLC.

Bảng 1.2: Thống kê địa chỉ trên phần mềm KepserverEX đối với PLC Mitsubishi

No	Miền nhớ	Địa chỉ KEPServerEX	Định dạng	Quyền hạn
1	Inputs (cổng vào logic)	X000-X1FFF(Hex)	Boolean	Đọc/Ghi
		X000-X1FF0 (Hex)	Short, Word, BCD	Đọc/Ghi
		X000-X1FE0 (Hex)	Long, DWord, LBCD	Đoc/Ghi
2	Outputs (cổng ra logic)	Y000-Y1FFF (Hex)	Boolean	Đọc/Ghi
		Y000-Y1FF0 (Hex)	Short, Word, BCD	Đọc/Ghi
		Y000-Y1FE0 (Hex)	Long, DWord, LBCD	Đọc/Ghi
3	Link Relays (cờ chuyên dùng cho truyền thông)	B000-B1FFF (Hex)	Boolean	Đoc/Ghi
		B000-B1FF0 (Hex)	Short, Word, BCD	Đọc/Ghi

		B000 B1FE0 (Hex)	Long, DWord, LBCD	Đọc/Ghi
4	Internal Relays (cờ nhớ)	M0000-M8191	Boolean	Đoc/Ghi
		M0000-M8191	Short, Word, BCD	Đọc/Ghi
		M0000-M8160	Long, DWord, LBCD	Đọc/Ghi
5	Special Relays (cờ chuyên dùng)	M9000-M9255	Boolean	Chỉ đoc
		M9000-M9240	Short, Word, BCD	Chỉ đọc
		M9000-M9224	Long, DWord. LBCD	Chỉ đoc
6	Latch Relays (cờ có nhớ)	L0000-L8191	Boolean	Đọc/Ghi
		L0000-L8176	Short, Word, BCD	Đọc/Ghi
		L0000-L8160	Long, DWord, LBCD	Đọc/Ghi
7	Annunciator Relays (cờ thông báo)	F0000-F2047	Boolean	Đọc/Ghi
		F0000-F2032	Short, Word, BCD	Đọc/Ghi
		F0000-F2016	Long, DWord, LBCD	Đọc/Ghi
8	Timer Contacts (công tắc bộ định thì)	TS0000-TS2047	Boolean	Đọc/Ghi
		TS0000-TS2032	Short, Word, BCD	Đọc/Ghi
		TS0000 TS2016	Long, DWord, LBCD	Đọc/Ghi
9	Timer Coils (cổng ra bộ định thì)	TC0000-TC2047	Boolean	Đọc/Ghi
		TC0000-TC2032	Short, Word, BCD	Đọc/Ghi
		TC0000-TC2016	Long, DWord, LBCD	Đọc/Ghi
10	Counter Contacts (công tắc bộ đếm)	CS0000-CS1023	Boolean	Đọc/Ghi
		CS0000-CS1008	Short, Word, BCD	Đọc/Ghi
		CS0000-CS0992	Long, DWord, LBCD	Đọc/Ghi
11	Counter Coils (cổng ra bộ định thì)	CC0000-CC1023	Boolean	Đoc/Ghi
		CC0000-CC1008	Short, Word, BCD	Đọc/Ghi
		CC0000-CC0992	Long, DWord, LBCD	Đọc/Ghi
12	Timer value (giá trị bộ định thời)	TN0000-TN2047	Short, Word, BCD	Đoc/Ghi

13	Counter value (giá trị bộ đếm)	CN0000-CN1023	Short, Word, BCD	Đọc/Ghi
14	Data registers (thanh ghi dữ liệu)	D0000-D8191	Short, Word, BCD	Đọc/Ghi
		D0000-D8190	Long, DWord, LBCD, Float, Date	Đọc/Ghi
		D0000-D8188	Double	Đọc/Ghi
15	Data Register Bit Access (thanh ghi dữ liệu truy xuất bit)	D0000.00-D8191.15	Short, Word, BCD, Boolean	Đọc/Ghi
		D0000.00-D8190.31	Long, DWord, LBCD	Đọc/Ghi
16	Thanh ghi dữ liêu dạng string từ byte cao đến thấp (HiLo Byte Ordering)	DSH00000.002-DSH08190.002	String	Đoc/Ghi
		DSH00000.128-	String	Đọc/Ghi

Trong chương sau, khi thiết kế quy trình điều khiển trình tự cho robot dịch vụ, các biến điều khiển cũng như các tham số của máy được sử dụng trên PLC sẽ được khai báo trên phần mềm KepserverEX thực hiện việc giám sát thường xuyên.

1.5 Kết luận chương

Chương này trình bày các khái niệm cơ bản hình thành nên một ứng dụng của robot trong lĩnh vực dịch vụ, bao gồm robot cộng tác, cấu hình cho OPC server (dùng phần mềm KepserverEX), các định dạng dữ liệu giữa bộ điều khiển khả trình PLC và dữ liệu giám sát trên phần mềm C#. Robot dịch vụ cần giám sát bao gồm 2 lớp: lớp realtime robot và lớp non-realtime robot. Trong đề tài này tập trung vào lớp non-realtime robot: đó là các thiết bị tự động hóa có thao tác cố định dùng PLC, không có các quỹ đạo phức tạp cần phải bám theo (tracking). Một hệ thống robot thuộc lớp non-realtime là phổ biến chiếm nhiều ứng dụng trong lĩnh vực dân dụng sẽ được dùng để minh họa khả năng điều khiển trong đề tài này.

Robot trong lĩnh vực dịch vụ có thể ứng dụng trong nhiều ngành, trong đó lĩnh vực ẩm thực, nhà hàng, ăn uống, dịch vụ (F&B - Food and Beverage Service) là một

minh họa tốt nhất cho các hoạt động dịch vụ nên được chọn làm mô hình thử nghiệm trong đề tài này.

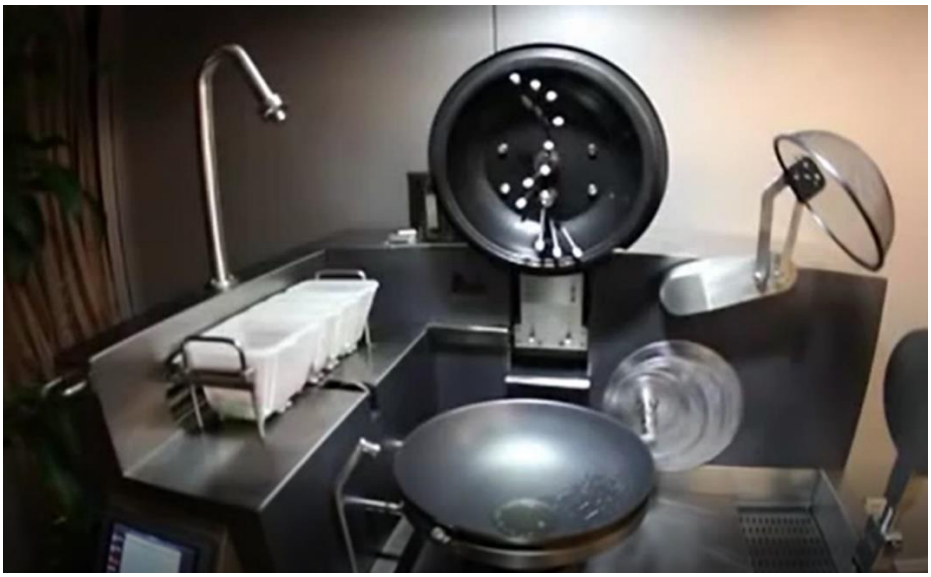
Hệ thống thu nhận dữ liệu này là cơ sở với khả năng mở rộng thêm nhiều robot dịch vụ hay nhiều thiết bị tự động khác (khi cần thiết). Ngoài ra, hệ thống này cũng dễ dàng tích hợp hệ thống đặt hàng POS (Point of Sale) một cách dễ dàng thay vì xuất dữ liệu cho máy in trong khu vực nhà bếp. Hệ thống POS không đề cập trong đề tài này.

CHƯƠNG 2: THIẾT KẾ I/O HỆ THỐNG ROBOT DỊCH VỤ

2.1 Mô tả hệ thống

Hệ thống robot dịch vụ giả định với các hoạt động tham khảo theo [14] là một trạm phục vụ bếp nấu tự động tích hợp của các cơ cấu chấp hành như sau *(Hình 2.1)*:

- Thiết bị nấu ăn dạng trộn
- Hệ thống van cấp dầu, van nước rữa xả
- Hệ thống gia nhiệt
- Hệ thống cấp liệu
- Thiết bị cánh tay robot thao tác cố định (không lập trình được)

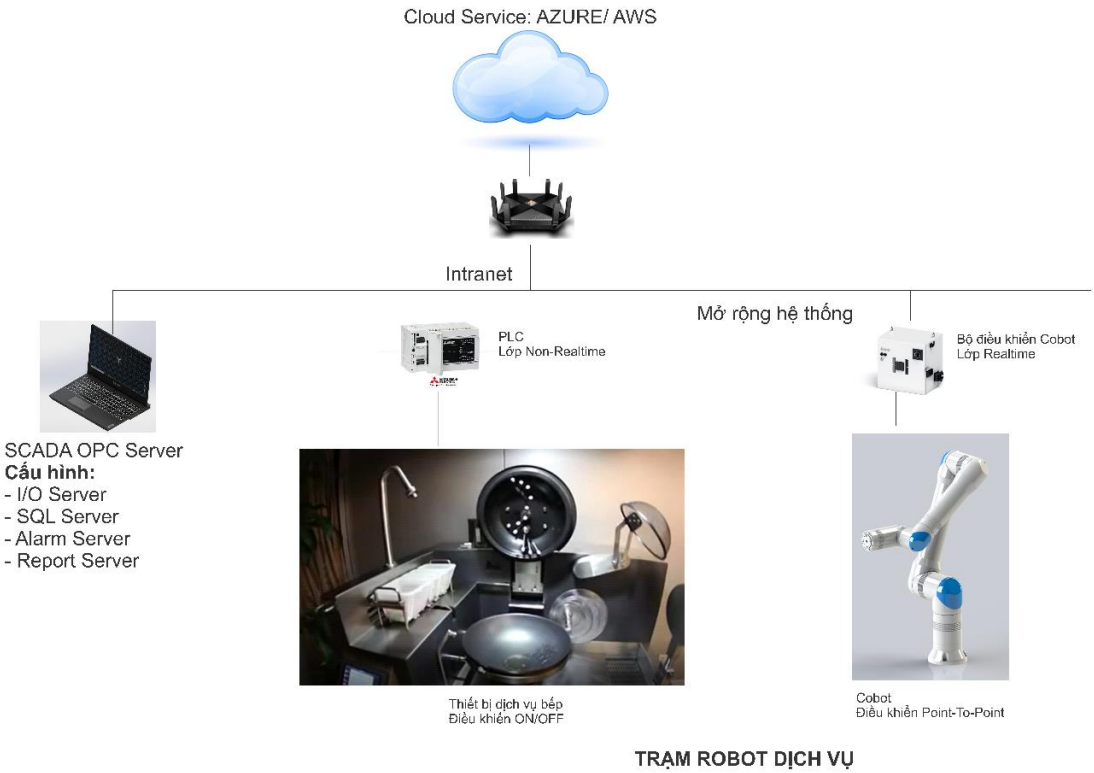


Hình 2.1: Hệ thống bếp nấu tự động [14]

Các tác vụ của cánh tay robot có hai loại: (1) loại cánh tay robot có chuyển động cố định đơn giản là dạng chuyển động phổ biến trong lĩnh vực dịch vụ, và (2) loại cánh tay robot có chuyển động quỹ đạo phức tạp có khả năng lập trình được ví dụ các thao tác trung chuyển, các thao tác trong không gian 3D; tuy nhiên không được khảo sát để giám sát trong đề tài này vì robot này cần một bước lập trình trung gian

và phần mềm chuyên dùng để thiết kế quỹ đạo thao tác của robot và vấn đề này có thể được nghiên cứu ở một để tài khác.

Mô hình kết nối hệ thống thu nhận dữ liệu cho robot dịch vụ giả định dạng cánh tay được khảo sát và trình bày như *(Hình 2.2)*. Quá trình điều khiển dạng ON/OFF thiết bị bếp chuyên dùng sử dụng bộ điều khiển logic khả trình công nghiệp PLC. Các biến I/O của PLC và các thanh ghi cấu hình hệ thống PLC được xây dựng thông qua các định nghĩa thông qua các Tag trên OPC-KepserverEX và lập trình C# chạy trên Windows.



(a) (b)

Hình 2.2: Mô hình kết nối hệ thống thu nhận dữ liệu cho robot dịch vụ (a) Robot có quỹ đạo cố định, (b) Robot có quỹ đạo phức tạp

2.1.1 Nhà hàng truyền thống

Bước 1: khi khách hàng vào nhà hàng, sẽ đến quầy đặt món ăn có nhân viên thu ngân thao tác trên phần mềm chuyên dụng để nhận được một số thứ tự *(Hình 2.3)*. Sau đó

khách hàng có thể ngồi vào bất cứ bàn bàn nào còn trống và nhân viên phục vụ sẽ cập nhật số bàn để sau đó có thể thực hiện phục vụ món đến đúng số bàn tương ứng với số thứ tự khách đã nhận được.

Hình 2.3: Cách thức đặt món và thanh toán tại quầy với nhân viên thu ngân

Bước 2: sau khi chọn món, khách hàng chọn chỗ ngồi ưng ý và đợi một khoảng thời gian sẽ có các món được nhân viên mang lên *(Hình 2.4)*. Nếu khách hàng muốn đặt thêm thì sẽ tiến lại đến quầy và làm như bước 1, khi đó khách hàng sẽ được tính vào bill món mới.

Hình 2.4: Khách hàng ngồi chờ sau khi đặt món

Bước 3: sau khi khách hàng đặt món xong các món của khách sẽ được hệ thống tổng hợp lại trong 1 hóa đơn.

Bước 4: Bill đó sẽ được tự động gửi cho bếp thông qua máy in để đầu bếp thực hiện nấu các món trong bill một cách thủ công.

Bước 5: Sau khi nấu xong tất cả các món trong bill, bếp sẽ đưa ra thông báo cho nhân viên để mang ra cho khách *(Hình 2.5)*.

Hình 2.5: Nhân viên mang thức ăn cho khách sau khi nấu xong

Bước 6: Sau khi dùng xong nhân viên thực hiện tính tiền với khách và thanh toán.

2.1.2 Nhà hàng thông minh

Về mặt giao tiếp với khách hàng, khi khách đến với nhà hàng, sẽ tiến thẳng đến bàn ăn có trang bị màn hình để chọn món, khách hàng sẽ chọn các món ăn mình muốn và lấy mã số thực đơn để đợi lấy món và nhấn nút thực hiện thanh toán. Với hình thức này, khách hàng sẽ tự chọn bàn cho mình tùy theo ý thích và nhà hàng sẽ tiết kiệm nhân lực cho khâu này, tiết kiệm được khoảng thời gian chờ đợi.

Hình 2.6: Giao diện đặt hàng của nhà hàng thông minh

Khi khách đã chọn món tại bàn, thực đơn sẽ được gửi tự động đến Webserver và gửi xuống khu nhà bếp, từ đây hệ thống bếp tự động nấu bằng hệ thống bếp nấu tự động mà không cần in thực đơn hoặc cần thêm người phục vụ ghi thực đơn. Món ăn của khách hàng sẽ hoàn thành trong thời gian tối ưu theo thiết kế mà vẫn giữ được vị ngon giống như đầu bếp chuyên nghiệp, vì công thức, nhiệt độ và thời gian để nấu từng món đã được nghiên cứu và khảo sát sao cho món ăn ngon nhất đã được thực nghiệm và lưu trên server của nhà hàng. Món ăn sẽ được nhân viên mang ra trên xe đẩy hoặc bằng robot di động *(Hình 2.7)*.

Hình 2.7: Robot di động BellaBot mang thức ăn cho khách [13]

2.2 Thiết kế hệ thống

Thiết kế hệ thống cơ khí bếp nấu phục vụ lĩnh vực phục vụ ăn uống F&B và phân tích hoạt động (điều khiển trình tự) của bếp để làm cơ sở từ đó cấp phát Input/Output cho toàn bộ hệ thống. Với các I/O đã được cấp phát như trên ta tiến hành lập trình PLC theo trình tự đã thiết kế và đề xuất các biến bộ nhớ (cờ M và thanh ghi D) trên PLC cần giám sát để khai báo cấu hình các Tags tương ứng trên phần mềm OPC-Server là KepServerEX. Các Tags này đóng vai trò quan trọng của hệ thống và được truy vấn trạng thái và giá trị của chúng thông qua một I/O Server dùng C# hình thành một hệ giám sát (SCADA) cho hệ thống.

Đề tài này chỉ tập trung vấn đề truy xuất dữ liệu hoàn chỉnh từ PLC về máy tính để hoàn thiện I/O server là vấn đề cốt lõi của bài toán SCADA và lập trình PLC cho bài toán điều khiển trình tự để có dữ liệu giám sát cho đề tài. Đề tài chưa tập trung phát triển các thành phần khác của SCADA như Database server, Report server, Trend server. Các vấn đề này cần được nghiên cứu chuyên sâu trong các đề tài khác do tính đặc thù cho từng ứng dụng cụ thể.

2.3 Nguyên lý vận hành hệ thống

Hệ thống dịch vụ là hệ thống nếu bếp tự động được trang bị các cơ cấu chấp hành và quy trình hoạt động của bếp tham khảo theo [14] để giải mã công nghệ với thiết kế được trình bày trong các mục sau.

2.3.1 Cấp phát I/O

1. Van cấp dầu định lượng **Valve_Oil** định lượng theo thời gian:

- **t_oil_1:** thời gian châm dầu lúc chế độ trộn thức ăn
- **t_oil_2:** thời gian châm dầu lúc chế độ lắc chảo

Hình 2.8: Hệ thống van cấp dầu [13]

2. Cơ cấu tác động đóng/mở nắp **Actuator_Cover** (On/Off)

Cơ cấu này dùng để đóng mở nắp, trên đó có trang bị cơ cấu quay để thực hiện tác vụ trộn nguyên liệu trong chảo và làm sạch chảo khi nấu xong món ăn

Hình 2.9: Cơ cấu đóng/mở nắp [14]

3. Cơ cấu tác động hệ thống gia nhiệt Actuator_Temperature

Cơ cấu này là cụm thiết bị chuyên dùng để gia nhiệt theo profile đảm bảo nón ăn được điều chỉnh nhiệt độ tương ứng với từng giai đoạn nấu. Trong trường hợp đề tài

chỉ giả định chỉ gia nhiệt với 3 giá trị cố định để đơn giản trong việc minh hoạ quy trình nấu ăn. Các tham số như sau:

- **T_1:** nhiệt độ nấu ăn 1
- **T_2:** nhiệt độ nấu ăn 2
- **T_3:** nhiệt độ nấu ăn 3 (dự phòng)

Hình 2.10: Hệ thống gia nhiệt dùng Gas [14]

4. Động cơ trộn **Motor_Blending**, tham số

- **speed_blending_i** (i=1-4): tốc độ quay
- **time_blending_i** (i = 1-4): thời gian quay

Động cơ này thực hiện việc trộn nguyên liệu trong quay trình với các chế độ khác nhau về tốc độ quay và thời gian quay tùy loại nguyên liệu được thêm vào.

Hình 2.11: Động cơ thực hiện thao tác trộn nguyên liệu [14]

5. Cơ cấu chấp hành chuyên cấp liệu **Actuator_Ingredients**, tham số

- **ingredient_1:** vị trí nguyên liệu 1
- **ingredient_2:** vị trí nguyên liệu 2
- **ingredient_3:** vị trí nguyên liệu 3
- **ingredient_4:** vị trí nguyên liệu 4

Lưu ý: tham số vị trí này có thể có thứ tự bất kỳ tùy theo quy trình nấu

Hình 2.12: Cơ cấu cấp liệu với 4 hộp nguyên liệu [14]

6. Cơ cấu chấp hành lật khay nguyên liệu **Ingredients_Push**

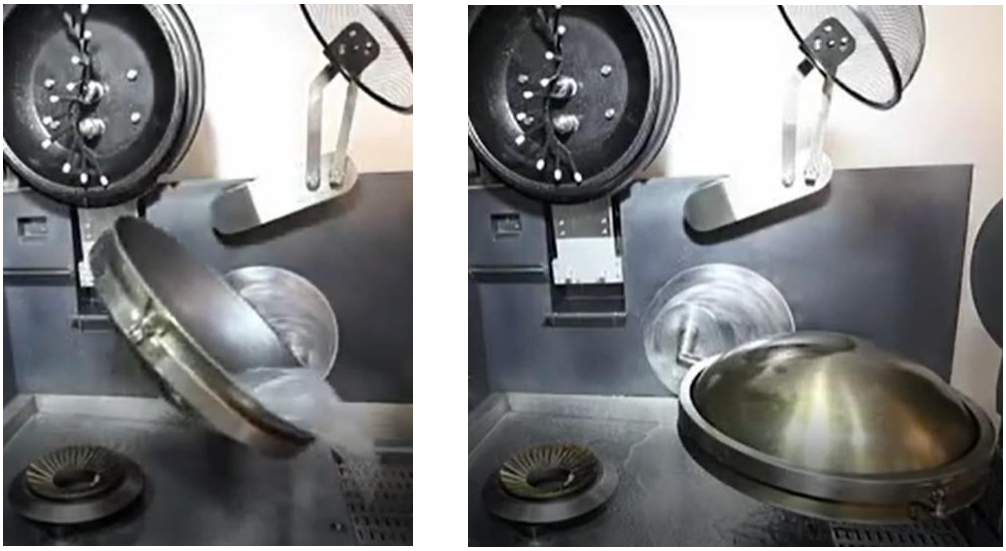
Hình 2.13: Cơ cấu lật khay nguyên liệu [14]

7. Cơ cấu chấp hành tay máy **Actuator_Hand** là sự phối hợp chuyển động của **Motor_PushPull** và **Motor_Sharking** (2 động cơ này không được khảo sát trong đề tài này) để tạo ra 3 thao tác phức tạp là xốc chảo, đổ thức ăn ra dĩa và đổ nước tráng chảo ra ngoài, với tham số

- **mode**: gồm 4 chế độ hoạt động như sau
  sharking_mode=1/ foodout_mode=2/ waterout_mode=3/ washing_mode=4
- **time_sharking**: thời gian xốc
- **time_foodout**: thời gian lấy thức ăn ra dĩa
- **time_waterout**: thời gian đổ nước rữa chảo ra bể
- **time_washing**: thời gian rữa chảo

Hình 2.14: Thao tác cánh tay xốc [14]

Hình 2.15: Thao tác cánh tay lấy thức ăn [14]



Hình 2.16: Thao tác cánh tay vị trí đổ nước rữa chảo [14]

8. Van cấp nước rữa chảo **Valve_Cleaning** định lượng theo thời gian: **t_cleaning**

9. Van cấp nước tráng chảo **Valve_Wasing** định lượng theo thời gian: **t_washing**

Hình 2.17: Thao tác mở van phun nước làm sạch lòng chảo và thanh trộn [14]

2.3.2 Quy trình mẫu nấu một món ăn tự động

Khi nhận mã số món ăn từ hệ thống POS, hệ thống bếp nấu tự động thực hiện điều khiển nấu ăn trình tự hoạt động như sau:

//**********

// *Khởi động quy trình nấu*

//**********

(0) **Bước 0 (S0):** Khởi tạo các biến liên quan đến điều khiển quy trình nấu,

(1) **Bước 1 (S1):** Nhấn nhút **START,** bật van định lượng cấp dầu **Valve_Oil** với liều lượng thời gian **t_oil_1**

(2) **Bước 2 (S2):** SET cơ cấu tác động hệ thống gia nhiệt **Actuator_Temp** với nhiệt độ **T_1** trong thời gian 2 giây (để đạt nhiệt độ mong muốn)

//**********

// *Cấp nguyên liệu 1 và thực hiện thao tác trộn*

//**********

(3) **Bước 3 (S3):** OUT cơ cấu chấp hành cấp liệu **Actuatua_Ingredients** với tham số **ingredient_1** trong thời gian 3 giây

(4) **Bước 4 (S4):** SET cơ cấu chấp hành **Tag_Ingredients_Push** lật khay nguyên liệu và RESET trong thời gian 2 giây

(5) **Bước 5 (S5):** SET **Actuator_Cover** đóng nắp lần 1 trong thời gian 2 giây

(6) **Bước 6 (S6):** SET **Motor_ Blending** trộn lần 1 với tham số

- Tốc độ **speed_blending_1**
- Thời gian **time_blending_1**

RESET **Motor_ Blending** dừng trộn lần 1 trong thời gian 2 giây

(7) **Bước 7 (S7):** RESET **Actuator_Cover** mở nắp lần 1 trong thời gian 2 giây

//**********

// *Cấp nguyên liệu 2 và thực hiện thao tác trộn*

//**********

(8) **Bước 8 (S8):** OUT cơ cấu chấp hành cấp liệu **Actuator_Ingredients** với tham số **ingredient_2** trong thời gian 3 giây

(9) **Bước 9 (S9):** SET cơ cấu chấp hành **Ingredients_Push** lật khay nguyên liệu trong và RESET trong thời gian 2 giây

(10) **Bước 10 (S10):** SET **Actuator_Cover** đóng nắp trong thời gian 2 giây

(11) **Bước 11 (S11):** SET **Motor_ Blending** trộn lần 2 với tham số

- Tốc độ **speed_blending_2**
- Thời gian **time_blending_2**

RESET **Motor_ Blending** dừng trộn lần 2 trong thời gian 2 giây

(12) **Bước 12 (S12):** RESET **Actuator_Cover** mở nắp lần 2 trong thời gian 2 giây

//**********

// *Cấp nguyên liệu 3 và thực hiện thao tác trộn*

//**********

(13) **Bước 13 (S13):** OUT cơ cấu chấp hành cấp liệu **Actuatua_Ingredients** với tham số **ingredient_3** trong thời gian 3 giây

(14) **Bước 14 (S14):** SET cơ cấu chấp hành **Ingredients_Push** lật khay nguyên liệu trong thời gian 2 giây và RESET trong thời gian 2 giây

(15) **Bước 15 (S15):** SET **Actuator_Cover** đậy nắp lần 3 trong thời gian 2 giây

(16) **Bước 16 (S16):** SET **Motor_ Blending** trộn lần 3 với tham số

- Tốc độ **speed_blending_3**
- Thời gian **time_blending_3**

RESET **Motor_ Blending** dừng trộn lần 3 trong thời gian 2 giây

(17) **Bước 17 (S17):** RESET **Actuator_Cover** mở nắp lần 3 trong thời gian 2 giây

//**********

// *Cấp nguyên liệu 4 và thực hiện thao tác trộn*

//**********

(18) **Bước 18 (S18):** OUT cơ cấu chấp hành cấp liệu **Actuatua_Ingredients** với tham số **ingredient_4** trong thời gian 3 giây

(19) **Bước 19 (S19):** SET cơ cấu chấp hành **Ingredients_Push** lật khay nguyên liệu trong thời gian 2 giây và RESET trong thời gian 2 giây

(20) **Bước 20 (S20):** SET **Actuator_Cover** đậy nắp lần 4 trong thời gian 2 giây

(21) **Bước 21 (S21):** SET **Motor_ Blending** trộn lần 4 với tham số

- Tốc độ **speed_blending_4**
- Thời gian **time_blending_4**

RESET **Motor_ Blending** dừng trộn lần 4 trong thời gian 2 giây

(22) **Bước 22 (S22):** RESET **Actuator_Cover** mở nắp lần 4 trong thời gian 2 giây

//**********

// *Thao tác xốc thức ăn*

//**********

(23) **Bước 23 (S23):** SET van định lượng cấp dầu **Valve_Oil** với liều lượng thời gian **t_oil_2**

(24) **Bước 24 (S24):** SET cơ cấu tác động hệ thống gia nhiệt **Actuator_Temperature** với nhiệt độ **T_2**

(25) **Bước 25 (S25):** SET cơ cấu chấp hành **Actuator_Hand** với tham số

- Mode s**harking_mode**
- Thời gian **time_sharking**

RESET cơ cấu chấp hành **Actuator_Hand** dừng xốc trong thời gian 2 giây

(26) **Bước 26 (S26):** RESET cơ cấu tác động hệ thống gia nhiệt **Actuator_Temperature** tắt gas trong vòng 2 giây

//**********

// Thao tác đổ thức ăn ra dĩa

*//**********

(27) **Bước 27 (S27):** SET cơ cấu chấp hành **Actuator_Hand** với tham số

- Mode **foodout_mode**
- Thời gian **time_foodout**

RESET cơ cấu chấp hành **Actuator_Hand** về vị trí **Home** trong thời gian 2 giây

*//**********

// Thao tác rữa lòng chảo và thanh trộn

*//**********

(28) **Bước 28 (S28):** SET **Actuator_Cover** đóng nắp lần 5 trong thời gian 2 giây

(29) **Bước 29 (S29):** SET van nước rửa chảo **Valve_Cleaning** với thời gian **t_cleaning**

(30) **Bước 30 (S30):** SET **Motor_ Blending** trộn lần 5 trong thời gian 5 giây

RESET **Motor_ Blending** dừng trộn lần 5 trong thời gian 2 giây

(31) **Bước 31 (S31):** RESET **Actuator_Cover** mở nắp trong thời gian 2 giây

*//**********

// Thao tác đổ nước rữa chảo ra ngoài và trán lại lòng chảo

*//**********

(32) **Bước 32 (S32):** SET cơ cấu chấp hành **Actuator_Hand** với tham số

- Mode **waterout_mode** (đổ nước ra ngoài)
- Thời gian **time_waterout**

RESET cơ cấu chấp hành **Actuator_Hand** về Home trong thời gian 2 giây

(33) **Bước 33 (S33):** SET cơ cấu chấp hành **Actuator_Hand** với tham số

- Mode **wasing_mode** (úp chảo và tráng nước cho chảo)

- Thời gian **time_wasing**

SET van nước tráng chảo **Valve_Wasing** với thời gian **t_wasing**

(34) **Bước 34 (S34):** RESET cơ cấu chấp hành **Actuator_Hand** về vị trí **Home** ban đầu trong thời gian 2 giây

//**********

// Kết thúc quy trình nấu

//**********

2.4 Thống kê thiết bị trong bài toán

Với trình tự thực hiện như mục trên danh sách các biến được thống kê và cấp phát tương ứng trên PLC và trên MX-OPC (xem bảng 2.1). Trong bảng 2.1, các các biến M0-M8 (tô màu xanh trong bảng) được liên kết với các cổng ra của PLC Y0-Y07 và Y10 để kích trực tiếp các cơ cấu chấp hành. Các biến này được giám sát trên hệ thống để biết được trạng thái các kích hoạt các cơ cấu chấp hành của hệ thống.

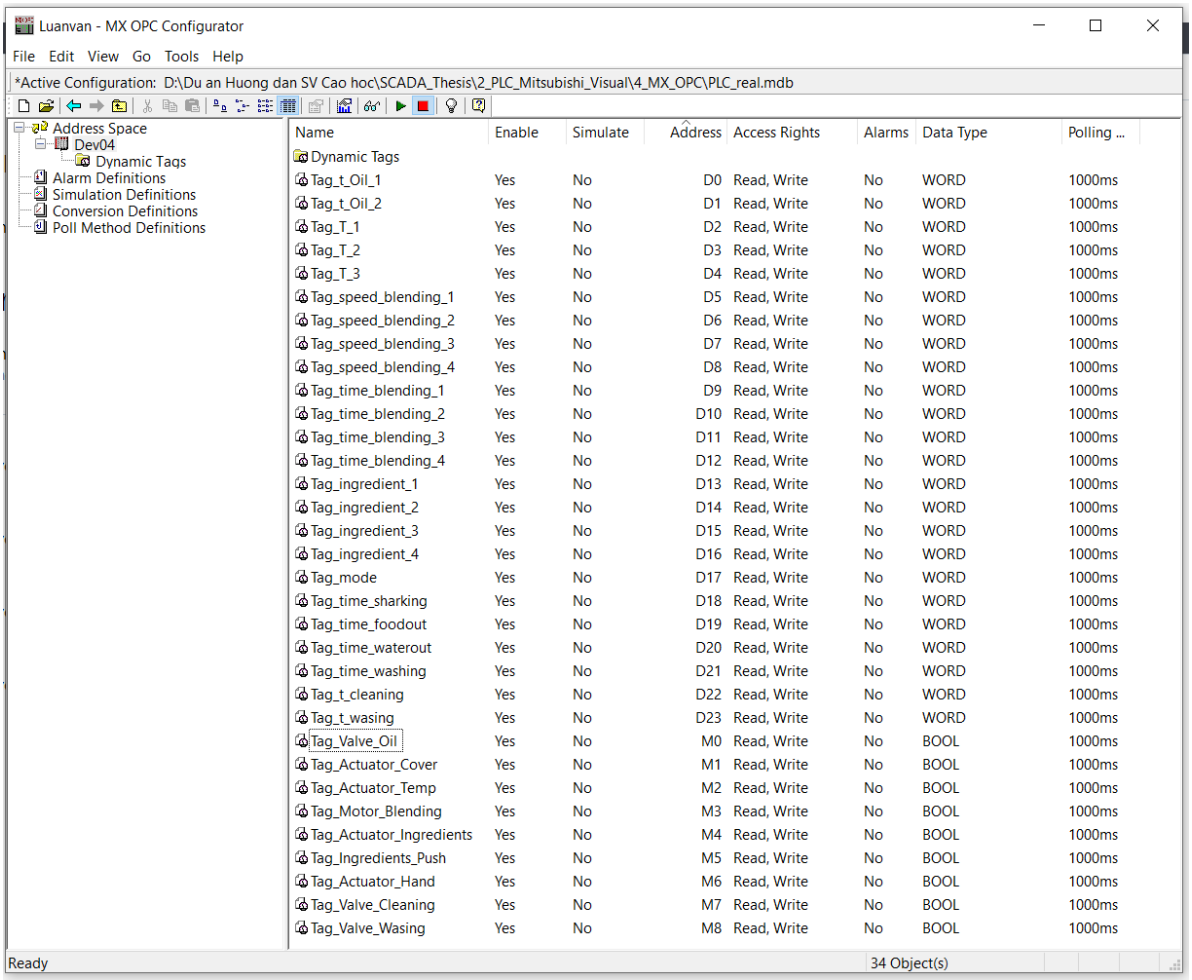
Bảng 2.1: Bảng cấp phát I/O trên PLC và cấp phát Tag trên MX-OPC

TT	Tên thiết bị	Định dạng	Địa chỉ	Tag (MX OPC)	Ghi chú
1	Valve_Oil	bool	M0→Y0	Tag_Valve_Oil	Van cấp dầu định lượng theo thời gian
2	t_oil_1	WORD	D0	Tag_t_Oil_1	Thời gian châm dầu cho chế độ trộn
3	t_oil_2	WORD	D1	Tag_t_Oil_2	Thời gian châm đầu cho chế độ lắc chảo
4	Actuator_Cover	bool	M1→Y1	Tag_Actuator_Cover	Đóng/mở nắp chảo
5	Actuator_Temp	bool	M2→Y2	Tag_Actuator_Temp	Đóng/mở bộ gia nhiệt
6	T_1	WORD	D2	Tag_T_1	Nhiệt độ nấu ăn 1
7	T_2	WORD	D3	Tag_T_2	Nhiệt độ nấu ăn 2
8	T_3	WORD	D4	Tag_T_3	Nhiệt độ nấu ăn 3 (dự phòng)
9	Motor_Blending	bool	M3→Y3	Tag_Motor_Blending	Đóng/mở động cơ trộn
10	speed_blending_1	WORD	D5	Tag_speed_blending_1	Tốc độ động cơ trộn nguyên liệu 1

11	speed_blending_2	WORD	D6	Tag_speed_blending_2	Tốc độ động cơ trộn nguyên liệu 2
12	speed_blending_3	WORD	D7	Tag_speed_blending_3	Tốc độ động cơ trộn nguyên liệu 3
13	speed_blending_4	WORD	D8	Tag_ speed_blending_4	Tốc độ động cơ trộn nguyên liệu 4
14	time_blending_1	WORD	D9	Tag_time_blending_1	Thời gian trộn nguyên liệu 1
15	time_blending_2	WORD	D10	Tag_time_blending_2	Thời gian trộn nguyên liệu 2
16	time_blending_3	WORD	D11	Tag_time_blending_3	Thời gian trộn nguyên liệu 3
17	time_blending_4	WORD	D12	Tag_time_blending_4	Thời gian trộn nguyên liệu 4
18	Actuator_Ingredients	bool	M4→Y4	Tag_Actuator_Ingredients	Đóng/mở cơ cấu cấp liệu
19	ingredient_1	WORD	D13	Tag_ingredient_1	Vị trí nguyên liệu 1
20	ingredient_2	WORD	D14	Tag_ingredient_2	Vị trí nguyên liệu 2
21	ingredient_3	WORD	D15	Tag_ingredient_3	Vị trí nguyên liệu 3
22	ingredient_4	WORD	D16	Tag_ingredient_4	Vị trí nguyên liệu 4
23	Ingredients_Push	bool	M5→Y5	Tag_Ingredients_Push	Đóng/mở lật khay nguyên liệu
24	Actuator_Hand	bool	M6→Y6	Tag_Actuator_Hand	Đóng/mở 3 thao tác tay cho chảo
25	mode	WORD	D17	Tag_mode	Chế độ thao tác: xốc, lấy thức ăn, rữa
26	time_sharking	WORD	D18	Tag_time_sharking	Thời gian xốc chảo
27	time_foodout	WORD	D19	Tag_time_foodout	Thời gian lấy thức ăn ra dĩa
28	time_waterout	WORD	D20	Tag_time_waterout	Thời gian đổ nước rữa chảo ra bể
29	time_washing	WORD	D21	Tag_time_washing	Thời gian úp chảo tráng nước lòng chảo
30	Valve_Cleaning	bool	M7→Y7	Tag_Valve_Cleaning	Đóng/mở van nước rữa chảo
31	t_cleaning	WORD	D22	Tag_t_cleaning	Thời gian châm nước rữa chảo
32	Valve_Wasing	bool	M8→Y10	Tag_Valve_Wasing	Đóng/mở van nước tráng chảo
33	t_wasing	WORD	D23	Tag_t_wasing	Thời gian châm nước tráng chảo

Do PLC Mitsubishi FX3U-32M được sử dụng trong để tài không có mô-đun truyền thông LAN để phần mềm KepserverEX giao tiếp lấy dữ liệu trực tiếp với PLC

nên các biến I/O trên PLC đó được giám sát qua một phần mềm trung gian OPC khác là Mitsubishi MX-OPC và các biến Tag được khai báo và cài đặt vào MX-OPC như trên *Hình 2.18*.



Hình 2.18: Các biến Tag tương ứng được cài đặt trên phần mềm MX-OPC

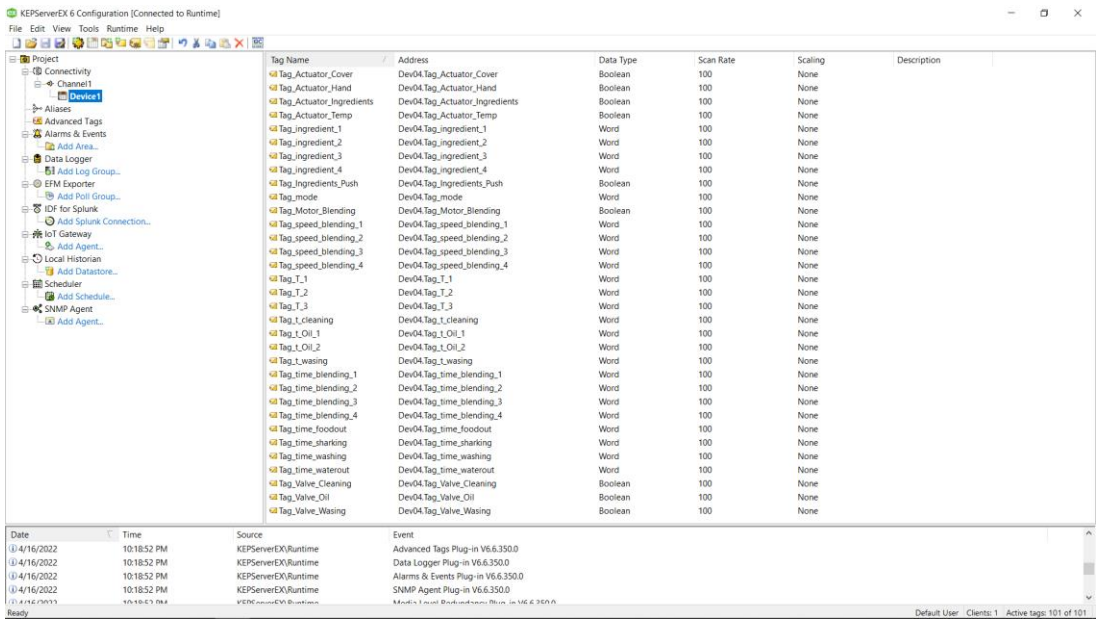
2.5 Lập trình điều khiển trên PLC

Với thiết kế quy trình điều khiển trình tự của máy nấu ăn tự động trong mục 2.2.2. chương trình PLC điều khiển trình tự chuyên dùng dùng cơ chế Step Ladder (STL) được lập trình và cài đặt.

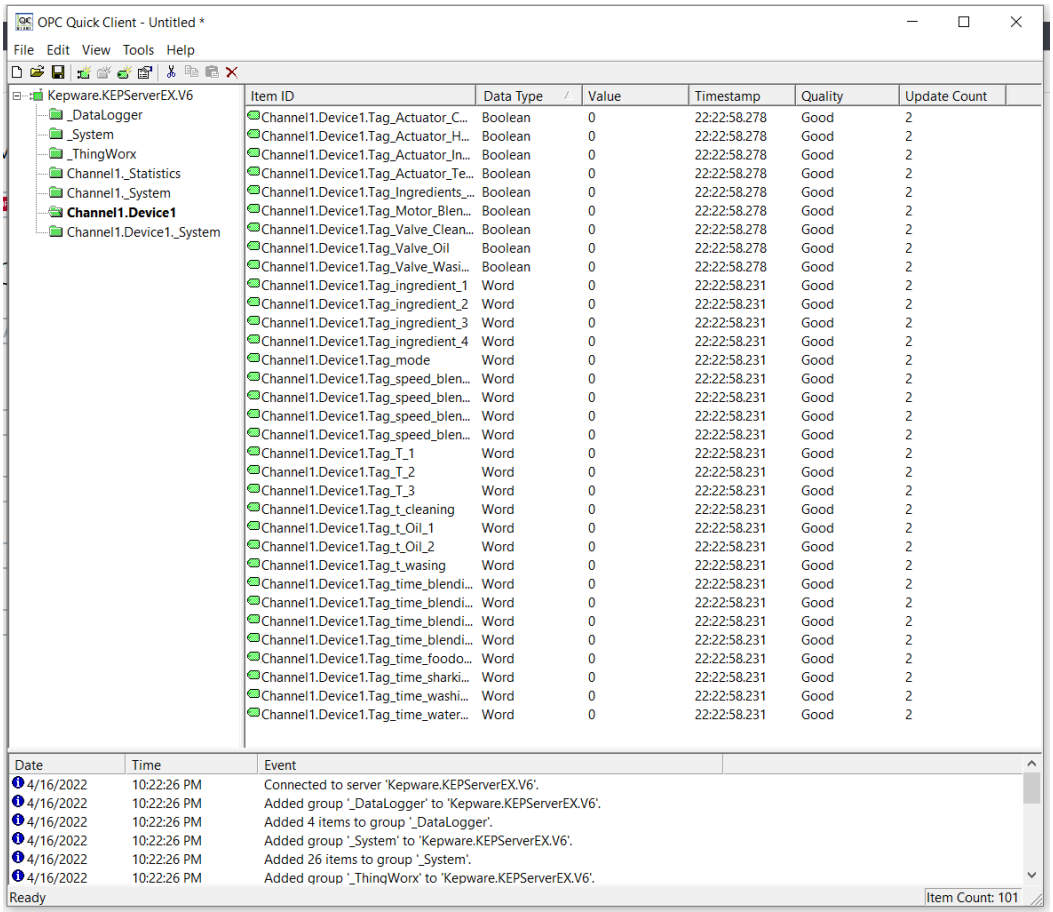
2.6 Khai báo Tag trong KepserverEX

Các biến Tag được khai báo tương ứng trên OPC KepserverEX để truy xuất các biến trên PLC, Các biến này được dùng để điều khiển và giám sát hệ thống hoạt

động. Khai báo Tag được cho trên (*Hình 2.19)*. Trạng thái kết nối với PLC (Quick Client) được cho trên (*Hình 2.20)*.



Hình 2.19: Các biến Tag tương ứng được cài đặt trên phần mềm KepserverEX



Hình 2.20: Các biến Tag tương ứng được hiển thị trên Quick client KepserverEX

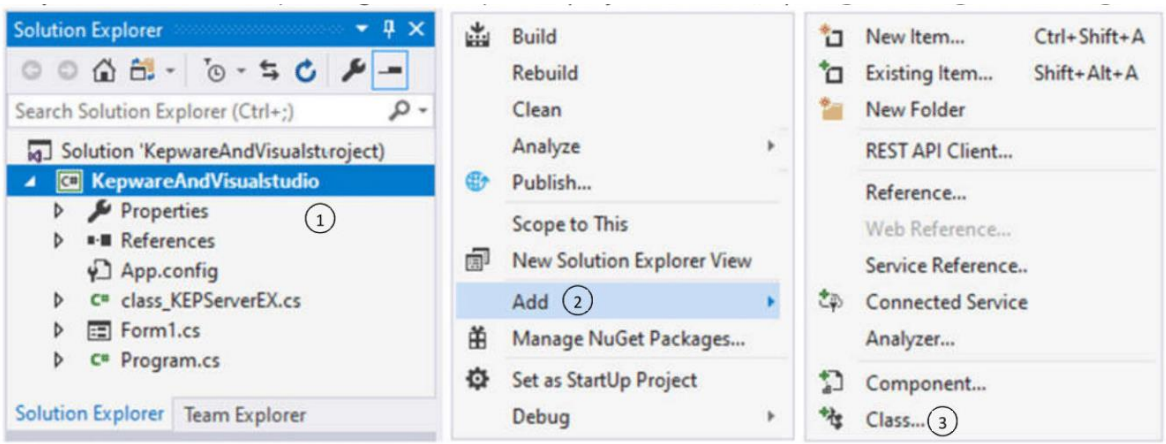
CHƯƠNG 3: HỆ THỐNG ĐIỀU KHIỂN VÀ THU NHẬN DỮ LIỆU

3.1 Thiết kế và cài đặt tác vụ truy xuất thông tin

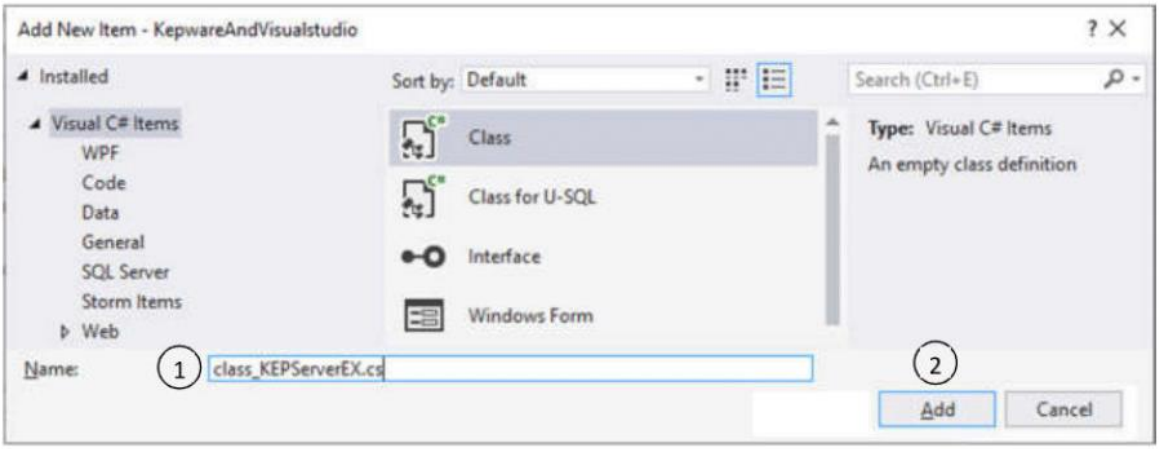
3.1.1 Tạo class "class_KEPServerEX.cs"

Class này là dùng khai báo chương trình con kết nối dữ liệu. Cũng như khi lập trình PLC, để rút gọn chương trình chính hoặc một số chức năng có thể dùng chung, người lập trình sẽ sử dụng các chương trình con để rút rút gọn và giảm dung lượng chương trình cũng như nâng cao tính logic cho chương trình PLC, từ đó nâng cao hiệu suất chương trình.

Bước 1: Thêm mới class



Bước 2: Đặt tên class và nhấn Add



Bước 3: Chương trình "class_KEPServerEX.cs"

Xin xem **PHỤ LỤC P2.1**

Bảng 3.1: Giải thích chương trình "class_KEPServerEX.cs"

1	public static string[] tagread(int tagnumber)
Chương trình con khai báo tag cho project, trong đó đầu vào của chương trình con là "tagnumber" với định dạng dữ liệu là int, số lượng tag trong projects, số lượng tag trong chương trình con này sẽ thay đổi theo project cụ thể khi lập trình giao diện Visual studio.	
2	string tagID_l = "Channell.Devicel.tagNamel"; string tagID_2 = "Channell.Devicel.tagName2"; string tagID_3 = "Channell.Devicel.tagName3"; string tagID_4 = "Channell.Devicel.tagName4";
Thực hiện khai báo tag với ID là 1 (taglD_l), đường dẫn của tag trong KEPServerEX sẽ là "Channel.Device.tên tag", trong trường hợp này tên channel là	

"Channel1", tên device là "Device1" và tên tag là "tag_Bool", ghép 3 thành phần này lại với nhau ta có ID tag đã khai báo trong KEPServerEX.	
3	string[] tags;
Tạo mảng (array) có tên "tags" với định dạng là String	
4	tags = new string[tagnumber];
Khai báo array "tags" với độ dài là số lượng tag của project "tagnumber".	

5	tags.SetValue(tagID_l, 1);
	tags.SetValue(tagID_2, 2);
	tags.SetValue(tagID_3, 3);
	tags.Setvalue(tagID_4, 4);

Đưa tên tag "taglD_l" vào mảng "tags" ở vị trí 1.	
6	public static Int32[] tagID(int tagnumber)
Chương trình này là mặc định cho toàn bộ project, do đó không cần sửa khi tạo các project khác nhau, mục đích của chương trình con này là tạo ra một mảng (array) ID của các tag đã khai báo để đưa vào OPC nhằm kết nối với KEPServerEX.	

3.1.2 Lập trình cho Form chính

Bước 1: Sử dụng hàm using thêm "OPCAutomation" vào projects

```
using OPCAutomation; //Thư viện OPC
```

Bước 2: Khai báo kết nối và định nghĩa các thuộc tính cho OPC

Xin xem **PHỤ LỤC 2.2** cho chương trình khai báo OPC

Bảng 3.2: Giải thích các khai báo OPC trong Form1

<table>
<tr><td></td><td></td></tr>
<tr><td>1</td><td>static int tagNumber = 69;</td></tr>
<tr><td colspan="2">Khai báo một biến số nguyên tĩnh (static integer) có tên "tagNumber" lưu số lượng tag trong project, trong trường hợp này số lượng tag là 4, vì ở class "class_KEPServerEX.cs" đã khai báo 4 tag ID (Lưu ý: Số lượng tag này phải khớp với số lượng tag ID đã liệt kê trong "class_KE PServerEX.es")</td></tr>
<tr><td>2</td><td>static int PLCscantime = 1000;</td></tr>
<tr><td colspan="2">Khai báo thời gian quét dữ liệu, biến có tên "PLCscantime", ở đây 1000 tương ứng với 1s, có thể tăng thời gian cập nhật dữ liệu bằng cách thay đổi giá trị millisecond (mili giây).</td></tr>
<tr><td>3</td><td>// Gọi các kết nối OPC
public OPCAutomation.OPCServer AnOPCServer;
public OPCAutomation.OPCServer OPCServer;
public OPCAutomation.OPCGroups OPCGroup;
public OPCAutomation.OPCGroup PLC;
public string Groupname;</td></tr>
<tr><td colspan="2">Khai báo các biến OPC Server và OPC group ở dạng Public, sử dụng để gọi hàm kết nối với KEPServerEX.</td></tr>
</table>

4	static int arrlength = tagNumber + 1;

Khai báo lấy độ dài mảng (array) cho các mảng khi kết nối dữ liệu với OPC.

5	Array OPtags = class_KEPServerEX.tagread(arrlength);

Lấy dữ liệu danh sách tag từ "class_KEPServerEX.cs" và đưa vào biến array "OPtags".

6	Array tagID = class_KEPServerEX.tagID(arrlength);

Lấy dữ liệu tag ID từ "class_KEPServerEX.cs" và đưa vào biến array "taglD".

7	Array WriteItems = Array.CreateInstance(typeof(object), arrlength); Array tagHandles = Array.CreateInstance(typeof(Int32), arrlength); Array OPCError = Array.CreateInstance(typeof(Int32), arrlength); Array dataType = Array.CreateInstance(typeof(Int16), arrlength); Array AccessPaths = Array.CreateInstance(typeof(string), arrlength);

Tạo các Array thực hiện chức năng truy xuất dữ liệu từ KEPServerEX.

Bước 3: Tạo chương trình con kết nối OPC KEPServerEX

Bảng 3.3: Giải thích chương trình kết nối với OPC

1	private void KEPServerEX_Connect()
Khai báo chương trình con "KEPServerEX_Connect", chương trình con này sẽ được gọi ở hàm form load, thực hiện việc khởi tạo kết nối với KEPServerEX.	
2	string IOServer = "Kepware.KEPServerEX.V6";
Khai báo tên phiên bản phần mềm Kepware (KEPServerEX), ở đây tên phiên bản phần mềm là "Kepware.KEPServerEX.V6", nếu các phiên bản khác của phần mềm KEPServerEX thì cần thay đổi V6 thành Vx (x là tên phiên bản của KEPServerEX).	
3	string IOGroup = "OPCGroup1";
Khai báo một OPC group (cố định)	
4	OPCServer = new OPCAutomation.OPCServer();
Sử dụng hàm thư viện "OPCAutomation" tạo mới một OPCServer (cố định).	
5	OPCServer.Connect(IOServer, "");

Thực hiện câu lệnh kết nối tới OPC Server (cố định).	
6	PLC = OPCServer.OPCGroups.Add(IOGroup);
Tạo một kết nối có tên group đã tạo (cố định).	
7	PLC.DataChange += new DIOPCGroupEvent_DataChangeEventHandler(dataScan);
Khi đọc dữ liệu từ OPC nếu giá trị tag thay đổi thì move dữ liệu vào chương trình con "dataScan" (chương trình con này sẽ sử dụng khi đọc/ghi dữ liệu từ KEPServerEX.	
8	PLC.UpdateRate = PLCscantime;
Khai báo thời gian cập nhật dữ liệu từ Kepware (cố định).	
9	PLC.IsSubscribed = PLC.IsActive;
Kiểm tra đã kết nối với Kepware chưa (cố định).	
10	PLC.OPCItems.DefaultIsActive = true;
Khai báo giá trị default của PLC là active (cố định).	

11	PLC.OPCItems.AddItems(tagNumber, ref OPtags, ref tagID, out tagHandles, out OPCError, dataType, AccessPaths);
Thực hiện đọc dữ liệu từ Kepware lên nhờ câu lệnh "Addltems" (cố định).	

Bước 4: Đọc dữ liệu tag từ KEPServerEX và hiển thị giá trị tag lên textbox

Tham khảo **PHỤ LỤC 2.3** để xem chương trình con "dataScan"

Bảng 3.4: Các biến trong chương trình con đọc dữ liệu từ KEPServerEX (dataScan)

No.	Tên biến	Giải thích
1	int ID	Biến số nguyên tên ID
2	int Numitems	Biến số nguyên số lượng tag tên Numltems
3	ref Array tagID	Biến mảng (Array) chứa ID tag tên TaglD
4	Ref Array ItemValues	Biến mảng (Array) chứa giá trị tag đọc lên tên ItemValues
5	ref Array Qualities	Biến màng (Array) chứa chất lượng tag tên Qualities
6	ref Array Timestamps	Biến mảng (Array) chứa thời gian trích mẫu tên TimeStamps

Bảng 3.5: Giải thích chương trình đọc dữ liệu từ PLC thông qua Tag

1	for (int ỉ = 1; i <= Numitems; i++);

Thực hiện vòng lặp cho tag từ 1 đến tổng số lượng tag của project (mặc định).	
2	int getTagID = Convert.ToInt32(tagID.GetValue(i));
Khai báo biến có tên "getTaglD" lấy dữ liệu ID của tag từ mảng "taglD" đồng thời chuyển đổi sang integer 32bit (mặc định)	
3	string tagValue = ItemValues.GetValue(i)?.ToString();
Khai báo biến có tên "tagValue" lấy giá trị từ mảng "ItemValues", đây là mảng chứa giá trị tag đọc lên từ KEPServerEX (mặc định). Lưu ý có dấu hỏi sau GetValue(i) mục đích để tránh lỗi phần mềm khi chưa kết nối được với KEPServerEX	
4	if (getTaglD == 1) { textBoxl.Text = tagvalue; };
Nếu giá trị biến getTaglD = 1 (tức là tag có ID 1, đã khai báo tag ID ở class "class_KEPServerEX.cs") thì đưa dữ liệu giá trị tag "tagValue" vào textboxl. Tương tự cho các tag ID khác.	

3.2 Tính năng WatchDog

3.2.1 Watchdog là gì?

Watchdog sử dụng để giám sát trạng thái kết nối giữa Visual Studio và PLC, nếu trong trường hợp quá thời gian cài đặt mà không có kết nối giữa PLC và Visual

studio thì phần mềm sẽ tự động connect lại cho đến khi trạng thái kết nối trở lại bình thường.

3.2.2 Nguyên tắc thực hiện

Ở PLC tạo 1 bộ đếm, bộ đếm này thực hiện chức năng Watchdog, bộ đếm sẽ đếm lặp lại từ 0-20, trong điều kiện kết nối bình thường thì Visual studio nhận dạng được bộ đếm luôn tăng sau mỗi giây tức là kết nối ổn định, trong một trường hợp nào đó dẫn đến Visual studio không thể kết nối được với PLC nữa, tức là lúc này ở phía visual studio bộ đếm không tăng, phần mềm sẽ phán đoán kết nối bị mất, lúc này phần mềm hiển thị cảnh báo mất kết nối, dễ dàng giúp người vận hành phát hiện và kiểm tra khắc phục sự cố.

OPC có một ưu điểm là tự động kết nối sau khi có kết nối lại, do đó không cần lập trình chương trình kết nối lại mà chỉ hiển thị cảnh báo kết nối bị mất hoặc có kết nối.

3.2.3 Cài đặt Watchdog

Bước 1: Lập trình trên PLC

Hình 3.1: Chương trình tạo bộ đếm cho tính năng WatchDog

Cờ M8013 là xung Clock l Hz, do đó ở chương trình này thì mỗi giây bộ đếm C0 sẽ tăng giá trị lên 1 và đếm đến giá trị 20. Bộ đếm được mặc định là làm đếm xung cạnh lên. Khi C0 đặt giá trị 20 thì logic C0 bằng 1 có tác dụng rest lại bộ nhớ C0 đếm trở lại ban đầu. Giá trị đếm của C0 được lưu vào thanh ghi

Bước 2: Thêm tag trong KEPServerEX

Thêm 1 tag có tên "Watchdog" đường dẫn tag "Channel1.Device1.Tag_WatchDog”.

Bước 3: Lập trình trên Visual Studio

Visual studio giám sát giá trị đọc về Watchdog, nếu trong thời gian đặt trạng thái kết nối bình thường thì nút nhấn "Connect" hiển thị màu xanh, khi mất kết nối thì nút nhấn hiển thị màu đỏ.

Bảng 3.6: Giải thích chương trình hiển thị trạng thái Watchdog

1	public static void WatchdogStatus(Button btt, string valnow)
Chương trình con **Watchdogstatus** với các biến đầu vào là tên nút nhấn (Button btt) và giá trị bộ đếm watch dog đọc lên từ PLC (string val_now). Chương trình này sẽ tiến hành so sánh một giá trị biến string có tên là valold với giá trị watch dog đọc về nếu 2 giá trị này khác nhau tức là watch dog vẫn đang đếm. Nếu valold khác valnow thì cho nút nhấn có nền màu xanh chữ màu trắng, ngược lại nếu valold bằng valnow tức là bộ đếm không đếm nữa, đã bị mất kết nối với PLC, lúc này cho nút nhấn hiển thị nền màu đỏ và chữ màu trắng.	

Thêm một Timer có tên là Timer_Watchdog với Interval = 3000 (3 giây). Event Tick của Timer như sau:

```
string Watchdog_value = "0";
private void Timer_Watchdog_Tick(object sender, EventArgs e)
{
  class_Watchdog.WatchdogStatus(toolStripStatusLabel1, Watchdog_value);            }
```

Bảng 3.7: Sử dụng chương trình hiển thị trạng thái Watchdog

1	string Watchdog_value = "0";
Tạo một biến định dạng string có tên Watch dog_value lưu giá trị tag watchdog, giá trị khởi tạo của biến này là "0".	
2	class_Watchdog.WatchdogStatus(toolStripStatusLabel1, Watchdog_value);
Gọi chương trình con "WatchdogStatus" trong class "class_Watchdog.cs" với biến đầu vào là nút nhấn connect (ID = bttConnect) và giá trị thực tế của tag watchdog.	

Xin xem **PHỤ LỤC 2.4** cho “class_Watchdog.cs”

Tại mục đọc về giá trị tag của PLC gán giá trị tag watchdog đọc về vào biến vừa tạo "Watchdog_value".

```
    case 34:
Watchdog_value = tagValue;
break;
```

Bảng 3.8: Giải thích chương trình cập nhật trạng thái Watchdog

1	case 34:
Lấy giá trị tag ID của watch dog đã khai báo ở class "class_KEPServerEX.cs" ID của tag này là 34.	
2	Watchdog_value = tagValue;
Gán giá trị tag đọc lên từ PLC sang biến "Watchdog_value".	

3.3 Tính năng quản lý người dùng

3.3.1 Hoạt động tính năng quản lý người dùng

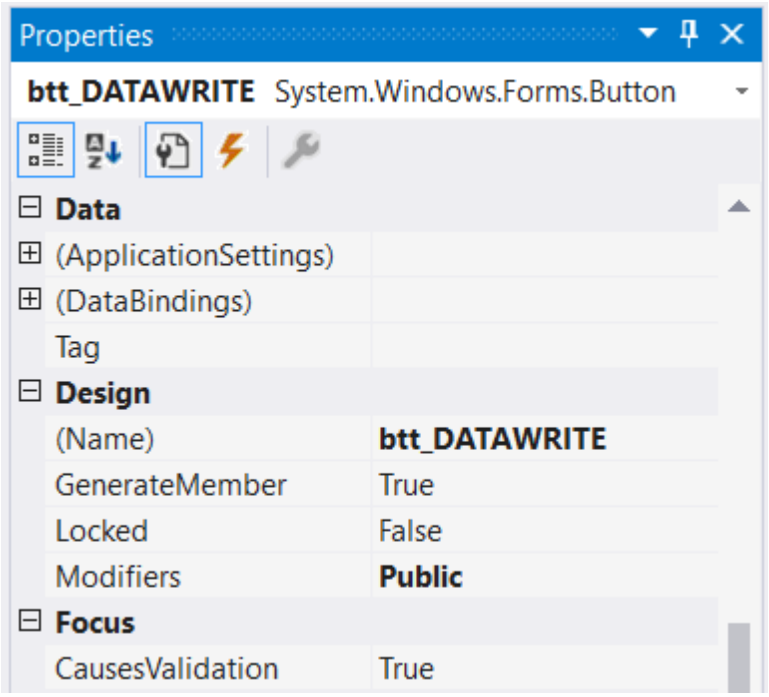
Người dùng có thể phân thành nhiều cấp khác nhau, nhưng trong đề tài giới thiệu 2 cấp, cấp quản lý và cấp thao tác giám sát. Xem bảng bên dưới

- Cấp quản lý (Administrator): có toàn quyền vận hành hệ thống
- Cấp giám sát (Operator): chỉ được thao tác các nút nhấn điều khiển vận hành, không được ghi dữ liệu
- Chưa đăng nhập: chỉ được xem, không được thao tác, không được ghi dữ liệu

3.3.2 Nguyên lý thực hiện

Bước 1: Chuyển các thành phần control cần quản trị sang chế độ Public.

Click vào toàn bộ các thành phần control cần quản trị trong Winform » chọn properties » Design » Modifiers chuyển thành public.



Bước 2: Tạo class chương trình con có tên "class_Login.cs" với nội dung như sau:

Xin xem **PHỤ LỤC P2.5**

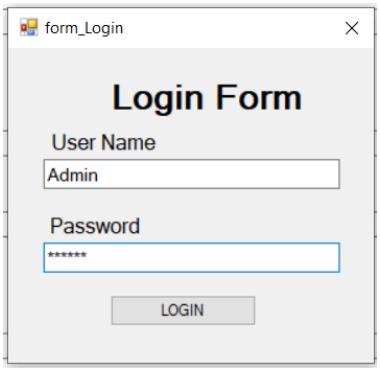
Bảng 3.9: Giải thích chương trình con quản lý người dùng

1	public void Not_Login ()
Chương trình con khi người dùng chưa đăng nhập hoặc đã logout, người dùng chưa đăng nhập nên thực hiện disable toàn bộ nút nhấn	

2	frm.btt_DATAWRITE.Enabled = false;
	frm.btt_Auto.Enabled = false;
	frm.btt_Manu.Enabled = false;
	frm.btt_Init.Enabled = false;
	frm.btt_Start.Enabled = false;
	frm.btt_Stop.Enabled = false;
	frm.btt_Y0.Enabled = false;
	frm.btt_Y1.Enabled = false;
	frm.btt_Y2.Enabled = false;
	frm.btt_Y3.Enabled = false;
	frm.btt_Y4.Enabled = false;
	frm.btt_Y5.Enabled = false;
	frm.btt_Y6.Enabled = false;
	frm.btt_Y7.Enabled = false;
	frm.btt_Y8.Enabled = false;

Disable nút nhấn có tên **btt_DATAWRITE** trong Form1 khi chưa đăng nhập. Tương tự với các nút nhấn khác	
3	public void admin_Control_Elements()
Chương trình con khi người dùng đăng nhập bằng quyền Admin, người dùng có quyền thao tác toàn bộ chức năng của hệ thống.	
4	public void Operator Control Elements()

Chương trình con khi người dùng đăng nhập bằng quyền điều khiển (Operator), người dùng có quyền thao tác một số chức năng



Hình 3.2: Cửa sổ đăng nhập

Trên cửa sổ đang nhập, nếu người dùng nhập vào đúng tên Admin và password (123456) thì gọi chương trình con login bằng quyền admin (fn_login.admỉn_Control_Elements();), nếu người dùng nhập vào tên Userl hoặc User2 và đúng password thì gọi chương trình con đăng nhập cho người vận hành (fn_login.Operator_Control_Elements();), nếu sai tên người dùng hoặc mật khẩu thì hiện thông báo "Sai tên người dùng hoặc mật khẩu".

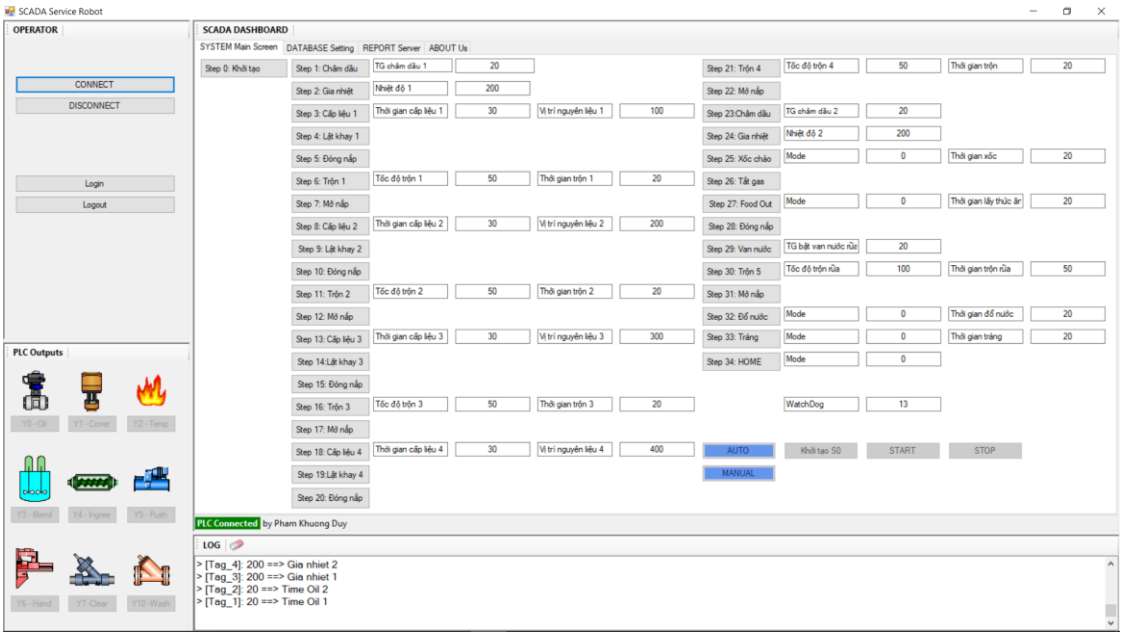
3.4. Các kết quả thử nghiệm

Xây dựng một I/O server dùng C# với các thiết kế class quan trọng đặc trưng cho 1 hệ thống SCADA như tính năng WatchDog, tính năng quản lý người dùng và một hàm quan trọng để đọc các Tags trong phần mềm Kepserver-EX là DataScan.

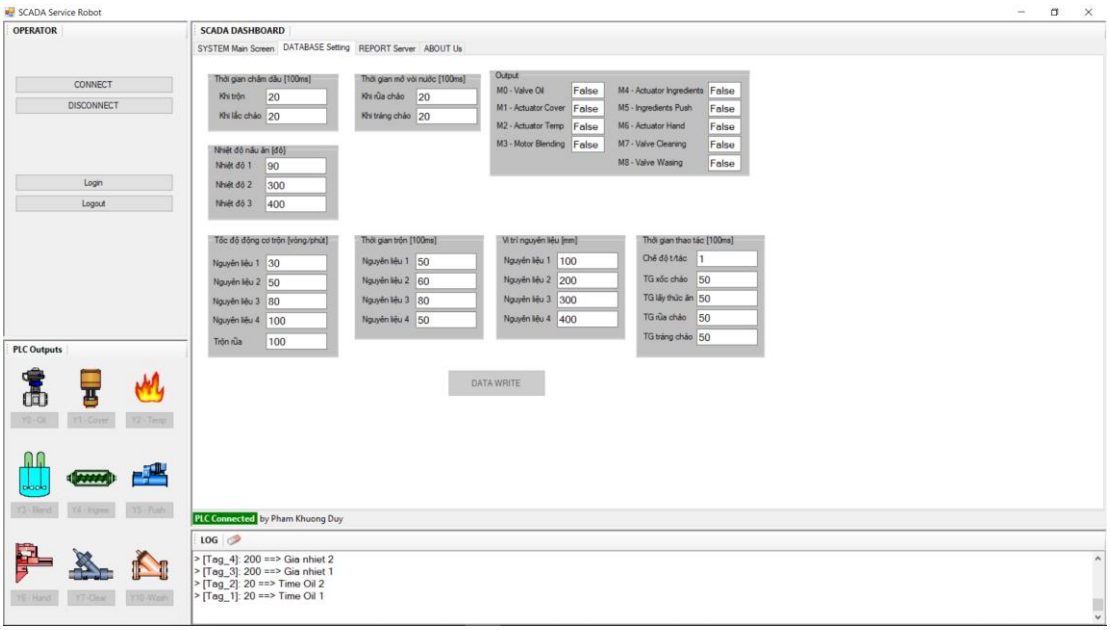
3.4.1 Kết quả hiển thị lên Window form (Visual Studio)

Thông qua các các khai báo Tag trên phần mềm OPC, hoạt động thu nhận dữ liệu thông qua chương trình con "datascan", hoạt động điều khiển thông qua chương trình con ghi vào biến PLC, cùng các tính năng Watchdog và quản lý người dùng

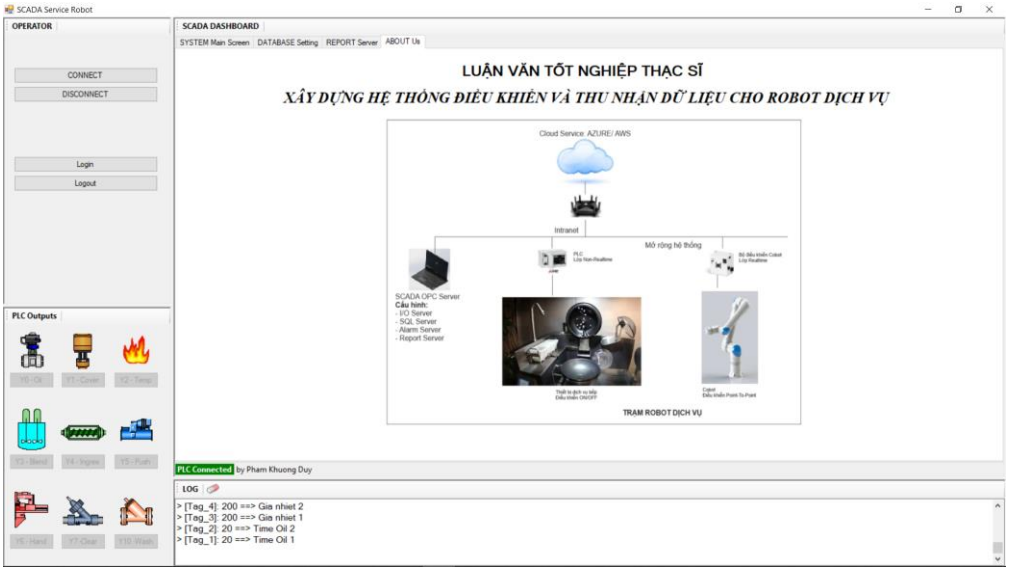
được tích hợp lập trình có thể quan sát toàn bộ các kết quả đọc ghi dữ liệu thông qua giao diện đồ họa như trên *Hình 3.3* và *Hình 3.4*.



Hình 3.3: Giao diện màn hình giám sát chính



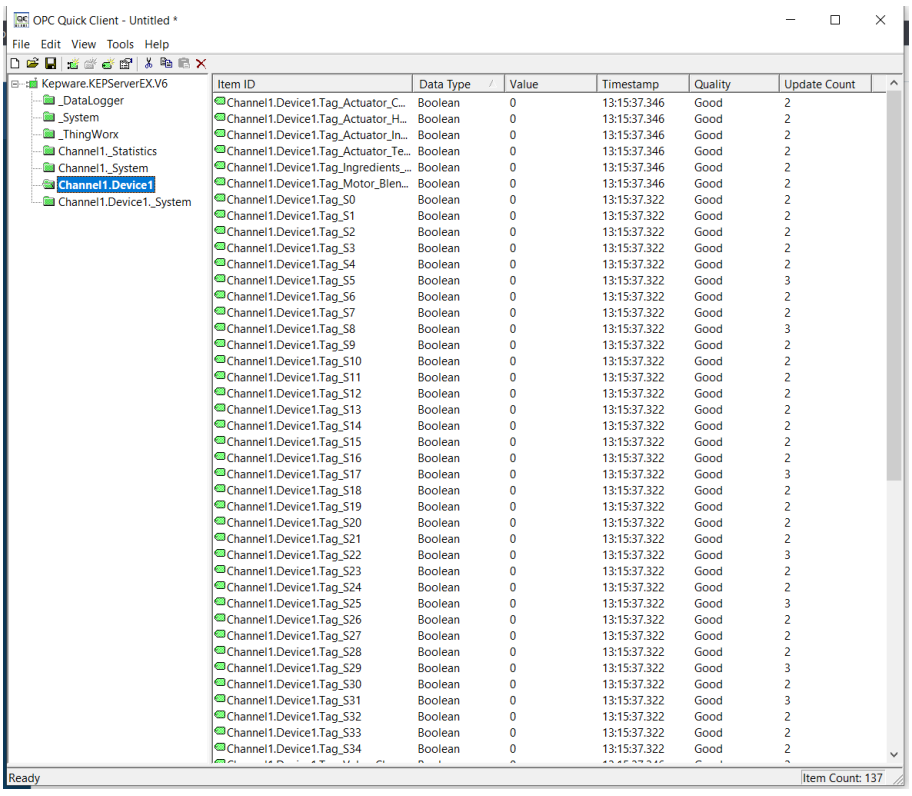
Hình 3.4: Giao diện màn hình cài đặt tham số cho PLC



Hình 3.5: Giao diện màn hình AboutUs

3.4.2 Kết quả trạng thái giá trị Tag trên phần mềm KepServerEX

Trạng thái giá trị tag KEPServerEX, ở cột "Value" ta thấy các giá trị tag hiển thị đúng với giá trị tag từ PLC cho trên *Hình 3.6*.



Hình 3.6: Các giá trị Tag cần giám sát online tương ứng trên KepServerEX

CHƯƠNG 4: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

4.1 Kết luận

Đề tài mang tính ứng dụng theo tinh thần của định hướng ứng dụng của luận văn Thạc sĩ. Luận văn tập trung giải mã công nghệ từ một thiết bị có trên thị trường và từ đó tìm cách cải tiến tính năng và hạ giá thành sản phẩm phù hợp với thị trường Việt Nam. Với hệ thống giám sát dùng OPC server, hệ thống dễ dàng mở rộng cho nhiều thiết bị bếp khác với các tính năng khác. Các ứng dụng thực tế dạng này không được các hãng sản xuất trên thế giới công bố rộng rãi về công nghệ, mã nguồn….nên đề tài đóng góp nền tảng ban đầu cho xu hướng chuyển đổi số và công nghệ 4.0 vào các lĩnh vực dịch vụ trong xu hướng thành phố thông minh đã và đang hình thành tại Việt Nam.

Do hạn chế trong việc điều khiển thiết bị nên đề tài sử dụng mô phỏng ở phần logic điều khiển PLC bằng phần mềm MX-OPC kết hợp với phần mềm lập trình PLC GX Work2. Các thao tác của bếp được thiết kế hoạt động giả định theo logic máy thật vẫn giữ được tính thực tế để nội dung có ý nghĩa thực tế. Các chương trình giám sát dùng C# thực hiện được việc truy xuất dữ liệu từ PLC để hoàn thành hệ thống giám sát.

Đề tài bao gồm 4 chương thực hiện một mục tiêu chính, đó là, xây dựng một hệ thống thu nhận dữ liệu từ các hoạt động của một trạm robot gồm các cơ cấu tác động thực hiện các thao tác nấu bếp với các cơ cấu chấp hành có chuyển động cố định. Hệ thống robot dịch vụ trên là sự tích hợp của các cơ cấu chấp hành như sau:

- Thiết bị nấu ăn dạng trộn
- Hệ thống van cấp dầu, van nước rửa xả
- Hệ thống gia nhiệt
- Hệ thống cấp liệu
- Thiết bị cánh tay robot thao tác cố định (không lập trình được)

Các công việc thực hiện như sau:

(1) Thiết kế quy trình nấu bếp tổng quát với các bước trình tự chạy tự động và cấp phát I/O cho hệ thống hoạt động được.

(2) Lập trình điều khiển trình tự trên PLC một quy trình cơ sở thao tác đã thiết kế

(3) Xây dựng Tag cần điều khiển và giám sát trên phần mềm KepserverEX

(4) Lập trình C# truy xuất các giá trị Tag trên KepserverEX để sử dụng cho mục đích điều khiển và giám sát hệ thống và sử dụng

(5) Hệ thống được kiểm chứng bằng cách cho chương trình bếp chạy trên PLC và giám sát trình tự đó trên phần mềm C# thông qua giao diện đồ họa trên máy tính.

4.2 Hướng phát triển

Đề tài đã thực hiện thành công quá trình thu nhận dữ liệu bao gồm các biến I/O sử dụng trên PLC ở điều khiển cấp thấp. Đây là bước quan trọng nhất của hệ thống thu nhận dữ liệu vì là bước trung gian giữa điều khiển cấp thấp và quản lý cấp cao của hệ thống. Sau khi dữ liệu được thu nhận đầy đủ về máy tính trung tâm thông qua phần mềm OPC server KepserverEX, các dữ liệu có thể được tiếp tục khai thác ở các bước tiếp theo để tăng giá trị của hệ thống khi đưa vào sử dụng. Các hướng phát triển của đề tài như sau:

- Xây dựng cơ sở dữ liệu (SQL Server) để dữ liệu hoạt động được lưu trữ và truy vấn

- Xây dựng tính năng báo cáo (Report Server) dựa trên truy vấn cơ sở dữ liệu để người chủ sở hữu có các báo cáo trong quá trình hoạt động kinh doanh của hệ thống.

- Xây dựng tính năng cảnh báo (Alarm server) vẽ đồ thị (Trend server).

- Xây dựng ngôn ngữ quy trình nấu món ăn khác để dễ dàng xây dựng quy trình mới

- Xây dựng hệ thống thu nhận dữ liệu cho robot dạng thao tác phức tạp lập trình được (ví dụ robot cộng tác) để mở rộng hệ thống toàn diện hơn cho lĩnh vực dịch vụ.

- Xây dựng hệ thống SCADA dùng Webserver và cài đặt trên Cloud để tăng tính linh hoạt quản lý trên các hệ thống.

DANH MỤC TÀI LIỆU THAM KHẢO

[1] *Robot dịch vụ,* [Online]. https://vneconomy.vn/covid-19-thuc-day-tu-dong-hoa-nganh-dich-vu-an-uong.htm

[2] *Robot dịch vụ nhà hàng*, [Online]. https://robbreport.com/food-drink/dining/silicon-valley-restaurant-robot-servers-1234629314/

[3] Avinash (2021), *Everything About PLC Programming* - Practical lessons on PLC programming using Allen Bradley, Siemens, and Mitsubishi PLCs with examples, Avinash Prakash Malekar.

[4] Boyer, Stuart A (2004), *SCADA: Supervisory Control and Data Aquisition*, ISA – The Instrumentation, System, and Automation Society, 3rd Edition.

[5] Stuart G. McCrady (2013), *Designing SCADA Application Software: A Pratical Approach* , Elsevier, 1st Edition.

[6] Khaled Kamel and Eman Kamel (2014), *Programmable Logic Controllers – Industrial Control* , McGraw-Hill Education, 1st Edition.

[7] Tianmiao Wang, Yong Tao, Yang Chen (2012), *Research status and development trend of service robot technology,* Chinese science, Information science.

[8] Hakan Gürocak (2016), *Industrial Motion Control, Motor Selection, Drives, Controller Tuning, Application* , John Wiley & Sons, Ltd.

[9] *Robot dịch vụ dùng cơ cấu tịnh tiến điều khiển trình tự*, [Online]. https://vention.io/designs/cartesian-palletizer-dual-pallet-w-infeed-machinemotion-2-132549

[10] *Robot cộng tác*, [Online]. https://robot3t.com/san-pham/r3t-03-robot-cong-nghiep-tai-6kg/

[11] L.A. Bryan and E.A. Bryan (1997), *Programmable Controller – Theory and Implementation*, Industrial Text Company Publication, second Edition.

[12] *Phần mềm KEPServerEX*, [Online]. https://www.kepware.com/

[13] *Robot phục vụ*, [Online]. https://www.youtube.com/watch?v=6-Jo9Lk3txI

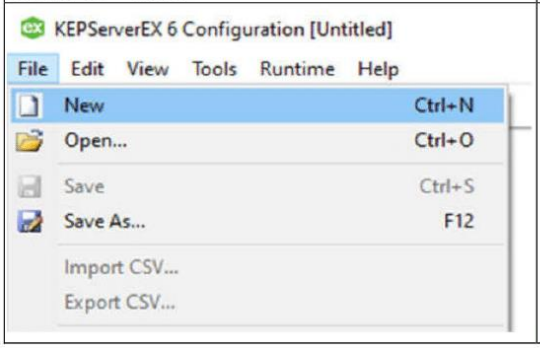
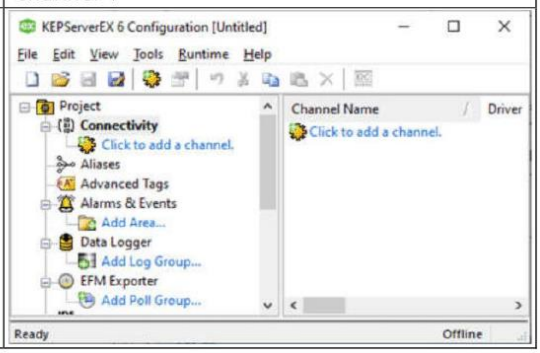
[14] *Robot nấu ăn*, [Online]. https://www.youtube.com/watch?v=8NTIciISVPA&t=1s

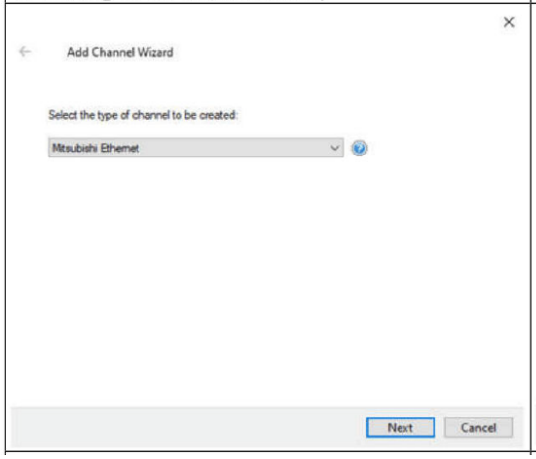
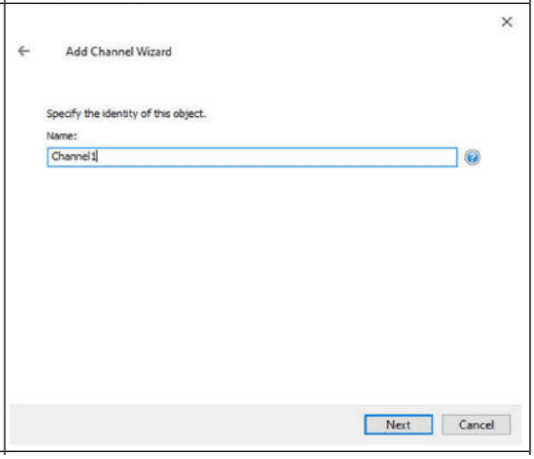
PHỤ LỤC

PHỤ LỤC P1: Cấu hình PLC Mitsubishi trên KEPServerEX

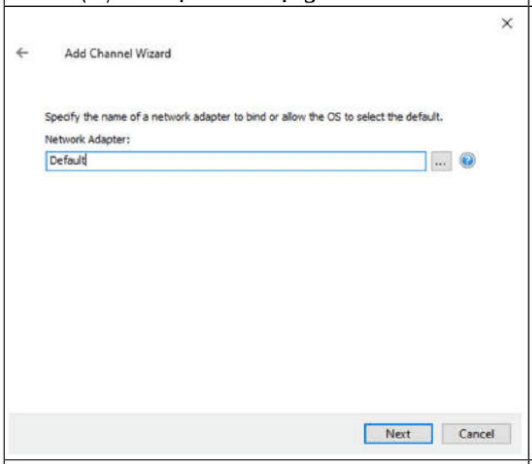
P1.1 Thêm mới Channel

Đối với các loại PLC có cổng Ethernet của hãng Mitsubishi, phương pháp thêm mới và cấu hình channel trong KEPServerEX tương đối giống nhau, do đó ở đây sử dụng FX3U làm ví dụ.

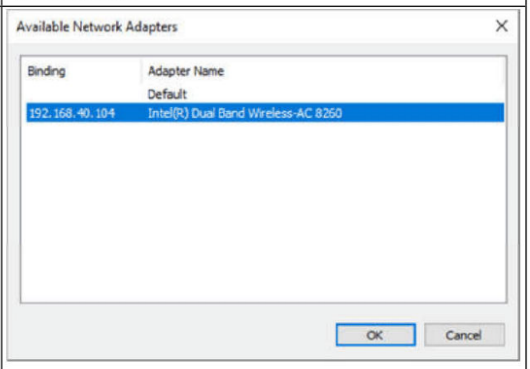
Step 1: Mở phần mềm KEPServerEX tạo project mới. File > New.	**Step 2:** Thêm channel mới "Click to add a channel".
	

Step 3: Chọn Mitsubishi Ethernet (hoặc dựa vào PLC đang có để chọn channel)	**Step 4:** Đặt tên cho channel, ở đây đặt tên "Channel1".
	

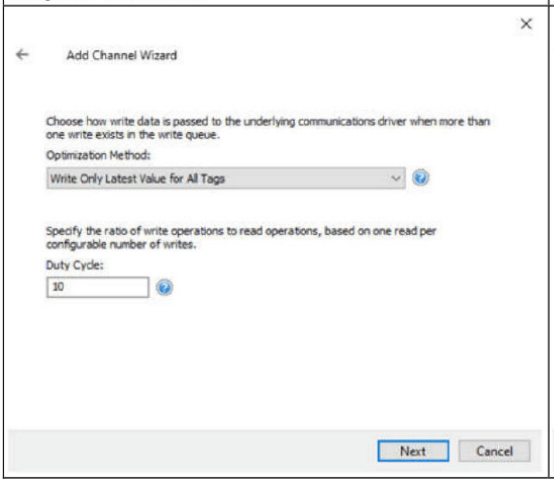
Step 5: Lựa chọn card mạng, nhấn vào nút 3 chấm (...) để chọn card mạng.



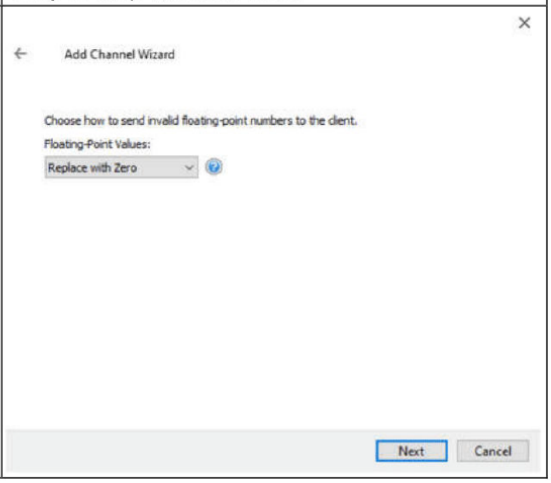
Step 6: Click vào card mạng đang có ở hiện tại và nhấn OK.



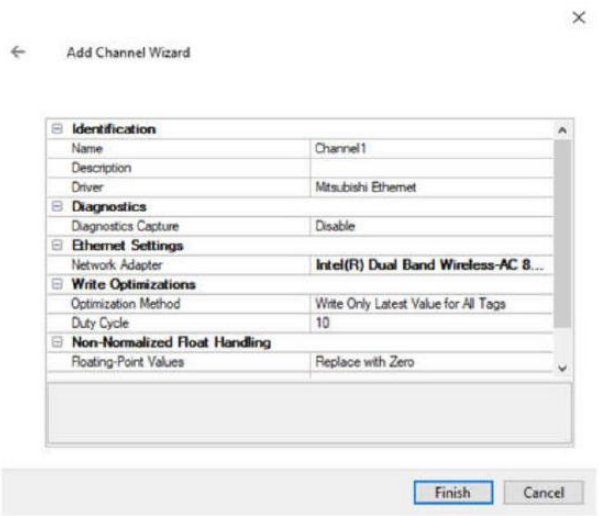
Step 7: Nhấn Next



Step 8: Tiếp tục nhấn Next

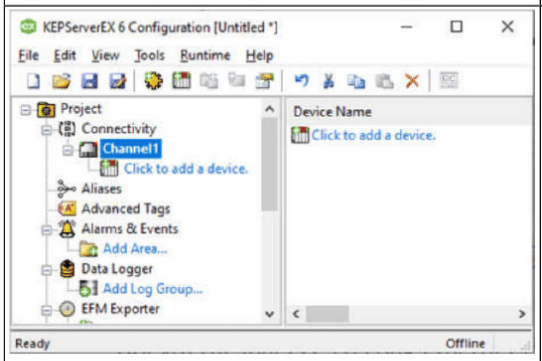
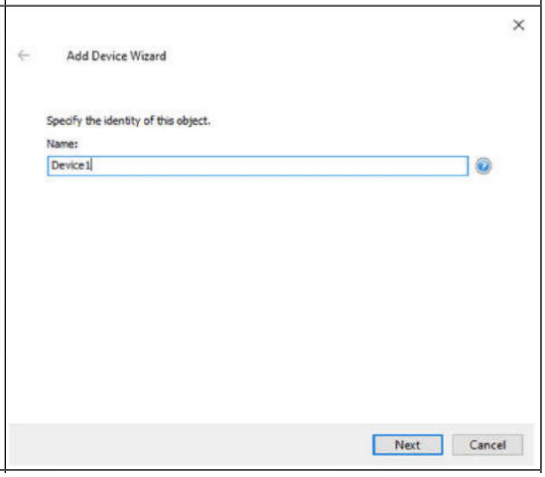


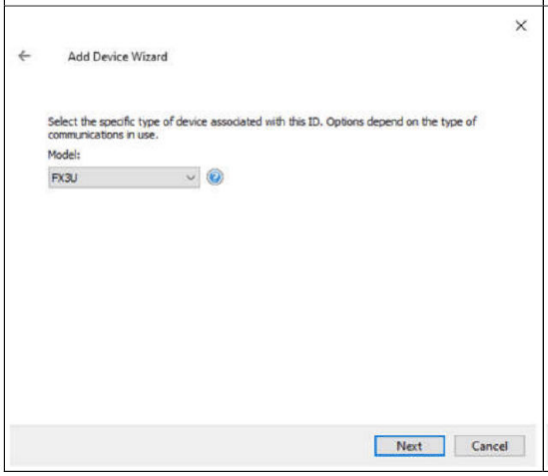
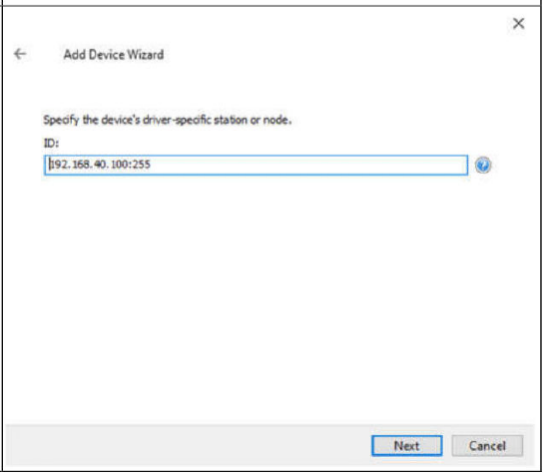
Step 9: Nhấn finish để kết thúc

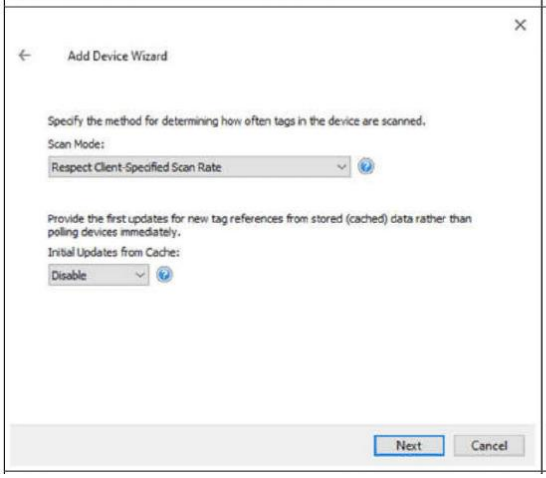
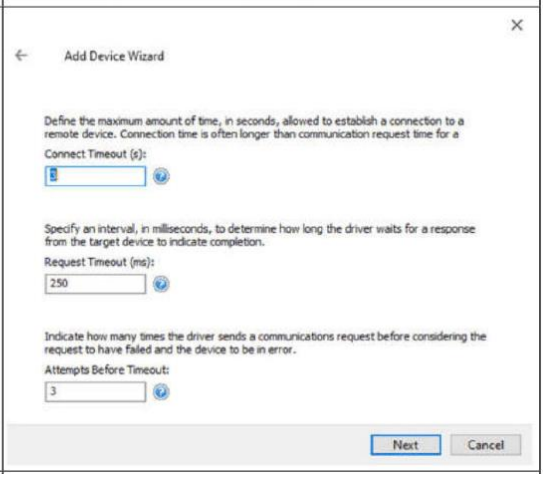


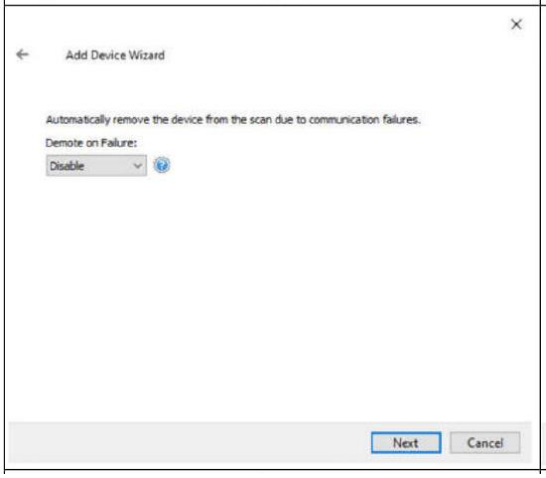
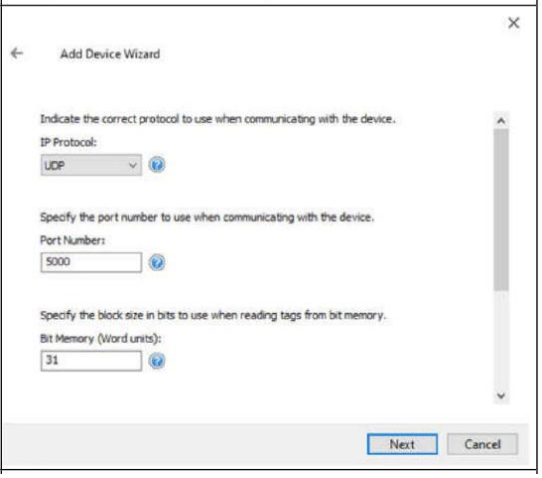
P1.2 Thêm mới Device

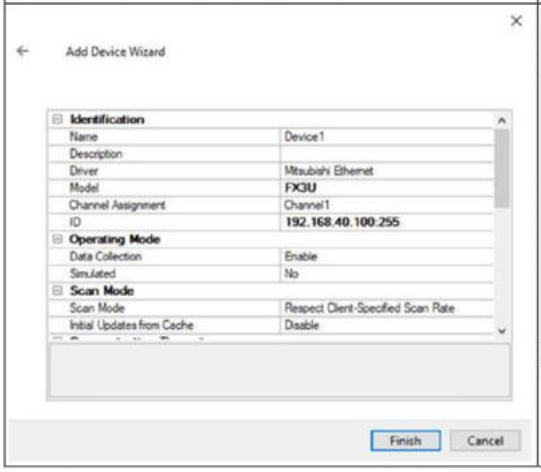
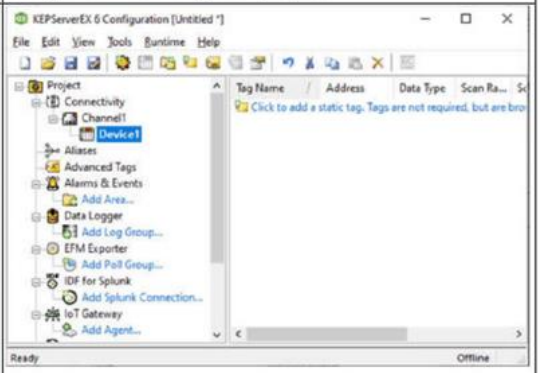
Đối với các loại PLC có cổng Ethernet của hãng Mitsubishi, phương pháp thêm mới và cấu hình device trong KEPServerEX tương đối giống nhau, do đó ở đây sử dụng FX3U làm ví dụ.

Step 1: Click chọn "Click to add a device".	**Step 2:** Đặt tên cho Device, ở đây đặt là "Device1"
	

Step 3: Lựa chọn PLC, ở đây chọn PLC dòng FX3U.	**Step 4:** Cài đặt địa chỉ IP cho PLC, cổng mặc định là 255.
	

Step 5: Nhấn Next để tiếp tục.	**Step 6:** Nhấn Next để tiếp tục.
	

Step 7: Nhấn Next để tiếp tục.	**Step 8:** Nhấn Next để tiếp tục.
	

Step 9: Nhấn Finish để kết thúc.	**Step 10:** Thêm tag cho project.
	

PHỤ LỤC P2: Chương trình nguồn

P2.1 Chương trình “class_KEPServerEX.cs”

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Drawing;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace SCADAServicesApp
{
   class class_KEPServerEX
   {
      //*************************************************************
      public static string[] tagread(int tagnumber)
      {// Class Khai báo tag
         string tagID_1 = "Channel1.Device1.Tag_t_Oil_1";
         string tagID_2 = "Channel1.Device1.Tag_t_Oil_2";
         string tagID_3 = "Channel1.Device1.Tag_T_1";
         string tagID_4 = "Channel1.Device1.Tag_T_2";
         string tagID_5 = "Channel1.Device1.Tag_T_3";
         string tagID_6 = "Channel1.Device1.Tag_speed_blending_1";
         string tagID_7 = "Channel1.Device1.Tag_speed_blending_2";
         string tagID_8 = "Channel1.Device1.Tag_speed_blending_3";
         string tagID_9 = "Channel1.Device1.Tag_speed_blending_4";
         string tagID_10 = "Channel1.Device1.Tag_time_blending_1";
         string tagID_11 = "Channel1.Device1.Tag_time_blending_2";
         string tagID_12 = "Channel1.Device1.Tag_time_blending_3";
```

```
string tagID_13 = "Channel1.Device1.Tag_time_blending_4";
string tagID_14 = "Channel1.Device1.Tag_ingredient_1";
string tagID_15 = "Channel1.Device1.Tag_ingredient_2";
string tagID_16 = "Channel1.Device1.Tag_ingredient_3";
string tagID_17 = "Channel1.Device1.Tag_ingredient_4";
string tagID_18 = "Channel1.Device1.Tag_mode";
string tagID_19 = "Channel1.Device1.Tag_time_sharking";
string tagID_20 = "Channel1.Device1.Tag_time_foodout";
string tagID_21 = "Channel1.Device1.Tag_time_waterout";
string tagID_22 = "Channel1.Device1.Tag_time_washing";
string tagID_23 = "Channel1.Device1.Tag_t_cleaning";
string tagID_24 = "Channel1.Device1.Tag_t_wasing";
string tagID_25 = "Channel1.Device1.Tag_Valve_Oil";        //M0
string tagID_26 = "Channel1.Device1.Tag_Actuator_Cover";   //M1
string tagID_27 = "Channel1.Device1.Tag_Actuator_Temp";    //M2
string tagID_28 = "Channel1.Device1.Tag_Motor_Blending";   //M3
string tagID_29 = "Channel1.Device1.Tag_Actuator_Ingredients";//M4
string tagID_30 = "Channel1.Device1.Tag_Ingredients_Push"; //M5
string tagID_31 = "Channel1.Device1.Tag_Actuator_Hand";    //M6
string tagID_32 = "Channel1.Device1.Tag_Valve_Cleaning";   //M7
string tagID_33 = "Channel1.Device1.Tag_Valve_Wasing";     //M8
string tagID_34 = "Channel1.Device1.Tag_WatchDog";         //WatchDog
string tagID_35 = "Channel1.Device1.Tag_S0";
string tagID_36 = "Channel1.Device1.Tag_S1";
string tagID_37 = "Channel1.Device1.Tag_S2";
string tagID_38 = "Channel1.Device1.Tag_S3";
string tagID_39 = "Channel1.Device1.Tag_S4";
string tagID_40 = "Channel1.Device1.Tag_S5";
string tagID_41 = "Channel1.Device1.Tag_S6";
```

```
string tagID_42 = "Channel1.Device1.Tag_S7";
string tagID_43 = "Channel1.Device1.Tag_S8";
string tagID_44 = "Channel1.Device1.Tag_S9";
string tagID_45 = "Channel1.Device1.Tag_S10";
string tagID_46 = "Channel1.Device1.Tag_S11";
string tagID_47 = "Channel1.Device1.Tag_S12";
string tagID_48 = "Channel1.Device1.Tag_S13";
string tagID_49 = "Channel1.Device1.Tag_S14";
string tagID_50 = "Channel1.Device1.Tag_S15";
string tagID_51 = "Channel1.Device1.Tag_S16";
string tagID_52 = "Channel1.Device1.Tag_S17";
string tagID_53 = "Channel1.Device1.Tag_S18";
string tagID_54 = "Channel1.Device1.Tag_S19";
string tagID_55 = "Channel1.Device1.Tag_S20";
string tagID_56 = "Channel1.Device1.Tag_S21";
string tagID_57 = "Channel1.Device1.Tag_S22";
string tagID_58 = "Channel1.Device1.Tag_S23";
string tagID_59 = "Channel1.Device1.Tag_S24";
string tagID_60 = "Channel1.Device1.Tag_S25";
string tagID_61 = "Channel1.Device1.Tag_S26";
string tagID_62 = "Channel1.Device1.Tag_S27";
string tagID_63 = "Channel1.Device1.Tag_S28";
string tagID_64 = "Channel1.Device1.Tag_S29";
string tagID_65 = "Channel1.Device1.Tag_S30";
string tagID_66 = "Channel1.Device1.Tag_S31";
string tagID_67 = "Channel1.Device1.Tag_S32";
string tagID_68 = "Channel1.Device1.Tag_S33";
string tagID_69 = "Channel1.Device1.Tag_S34";
```

```
string tagID_70 = "Channel1.Device1.Tag_INIT";
string tagID_71 = "Channel1.Device1.Tag_START";
string tagID_72 = "Channel1.Device1.Tag_STOP";
string tagID_73 = "Channel1.Device1.Tag_AUTO";
string tagID_74 = "Channel1.Device1.Tag_MANUAL";
string tagID_75 = "Channel1.Device1.Tag_EMER";
string tagID_76 = "Channel1.Device1.Tag_STEP_COMMAND";

string tagID_77 = "Channel1.Device1.Tag_Y0";
string tagID_78 = "Channel1.Device1.Tag_Y1";
string tagID_79 = "Channel1.Device1.Tag_Y2";
string tagID_80 = "Channel1.Device1.Tag_Y3";
string tagID_81 = "Channel1.Device1.Tag_Y4";
string tagID_82 = "Channel1.Device1.Tag_Y5";
string tagID_83 = "Channel1.Device1.Tag_Y6";
string tagID_84 = "Channel1.Device1.Tag_Y7";
string tagID_85 = "Channel1.Device1.Tag_Y10";

string[] tags;
tags = new string[tagnumber];
tags.SetValue(tagID_1, 1);
tags.SetValue(tagID_2, 2);
tags.SetValue(tagID_3, 3);
tags.SetValue(tagID_4, 4);
tags.SetValue(tagID_5, 5);
tags.SetValue(tagID_6, 6);
tags.SetValue(tagID_7, 7);
tags.SetValue(tagID_8, 8);
tags.SetValue(tagID_9, 9);
```

```
tags.SetValue(tagID_10, 10);
tags.SetValue(tagID_11, 11);
tags.SetValue(tagID_12, 12);
tags.SetValue(tagID_13, 13);
tags.SetValue(tagID_14, 14);
tags.SetValue(tagID_15, 15);
tags.SetValue(tagID_16, 16);
tags.SetValue(tagID_17, 17);
tags.SetValue(tagID_18, 18);
tags.SetValue(tagID_19, 19);
tags.SetValue(tagID_20, 20);
tags.SetValue(tagID_21, 21);
tags.SetValue(tagID_22, 22);
tags.SetValue(tagID_23, 23);
tags.SetValue(tagID_24, 24);
tags.SetValue(tagID_25, 25);
tags.SetValue(tagID_26, 26);
tags.SetValue(tagID_27, 27);
tags.SetValue(tagID_28, 28);
tags.SetValue(tagID_29, 29);
tags.SetValue(tagID_30, 30);
tags.SetValue(tagID_31, 31);
tags.SetValue(tagID_32, 32);
tags.SetValue(tagID_33, 33);
tags.SetValue(tagID_34, 34);    //WatchDog
tags.SetValue(tagID_35, 35);
tags.SetValue(tagID_36, 36);
tags.SetValue(tagID_37, 37);
tags.SetValue(tagID_38, 38);
```

```
tags.SetValue(tagID_39, 39);
tags.SetValue(tagID_40, 40);
tags.SetValue(tagID_41, 41);
tags.SetValue(tagID_42, 42);
tags.SetValue(tagID_43, 43);
tags.SetValue(tagID_44, 44);
tags.SetValue(tagID_45, 45);
tags.SetValue(tagID_46, 46);
tags.SetValue(tagID_47, 47);
tags.SetValue(tagID_48, 48);
tags.SetValue(tagID_49, 49);
tags.SetValue(tagID_50, 50);
tags.SetValue(tagID_51, 51);
tags.SetValue(tagID_52, 52);
tags.SetValue(tagID_53, 53);
tags.SetValue(tagID_54, 54);
tags.SetValue(tagID_55, 55);
tags.SetValue(tagID_56, 56);
tags.SetValue(tagID_57, 57);
tags.SetValue(tagID_58, 58);
tags.SetValue(tagID_59, 59);
tags.SetValue(tagID_60, 60);
tags.SetValue(tagID_61, 61);
tags.SetValue(tagID_62, 62);
tags.SetValue(tagID_63, 63);
tags.SetValue(tagID_64, 64);
tags.SetValue(tagID_65, 65);
tags.SetValue(tagID_66, 66);
tags.SetValue(tagID_67, 67);
```

```
    tags.SetValue(tagID_68, 68);
    tags.SetValue(tagID_69, 69);

    tags.SetValue(tagID_70, 70);
    tags.SetValue(tagID_71, 71);
    tags.SetValue(tagID_72, 72);
    tags.SetValue(tagID_73, 73);
    tags.SetValue(tagID_74, 74);
    tags.SetValue(tagID_75, 75);
    tags.SetValue(tagID_76, 76);

    tags.SetValue(tagID_77, 77);//Y0
    tags.SetValue(tagID_78, 78);//Y1
    tags.SetValue(tagID_79, 79);//Y2
    tags.SetValue(tagID_80, 80);//Y3
    tags.SetValue(tagID_81, 81);//Y4
    tags.SetValue(tagID_82, 82);//Y5
    tags.SetValue(tagID_83, 83);//Y6
    tags.SetValue(tagID_84, 84);//Y7
    tags.SetValue(tagID_85, 85);//Y10
    return tags;
}
//*****************************************************************
public static Int32[] tagID(int tagnumber)
{// Class tạo array đọc ID tags - mặc định không đổi
    Int32[] cltarr;
    cltarr = new Int32[tagnumber];
    for (int i = 1; i < tagnumber; i++)
    {
```

```
            cltarr.SetValue(i, i);
        }
        return cltarr;
    }

  }

}
```

P2.2 Chương trình khai báo kết nối OPC

```
//*************************************************************************
//KEPServerEX CONNECT
//*************************************************************************
    static int tagNumber = 85;      // Cài đặt số lượng tag của project
    static int PLCscantime = 1000;  // Cài đặt thời gian quét PLC
    // Gọi các kết nối OPC
    public OPCAutomation.OPCServer AnOPCServer;
    public OPCAutomation.OPCServer OPCServer;
    public OPCAutomation.OPCGroups OPCGroup;
    public OPCAutomation.OPCGroup PLC;
    public string Groupname;

    static int arrlength = tagNumber + 1;
    Array OPtags = class_KEPServerEX.tagread(arrlength);
    Array tagID = class_KEPServerEX.tagID(arrlength);
    Array WriteItems = Array.CreateInstance(typeof(object), arrlength);
    Array tagHandles = Array.CreateInstance(typeof(Int32), arrlength);
    Array OPCError = Array.CreateInstance(typeof(Int32), arrlength);
    Array dataType = Array.CreateInstance(typeof(Int16), arrlength);
    Array AccessPaths = Array.CreateInstance(typeof(string), arrlength);
```

P2.3 Chương trình đọc dữ liệu từ OPC lên C#

```
private void dataScan(int ID, int NumItems,
ref Array tagID, ref Array ItemValues, ref Array Qualities, ref Array TimeStamps)
    {
       for (int i = 1; i <= NumItems; i++)
       {
          // Khai báo biến chung
          int getTagID = Convert.ToInt32(tagID.GetValue(i));
```

```
            string tagValue = ItemValues.GetValue(i)?.ToString();
            switch (getTagID)
            {
               case 1:
                  tbx_TG_ChamDau1.Text = tagValue;
                  DisplayMessage(LOG_txtMessage, "[Tag_1]: " + tagValue);
                  break;
               case 2:
                  tbx_TG_ChamDau2.Text = tagValue;
                  DisplayMessage(LOG_txtMessage, "[Tag_2]: " + tagValue);
                  break;
               case 3:
                  tbx_GiaNhiet1.Text = tagValue;
                  DisplayMessage(LOG_txtMessage, "[Tag_3]: " + tagValue);
                  break;
                              …
                              Default:
                                 break;
                        }
                     }
}
```

P2.4 Chương trình “class_Watchdog.cs”

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace SCADAServicesApp
{
   public static class class_Watchdog
   {
      static string valold = "";
      //*************************************************************
      //Chương trình con Watchdog
      //*************************************************************
      public static void WatchdogStatus(Button btt, string valnow)
      {
         if (valnow != valold)
         {
            btt.BackColor = Color.Green;
            btt.ForeColor = Color.White;
         }
         else
         {
            btt.BackColor = Color.Red;
```

```
            btt.ForeColor = Color.White;
        }
        valold = valnow;
    }
    //*************************************************************
    public static void WatchdogStatus(ToolStripStatusLabel lb, string valnow)
    {
        if (valnow != valold)
        {
            lb.BackColor = Color.Green;
            lb.ForeColor = Color.White;
        }
        else
        {
            lb.BackColor = Color.Red;
            lb.ForeColor = Color.White;
        }
        valold = valnow;
    }
    //*************************************************************************
  }
}
```

P2.5 Chương trình “class_Login.cs”

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace SCADAServicesApp
{
   public class class_Login
   {
      //*********************************************************************
      //Disable toàn bộ khi chưa đăng nhập/ hoặc nhấn đăng xuất
      //*********************************************************************
      public void Not_Login ()
      {
         Form1 frm = (Form1)Application.OpenForms["Form1"];
         // Các object cần Enable/Disable
         frm.btt_DATAWRITE.Enabled = false;//Admin
         frm.btt_Auto.Enabled = false;
         frm.btt_Manu.Enabled = false;
         frm.btt_Init.Enabled = false;
         frm.btt_Start.Enabled = false;
         frm.btt_Stop.Enabled = false;
         frm.btt_Y0.Enabled = false;
         frm.btt_Y1.Enabled = false;
         frm.btt_Y2.Enabled = false;
         frm.btt_Y3.Enabled = false;
```

```
    frm.btt_Y4.Enabled = false;
    frm.btt_Y5.Enabled = false;
    frm.btt_Y6.Enabled = false;
    frm.btt_Y7.Enabled = false;
    frm.btt_Y8.Enabled = false;


}
//*********************************************************************
//Đăng nhập bằng quyền Admin
//*********************************************************************
public void admin_Control_Elements()
{
    Form1 frm = (Form1)Application.OpenForms["Form1"];
    // Các object cần Enable/Disable
    frm.btt_DATAWRITE.Enabled = true;//Admin
    frm.btt_Auto.Enabled = true;
    frm.btt_Manu.Enabled = true;
    frm.btt_Init.Enabled = true;
    frm.btt_Start.Enabled = true;
    frm.btt_Stop.Enabled = true;
    frm.btt_Y0.Enabled = true;
    frm.btt_Y1.Enabled = true;
    frm.btt_Y2.Enabled = true;
    frm.btt_Y3.Enabled = true;
    frm.btt_Y4.Enabled = true;
    frm.btt_Y5.Enabled = true;
    frm.btt_Y6.Enabled = true;
    frm.btt_Y7.Enabled = true;
```

```
            frm.btt_Y8.Enabled = true;
        }
        //*********************************************************************
        //Đăng nhập bằng quyền người dùng Operator
        //*********************************************************************
        public void Operator_Control_Elements()
        {
            Form1 frm = (Form1)Application.OpenForms["Form1"];
            // Các object cần Enable/Disable
            frm.btt_DATAWRITE.Enabled = false;//Admin
            frm.btt_Auto.Enabled = true;
            frm.btt_Manu.Enabled = true;
            frm.btt_Init.Enabled = true;
            frm.btt_Start.Enabled = true;
            frm.btt_Stop.Enabled = true;
            frm.btt_Y0.Enabled = true;
            frm.btt_Y1.Enabled = true;
            frm.btt_Y2.Enabled = true;
            frm.btt_Y3.Enabled = true;
            frm.btt_Y4.Enabled = true;
            frm.btt_Y5.Enabled = true;
            frm.btt_Y6.Enabled = true;
            frm.btt_Y7.Enabled = true;
            frm.btt_Y8.Enabled = true;

        }
    }
}
```

BẢN CAM ĐOAN

Tôi cam đoan đã thực hiện việc kiểm tra mức độ tương đồng nội dung luận văn/luận án qua phần mềm Kiểm tra tài liệu (https://kiemtratailieu.vn) một cách trung thực và đạt kết quả mức độ tương đồng **8%** toàn bộ nội dung luận văn/luận án. Bản luận văn/luận án kiểm tra qua phần mềm là bản cứng luận văn/luận án đã nộp bảo vệ trước hội đồng. Nếu sai sót tôi xin chịu các hình thức kỷ luật theo quy định hiện hành của Học viện.

TP. Hồ Chí Minh, ngày 04 tháng 5 năm 2022

Học viên thực hiện luận văn

Phạm Khương Duy

KiểmTraTàiLiệu

BÁO CÁO KIỂM TRA TRÙNG LẶP

Thông tin tài liệu

Tên tài liệu:	Xây dựng hệ thống điều khiển và thu nhận dữ liệu cho robot dịch vụ
Tác giả:	Phạm Khương Duy
Điểm trùng lặp:	8
Thời gian tải lên:	22:09 04/05/2022
Thời gian sinh báo cáo:	22:16 04/05/2022
Các trang kiểm tra:	50/50 trang

Kết quả kiểm tra trùng lặp

Nguồn trùng lặp tiêu biểu

123docz.net vi.wikipedia.org

Học viên **Người hướng dẫn khoa học**

Phạm Khương Duy **TS. Chung Tấn Lâm**