

Routing/Linking Communication Networks of Multi-Brand PLCS by Port Incorporation Employing SCADA

Laiba Mughal^{a,*}, Bilal Iqbal^b, Syed Abbas Ali^c, Muhammad Asad Hussain^d

^a Department of Computer Science, UIT University, Karachi, Pakistan

^b Department of Engineering Technology, UIT University, Karachi, Pakistan

^c Department of Computer & Information System Engineering, NED University of Engineering & Technology, Karachi, Pakistan

^d Department of Electrical Engineering, UIT University, Karachi, Pakistan

* Corresponding author: Laiba Mughal, Email: lmughal@uitu.edu.pk

Received:03 September 2025, Revised:18 September 2025, Accepted:29 September 2025, Published:01 October 2025

KEY WORDS

SCADA

Multi-Brand PLCs

M2M communication

ABSTRACT

Without new technologies, our industry would not be where it is now. Almost every industrial site contains PLCs and other controllers of various brands and models when it comes to automation. The use of programmable logic controllers (PLCs) in industry has accelerated in recent years. But when comes to hardware components and their compatibility, there are certain limitations that causes problems on large scale. With the growing use of PLCs in industries, the limitation of communication also rises. In this paper, we present an automatic control structure that removes that stumbling block and gives industries an advantage by permitting the use of numerous PLCs from multiple brands and allowing comprehensive and extensive communication between them. The objective of this project is to remove the hassle of communication by creating a system consisting of a server translating the information between the two PLCs, helping them communicate regardless of what communication protocol they use. This is accomplished with a server that acts as a translation layer between the PLCs, translating data and allowing the various machines to communicate and harmonize as one with minimum human intervention, comparable to M2M communication. Our final deliverable is a system consisting of control system architecture i.e. a SCADA, connected to a server. The server would be further connected to clients controlling different processes programmed to perform through multi brand PLCs and monitoring their data in one place. These processes will be a water level sensor and a temperature controller done with minimal human intervention.

1. Introduction

The use of PLCs (Programmable Logic Controller) in industry has risen dramatically. It's a device that's intended to monitor and control building systems and manufacturing operations in a wide range of sectors. Except under real-time constraints and with greater reliability and performance, a PLC is meant to survive the challenges of severe industrial environments and are often capable of withstanding extreme industrial conditions. But that too comes with

the limitations. Sometimes conflicts arise when we integrated multi-brand PLCs in a system. The issue arises due to the conflict among different communication protocols, on which these PLCs process. Interaction between various brands or manufacturers' control devices, such as PLCs, RTUs (Remote Terminal Unit), drivers, sensors etc, is one of the most common issues in Industrial Integration. While most control devices have communication mechanisms, and certain industry standards have helped, they don't all communicate in the same way,

and even if they have the same serial or Ethernet cabling, the differences in communication protocols prevent them from transferring information. The most common example through which one can understand this issue is to communicate with a person who doesn't speak your language. Same goes for these controlled devices. The conflict arises due to the protocols they work on. When multiple PLCs are unable to interact with one another, things become more complicated because it is impossible to keep all the variables in mind, and there is no way to operate our system without it. Instead of needing to employ only PLCs that are compatible with one another, this research allows industries to use many PLCs of different brands while still being able to completely communicate with one another [1]. This task would be accomplished by a server acting as a translator or a layer between the PLCs, translating data and assisting two completely different machines to communicate and harmonize as one, analogous to M2M communication, as well as being able to control each other's interdependent operations without human involvement. PLC to PLC communication can be accomplished in a variety of ways. Which include:

1. Hard-wiring one controller to a different one.
2. Using protocol converters to connect the controllers.
3. By employing the HMI (Human-Machine Interface) as a communication hub for all controllers.

And even these methods hold limitations in regard to making modifications; requiring external wiring and additional I/O modules, and only applicable to a few certain applications. This process could also be possible using hardware to hardware approach, by integrating the modules such as hardware/protocol converters. That too comes with the detriment of cost incompetency and the maintenance every now and then. If there are any redundant systems involved, the complexity and expense all increase.

In this paper, we propose the software-to-software approach in order to resolve this distressing situation. This is advantageous since it is less expensive, more sustainable, and more powerful. With the help of control system architecture, we have proposed a solution in which a system that will integrate the SCADA (Supervisory Control and Data Acquisition) of various controlled process. Our control system architecture would include a graphical user interface (GUI) that will operate as a "protocol bridge" between numerous PLCs, allowing data to be transmitted between them.

In order to experiment the research, we simulated an industrial environment using a software approach

and includes two key and common industrial level control processes: water level indicator and control, and star-delta starter, as well as a SCADA-styled control system architecture. The framework is set up to run the control processes via SCADA, since the processes are physically connected, they will have the potential to start up on their own and can be operated manually in parallel. The operation is subsequently taken over by the control system architecture, running on both ends, starting in either direction. The processes accept the appropriate inputs, which the PLC then processes according to its programming and switches on specific output coils. These are then linked to their respective process indicators. In this scenario, a bridge can be created. When we are unable to communicate with the person who doesn't speak our language, we use translators. They work as a bridge between the two individuals. A protocol bridge, i.e. the linking server, communicates with this system and its data. The linking server is responsible for data reading and writing, as well as data interchange between the control processes and the SCADA [2]. The procedure of port integration can be used to make this connection possible. The SCADA collects the flux of input and output data, processes it in real time, and allows it to be watched and managed from a single distant location. This can be accomplished by including the ability to pause a control process abruptly using the emergency stop button, which can be accessed via SCADA or a physical connection to the control process. Some beneficial factors of this study include:

- The system will make it easier for companies to interface different brands of PLC's easily and efficiently.
- It will allow the PLCs to communicate worldwide through Ethernet being the portal to the Internet.
- It will make it possible to transmit data over a long distance.
- It is a highly reliable system requiring minimal maintenance.
- It will require no additional communication for the system to be installed.
- It is able to control processes on different ends as well as interdependent processes.
- It allows reliable communication over a network to other systems.
- It provides an interface to the user to control the process in real-time.

This module collects data from PLCs over the network and send it along with the newly created network, allowing industries to use multiple PLCs from different brands while still being able to fully communicate with one another.

Our final deliverable is a system consisting of control system architecture i.e. a SCADA, connected to a server. The server would further be connected to clients controlling different processes programmed to perform through multi brand PLCs and monitoring their data in one place.

2. Implementation Framework

2.1 Proposed System

Since the goal of this research is to create a system that provides an interface that allows communication across multiple brands of PLCs. This work can be accomplished by a server acting as a translator or a layer between the PLCs, translating information and assisting two completely different machines in communicating and synchronizing as one, as well as controlling each other's interdependent processes without the need for human intervention, similar to the M2M communication. Both PLCs will be interfacing with the SCADA that is working as a server and the PLCs are client [3], [4]. The linking software will be creating a bridge between SCADA and PLCs. A data base server will be used to save the data of both processes. The aim of this study is to present an initiative of a system that provides us with an interface to make communication possible between multi-brands of PLCs, translating the data among them to obtain a communication system with minimal human intervention, to produce a middle layer between the PLCs which can work as a translator among them, the connection and interface of multiple PLCs through the control system architecture of choice. This system will gather data over the network from the PLCs and send that information along with the created network and enable the industries to use multiple PLCs of different brands whilst still being able to fully communicate with one another rather than having to use only the PLCs which are compatible with each other. Figure below represents the architecture of proposed controlled system [5].

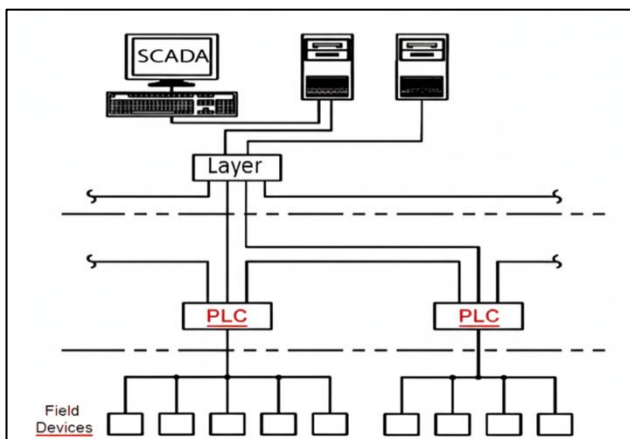


Fig. 2. Process Control via different brands of PLC with SCADA monitoring

2.2 Hardware execution

For the demonstration of our study, the plan is intended to create a replica of an industrial setting that enables the industries, especially small-scale businesses to be able to communicate with different brands of PLCs using different protocols. The communication was accomplished through the use of a connecting server, which served as a translator between two different brands of PLCs. For this demonstration, two commonly used processes of industries are implemented i.e. the water level sensor and indication control block, and the star- delta starter control block.

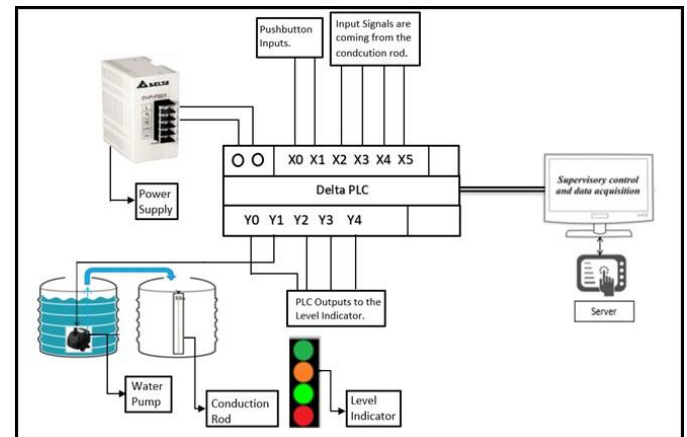


Fig. 1. Water Level Sensor and Indicator Control Block Diagram

The PLC utilized for the monitoring and controlling of water level sensor block is Delta DVP-14SS2. Following is the specification table;

Table 1

Delta DVP-14SS2 Specifications

Parameter	Value
Digital Inputs	8
Digital Outputs	6
Input Signal	24 VDC
Output	Relay type
Power Supply	24V DC

A PLC controls and monitors inputs and makes decisions based on the designed ladder logic to control the condition of the outputs [6], [7], [8]. The PLC is powered by a 24V DC supply. The purpose of the conduction rod is to show the level of water in the main tank. The level of water is communicated to the PLC through a conduction rod.

In this process, six inputs are used. X0 and X1 are used to stop and start the process respectively. X2, X3, X4, and X5 are the conduction rod representing the levels i.e. low level (25%), medium level (50%), high

level (75%), and top level (100%). The ladder logic of this process is programmed in such a way that the PLC takes input, processes it, and then turns on the outputs, lighting the signalling LEDs and filling the tank to ensure it never falls below a set level. The pump will automatically switch on to fill the tank if the water level falls below a specific level, and the LEDs will indicate the water level in the tank. The pump is automatically turned off if the tank is full or indicates a high level. Block diagram of the process controlled by DELTA PLC is given in Fig. 2.

Table 2

SIMATIC S71200 Specification

Parameter	Value
Digital Inputs	8
Digital Outputs	6
Analog Inputs	2
Output	Transistor type
Power Supply	24V DC
Input Signal	2 VDC

The star delta starter is an electromechanical reduced voltage method. This starter is structured in a way that the motor windings are connected in a star pattern during start-up. Because the voltage is lower in the Star arrangement, we use it to start the motor. When the motor increases speed, the windings are reconfigured in delta and the motor runs normally. Three contactors i.e. the main contractor, star contactor, and delta contactor, and a three-phase motor are used to design the star-delta starter [9].

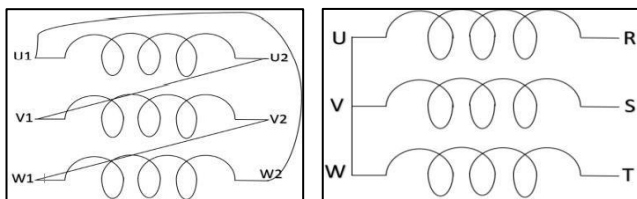


Fig. 3. Star Connection and Delta Connections

The ladder logic of this process is done in the respective manner. The main contactor is responsible for supplying power and must remain operational at all times. Star contactor with one end of all three windings, shorted to each other and the other end of the windings connected to the power source. Windings are connected in a loop in delta connections.

Two inputs are used in this pro I.O input that is used to initiate the process. I.1 is used as an emergency stop to protect machinery or limit any kind of risk. The main and star contactor are initially closed during the

process. Delta Contactor will close after a period, and star contactor will open. The PLC is controlling the operation. The timer block is used in the ladder logic to determine how long the star contactor has been open and how long the delta contactor has been closed. Once the main contactor is powered, the timer begins counting until the time surpasses the predetermined time, at which point the circuit is switched from star to delta.

The block diagram of star delta starter process is given below.

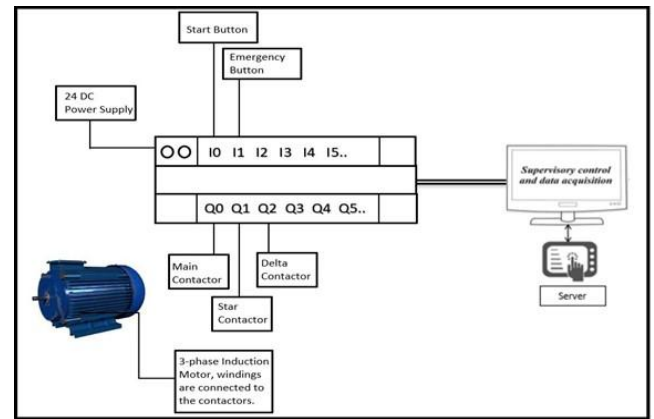


Fig. 4. Star Delta Starter

2.3 Software Integration

The first and the most important part starts from the programming of PLCs. The flow of software integration is divided into the phases and implemented in this order of PLC programming, fabrication of linking layers, addressing of PLCs, and designing of SCADA respectively. Since the first and foremost step is to program the PLC according to the processes. We can employ many inputs and output coils of a PLC by giving power to those terminals based on whatever logic is programmed into the controller. The graphical scripture on which the PLC programming of this study is based on is Ladder Logic, which is well-known for its programming convenience and efficiency.

2.3.1 PLC Programming

As this research is based on the two different brands of PLCs i.e. Delta DVP-14SS2 and SIMATIC S7-1200. Hence, different software are used for the programming of both blocks.

To program the Delta DVP-14SS2 PLC, we used WPLSoft, a program editor specifically built for the Delta DVP. As mentioned above, it is crucial to construct an application-based architecture for our control process, in which hitting the system start push button initiates the process's operation by starting the water pump [10]. The state of the water as it reaches each predetermined level is represented by the led indicators. The pump stops running when it reaches the highest level, and the process ends.

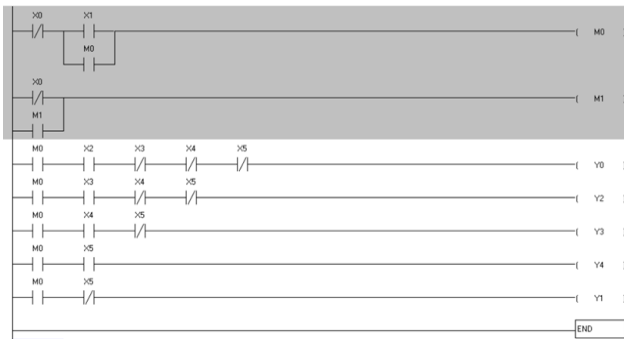


Fig. 5. Ladder Logic for the DELTA DVP-14SS2 PLC

The PLC used for the control block of star-delta starter is Siemens SIMATIC S7-1200, CPU 1212C DC/DC/DC. To program Siemens PLC, we used Tia Portal, a powerful and advanced object-oriented software architecture, specifically developed to program Siemens devices. Siemens follows the same approach as Delta, but not the same programming software. In the second control process, we first design an application-based architecture, such as connecting an induction motor to a star connection, which is connected through a delta connection once it has reached the desired speed [11,12].

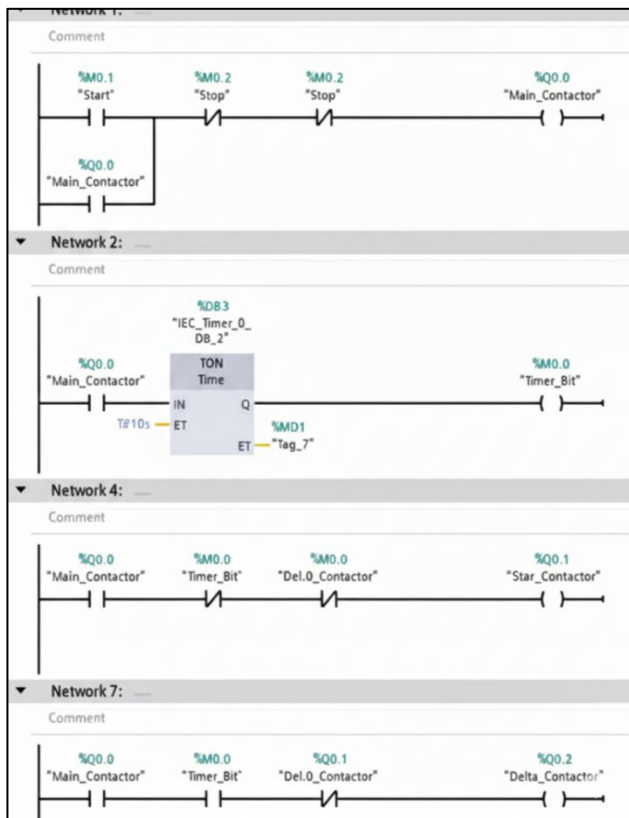


Fig. 6. Ladder Logic for the Process on Siemens SIMATIC S7-1200.

The code block is then compiled and downloaded to their respective PLCs after being created using one of the programming software. Before initiating the downloading process, we must first configure the communication system, which includes certain operations such as selecting hardware models which are being used, and modifying their parameters, the processor model, communication style, transfer

arrangement, port, and other options etc. The program is then put online for simulation to check for faults prior to getting downloaded to the hardware via each specialized communication link.

2.3.2 Fabricating Linking Layer

This step has the core importance in this project. After all the programming, it all comes to the conflict of communication among different protocols. The Linking layer/server is a platform that translates multiple PLCs' hardware communication protocols into a common communication standard that allows data to flow. Under their communication protocols, our multi-brand PLCs carry out all types of communication.

In this regard, the Delta PLC works on the Modbus ASCII serial, meanwhile Siemens works on Ethernet TCP/IP. Different channel is created and devices are added in order to configure them on the single platform. After multiple channels are made for each PLC and their devices are added to them, the configuration platform settings are updated to match the separate hardware protocols for the creation of the linking server. These communication protocols are then translated to a single communication protocol for data transfer and reception, which in our instance is the Data Access (DA) Protocol. Some modifications are made in order to establish the connection. For delta, we modify the device driver, device model, stop bits, parity, data bits, baud rate, and com id. Modifications are also made for Siemens in a device driver, device model, rack and slot value, and device id, which is the allocated IP address [13,14].

After the linking layer is established, there should be passages for communication across networks. This task is achieved by the port incorporation. Port incorporation is the complete inclusion of the serial and Ethernet port to enable multi-protocol communication. This process of port incorporation is so crucial to the system that it occurs multiple times throughout the system. The first time it happens is when we configure a serial port (COM) to be used for the hardware during PLC programming. The bit values must be set appropriately in this case, and the usage of a serial port is even more important for delta because Siemens utilizes an Ethernet interface. When configuring the linking server, the second instance of port inclusion occurs. This occurrence is related to the COM ID and the requirement for it to match the actual port number. The third event occurs after the tag integration process and a little distinction in that is, it does not require any configurations; instead, it is the visible use of the ports when connecting the linking server to our hardware. For Siemens, the Ethernet TCP/IP port is used, while for Delta, any actual serial

COM port is used [15]. The important point is having all relevant ports vacant for any purpose.

After this step, comes the tag integration. Tags are memory registers defined in the linking server to read and write real- time data from the constructed device and exchange data with the SCADA. Every tag of that device is labelled, and addresses are allocated during the integration of tags in the linking server. The tag names are either identical names or the same as their addresses for the easy recognition. Client access of all input and output tags are modified as read/write and their data type is set to Boolean. Depending on the communication protocol used by a PLC, the addressing remains the same or is transformed to specific address types.

2.3.3 Addressing

The addressing in Siemens is simple as the address are allocated same as the PLC tags in Tia Portal.

Table 3

Tag Addressing in Siemens

Name	PLC Address	Tag Address
Start	I0.0	I0.0
Stop	I0.1	I0.1
Main	Q0.1	Q0.1
Star	Q0.2	Q0.2
Delta	Q0.3	Q0.3

In Delta, the addressing is different. Before they can be used as tag addresses, PLC addresses must first be converted to Modbus addresses.

Each type of address has its own format and range of values. The initial bits of the I/O range indicate the first two bits of the format. The remaining bits of the format are represented by the converted DVP address. This conversion is done by taking the DVP address of the PLC address as a hexadecimal value and then converting it to its decimal value.

Table 4

DVP address range g Addressing in Siemens

Device	DVP address (Hex)
M (0-255)	0800-08FF
X	0400-04FF
Y	0500-05FF

For M0,

DVP address= 0800

convert into decimal:

= 2048

As M0 is the first address in the range so we add 1 to the converted value

= 2048 + 1

Modbus Address = Tag Address = 2049

For M1,

DVP address= 0801

convert into decimal:

= 2049

Add 1 to the converted value:

= 2049+ 1

Modbus Address = Tag Address = 2050

Similarly; for X0,

DVP address= 0400

convert into decimal:

= 1024

Add 1 to the converted value:

= 1024 + 1

=1025

For read contacts, we use the range of 10001-20000

Modbus Address = Tag Address = 101025

For X1,

DVP address= 0401

convert into decimal:

= 1025

Add 1 to the converted value:

= 1025 + 1

=1026

For read contacts, we use the range of 10001-20000

Modbus Address = Tag Address = 101026

For X2,

DVP address= 0402

convert into decimal:

= 1026

Add 1 to the converted value:

= 1026 + 1

=1027

For read contacts, we use the range of 10001-20000

Modbus Address = Tag Address = 101027

For X3,

DVP address= 0403

convert into decimal:

= 1027

Add 1 to the converted value:

= 1027 + 1

=1028

For read contacts, we use the range of 10001-20000

Modbus Address = Tag Address = 101028

For X4,

DVP address= 0404

convert into decimal:

= 1028

Add 1 to the converted value:

= 1028 + 1

=1029

For read contacts, we use the range of 10001-20000

Modbus Address = Tag Address = 101029

For X5,

DVP address= 0405

convert into decimal:

= 1029

Add 1 to the converted value:

= 1029 + 1

=1030

For read contacts, we use the range of 10001-20000

Modbus Address = Tag Address = 101030

Similarly; for Y0,

DVP address= 0500

convert into decimal:

= 1280

Add 1 to the converted value:

= 1280 + 1

Modbus Address = Tag Address = 1281

For Y1,

DVP address= 0501

convert into decimal:

= 1281

Add 1 to the converted value:

= 1281 + 1

Modbus Address = Tag Address = 1282

For Y2,

DVP address= 0502

convert into decimal:

= 1282

Add 1 to the converted value:

= 1282 + 1

Modbus Address = Tag Address = 1283

For Y3,

DVP address= 0503

convert into decimal:

= 1283

Add 1 to the converted value:

= 1284 + 1

Modbus Address = Tag Address = 1284

For Y4,

DVP address= 0504

convert into decimal:

= 1284

Add 1 to the converted value:

= 1284 + 1

Modbus Address = Tag Address = 1285

Final addressing table of Delta DVP is follow:

Table 5

Tag Addressing

Name	PLC Address	Tag Address
Memory Bit 1	M0	002049
Memory Bit 2	M1	002050
Start	X0	101025
Stop	X1	101026
Low Level Input	X2	101027
Mid-Level Input	X3	101028
High Level Input	X4	101029
Top Level Input	X5	101030
Low Level LED	Y0	001281
Water Pump	Y1	001282
Mid-Level LED	Y2	001283
High Level Input	Y3	001284
Top Level Input	Y4	001285

Then, we begin the process of interfacing the linking server with the communication networks after completing the appropriate tag integration and platform configuration settings. This entails a port incorporation process in which we first ensure that the appropriate ports are available. We have full liberty to terminate any task that is occupying a required port for the port's accessibility.

After releasing the ports, we connect our controllers to the linking server using their respective communication cords. The hardware and linking layer have been successfully integrated, allowing real-time data acquisition of both PLCs on the linking server at the same time. An easy validation of the interface process is the data quality displayed as "Good" for both devices in their respective channels at the same time.

Item ID	Data Type	Value	Timestamp	Quality	Update Count
DELTA.DVP-14s2.M0	Boolean	1	20:28:17.310	Good	1
DELTA.DVP-14s2.M1	Boolean	1	20:28:17.310	Good	1
DELTA.DVP-14s2.X0	Boolean	1	20:28:17.326	Good	1
DELTA.DVP-14s2.X1	Boolean	1	20:28:17.326	Good	1
DELTA.DVP-14s2.X2	Boolean	1	20:28:17.326	Good	1
DELTA.DVP-14s2.X3	Boolean	0	20:28:17.326	Good	1
DELTA.DVP-14s2.X4	Boolean	0	20:28:17.326	Good	1
DELTA.DVP-14s2.X5	Boolean	0	20:28:17.326	Good	1
DELTA.DVP-14s2.Y0	Boolean	1	20:28:17.342	Good	1
DELTA.DVP-14s2.Y1	Boolean	1	20:28:17.342	Good	1
DELTA.DVP-14s2.Y2	Boolean	0	20:28:17.342	Good	1
DELTA.DVP-14s2.Y3	Boolean	0	20:28:17.342	Good	1
DELTA.DVP-14s2.Y4	Boolean	0	20:28:17.342	Good	1

Fig. 7. Data Acquisition from the Siemens PLC to the Linking Server.

Item ID	Data Type	Value	Timestamp	Quality	Update Count
SIEMENS.S7-1200_Rack	Byte	0	20:28:17.358	Good	1
SIEMENS.S7-1200_Slot	Byte	1	20:28:17.358	Good	1
SIEMENS.S7-1200.DeltaContactor	Boolean	1	20:28:20.364	Good	1
SIEMENS.S7-1200.MainContactor	Boolean	1	20:28:20.364	Good	1
SIEMENS.S7-1200.StarContactor	Boolean	0	20:28:20.364	Good	1
SIEMENS.S7-1200.Start	Boolean	1	20:28:20.364	Good	1
SIEMENS.S7-1200.Stop	Boolean	0	20:28:20.364	Good	1

Fig. 8. Data Acquisition from the Delta PLC to the Linking Server.

2.3.4 SCADA Development

SCADA stands for Supervisory Control and Data Acquisition system, a system that allows you to monitor and control the controlled processes in remote areas. Its major objective is to monitor and control various equipment in industrial operations for both commercial and public entities [16].

In reality, SCADA systems may be found practically everywhere in today's world. Industrial plants, manufacturing, transportation, oil and gas, power distribution, water control, and many other sectors has been utilizing SCADA [17,18]. The SCADA system is comprised of three major elements that must all work together to ensure smooth monitoring and control of control activities. These are the front panel, connector panel, and rear panel/block diagram. The front panel is an application layer made up of the controls and indicators that serves as a user interface [19,20,21]. The connector panel is where tags are imported, and it is this workspace that allows real-time data capture from the connection layer into the SCADA. Finally, the block diagram, which is the graphical source code including cases of structures and functions, is displayed on the back panel. All three levels must be developed in order for the SCADA to be fully completed, and none of them can function without the others [22,23]. The process of development starts with the creation of an I/O server and its configuration parameters, two of which are very critical alteration in the update pace; this varies from project to project, and the choice of server that has been registered in this case, according to our project we choose the update rate of 500- 1000 ms and the update rate of 500-1000 ms. This results in the creation of a medium that allows to communicate by importing the tags from the linking layer as variables and use them in our code. For the animation aspect, SCADA was used. Our connector panel is the medium.

After completing the first phase of designing, which is to interface the linking server with the LabVIEW software for the future SCADA; to put it another way, we create the bound variables. These are integrated tags that we devised in our linking server for both processes, and we perform this task by using a LabVIEW platform through a registered server. Then these variables are dragged onto the workspace of LABVIEW to link them to the graphics' Boolean values. This workspace serves as the connector panel

of the virtual instruments of platform, specifying the inputs and outputs nodes that were constructed in the block diagram. A tag is interfaced on the back-end of every Boolean that we produced to represent our animation. Either it is a button, level, tank, or indication, which is imported from the connecting layer and boxed in a while loop. This procedure is followed only until variables are introduced into the workspace. All the remaining connections of the animation with produced graphics and their Boolean values are completed after the design process [24].

The creation of the front panel, as well as the back panel, is the next step in the connector panel manufacturing process. The front panel functions as a graphical user interface, while the back panel serves as a conduit enabling the terminals and nodes established in the front panel to be used with variables imported in the connector panel. Begin the process by using graphic counterparts of the bound variables or creating them from scratch to fabricate the front panel. The graphic counterparts are accessed via the numeric controls in the primary palette. Each graphic's characteristics can be modified, and this is how new visuals are created from beginning. To generate a customized graphic for the interface, multiple elements from the palette 'controls and indicators', such as scale, range, and look, can be modified. They have a distinct palette for buttons and switches, which consists of five primary categories. It all works on a true- false basis, and the main differences between them are in their appearance, operation, and function depending upon the characteristic of the animation. The front panel was designed according to the processes used in this project. In the designed SCADA front panel, the representation of all of the palette visuals are combined with library graphics can be shown.

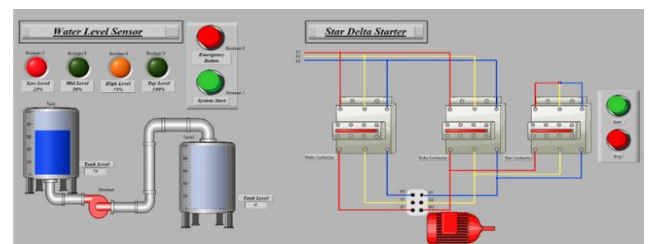


Fig. 9. Front Panel of designed SCADA



Fig. 10. Back Panel of designed SCADA

Finally, we design the back panel. Back panel is basically a graphical source code, just like a block diagram. In the back end, all of the animation we

created on the front panel are shown as terminals. With these terminals, structures are also created, which comprise functions, variables, and nodes for the control processes' animation and to accomplish the operations at hand. These structures are created as subroutines for a bigger whole, which in this case is the totality of the control processes, in order to administer the controls and supply data for the indicators. For the feasibility of a continuous process, these case structures are bound in loops.

After completing the designing process, proceed to the last step. This will detach the connecting layer, store any recent tag adjustments made in the LabVIEW environment, and reconnect the layer with the PLCs, control processes, and selected SCADA [25]. The real-time animation with simultaneous data collecting begins when the SCADA is turned on. As the physical control procedures are carried out, their virtual representation begins in the SCADA. Any actions taken in the SCADA have a physical effect, providing for master control and override authority.

3. Results and Analysis

3.1 Findings

The program began by activating the system enabling the SCADA, which allowed the project to begin.

SCADA, database, and linking layer was simultaneously enabled by the system. Following this, the power source is plugged in to energize and power up the PLCs, each of which required 24 volts. The database initially displayed the different channels; Siemens and Delta, as well as their corresponding connections, during runtime. Now without reconnecting the linking layer, these connections were then extended to determine the address, and data type for all the tags involved in each channel configured in the layer.



Tag Name	Address	Data Type	Scan Rate	Scaling
Y4	001285	Boolean	100	None
Y3	001284	Boolean	100	None
Y2	001283	Boolean	100	None
Y1	001282	Boolean	100	None
Y0	001281	Boolean	100	None
X5	101030	Boolean	100	None
X4	101029	Boolean	100	None
X3	101028	Boolean	100	None
X2	101027	Boolean	100	None
X1	101026	Boolean	100	None
X0	101025	Boolean	100	None
M1	002050	Boolean	100	None
M0	002049	Boolean	100	None

Fig. 11. Database of Delta DVP-14ss2



Tag Name	Address	Data Type	Scan Rate	Scaling
Stop	I0.1	Boolean	100	None
Start	I0.0	Boolean	100	None
StarContactor	Q0.1	Boolean	100	None
MainContactor	Q0.0	Boolean	100	None
DeltaContactor	Q0.2	Boolean	100	None

Fig. 12. Database of Siemens S7-1200

By preserving the changes made to the linking layer, a strong and safe connection was established. After that, the quick client was used to check the status of the tags to see if they were good, bad, or unknown. Any tags that were ruled out because of uncertain status were all changed until each channel's tags appeared to be good. After that, we started the SCADA application, which first displayed the project's running front page. The day/date/time were utilized to assess whether it was presenting the data in real time for the verification of its operability.

After that, we proceed to the process page. The water pump began operating as soon as the process was started, filling the empty main tank with water from the storage tank. By changing the color on the SCADA screen, this running water pump was graphically depicted to be in on state. The relevant LED turned on as the tank filled up and reached the various level on the conduction rod that served as a level sensor. Simultaneously, new data was recorded to database tags, which were subsequently acquired by the animation in the SCADA, which also exhibited the switched on same led, complete with its own visual representation of the tanks and numeric level display as it happened in real-time.

All the results obtained from the method discussed in the paper can be further expanded to include multiple processes of a wide variety designed on various contrasting devices by adding and configuring channels onto the linking layer and adding more pages to the SCADA screen, while still capable of simultaneously acquiring data for each process in one place in the database.

3.2 Analysis

Different PLCs interact with various systems and software, and the communication is proprietary, therefore a dedicated SCADA is normally present for each PLC controlling the processes [26], [27]. When industries try to make them communicate with each other in a particular region, challenges occur due to communication protocol incompatibility and conflicts. For this problem, usually hard wiring or physical equipment such as protocol converters are generally used to solve compatibility difficulties. There was a little percentage of error in terms of information latency when we compared our software-to-software approach to the commonly used hardware-based approach. The reason is linking layer, which is acting as a server, with PLCs of various communication protocols acting as clients, and all data manipulation, reading, and writing of data took place directly through conversion to related property, the data access protocol which cause the Siemens PLC experienced a half-second delay. Any changeover that happened in real time in the process happened after a

little delay. Aside from that minor predicament, the use of a direct interface proved to be a very accurate strategy in terms of data collecting. All of the data and information was accurately gathered at a checkpoint in the database and was efficiently integrated into the SCADA [28].

4. Conclusion

In this paper, we propose an optimum solution for a snag that is commonly experienced by the large-scale industries while integrating the multi-brand PLCs. There can be numerous motives for addressing this problem. What if an organization is on a tight budget and can only afford a collection of PLCs that don't communicate with one another or work on different protocols or acquired some really costly PLCs that aren't compatible by accident? What if you have two distinct PLCs doing two interdependent jobs, but they use different communication protocols, and your system is now unable to communicate between the modules? There could be a slew of problems like this that need to be addressed. The objective of this project is to present an initiative of a system that provides us with an interface to make communication possible between multi-brands of PLCs, translating the data among them to obtain a communication system with minimal human intervention, to produce a middle layer between the PLCs which can work as a translator among them, the connection and interface of multiple PLCs through the control system architecture of choice. This system will gather data over the network from the PLCs and send that information along with the created network of PLCs.

For creating such architecture, an interface is designed to make communication possible between multi brands of PLCs in one place through control system architecture with minimal human intervention.

It functions as a graphical user interface with the addition of data acquisition, reflecting real-time data from the processes, allowing control and overwriting the commands. This contains an internal translating and linking layer that converts the various protocols of various devices into a common protocol and allows communication between physical processes and the SCADA. It is basically designed as an imitation of a typical industrial setting, with two conventional processes commonly used by industries, i.e. water level indicator and control, and star-delta starter, developed individually on each of the PLCs. The water level indicator and control system is made up of a conduction rod and a sensory mechanism with four levels that detect varying degrees of water levels in order to maintain the main tank full via an overhead tank. The star-delta starter is the second process. It is controlled by a start button, which allows the programmed PLC to deliver electric impulses to the

output relays. The star-delta starter is the second process. It is controlled by a start button, which allows the programmed PLC to deliver electric impulses to the output relays, and a three-phase induction motor connected on the other end. The system and approach would be especially useful for small-scale enterprises on a tight budget who want to merge different processes running on distinct SCADA screens onto a single screen without having to spend thousands of rupees on expensive hardware-based protocol converters. It could also benefit educational institutions that build similar trainers or systems to use SCADA for more effective learning among students.

APPENDIX A

Appendix A1: Range of Resources

Every invention begins with an idea, but many ideas never make it past the framing stage due to the numerous elements that must be considered in order to turn them into a tangible form. The twists that develop into grounds for failure are usually things like base structures, plans, budgets, and, more commonly, gathering all kinds of resources.

Aside from these types of resources, external technical and labour assistance from human resources and specialists such as technicians, mechanics, and electricians may also be useful in achieving the results.

Human Resource			
S. No.	Discipline	Resource	Description
1	Woodworking	Carpenter	Due to the fact that we were not familiar with woodwork as engineering students, some labor support was acquired from a carpenter to help build the wooden casing with an acrylic top for the project.
Reference Literature			
S. No.	Discipline	Resource	Description
1	Engineering	Articles/ Research Papers	Literature to understand the basics behind Industrial Automation and PLCs.
Material/Equipment			
S. No.	Discipline	Resource	Description
1	Engineering (Software)	LabVIEW	LabVIEW is licensed designing software used to design a gui-esque SCADA to acquire the data from the processes and act as a supervisory control unit.
2	Engineering (Software)	TIA Portal	Tia Portal is a licensed programming software used to program the Siemens Simatic S7-1200 PLC with ladder logic designed specifically for the star-delta starter.
3	Engineering (Hardware)	PLC (Delta DVP-14SS2, SIEMENS SIMATIC S7-1200)	PLC is a controller programmed with the respective ladder logic used to run our replication of two standard industry processes i.e. water level indication and control and star-delta starter.

Appendix A2: Consequences to Society and Environment

There aren't per se any negative consequences to the environment with the implementation of this project. This is because it is based on a software approach. That being the case, there is not a ton of power usage; the only such usage would be to power the system on which the interface is running and the control processes. And because these control processes are more of a side venture and the interface is meant to be solely run as a standalone system when adapted in a professional industrial environment, it further decreases power consumption by a large percentage. Other than that, it does not contribute to any dangerous emissions or waste that may harm the environment either.

Appendix A3: Level of Interaction

During the implementation phase, the project underwent a couple of different techniques after the original solution of using a network extension failed as it was not compatible with one of the device's protocols. Other solutions proved to be accessible for all kinds of protocols but were limited to solely simulation over a physical connection which is why those were too rejected. To overcome the barrier the approach of configuring a linking layer with a virtual port extension of channels was implemented which proved to be a successful method of resolution.

The other complication that emerged, later on, was the availability of a license for LabVIEW for the SCADA. Due to the extensive work and the realization that a six-month time as a free trial period would not be sufficient, we decided that it would be best to purchase a full-fledged licensed version of the software.

5. References

- [1] M. Tyrovolas and T. Hajnal, "Inter-communication between Programmable Logic Controllers using IoT technologies: A Modbus RTU/MQTT Approach," in *arXiv preprint arXiv:2102.05988*, Feb. 2021.
- [2] K. Alremeithi, H. Almaeeni, and W. Sealy, "Virtualized digital twin (DT) of a reconfigurable programmable logic controller (PLC)," in Proc. 2024 6th Int. Conf. Reconfigurable Mechanisms and Robots (ReMAR), Jun. 2024, pp. 349–354.
- [3] M. Gaffurini, P. Bellagente, A. Depari, A. Flammini, E. Sisinni, and P. Ferrari, "Virtual PLC in industrial edge platform: Performance evaluation of supervision and control communication," *IEEE Trans. Instrum. Meas.*, 2024, doi: [insert DOI if available].
- [4] K. Charoensuk, T. S. N. Ayuthaya, J. Nangtin, and K. Pawanant, "The monitoring and controlling of wireless communication with PLC via internet access," in Proc. 2024 12th Int. Electr. Eng. Congr. (IEEECON), Mar. 2024, pp. 1–7.
- [5] PLC, "How PLCs control the program," [Online]. Available: <https://instrumentationtools.com/basics-of-industrial-communication-networks/>. [Accessed: Jan. 3, 2025].
- [6] R. A. Rajput, D. S. Nikam, and D. Bhowmick, "PLC & SCADA for efficient load management and fault monitoring," in Proc. 2024 IEEE Int. Students' Conf. Electr., Electron. Comput. Sci. (SCEECs), Feb. 2024, pp. 1–5.
- [7] K. Vadivukkarasi and S. Bhargava, "Mixing and filling operations using PLC, SCADA and IoT," in Proc. 2024 7th Int. Conf. Circuit Power Comput. Technol. (ICCPCT), vol. 1, Aug. 2024, pp. 635–639.
- [8] R. Bansal and A. K. Dubey, "Communication protocols used for industrial automation," in *Comput. Intell. Ind. 4.0*, CRC Press, 2024, pp. 73–94.
- [9] Star-Delta Starter, "Wiring of star-delta starter," [Online]. Available: <https://realpars.com/star-delta-part-5/>. [Accessed: Jan. 3, 2025].
- [10] K. H. Kaittan and S. J. Mohammed, "PLC-SCADA automation of inlet wastewater treatment processes: Design, implementation, and evaluation," *J. Eur. Syst. Automatisés*, vol. 57, no. 3, 2024.
- [11] R. A. Rajput, D. S. Nikam, and D. Bhowmick, "PLC & SCADA for efficient load management and fault monitoring," in Proc. 2024 IEEE Int. Students' Conf. Electr., Electron. Comput. Sci. (SCEECs), Feb. 2024, pp. 1–5.
- [12] L. M. Kadhum, S. J. Mohammed, and Q. Al-Gayem, "Advanced SCADA-PLC S7-1200 communication system for aeration unit in wastewater treatment stations," *J. Eur. Syst. Automatisés*, vol. 58, no. 4, p. 747, 2025.
- [13] Delta PLC, "Specifications of DELTA DVP14SS2," [Online]. Available: https://www.researchgate.net/figure/Specifications-of-delta-DVP14SS2_tbl1_316342684. [Accessed: Jun. 3, 2025].
- [14] V. D. Cong, "Design and development of a cost-efficiency robot arm with a PLC-based robot controller," *FME Trans.*, vol. 52, no. 2, 2024.
- [15] Y. Yan, Z. Liu, and S. Liu, "Research and application of automatic obstacle avoidance

- and handling system based on PLC,” in Proc. 2024 36th Chin. Control Decis. Conf. (CCDC), May 2024, pp. 3325–3330.
- [16] SCADA, “What is SCADA,” [Online]. Available: <https://www.watelectronics.com/scada-system-architecture-types-applications/>. [Accessed: Jul. 3, 2025].
- [17] S. Prasad, S. Kadam, T. Dhanadhya, D. Kanase, and S. Jadhav, “PLC-SCADA based online monitoring and conditioning of laboratory,” in Proc. 2024 Int. Conf. Intell. Syst. Adv. Appl. (ICISAA), Oct. 2024, pp. 1–8.
- [18] C. A. Fonseca, “SCADA system of pipelines,” in Handbook Pipeline Eng., Cham: Springer Int. Publishing, 2024, pp. 1–28.
- [19] S. Ferlito, S. Ippolito, C. Santagata, P. Schiattarella, and G. Di Francia, “A study on an IoT-based SCADA system for photovoltaic utility plants,” *Electronics*, vol. 13, no. 11, p. 2065, 2024.
- [20] D. Upadhyay, M. Zaman, R. Joshi, and S. Sampalli, “An efficient key management and multi-layered security framework for SCADA systems,” *IEEE Trans. Netw. Serv. Manag.*, vol. 19, no. 1, pp. 642–660, 2021.
- [21] A. Tokos, C. Bartha, M. Jipa, D. D. Micu, and I. Lingvay, “SCADA systems for wastewater treatment plants,” *Electrotehnica, Electronica, Automatica*, vol. 69, no. 3, pp. 39–45, 2021.
- [22] SCADA, “How SCADA system works,” [Online]. Available: <https://www.dpstele.com/scada/how-systems-work.php> [Accessed: Jul. 3, 2025].
- [23] Siemens, “Communication between SIMATIC S7-CPUs on PROFIBUS and Allen-Bradley controllers via SST PROFIBUS scanner,” Nov. 14, 2006. [Online].
- [24] J. B. Heluany, “PLC honeypots: Enhancing interaction-level assessment,” *Electronics*, vol. 13, no. 20, p. 4024, 2024.
- [25] H. Altaleb and R. Zoltán, “A comprehensive analysis and solutions for enhancing SCADA systems security in critical infrastructures,” in Proc. 2024 IEEE 11th Int. Conf. Comput. Cybern. Cyber-Med. Syst. (ICCC), Apr. 2024, pp. 1–6.
- [26] I. Vujović, M. Koprivica, and Ž. Đurišić, “Infrastructure for distributed PV systems monitoring and management using SCADA,” in Proc. 2024 32nd Telecommun. Forum (TELFOR), Nov. 2024, pp. 1–4.
- [27] T. Covrig, L. Miclea, and A. Ciobotaru, “Development of a simulated industrial process and its control with a PLC,” in Proc. 2024 IEEE Int. Conf. Autom., Qual. Test., Robot. (AQTR), May 2024, pp. 1–4.
- [28] M. B. Khaleeq, A. K. Yadav, M. Sarfraz, M. K. Siddiqui, and M. Shakir, “PLC based humidity and carbon dioxide monitoring and control system for industries using SCADA technology,” in Proc. 2024 2nd Int. Conf. Comput. Characterization Techn. Eng. Sci. (IC3TES), Nov. 2024, pp. 1–6.