FH AACHEN
UNIVERSITY OF APPLIED SCIENCES

CIDESI®

Design and implementation of a control system for robotic assembly station oriented to Industry 4.0

A Thesis

Submitted to the Fachhochschule Aachen, Germany
and Centro de Ingeniería y Desarrollo Industrial, Mexico

BY

Luis Humberto Rodríguez Verástegui

In Partial Fulfillment of the requirements for the degree of Master of Science in Mechatronics

Santiago de Queretaro, Qro., Mexico, February 2018

Declaration

I hereby declare that this thesis work has been conducted entirely on my own accord.
The data, the software employed and the information used have all been utilized in complete agreement to the copyright rules of concerned establishments.

Any reproduction of this report or the data and research results contained in it, either electronically or in publishing, may only be performed with prior sanction of the University of Applied Sciences and myself, the author.

Queretaro, February 2018

Luis Humberto Rodríguez Verástegui

Acknowledgments

I would like to extend my most cordial and sincere gratitude to everyone who gave me their support during the realization of the project. Without them, it would not have been possible to complete the task.

To CONCAYT for the economic support provided throughout the Master's degree both in Mexico and in my stay in Germany.

To Prof. Dr.-Ing. Jörg Wollert for the opportunity to collaborate in the realization of this type of automation projects.

To Dr. Leonardo Baldenegro for his tutoring in the elaboration of the Thesis.

To CIDESI and the personnel in charge of the Mechatronics Master's Degree for all the facilities and attention for obtaining the degree.

To the brand Wago® for providing the materials for the realization of the project.

To Sebastian Rau, Florian, Felix, Mark, Sebastian and the colleagues of the embedded systems laboratory for their advice and facilities provided in the long days of work and design meetings.

To Jaime, Ricardo, Jose, and Osvaldo for this brotherhood we built, friends and teammates. Thanks to their wisdom, experience, and patience we could achieve many things together.

To my friends in Germany Camilla, Jens, Aileen, Antonio, Chizuru and Emiko who welcome me into their lives, for all the experiences shared and good times together.

To Ricardo, Alfredo, Fabian, Dr. Fabian and Felix Schreiber for allowing me to continue growing out of college and consider myself part of their projects and life for much longer.

Dedication

To my brother Fernando, the engine that drives me to keep moving forward

To my grandparents, who have been my role model and a great inspiration

To my mother, who is the heart and soul that I always carry with me

Abstract

This Thesis aims to develop an industrial control system for a LEGO® bricks vehicle assembly station oriented to demonstrate the capabilities and concepts of Industry 4.0. The assembly station must be capable of being transported for display at demonstrations and conventions. The assembly station is designed to house two robots, one for assembling and the other for supplying materials. The robot proposed for the assembly of these bricks is a low-cost cylindrical robot with 3 degrees of freedom, for which the control system is designed.

The control system is based on a PLC that executes the control logic for the robot and communicates with an ERP system that manages the product models that must be assembled. The safety system for the assembly station and the electrical panel are also designed.

The methodology used in the development of the project is based on the V model of the VDI2206 standard, starting with the definition of the system requirements, the elaboration of a general concept and specific concepts until the development of the prototype.

In the developed assembly station, the integration of components is done in a modular way, in such a way that new adaptations or changes in the system can be made directly.

The management of resources and tools within the station is designed considering the scalability of the system and the possible integration into a network of similar assembly stations based on mass customization and flexible assembly technologies.

Resumen

Esta Tesis tiene como objetivo el desarrollar un sistema de control industrial para una estación de ensamble de vehículos compuestos por ladrillos LEGO® orientada a la demostración de capacidades y conceptos relacionados a la Industria 4.0. La estación de ensamble debe ser capaz de ser transportada para su exhibición en demostraciones y convenciones. La estación de ensamble está concebida para albergar 2 robots uno de ensamble y otro para abastecimiento de materiales. El robot propuesto para el ensamble de dichos ladrillos es un robot cilíndrico de 3 grados de libertad de bajo costo, para el cual se diseña el sistema de control.

El sistema de control está basado en un Controlador Lógico Programable que ejecuta la lógica de control para el robot y se comunica con un sistema ERP quien administra los modelos de productos que deben ser ensamblados. El sistema de seguridad para la estación de ensamble y el tablero eléctrico también son diseñados.

La metodología empleada en el desarrollo del proyecto está basada en el modelo V del estándar VDI2206 comenzando en la definición de los requisitos del sistema, la elaboración de un concepto general y conceptos particulares hasta llegar al desarrollo del prototipo.

En la estación de ensamble desarrollada la integración de componentes se hace de forma modular, de tal manera que nuevas adaptaciones o cambios en el sistema se pueden hacer de forma sencilla.

La administración de los recursos y herramientas dentro de la estación está diseñada considerando la escalabilidad del sistema y la posible integración a una red de estaciones de ensamble similares basada en las tecnologías de la industria 4.0.

Kurzfassung

Ziel dieser Masterarbeit ist es, ein industrielles Steuerungssystem für die Montage eines Fahrzeugs aus LEGO® Steinen zu entwickeln, das die Fähigkeiten und Konzepte von Industrie 4.0 demonstriert.

Hierbei wurde beachtet das die Montagestation für Demonstrationen auf Messen und Kongressen transportiert werden kann.

In der Montagestation sind zwei Roboter untergebracht, einer für die Montage und der andere für die Materialversorgung. Der für die Montage dieser Steine verwendete Roboter ist ein kostengünstiger zylindrischer Roboter mit 3 Freiheitsgraden, für den das Steuerungssystem ausgelegt ist.

Das Steuerungssystem basiert auf einer SPS, die die Steuerlogik für den Roboter ausführt und mit einem ERP-System kommuniziert, das die zu montierenden Produktmodelle verwaltet. Das Sicherheitssystem für die Montagestation und die Schalttafel sind ebenfalls für das Steuerungssystem ausgelegt.

Die bei der Entwicklung des Projekts verwendete Methodik basiert auf dem V-Modell des VDI2206-Standards, beginnend mit der Definition der Systemanforderungen, der Ausarbeitung eines allgemeinen Konzepts und bestimmter Konzepte bis zur Entwicklung des Prototyps.

In der entwickelten Montagestation erfolgt die Integration von Komponenten modular, so dass neue Anpassungen oder Änderungen im System direkt vorgenommen werden können.

Die Verwaltung der Ressourcen und Werkzeuge innerhalb der Station erfolgt unter Berücksichtigung der Skalierbarkeit des Systems und der möglichen Integration in andere dem Netzwerk ähnliche Montagestationen auf der Grundlage von Massenanpassungen und flexiblen Montagetechnologien.

Contents

List of Figures

List of Tables

List of Abbreviations

FH Aachen	Fachhochschule Aachen
IEC	International Electrotechnical Committee
DIN	Deutsches Institut für Normung
SysML	System Modeling Language
ICS	Industrial Control System
SCARA	Selective Compliance Assembly Robotic Arm
PLC	Programmable Logic Controller
CoDeSys	Controller Development System
ERP	Enterprise Resource Planning
IDE	Integrated Development Environment
3DOF	Three Degrees of Freedom
5DOF	Five Degrees of Freedom
HMI	Human Machine Interface
GUI	Graphical User Interface
LAN	Local Area Network
IP	Internet Protocol
TCP	Transmission Control Protocol
HTTP	Hypertext Transfer Protocol
XML	Extensible Markup Language
URL	Uniform Resource Locator
JSON	JavaScript Object Notation
AS-i	Actuator Sensor Interface
EDM	External Device Monitor
OSSD	Output Signal Switching Device
TFT	Thin Film Transistor
LCD	Liquid Crystal Display
LED	Light Emitting Diode

1. Introduction

Problem Definition

In the school environment, the generation of specialized technical skills by students is a topic that university professors must deal with generation after generation. An efficient and illustrative way, mainly in the engineering colleges, to generate this type of capabilities is using didactic workstations, whereby reducing a system to its purest form, students can understand its functioning under conditions defined.

That is why the University of Applied Sciences of Aachen (FH-Aachen), through the professors of the Faculty of Mechanics and Mechatronics, develops projects for the design and manufacture of said teaching stations.

Justification

Based on this philosophy of capabilities building, the Department of Mechatronics and Embedded Systems works as a link between students and emerging technologies related to Industry 4.0; which is a trend that in recent years has taken quite a lot of force in the automation of processes and management of information technologies. Encouraging students to design and build teaching stations to demonstrate concepts related to the topic for other students, and on the other hand, that can be exhibited at fairs and conventions.

Using tools such as LEGO® DUPLO® bricks, is possible to simulate the fundamental behavior of an assembly cell, so that if this tool is coupled with commercial, industrial components that are used in process automation; Students generate knowledge and gain direct professional experience.

Therefore, there is the opportunity to design a low-cost assembly station that, together with other stations designed by other students in the past, can demonstrate the concepts related to industry 4.0 and intelligent factories (flexible manufacturing cells).

The development of this machine is part of a more extensive project that consists of the creation of a network of several LEGO® car assembly stations with different assembly capabilities. The network is also connected to an enterprise resource planning platform which is responsible for providing support for production planning, purchasing, and inventory management, as shown in Figure 1-.

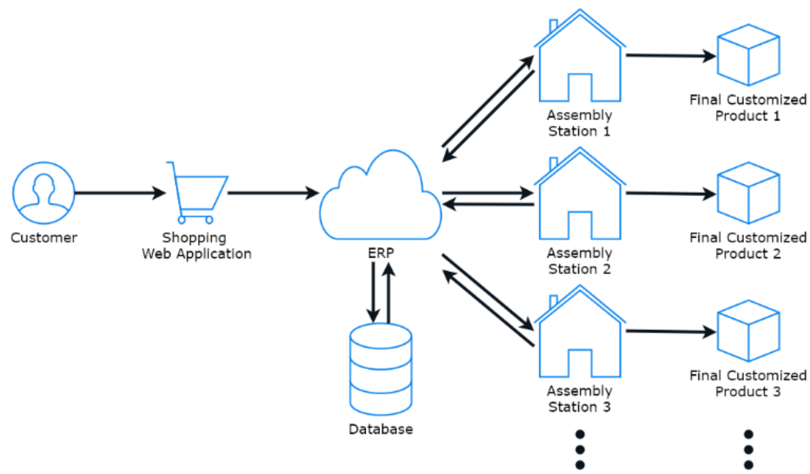


Figure 1-1 General Network topology

This concept of topology aims to improve the efficiency of the elaboration of the cars identifying the station that has higher availability and capacity to execute this task, thus reducing the delivery time of the product and diversifying the customization of the finished product.

Within the sister assembly stations described in this document, there are two that are already integrated into the entire system and two more stations that are under development and close to joining the network. Each of them has a different method to assemble the final product and features regarding its logic and components that make each of them can assemble different configurations of the final product.

The first one, which is already integrated into the network, has a Cartesian robot, composed of components of the IGUS® brand, as shown in Figure 1-2. With which it takes the pieces stored in small slides with a small inclination, and transports them to a conveyor belt where the robot performs the assembly of the product. Once the assembly is finished, the band transports the product to an exit tray.

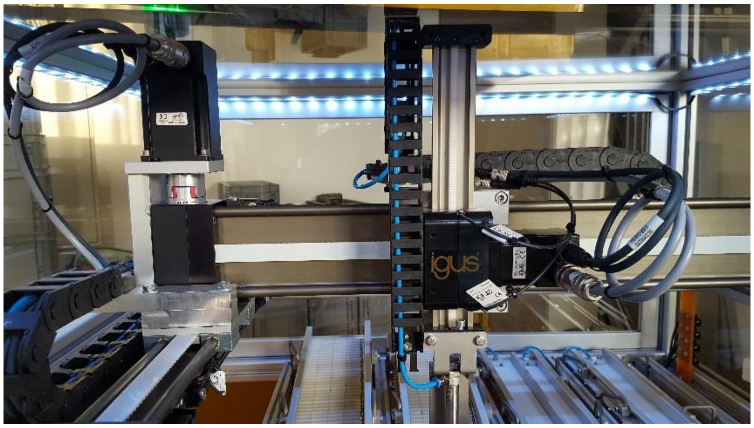


Figure 1-2 Cube Assembly Station

This station has a pneumatic system that injects pressurized air into the slides to align the stored parts; it also supplies air for a cylinder that it uses as a final effector. The control of this station is based on a PLC of the B&R® brand as well as the rest of the electrical components. It also has a computer in which a portable version of the ERP system is integrated to facilitate demonstrations outside the university facilities.

The second station integrates a parallel robot of the ABB® Flex Picker® line; this station has a system of conveyor belts -without inclination- which position the pieces inside the robot workspace, which once the assembly is finished, takes the final product and transfers it to a finished product exit slipper.

This station also has a pneumatic system that supplies air for a suction cup that the robot uses as a final effector. The control of this station uses the ABB's robot driver; Due to the robustness of the robot that is implemented in this station, as can be seen in Figure 1-3, an industrial safety system based on the AS-i protocol is also included. For reasons of dimensions, this station is only available for demonstrations within the university facilities, its integration into the network is still under development.

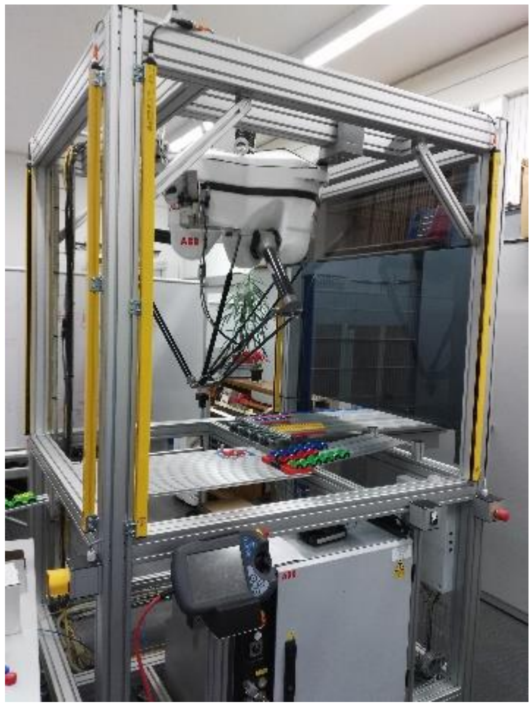


Figure 1-3 ABB® FlexPicker® Assembly Station

The third station also has a Cartesian robot for the transfer and assembly of the product, the difference between this station based on B&R® lies in the components of the control system. This station, shown in Figure 1-4, uses electronic components of the non-industrial level and the controller is an embedded Arduino system, and the control logic is made in LabVIEW. All the structural components were manufactured by additive manufacturing. This station is not integrated into the network but is available for demonstration.

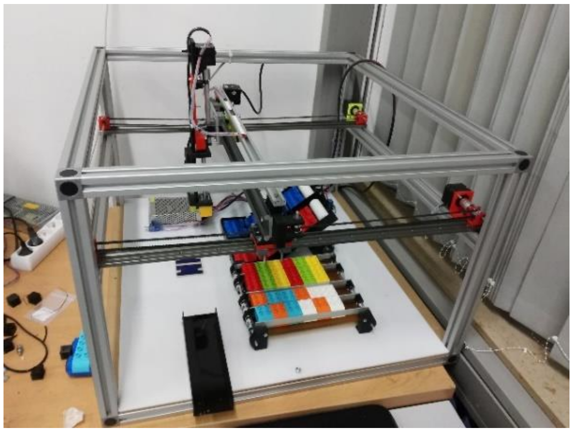


Figure 1-4 3D Printed Assembly Station

The last type of station that is connected to the network does not integrate manipulator robot for the assembly of the product, on the contrary, it is a manual operation station, and the operator is assisted by an augmented reality system that gives indications during the assembly. This station uses conveyors and racks to store the pieces that will be used, in addition to being a manual assembly, shown in Figure 1-5. With this type of station is possible to make orders that include unique pieces and complex configurations.

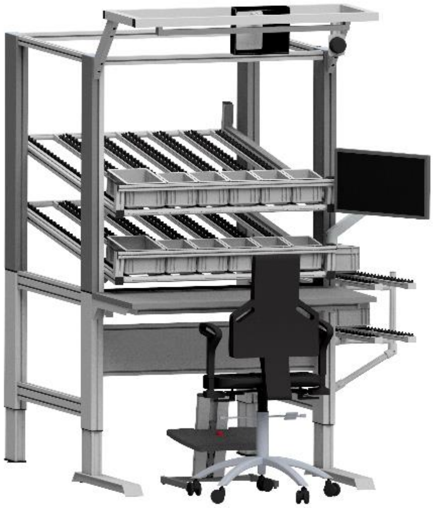


Figure 1-5 ARMW Augmented Reality Manual Workstation

Objectives

This compact LEGO® bricks car assembly station is meant to show and demonstrate the capabilities of the Industry 4.0. It involves the next specific tasks:

- Development of a robotic system to custom-made assembly cars made of LEGO® Duplo® bricks
- Design and implementation of the electrical system for the controller of the robot
- Implementation of the control system for the robotic system aiming the Industry 4.0 philosophy
- Development of a Human Machine Interface for the operation of the machinery

Fundamentals

Assembly Stations

Assembly lines date back to early in the twentieth century, initially using the patent filed by Ransom Olds, the founder of Oldsmobile, in 1901, but more importantly because of Henry Ford who perfected the concept in 1913 by adding conveyor belts. Since then, the concept of pure assembly lines has evolved into an effort to adapt to the ever-increasing sophistication of production systems.[1]

In today's competitive world, manufacturing industry is forced to be in constant evolution in response to the market requirements of shorter product life-cycles, increased productivity with the same amount of resources and shorter time-to-market. [2] They also find the need to make their systems flexible to counter external demand variability, while dealing with internal process variability.

In response to these requirements, the assembly system configuration has evolved from Dedicated Assembly Systems (DAS) where automated and rigid systems allow high efficiency in the mass production era, to Reconfigurable Assembly Systems (RAS) where quick production adjustment system is leading into mass customization era.[3]

Even talking about customization in the assembly process, it is possible to classify between mass customization and simple customization, for this Flexible Assembly Systems (FAS) are defined to deal with different types of products, with increased flexibility regarding a variety of marts/models to manage. If the parts or models that need to be managed to make more complex products, Manual Flexible Assembly Systems (MFAS) are charged to handle by workers means this kind of required tasks. In the Figure 1-6, is shown a comparative chart between these different types of Assembly systems according to production volume and variety.

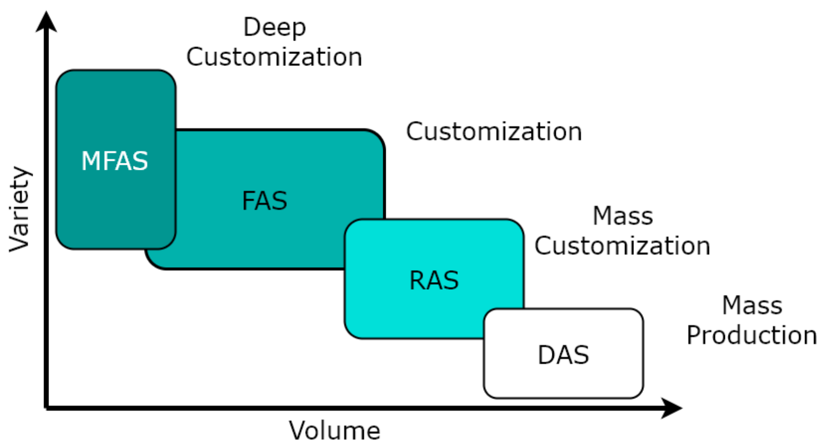


Figure 1-6 Comparative Graph between Assembly Systems

Of all the assembly systems mentioned, those that have the highest level of flexibility regarding the number of assembled models. Because of this ability of simultaneously assembling a variety of product types of small to large batches, they can be divided into two types:

- Robotic Assembly Lines (RAL): flow type systems constituted by a series of robotic assembly stations connected through an automated material handling system.[4]
- Robotic Flexible Assembly Cells (RFACs): highly modern systems, structured with industrial robots and assembly stations, capable of assembling several types of products in small quantities[4].

From these two categories, RFAC has several advantages over RAL, particularly in flexibility and dexterity to assemble a variety of products using the same equipment, plus RFAC is easy to modify and reconfigure.[5] In Figure 1-7 is shown an example of the layout of an RFAC.

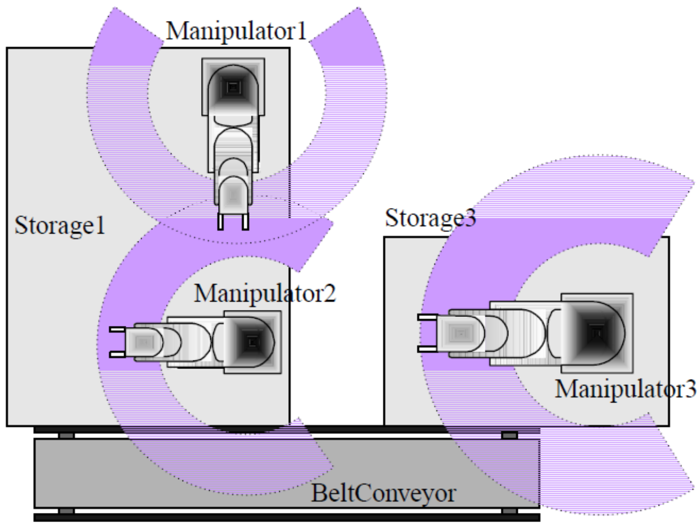


Figure 1-7 Robotic Assembly Cell Example [6]

Industry 4.0

The philosophy of the industry 4.0 in recent years has been a trend established by Germany that more and more countries have been adopting. Industry 4.0 is considered the fourth industrial revolution emerging from an industrial revolution, which is led by intelligent manufacturing.[7] The primary purpose of industry 4.0 is to create a highly flexible production model of client-oriented customized products and services through real-time bonds between all the stakeholders of the production process.

In this way production systems tend to evolve in more intelligent systems using digital systems, these production systems that are going through this transformation have acquired the name of factories based on knowledge and are characterized by having significantly improved measure their efficiency and competitiveness.

As a summary, in the evolutionary history of technology, there are three chapters known as revolutions, which have forged what up to now are the production processes. The first industrial revolution was characterized by the implementation of energy generated by steam and the development of machine tools; The second revolution brought with it the use of electricity and the creation of assembly lines; the third revolution managed to improve manufacturing processes by integrating automated systems and information technologies. Thus, the fourth revolution proposes the unification of the world of information with the real world using SCP. This evolution can be described in detail in the Figure 1-8.

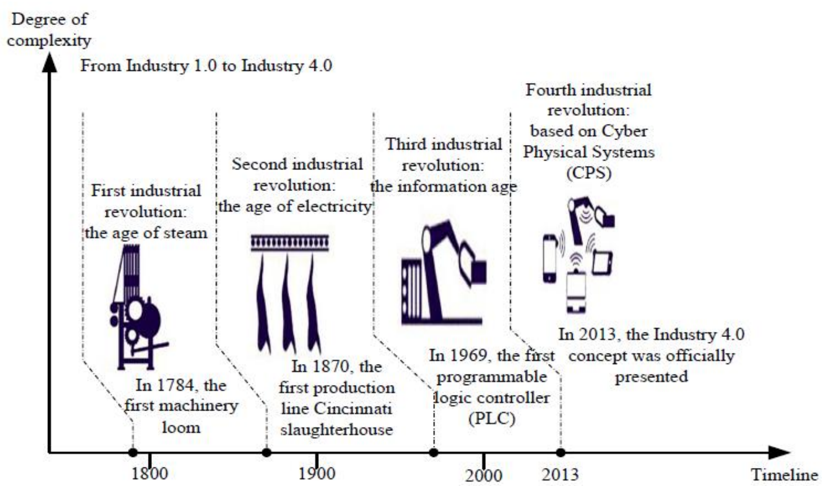


Figure 1-8 Industry Evolution Timeline[7]

Industry 4.0 is a sophisticated and flexible concept that covers various technological areas. Therefore, this philosophy is based on four sets of technologies, which can be observed in the Figure 1-9.

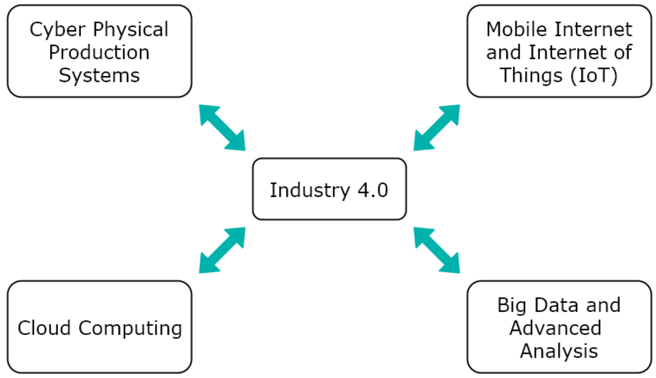


Figure 1-9 Industry 4.0 Key Technologies

Cyber-Physical Production systems: Industry 4.0 conceives a platform that connects the virtual space with the real world, thus allowing the equipment inside a factory to become intelligent and in this way, create better production conditions.

Mobile Internet and Internet of Things: Industry 4.0 strategy makes constant use of the Internet to create the interaction between men and machines giving way to smart manufacturing and production processes. Today several industrial processes can easily be controlled and monitored by applications on smart mobile devices.

Cloud Computing: This technology refers to the possibility of using low cost and high-performance platforms for computing engines through internet services. Within the range of tools covered by cloud computing are virtualization technologies or resource sharing.

Big Data and Advanced Analysis: Using new processing modes to quickly extract relevant information from different types of data, to obtain decision making based on more in-depth knowledge.

This combination of technologies leads to new opportunities and applications in production systems.[8] If all these technologies are applied to the assembly systems, it should be affected in the following ways:[3]

- Reduce costs of automation
- Assembly stations would be managed as software agents
- reduces setup cost and speed up the learning curve of the process
- Allow assembling different product families in the same system
- Improve error identification and handling
- Enable smarter material handling systems

It can be said that the end-result of Industry 4.0 is the creation of smart factories which have decentralized manufacturing logic through intelligent products in flexible integrated production systems for consistent engineering along the value chain. In the Figure 1-10, is shown the general concept of a smart factory.

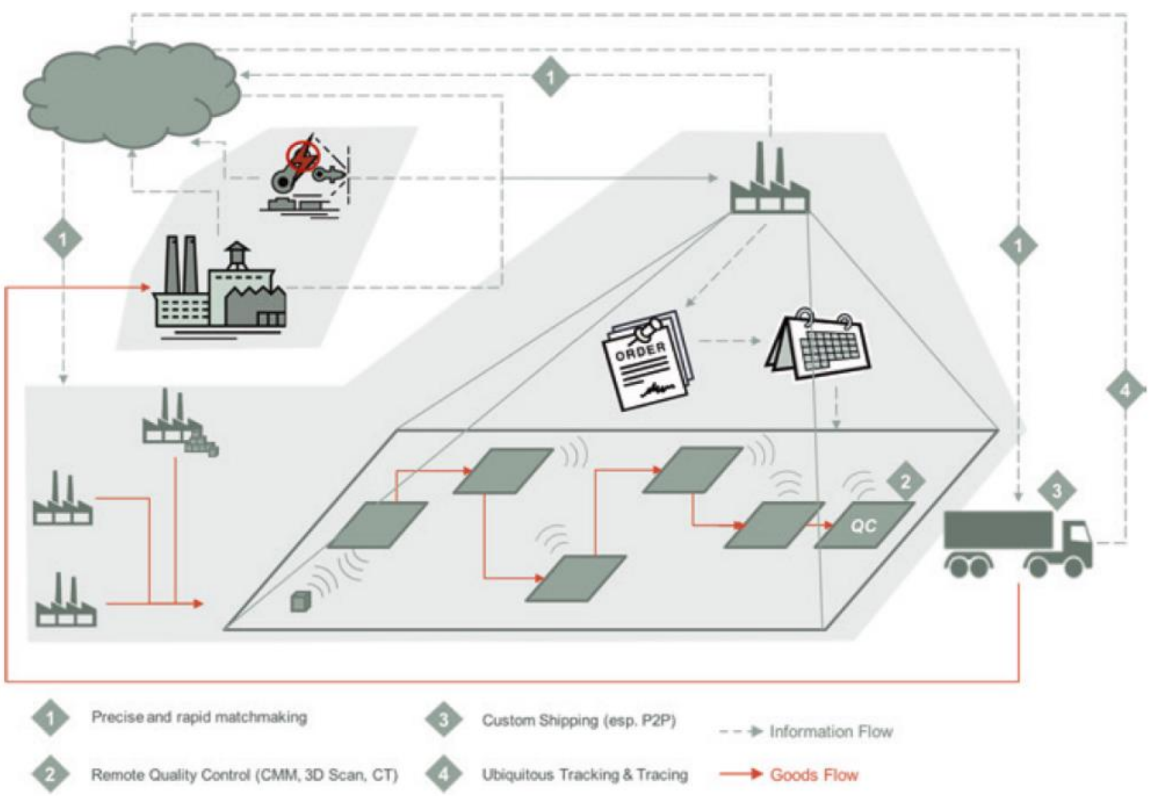


Figure 1-10 Smart Factory Concept[9]

Robot Manipulators

In the past decades, robot manipulators have been increasingly and widely applied in several production processes and industrial environments for autonomous manipulation tasks. The pick-and-place task is one of the most requested operation modes in autonomous robot manipulation. [10] In this task, the mobile manipulator must collect a set of parts to assemble a product from the storage area or pick items from a conveyor and place them in boxes located in another conveyor [11]. Pick-and-place tasks also involve the basic operation of assembly processes, such as inserting an edge connector socket into a printed circuit board and classifying workpieces of sorting systems. Pick-and-place systems for more prominent production lines usually consist of several robots or manipulators that can work collaboratively to accomplish the task.

In recent years, the customers demand of productivity and flexibility for their production lines has mostly increased. Therefore, robots and robotic pick & place cells are more and more present in some industrial fields such as the food industry. [11]

Planar robots are one configuration of autonomous manipulators that suit perfectly for roles in this type of tasks, mostly because of their velocity, high accuracy, and repetitive. The first configuration of a planar robot consists of one fixed link and the next movable links that move within the plane. All the links can be connected by either revolute or prismatic joint; this means that the general workspace of the arm is parallel to the plane of the links. There is no closed-loop kinematic chain; hence, it is a serial link mechanism.

As one kind of the planar robot, the selective compliance assembly robot arm (SCARA) robot is pliable in planar and rigid in the Z axis. The SCARA robot features:

- Simpler structure
- Lighter mass
- Faster response

- Higher accuracy compared to most of the robots.

Thus, the SCARA robot is widely applied to assembly industry. [12] In current high-performance applications, one of the world's fastest SCARA robots is Omron® AdeptOne, who is a four-axis SCARA robot. Joints 1, 2, and four are rotational, and Joint 3 is translational. See Figure 1-11 for a description of the robot joint locations. It has several times faster joints than other robots, with speeds up to 10 m/s of the end effector and achieve less than +/- 0.02 mm repeated accuracy. [13]

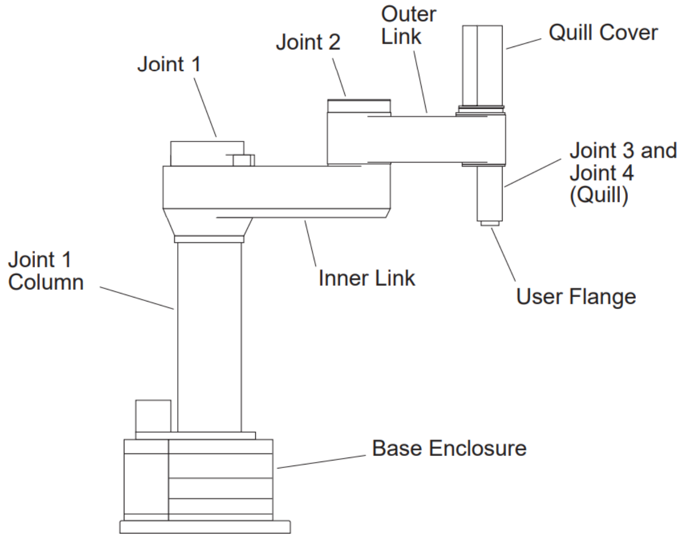


Figure 1-11 Adeptone® SCARA Joints Definitions[13]

For the matters of this project, the pick and place task of the robot arm is to build an assembly from a set of individual mechanical parts. It can be assumed that the task could be described as both a set of models of the parts, including their geometry and some physical properties and a sequence of spatial relations. The Figure 1-12 Illustrates this description with a simple construction task.

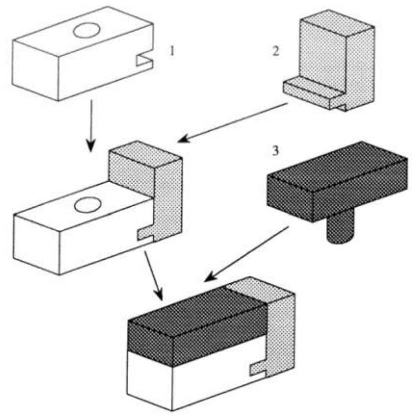


Figure 1-12 Simple Assembly Sequence[14]

The sequence determines the order in which the parts are to be assembled. Each relation would describe the position of a new part concerning the current sub-assembly. To assemble the parts, according to the input relations the robot must develop these three stages successively: grasp each part, transfer it to the current sub-assembly, and place it on the sub-assembly; for each stage are some conditions that are to be considered:

- Grasping the part: the position of the robot's gripper on the object must be selected and a path to this position to be generated. The grasp position must be accessible; it must also be stable and robust enough to resist some external forces. Sometimes, a satisfactory grasping position requires the part to be grasped, put down, and re-grasped.
- Transferring the part: the geometry of a path for the arm must be computed. Since the motion will typically be executed at high speed, the path should avoid any contact with obstacles. Due to imprecision in motion control, it may even have to guarantee some minimal clearance.
- Part mating or Placing it into the sub-assembly: the motion commands must be selected for achieving the target position with high precision in a very constrained space. In general, due to uncertainty, a single motion command is not enough; instead, a motion strategy involving several commands and using sensory inputs is necessary.

All these conditions and prior knowledge such as: knowing where the obstacles are located, of the spatial arrangement of the robot workspace, are needed for proper planning of the motion operations. [14]

Collaborative Robots

One reason because the market is changing in shorter and shorter cycles in the recent years is the exponential introduction in our daily lives of digital technologies like mobile phones, cyber-physical systems, IoT, social networks, big data, artificial intelligence, and so forth. [15] As an example of this, the Figure 1-13 shows the increase of people worldwide, from 2014 to 2021, who buys any good or service on the internet, it is expected that in 2021 over 2.14 billion people worldwide buy goods and services online. [16]

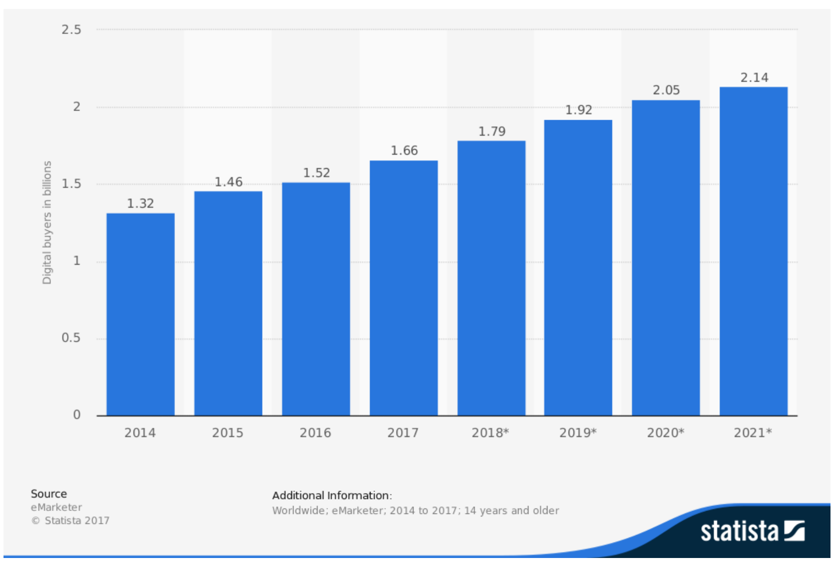


Figure 1-13 Number of Digital Buyer Worldwide from 2014 to 2021 (in billions)[16]

This growing individual demands of customers to access goods and services fuels their appetence for them in ever shorter delays to satisfy a demand for immediate gratification. [15] Increasing the variant diversity in companies and the typical made-to-order or small batch production with its client-specific adaptation of products. [17]

In response to these demands, manufacturers have been wearied of implementing robots just because of safety concerns. Industrial robots are usually preferred in large manufacturing plants for activities like an assembly line, dispensing, welding and even processing. [18] All these aspects of the evolution of the market and the industry as well have been considered an essential part of the concept of the fourth industrial revolution, Industry 4.0. [15]

Despite these efforts of automating processes made by the industry, in today's final assembly lines are still many assembly operations, such as maneuvering heavy parts, that are performed manually by humans although lit sizes are in principle high enough for automation solutions. [17] During these operations they are exposed to the risk of muscle-skeletal injuries. Robot assistant can relieve part of the human physical effort and related stress. [19]

These assistant robots are commonly known as collaborative robots or Co-bots, who can be considered as multi-robot systems working together, with or without human participation, for the same industrial task such as robotic assembling. [20]
For each kind of collaboration with robots exist already examples of commercial robots or research projects that can be found around the world as Robotic soccer teams compete against each other in RoboCup competitions every year, Figure 1-14, as an example of robots collaborating with other robots.

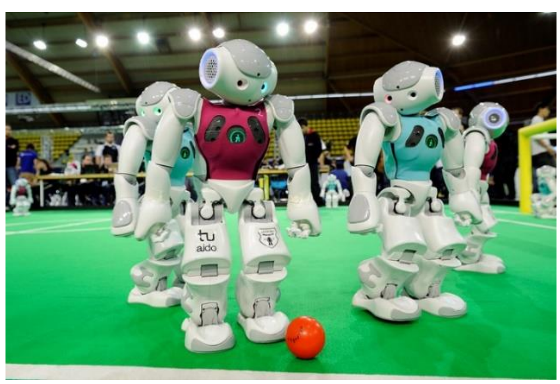


Figure 1-14 Nao Robots in RoboCup Competition[21]

On the other hand, in the industry can be found smart robots Baxter (Figure 1-15) and Sawyer are newly developed by Rethink Robotics® to perform tasks like humans do [20], who are helping workers to do more and more tasks in their workspace.

Figure 1-15 Baxter Robot in Factory[22]

One of the main challenges of this technological achievement is to ensure safety to the personnel which will be working around these robots because most of the time they are used in unstructured and dynamic environments and must manage many real-time constraints such as co-location with humans. Considering that most commercial collaborative robots that are now in operation already include some safety actions to be executed, like stopping or even triggering an emergency fault when such a limitation is reached. [23]

Co-bots usually adopt high-power motors and speed reducers (harmonic drives) to improve the weight-to-payload ratio, increasing the manufacturing cost of a robot and its energy consumption to perform tasks [24]. Plus the fact that they require a safety cage or enclosure to avoid any contact, especially while co-working with humans. [18]

Is well known that manufacturing systems are managed by different kinds of controllers, each one of them devoted to a specific task or resource and even divided according to different temporal horizons. [25]

Collaborative robotics represents multiple robotic nodes assisting each other to perform a task. Thus, multi-agent systems not only speed up complex tasks but also increase robustness. [26] Roughly, the robustness of a production control system can be defined as its ability to "absorb" perturbation, and perturbations can be defined as "unpredictable events." Considering two kinds of perturbations:

- The ones that can be identified but never know when they can happen, like tool wearing due to process activities.
- The ones that are not known a priori, like changes in the production context.

To define the collaborative control of a production system is needed first to specify the global decision system and the way it is going to be distributed among all controllers, in Figure 1-16 are shown two types of architectures that can be used:

- **Task-driven architecture**: Defining a controller for each task also capable of controlling different resources.
- **Resource-driven architecture**: Defining a controller for each resource also capable of developing different tasks.

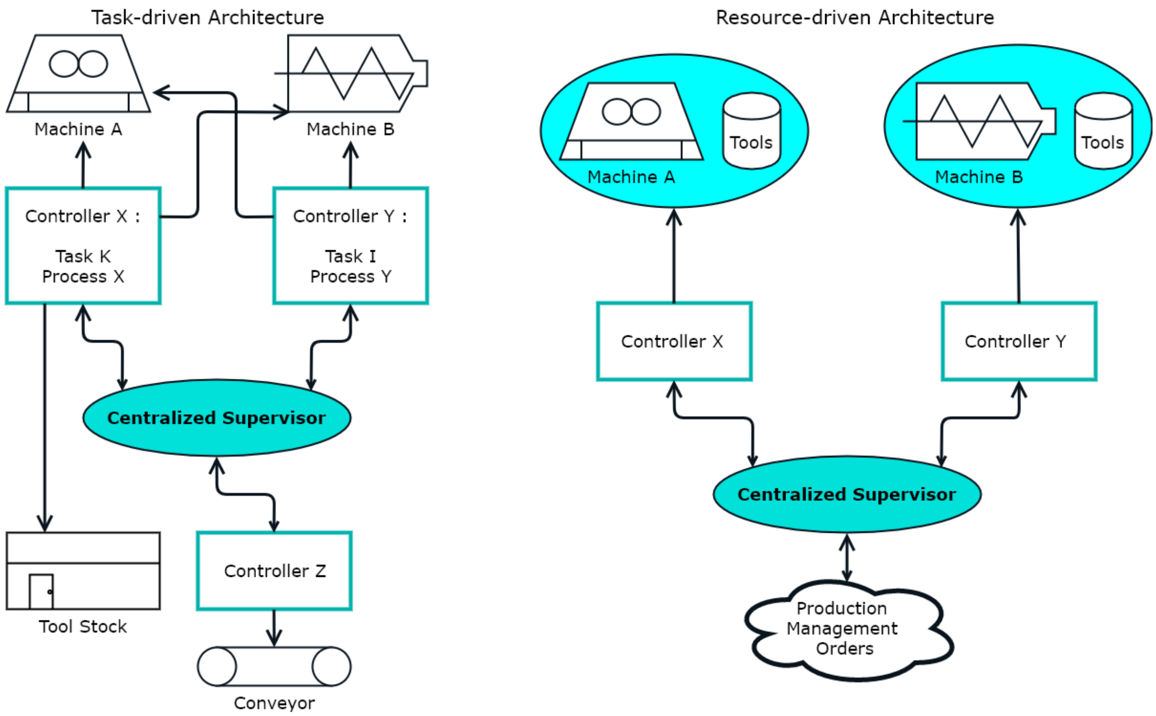


Figure 1-16 Decision Architectures for Collaborative Processes

Communication and collaboration play an essential role in robot systems, especially in multi-robot systems, enhanced by the human to complement individual skills. [26] This collaboration between in a collaborative environment can be achieved in 3 different ways:

- **Directly**: Using the output of a controller as an input event for the others, making the reaction time-dependent to the number of active controllers connected.
- **Indirectly**: Sharing information via the environment to whom the controllers are connected, depending on the inertia of the entire environment and reacting over a significant event.
- **At Knowledge level**: Sharing information and knowledge about the system, thereby each controller can anticipate changes in the shared environment.

Human-co-robot interaction also brings emphasis to the social challenges of robot design and their consequences for both next task performance and the long-term adoption and acceptance of co-robotic technologies. [27] Therefore more and more investigators are dedicating effort in the social part of it.

IEC 61131-3 Standard

To implement all the technologies already mentioned is necessary to implement a standard frame that would make the integration of all the parts to work smoothly together; in this case what suits better for this is the standard IEC 61133-3 and PLCOpen.

The PLCOpen activities are based upon the IEC 61131-3 standard which is by the moment the single global standard for industrial control programming. It harmonizes the way people design and operate industrial controls by standardizing the programming interface allowing people with different backgrounds and skills to create different elements of a program during different stages of the software lifecycle:[28]

- specification
- design
- implementation
- testing
- installation
- maintenance

The standard includes the definition of the Sequential Function Chart (SFC) graphical language, that looks like a flowchart that organizes your system progresses through defined steps or states. The standard also includes four inter-operable programming languages: Instruction List (IL), Ladder Diagram (LD), Function Block Diagram (FBD) and Structured Text (ST). The idea of the implementation of these languages and techniques is to increase user efficiency and re-usability and modularization of the programs

Within the sections of the standard, several aspects of machine design and development are taken into count, from programming languages as already said to safety conditions (SafetyOpen) or motion control (MCPLCOpen)

LEGO® Bricks

LEGO Group is one of the most well-known, world's leading manufacturers of play materials, based on their iconic LEGO® Bricks; they motivate children to reason systematically and think creatively allowing them to build anything they can imagine.[29]

LEGO® bricks are hollow blocks adapted to be connected using extruded pins from the faces of the elements and arranged so to engage protruding portions of an adjacent element when two such elements are assembled. The coupling method is made by clamping such building bricks together in any desired relative position thus providing for a vast variety of combinations of the bricks for making toy structures of many different kinds and shapes. [30]

A single building brick has two different sets of extruded pins or projections. The primary one that is extended from the top face and the second is extended from the cavity of the hollow block; both the primary and the secondary projections have a cylindrical shape, and the circular cross-section of the secondary projections will this touch the four-circular cross of the primary projections.

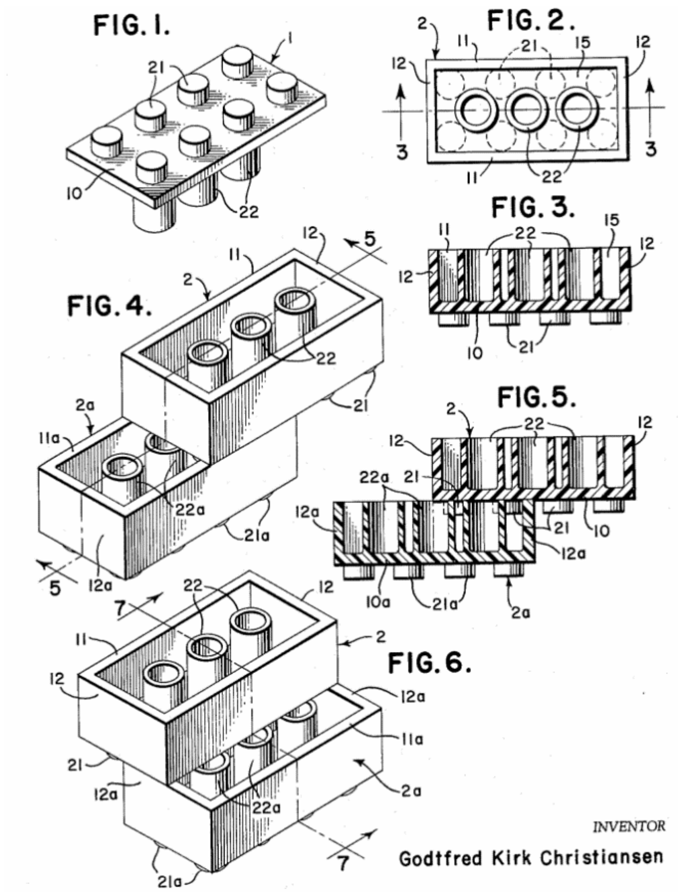


Figure 1-17 Views of the Assembly of LEGO® Bricks[30]

Thus, any of the elements illustrated in Figure 1-17 may be combined with any other element in any desired relative position. Either by clamping one or more secondary projections of one element or by clamping a pair of primary projections of one element between the one-second projection and their inner face of an end or side wall of the other element. [30]

LEGO® DUPLO® is a Branch of the LEGO® products, which aims to children from 2-5 years old. Therefore the bricks should be of a size bigger enough, compared to standard bricks as shown in Figure 1-18 so that they can be manipulated easily and hard to get stocked in their mouths.

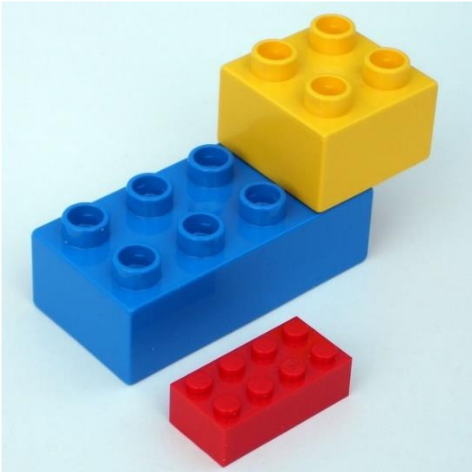


Figure 1-18 Size comparison between LEGO® standard bricks and Duplo® [31]

From the engineering point of view, this implies that their mechanical properties change such as tolerance for the assembly, resistance to impact, maximum compression load, resistance to traction, compression load for the assembly; all these mechanical properties can be seen in Table 1-1.

Table 1-1 Mechanical Properties of LEGO® DUPLO® Bricks

Property	**Value**
Tolerance	0.004mm
Resistance to impact	1 Kg @ 120 mm
Maximum compression load	50 Kg
Resistance to traction	11.8 Kg
Assembling Compression Load	15 Kg

*Data extracted from [32]

2. Project Methodology

Methodology

The design of mechatronic systems is entirely different from the design of products which are mainly coined by one engineering domain. Mechatronic systems are extraordinarily complicated due to the high number of connected elements that are realized furthermore in different engineering domains. [33]

Due to the complexity of these projects, it would be appropriate to use the development methodology based on the VDI 2206 guidelines. The design methodology proposed by the guideline defines two levels of the process. The first one is called micro-level supports the designer with tools like structuring sub-tasks and procedures to react to unforeseen situations when problem-solving is required; the other, called macro-level, helps to supervise the whole process setting milestones, planning, and controlling every phase of the product.

The V-model, whose name comes from validation and verification model, is the macro-level of methodology, it consists of a sequential path of execution of processes, and each phase must be completed before the next phase begins. Initially, this model was intended for software development, area in which has been proved for a long time.

Nevertheless, this model can also be applied to mechatronic systems development because of 3 reasons for the model:

- It combines in an obvious way two system approaches top-to-down (system design: dividing into sub-functions) and down-to-top (system integration: integrating the results into the overall system).

- Allows a regular verification/validation routine between the requirements (left side) and the actual system (right side).
- Because of the increased acceptance of the guideline, this model is already starting to be implemented in the context of mechatronic systems development.

In other words, describes the general procedure for designing systems which must be specified in individual design tasks (Figure 2-1).

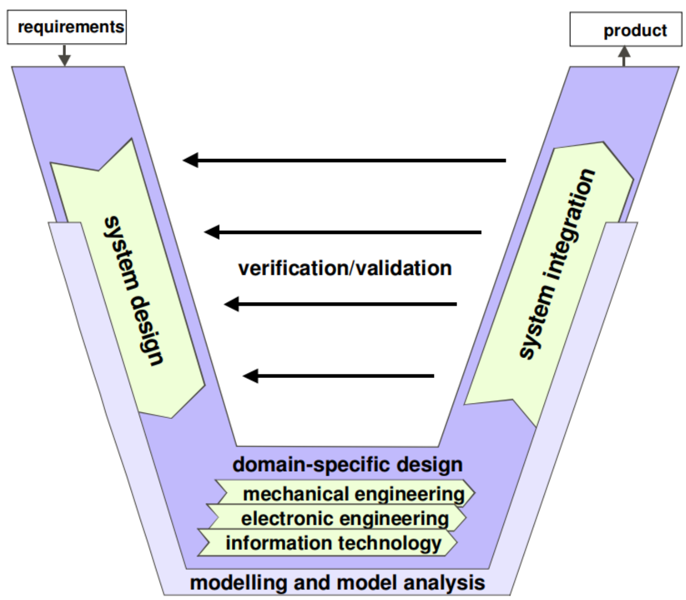


Figure 2-1 V-Shape Model on Macro Level [33]

Some advantages of the implementation of this model are:

- It is simple and easy to use.
- Saves plenty of time because testing activities such as planning, or test design occur before coding.
- Defects are found at early stages of development.
- It avoids the downwards of the defects.
- Works well for projects where requirements are easily understood.

On the other hand, the disadvantages of this model are:

- It is very rigid and least flexible.
- The system is developed during the implementation phase. Therefore, there is no early prototype is produced.
- If any changes happen in midway, then the test documents along with required documents must be updated.

Product Definition

Before being able to define the requirements of the machine that will be designed, it is necessary to describe the primary task that this machine must perform in this case is the assembly of the previously mentioned vehicles. The following describes how the configurations made by customers about their products are interspersed.

To homogenize the way in which all the stations receive information about the configuration of bricks that they must use to complete a product, a system of three-dimensional coordinates was defined to indicate the position of each brick in the assembly. This coordinate system was defined by the colleagues who are involved in the development of the resource planning system, as well as those who work with the web application.

It is essential to know how this coordinate system that uses the ERP system behaves since the information that the assembly station will receive contains data concerning this system. This prevents the system from having to define them based on the configuration of each of the stations, and this would result in something quite complicated.

In such a way that this coordinate system has a reference origin and a part origin. Because all possible configurations of products to be manufactured are vehicles, they all require a chassis piece on which all other bricks are assembled vertically. That is why the coordinate system of reference is linked to this piece. As shown in Figure 2-2, the origin of this coordinate system is in the center of the piece defining the X-Y plane. The X-axis is aligned in parallel with the most significant edge of the chassis, and the Z-axis is defined as the vertical axis concerning which the blocks will be stacked.

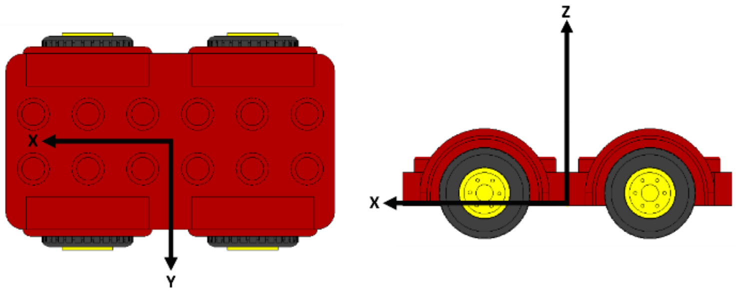


Figure 2-2 Origin Definition of Base Element

The second coordinated system is attributed to each of the pieces, which, as shown in Figure 2-3, locates the center of the piece on the underside of the brick and in the center of the cross-section of the same.

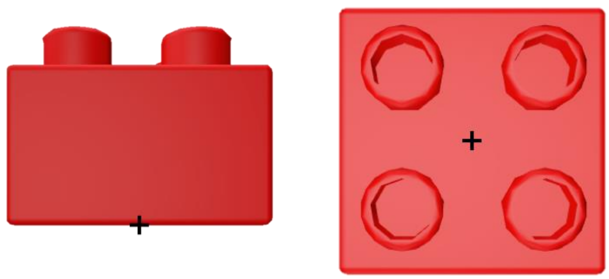


Figure 2-3 Origin Definition for Bricks

Although the regular use of the LEGO® bricks allows defining a coordinate system taking as reference the extruded pins of each brick, this system does not satisfy all the possible configurations to place all the pieces that define the shape of a vehicle. However, this attribute is used as a reference to identify the different brick families, for example, 2x2, 3x2, 4x2, and so forth.

Because this system does not include necessary positions, for the XY plane, a scale with detailed graduation was defined whose unit is equivalent to half the distance between the centers of 2 extruded pins as shown in Figure 2-4. This way is possible to get a grid of 12 units per side leaving the origin in the center. In part b of this figure can also be seen when a brick is positioned in the coordinate (0,0) in the X-Y plane; in section c the same brick is observed, displaced one unit on the X-axis and in part d the piece has been traversed two units concerning the X-axis. Although some pieces do not seem to fit naturally in all the coordinates as would be the case of part c, these coordinates are used for larger pieces of the same origin.

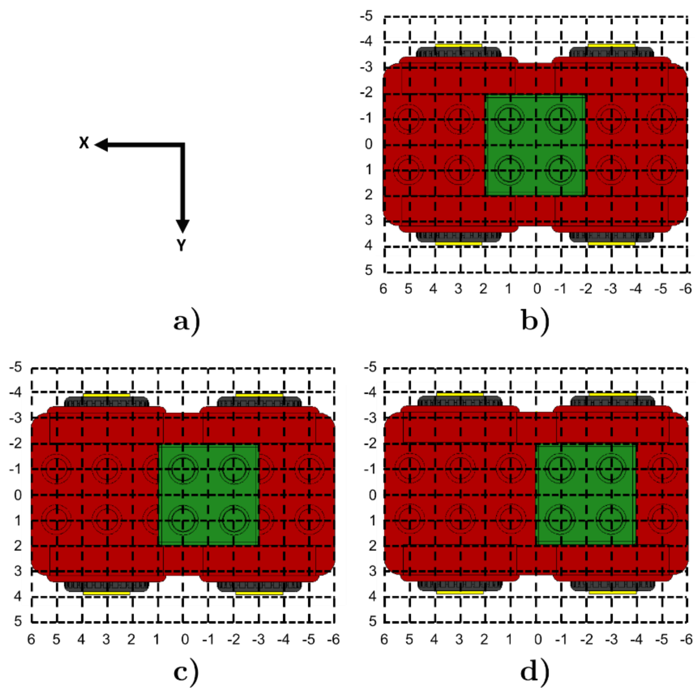


Figure 2-4 X-Y Plane's Grid Definition

Concerning the Z-axis, the scale is based on the height of the smallest brick that can be used in the LEGO® DUPLO® family of bricks, leaving six pieces of this height as the upper limit. In the Figure 2-5, can be seen this more detailed vertical distribution of parts.

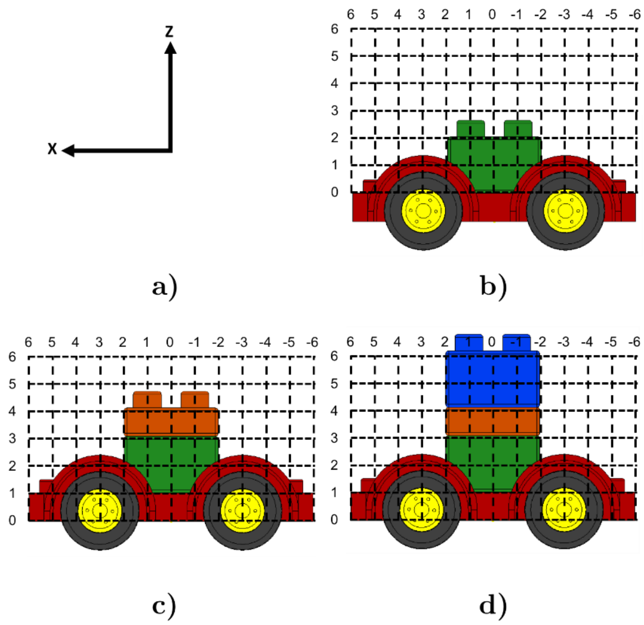


Figure 2-5 Z-Axis Positioning for Bricks in Assembly

In addition to the different configurations of the cross-section for the parts that are used in the assembly of the vehicles. There are pieces whose faces have different finishes; this allows the finish of the product to be more stylized. The customer can select the orientation of these pieces, for this, a rotation parameter is added concerning the Z-axis. These rotations are available every 90 degrees as can be seen in Figure 2-6, to allow the piece can be assembled with the previous pieces.

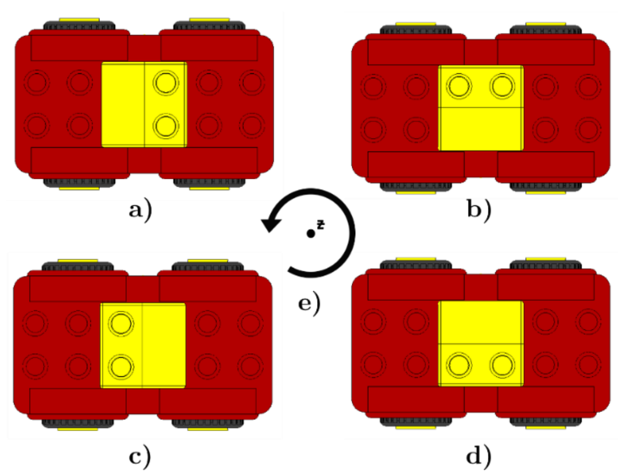


Figure 2-6 Available Rotations for Bricks in Assembly

All these data about the position, orientation, color and quantity of pieces are the parameters that the user can configure and modify in the web application. An example of a custom configuration can be seen in Figure 2-7.

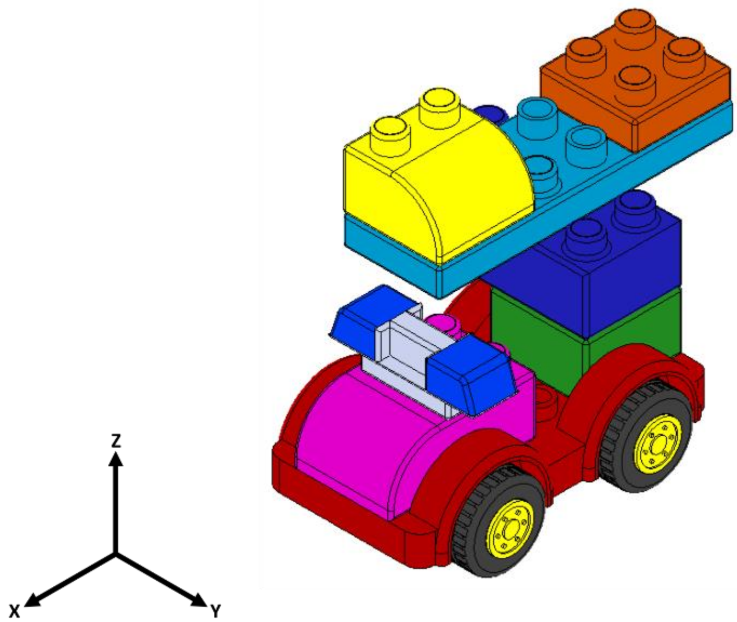


Figure 2-7 Example of a Customized Car to be Assembled

As mentioned before, all these custom product configurations are made through a web application called Opencart; in which, when registering the user can make the configuration of their products through a 3D viewer that is also being developed by students involved in the project. Below is a screenshot of how that plugin is seen mounted in the Opencart application in the Figure 2-8.

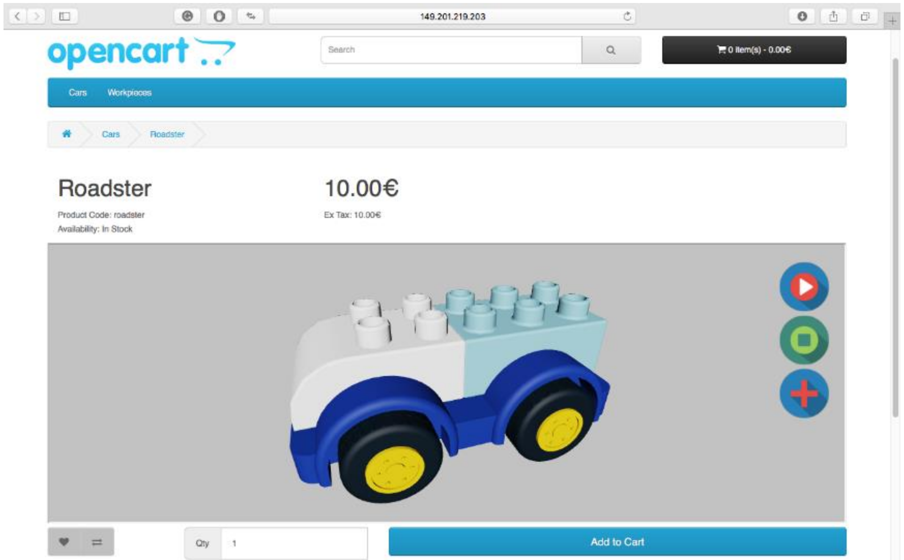


Figure 2-8 Opencart Web Application for Configuration of the Product

Up to this moment, the web application continues to be developed, for this reason, the application is enabled only to configure some predefined "recipes" of vehicle models: a roadster, a sedan, a station wagon, and a truck. Of these recipes, the proposed assembly station will focus on building recipe orders for roadster, sedan and station wagon. In Figure 2-9 these predefined models can be observed.

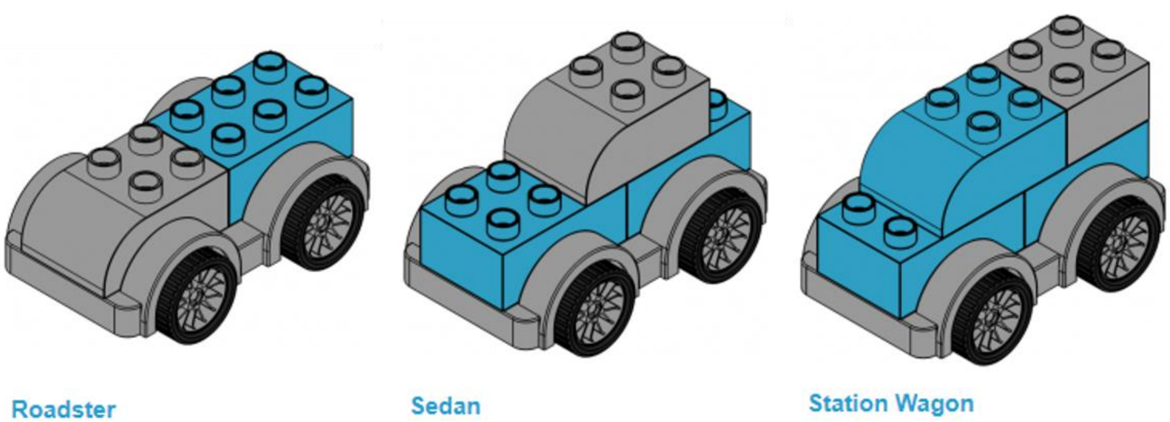


Figure 2-9 Car Models Available for Assembly

Requirements Specification

The primary function of the system is to use LEGO® bricks to assemble customized vehicles, based on the specifications of each customer, following the philosophy of the industry 4.0. This entire process has to be fully automated from the supply of parts to the storage area, up to the delivery of the finished product; and the customer would be able to configure this product, the LEGO® car, through a web application remotely.

The customer will be able to make orders through a web application that will show a 3D preview of the car. This application will be developed and adapted such that the orders made in it are compatible with the ERP system that is already implemented by the Laboratory.

The diagram in Figure 2-10 shows the structure of requirements that were specified and must be fulfilled by the system to perform all its functions. As indicated by the SysML language, the structure of the diagram is hierarchical, from father to son, taking the requirement of the higher level as the main functionality of the system. Likewise, each of the requirements of lower levels can have several requirements as children. [34]

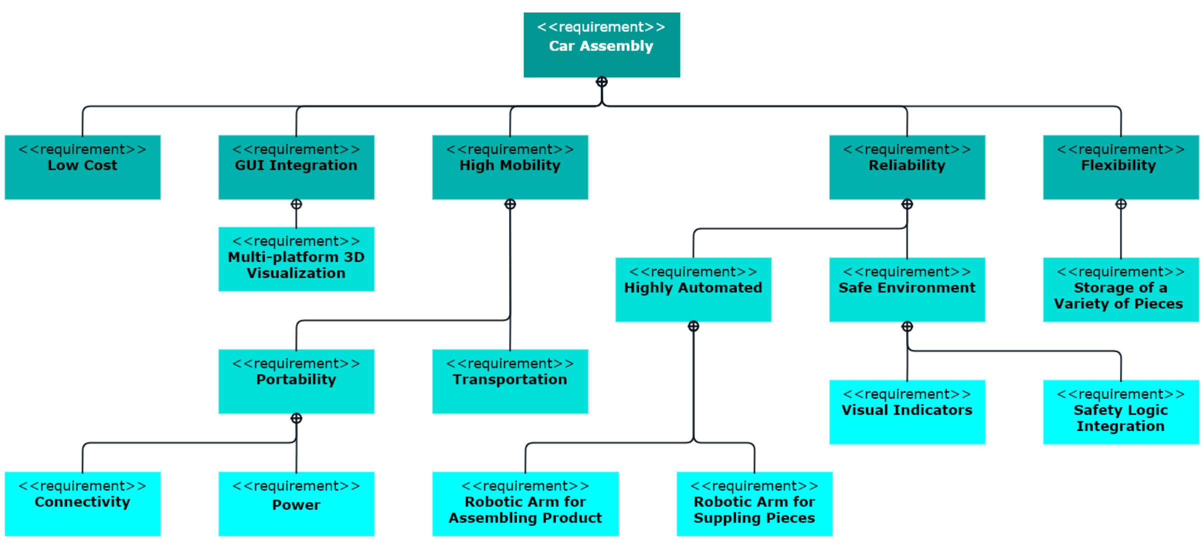


Figure 2-10 Requirements Diagram

In the case of this system, the primary functionality is to be able to assemble cars made from LEGO bricks. To meet this requirement, the following requirements must be met:

- **Low Cost**: The construction of the entire machine must be within the established budget, for this, the electromechanical components to be used must be of the IGUS® brand.
- **Integration of a GUI**: Because the machine is connected to an ERP system, an interface must be integrated with which the customer can configure and request a product.
- **High Mobility**: The machine must have the possibility of moving to different places since one of the objectives of the development of this system is the presentation of the same in different conventions and demonstrations. To do this two requirements must be fulfilled:
  - **Portability**: Anywhere the machine is transported, the optimal conditions of connectivity and power supply must be guaranteed to demonstrate the operation of the machine.
  - **Transportation**: The machine must be light enough so that it can be moved, it must also be able to pass through spaces of 800 mm wide, the equivalent to the width of a door under the German standard.
- **Reliability**: During the operation of the machine, it must be guaranteed that all steps of the assembly process can be replicated at any time, so it is convenient that the machine includes:
  - **High level of automation**; for which the movements of assembly are required, as well as those of supply of pieces either by robotic arms.
  - **Safe Environment**: Implement safety conditions for the operator and the people who are present during the operation of the machine, using visual indicators and implementing a safety logic.
- **Flexibility**: Satisfying some aspects that characterize the industry 4.0, the final product must be highly customizable by the client, for it the machine must store as many brick types as possible to offer a flexible product.

General Concept

Once the requirements that the system must satisfy have been defined it is possible to start talking about a concept for the machine, in Figure 2-11, a package diagram that decomposes, at its highest level, the concept of the machine can be seen. This first division can be done in central tree packages: mechanical components, electrical components, and the logical components, these, in turn, can also be broken down into other subsets a little more detailed that define the elements that will constitute the system. Subsequently, each of the three main packages mentioned above will be described.

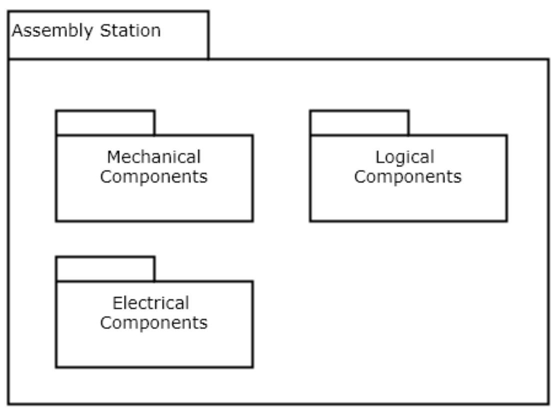


Figure 2-11 Package Diagram of the Top-Level Definition of the System

Starting with the package of mechanical components covers all physical components that involve direct manipulation of the materials used to assemble the final product. Figure 2-12 separates these components into three other categories:

- **Structure:** Refers to the elements that make up the structure that will contain the rest of the components of the machine.
- **Assembly mechanism:** The second refers to the elements whose primary task will be the assembly of the final product using the necessary materials.
- **Supply mechanism:** Refers to all the elements focused on supplying materials, bricks, to the assembly mechanism.

Each of these last two categories comprises two main components; the first is the final effector that will have direct interaction with the bricks before assembly, as well as with the finished product. On the other hand, the actuators are essential to facilitate the final effector's performance of his task.

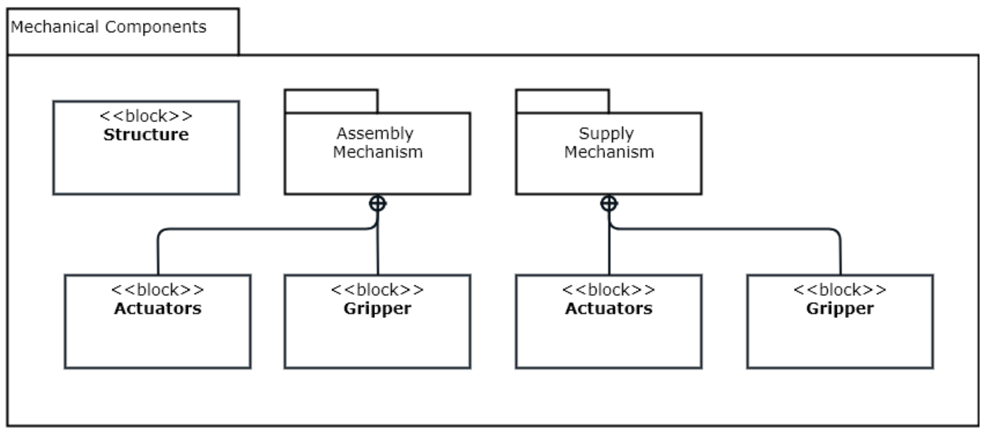


Figure 2-12 Package Diagram of the Mechanical Components

In the part of the electrical components the physical elements of the machine controller are contemplated, these, in turn, can be divided into four packages, which are described in the package diagram of the Figure 2-13.

- **General Components** are all the essential elements needed for any machine to work from the electrical point of view.
- **Controller for the Assembly** are the electrical components that will be focused on executing the control logic for the said task of the final product.
- **Driver for the supply** of bricks are the electrical components that will be used to execute the control logic for the brick supply tasks.
- **Safety components** those electrical components based on specific rules of this topic for the implementation of a subsystem that monitors the safety conditions in the machine during its operation.

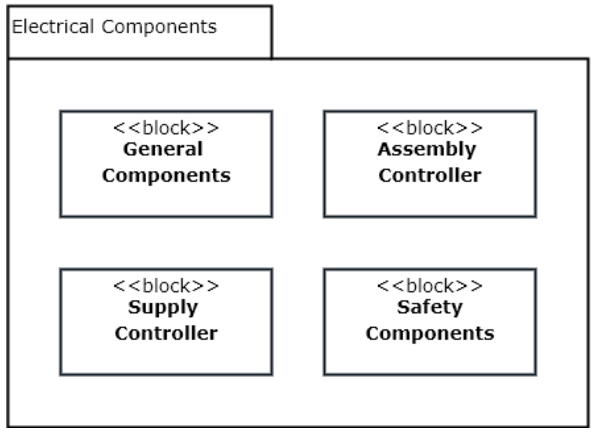


Figure 2-13 Package Diagram of the Electrical Components

Finally, the software category comprises the intangible elements the routines, which form the logic of complete system control and can be separated into three packages as shown in Figure 2-14. These three packages are:

- **HMI**: These are the software elements that are focused on communicating the status of the machine directly to the operator.
- **Security logic**: the routine that will be executed by the security elements parallel to the central logic.
- **Main logic**: constituted by the different subroutines necessary to execute by the machine to carry out its principal tasks.

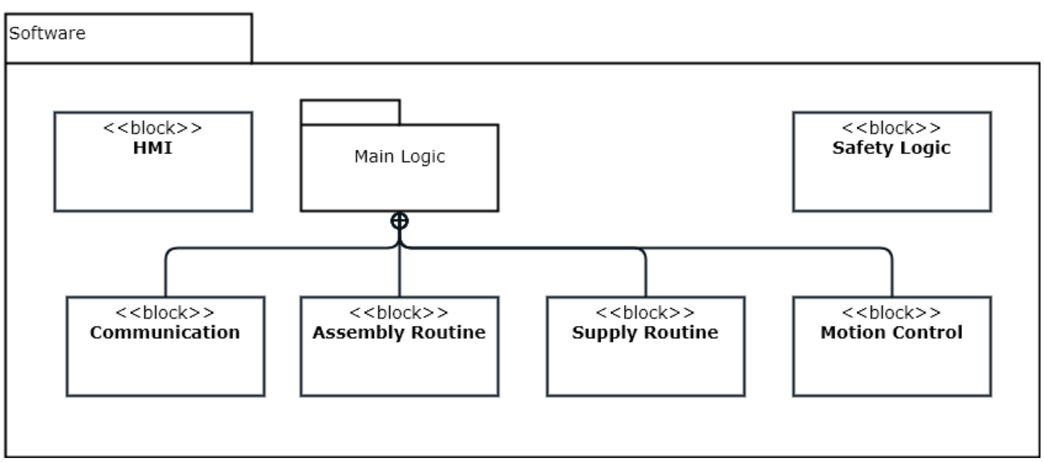


Figure 2-14 Package Diagram for the Software

In turn, the central logic can also be considered a package of elements that can be separated into sub-elements of it as follow:

- Communication Routine is exclusively responsible for establishing a connection with the server, receiving information about the requests for products to be made and returning information about the current state of the machine, in such a way that the server has feedback and executes its selection and distribution actions of work.
- Assembly Routine will be responsible for processing the information received from the server and establish the trajectories that the assembly robot must execute to complete the requested product.
- Supply Routine is the one that monitors the number of existing bricks in the inventory of the machine, and in case some piece begins to be scarce, it will be the one that indicates to the supply robot the trajectories that it must follow to comply with its function.
- Motion Control Routine dedicated to converting the trajectories, assigned to each robot, into functions and instructions in signals and commands for the controllers of each of the motors of each robot.

All the elements that make up each of the packages will not work independently, or in isolation, on the contrary, they will have to interact with elements of the other categories to guarantee the correct functioning of the whole machine.

Once the elements that make up the concept of the machine have been defined, one can begin to define actors that will interact with the station, as well as the cases in which these will be related to the general system. All these aspects can be observed more clearly in the case-use diagram of Figure 2-15.

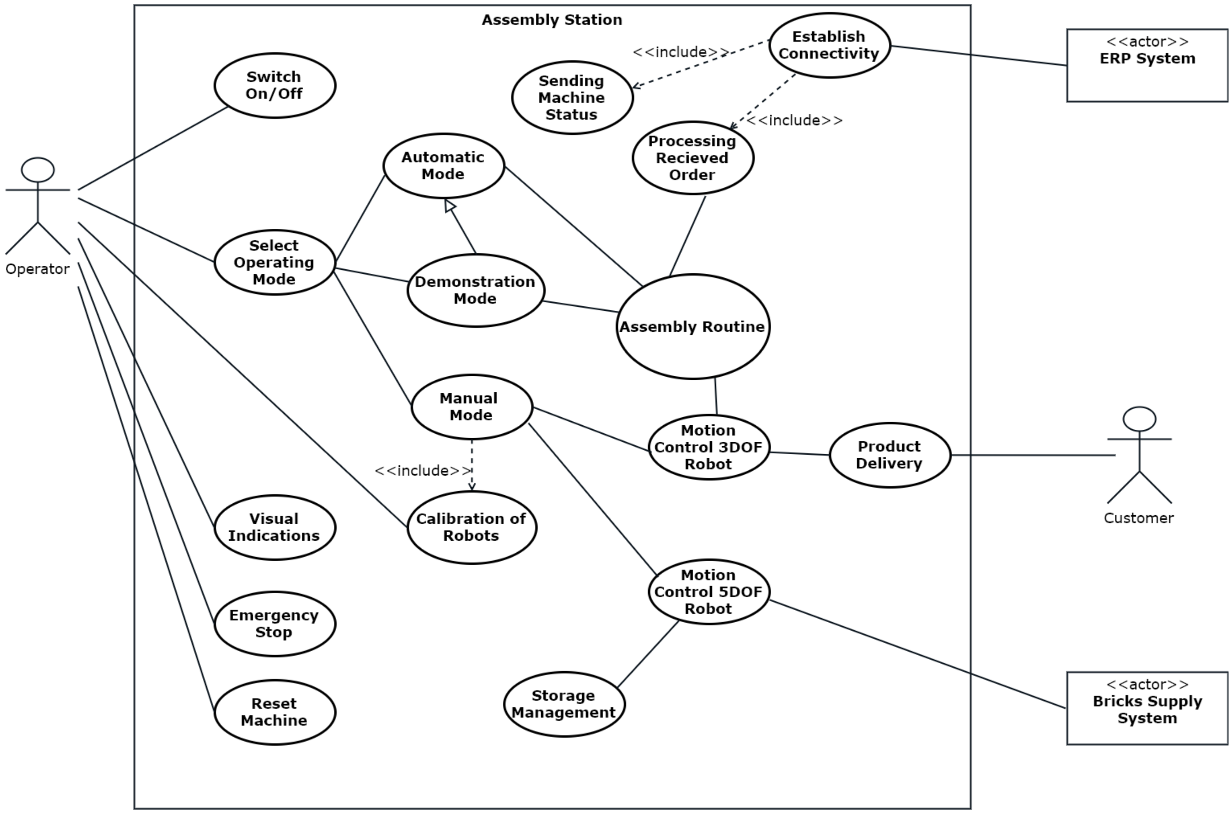


Figure 2-15 Use Case Diagram of the Assembly Station

Within the actors that can found:

- In the case of the operator, he will be mainly involved in configuring and establishing the initial conditions of the machine so that it can be operated. The operator will turn on or off the machine; select among the different modes of operation and in case If necessary, perform calibrations and adjustments to the actuators inside the machine.
- The assembly station will communicate with the ERP system to receive data about the order to be processed and respond with information about its status.
- The interaction that the machine will have with the client is only for the delivery/reception of the product once the assembly process has concluded.
- The last actor with which the assembly station will have to live will be a system of feeding materials for all stations, once it is implemented.

Mechanical Description

Structure

The structure that must be used must comply with the transportability, weight and moon requirements of the most critical dimensions, as it is specified in the requirements, it must be able to pass through a standard door with a clear of 800 mm. This is a critical requirement since it defines the geometric configuration of the entire system since the actuators to the control system must be contained or at least close to the structure.

The other important aspect for transportability is the weight since the laboratory where this machine will be sheltered most of the time is on a second floor and without an elevator available. Then it must be considered that several times this machine will have to be loaded by laboratory personnel.

For the assembly of a structure, different configurations and materials can be used, but a material that is frequently used for the construction of structures that must be mobile and light is the profile of extruded aluminum. This material among other features how much with:

- Design time reduction
- Adaptability of components
- Easy handling
- Robustness and durability thanks to its cross section
- Simple finishes of good quality

For these reasons it is convenient to use this type of material for the design of the structure, it is also easy to adapt casters to improve portability.

The structure must be able to store a certain amount of bricks to assemble several vehicles during the demonstrations. From this, it is useful to design mechanisms that store and classify the different variants of parts that can be used for the assembly of a vehicle. One of the techniques that are most used in the other assembly stations is by active or passive conveyor belts. These transport the materials to the nearest point so that the assembly mechanism - usually a robot - grabs them and places them in the assembly area. Therefore, it is proposed to use this type of mechanism for the storage of the bricks that the machine will use for the assembly of the final product. The proposed slide concept is shown in Figure 2-16.

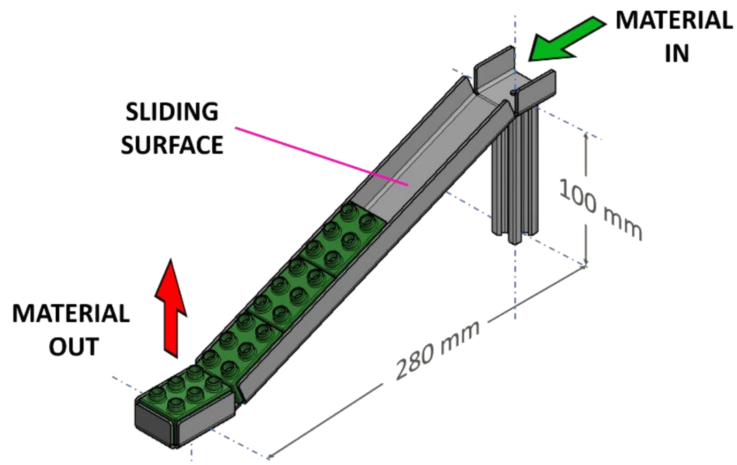


Figure 2-16 Slide Design Concept

Now, it is necessary to determine the quantity and types of pieces that will be stored in the assembly station. The pieces that are used for all seasons can be divided into four categories according to the dimensions of their base and the number of extruded pins they have:

- 2x2 base bricks
- 2x3 base bricks
- 2x4 base bricks
- Special bricks

This last category includes bricks with irregular shapes that refer to specific details of a vehicle such as the case of an engine or the chassis. Other unique pieces will not be considered for the design of this machine. Some examples of unique pieces can be seen in Figure 2-17.

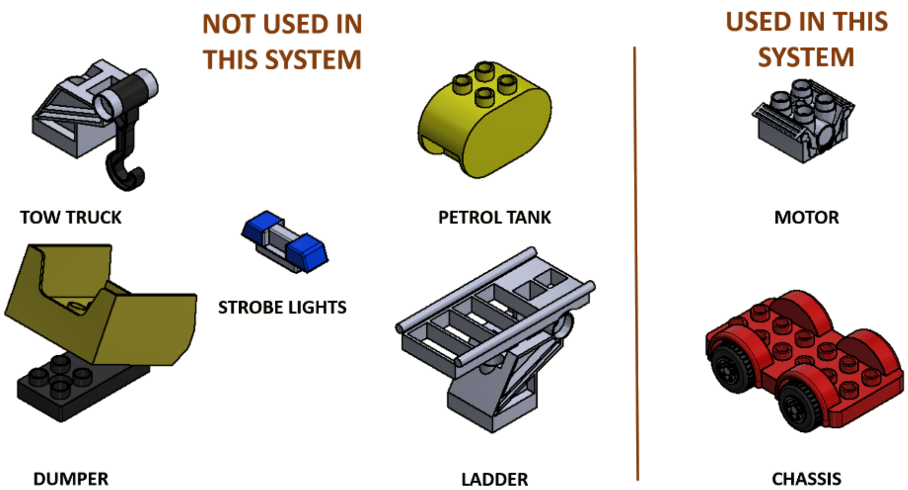


Figure 2-17 Special Shape Bricks

As mentioned above, most of the bricks used in the project have a side face that has the same length for all of them. From this, the categories start with "2x" as shown in Figure 2-18. This general characteristic will allow the design of the slides to be a little more generic by varying the measurements of the inlet and outlet trays of each type of slide.

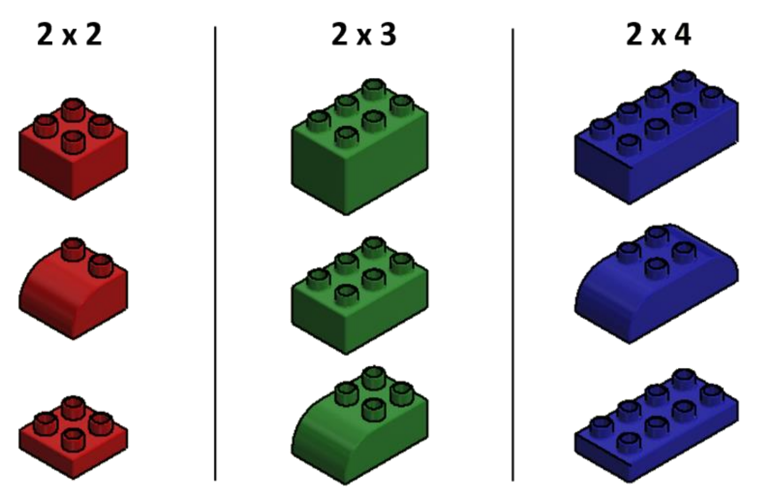


Figure 2-18 Examples of Bricks Available for the System

Considering that the assembly mechanism will be a cylindrical robot, it turns out that the distribution with which the most significant number of slides can be incorporated is circular or half-moon. Also, to increase the amount or variety of bricks available, it is possible to add a second row of slides. In Figure 2-19, the concept of the distribution of the slides in the machine can be observed.

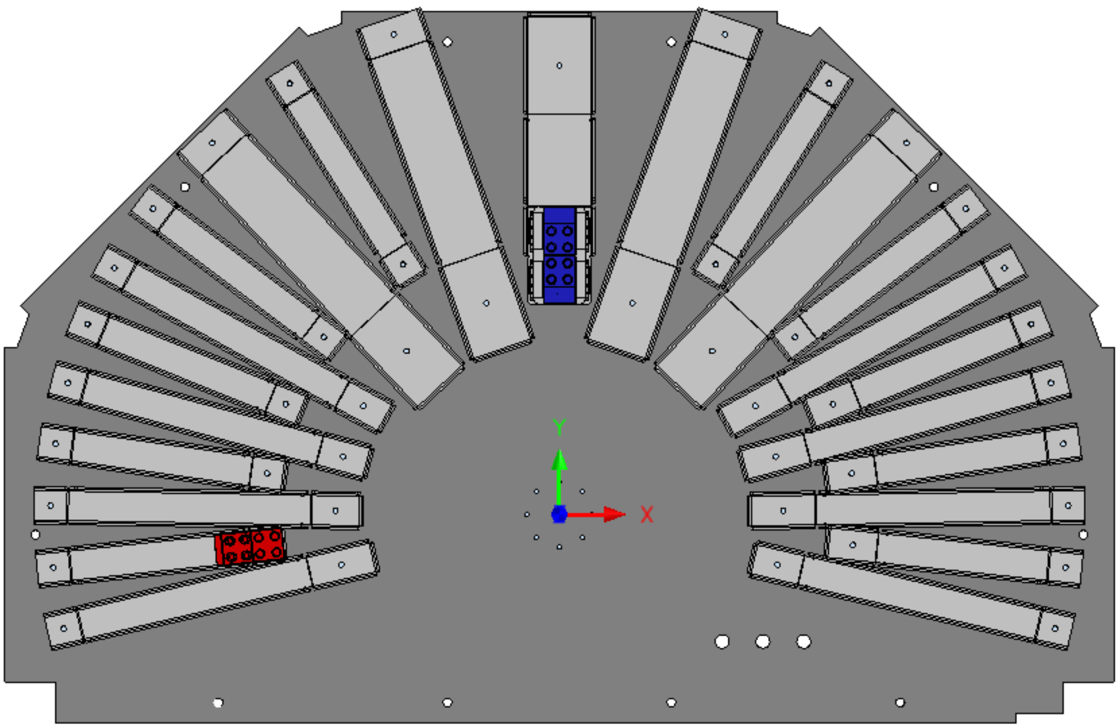


Figure 2-19 Layout of Slides inside the Assembly Station

Table 2-1 lists all the brick models that will be integrated into the system, as well as their distribution in the slides. It is important to mention that the identification of the slides is made counterclockwise starting with the bottom row. Also, in the center of the half moon is the platform den assembly and delivery of finished product.

Table 2-1 Slides with Type of Bricks

Slide Number	Slide Type	Color	Serial Number	Quantity
1	Low Feeder 3x2	Dark Green	4559584	6
2	Low Feeder 4x2	Light Green	4166924	5
3		Red	0301121	5
4		Dark Yellow	4290060	5
5	Chassis Feeder	Dark Blue	6048908	3
6		Dark Green	6048909	3
7		Red	6138944	3
8		Dark Yellow	6138858	3
9	Low Feeder 3x2	Yellow	4558884	6
10		Light Blue	4613700	6
11	Low Feeder 3x2 One side rounded rotated 180°	Light Blue	4541728	6
12	Low Feeder 4x2 Both sides rounded	Blue	6138928	5
13	Up Feeder 3x2 One side rounded	Orange	6055103	6
14		White	6172232	6
15		Light Blue	4541728	6
16	Up Feeder 2x2	Dark Blue	4221630	7
17		Grey	4210953	7
18	Up Feeder 2x2 Motor	Grey	4546220	7
19	Up Feeder 2x2	Red	0343721	7
20		Dark Green	0343728	7
21		Dark Yellow	0343724	7
22		Light Yellow	4648231	7
23	Exit Tray	-----	------	-----

Now, because the work area is in the form of a half moon, the base where they must be mounted can be of a non-square shape. Otherwise, will be wasted much space in the machine. From this, it follows that the shape of the structure must have a polygonal shape in such a way that the material to be used is used and therefore the structure must have the same shape. Figure 2-20 shows the front view of the concept of the complete structure.

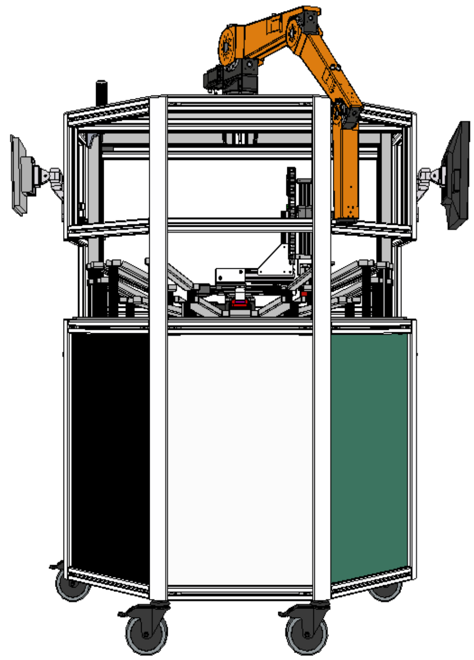


Figure 2-20 Front-View of the Assembly Station

Assembly Mechanism

To satisfy the requirement of a mechanism that performs the assembly, several configurations of robotic solutions -Cartesian Robot, Spherical Robot, and so forth.- are feasible for this application. However, one of the main limitations is the reduced space due to the overall dimensions of the entire structure. That is why the most viable configuration for this application is the implementation of a cylindrical robot.

The assembly robot is proposed with this topology, because its workspace includes, using the maximum extension of its joints, the shape of a cylinder as shown in Figure 2-21. This workspace is convenient for the distribution of the slides that will store the bricks. The robot has three articulations of which the first one is of rotational movement or revolute type, rotating around the z-axis on the base of it. The second joint is of linear movement or prismatic type and moves along a coplanar axis to the XY plane, and the third joint is also of primitive type and moves along the z-axis, concerning the coordinate system of space.

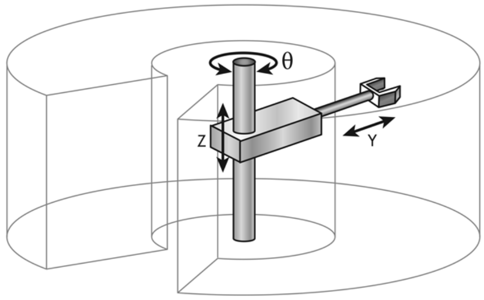


Figure 2-21 Generic Workspace for a Cylindrical Robot[35]

Currently, in the market, there are several brands that offer ready-made solutions to be integrated into the work area, and although an industrial SCARA type robot can be implemented; One of the constrains of the project is the budget allocated. That is why for this project it was proposed to use products of the IGUS® brand since within their products they offer robotic solutions.

Within the options available for the application, the configuration that best meets the requirements of the system is the design of the robot using joints and actuators and integrate them in a way that suits the proposed configuration, in Figure 2-22 is shown the topology of the DOF that the assembly robot has.

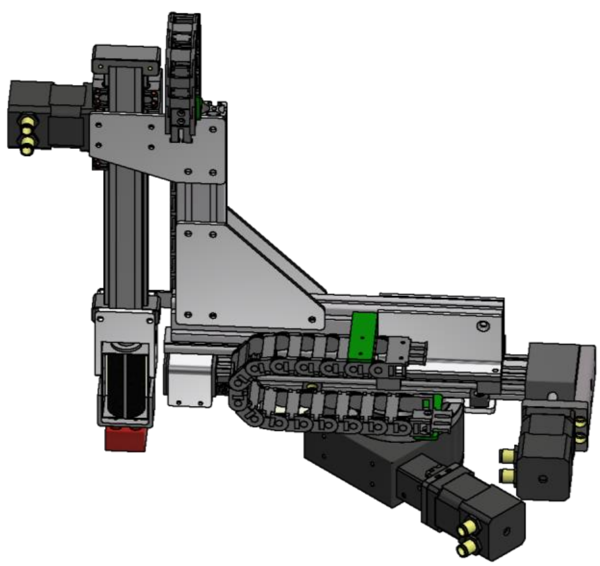


Figure 2-22 Assembly Robot Concept

All the actuators are driven by stepper motors and include an incremental type encoder to feedback their position, in Table 2-2 the technical specifications of the motors used for each axis are shown.

Table 2-2 Joints Technical Specifications

Specification			
Joint Number	1	2	3
Joint Type	revolute	prismatic	prismatic
Max Range	360 degrees	200mm	150mm
NEMA Size	17		
Voltage	24-48 VDC		
Current	1.8 A		
Holding Torque	0.5 Nm		
Repose Torque	0.22 Nm		
Resolution	1.8°/step		

Supply Mechanism

In this stage of development of the project, the integration of a spherical robot is proposed to execute the task of supplying materials. This proposal is made based on the working space of the robot, and the requirement that the system must be easy to transport.

The supply robot is proposed with this configuration, because its workspace resembles the shape of a sphere concerning the coordinate system of space, as shown in Figure 2-23. This workspace can be enough to reach the points of interest inside the machine - entrance section of the slides - and a specific point where the bricks can be taken. The robot has five revolutionary joints. From which the first 4, numbering from the base of the robot to the end effector. are known as position joints, because with these joints the robot will reach the desired final position. The last joint is known as orientation joint, with which the robot will orient the end effector most appropriately so that it performs its task in the indicated position.

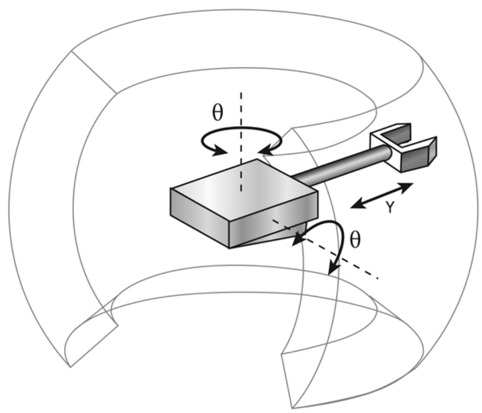


Figure 2-23 Generic Workspace for a Spherical Robot[35]

For this application, a feasible solution is presented through the IGUS® robot line, which offers robots with 5 degrees of freedom that can be conditioned for this system. Due to the configuration of the machine, the best option to incorporate this robot is placing it on top of it. In this way, the robot will share the same axis of reference so that it is possible to reach all the desired points. Figure 2-24 shows the concept of this robot. This configuration was made with a tool provided by the same company to select the appropriate elements for the application.

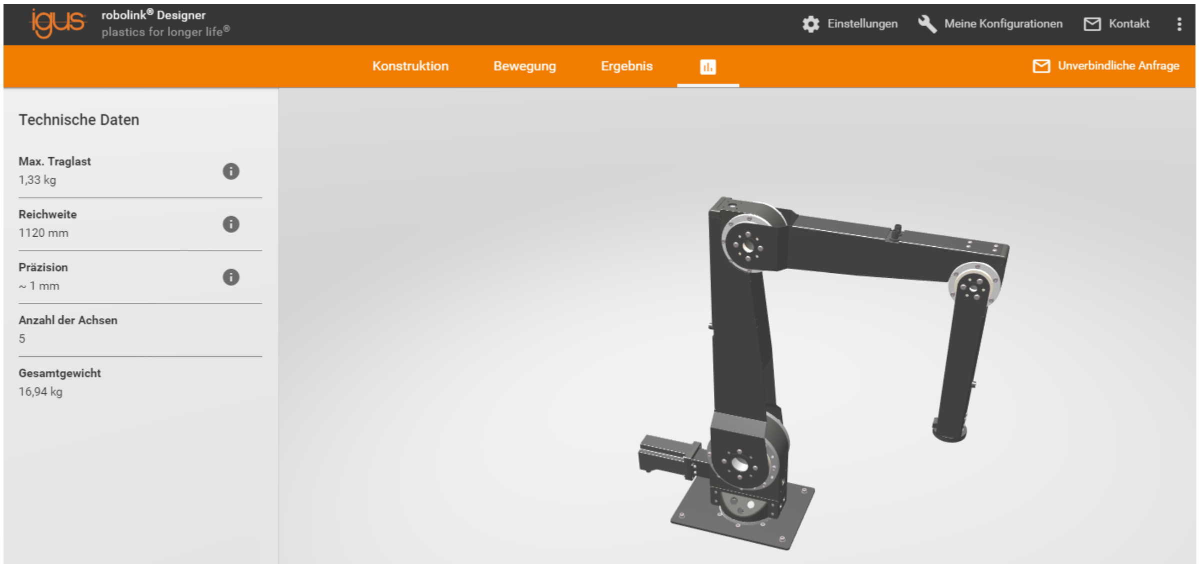


Figure 2-24 Configuration of the Supply Robot

All the actuators are driven by stepper motors and an incremental type encoder to feedback their position, in Table 2-3 the technical specifications of the motors used for each axis are shown.

Table 2-3 Supply Robot's Stepper Motors Technical Data

Specification				
Joint	5	4	2,3	1
Motor Size	NEMA 11	NEMA 17	NEMA 23	NEMA 23XL
Nominal Voltage	24-48 VDC			
Nominal Current	1.0 A	1.8 A	4.2 A	4.2 A
Holding Torque	0.12 Nm	0.5 Nm	2.0 Nm	3.5 Nm
Repose Torque	0.004 Nm	0.022 Nm	0.068 Nm	0.075 Nm
Resolution	1.8°/step			

A general view of the concept of both robots mounted in the assembly station can be seen in Figure 2-25. From this view is possible to see the complete setup of the mechanisms that would develop the core tasks of the machine.

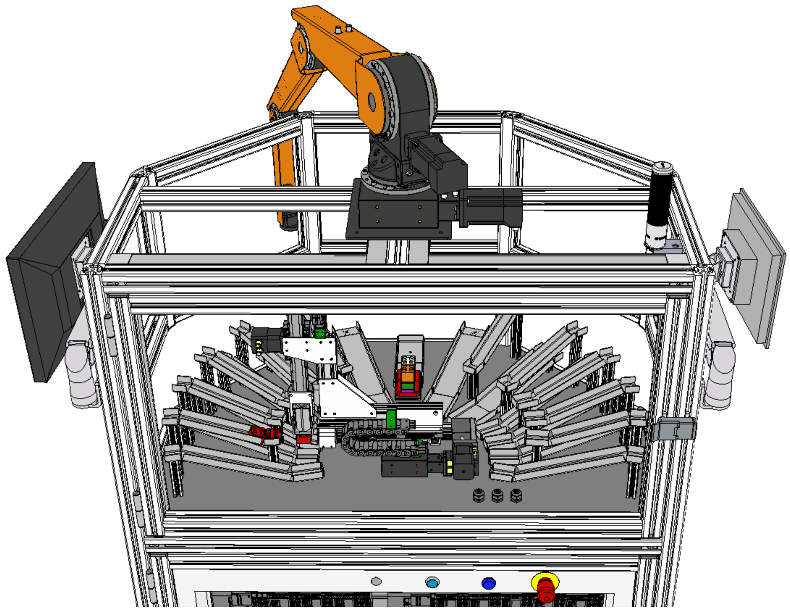


Figure 2-25 View of the upper section of the Assembly Station

In both robots, each integrated joint generates only one degree of freedom (DOF). That means, each joint can move only on one axis regardless of the type of movement, linear or rotational; therefore, the total amount of degrees of freedom that each robot has is 3 and 5 degrees respectively, from this also arises the reference name for its various control components.

Electrical Description

General Components

The components that make up the electrical system of a machine must be placed inside a cabinet so that they are protected from dust, water or any other agent that could affect its proper functioning. These cabinets are installed near the place where the machine will frequently operate, in the case of the station that was designed will not be fixed in one place, because it must be transported to different places for its demonstration. Therefore, the cabinet must be installed inside the machine, thus ensuring portability and operability in any place.

Electrical enclosures are usually made from rigid plastics, metals, mainly stainless steel, carbon steel, and aluminum. Inside the cabinet where the elements are mounted on a metal plate using DIN rail sections. This rail is a metal rail of a standard type used for mounting industrial control equipment and are typically made from cold rolled carbon steel sheet with a zinc-plated polished surface finish. The term derives from the original specifications published by Deutsches Institut für Normung (DIN) in Germany.

Due to the requirement of the portability of the machine the electrical power supply is a critical point, in such a way that to supply power to the whole machine, a standard voltage level must be used. That is why it is proposed to use 220V of alternating current, thus ensuring that the machine can be used in most of Europe.

Usually, for industrial control systems the essential components that come with are:

- Buttons and switches
- Lights and Signal Towers
- Power Suppliers or Power Converters
- Control Devices
- Network Devices
- Actuator Drivers
- Sensors

For the selection of the buttons and lights, which the operator must be in contact must of the time, the recommendations of the European standard EN 60204-1:2006, Safety of machinery - electrical equipment of machines, are to be considered.

Both in the GUI and in the cabinet, push and latched buttons should be color-coded with the Table 2-4.

Table 2-4 Color-coding and explanation for Push-buttons[36]

Color	Meaning	Explanation	Examples of Application
RED	Emergency	Actuate in the event of a hazardous situation or emergency	Emergency Stop Initiation of emergency function
YELLOW	Abnormal	Actuate in the event of an abnormal condition	Intervention to suppress abnormal condition Intervention to restart an interrupted automatic cycle
BLUE	Mandatory	Actuate for a condition requiring mandatory action	Reset function
GREEN	Normal	Actuate to initiate normal conditions	
WHITE	No specific meaning assigned	For general initiation of functions except for emergency stop	START / ON (preferred) STOP / OFF
GREY			START / ON STOP / OFF
BLACK			START / ON STOP / OFF (preferred)

Indicator lights and displays serve to give the following types of information to the operator:

- An indication that a specific task should be performed. Typically the colors used are RED; YELLOW, BLUE, and GREEN.
- Confirmation of a command or the termination of a transition period. Usually, the colors used are BLUE or WHITE, and in some cases, GREEN may be used as well.

In Table 2-5, the relation between the status of the machine and the colors of the indicator lights is explained.

Table 2-5 Color-coding for Indicator Lights and their meanings[36]

Color	Meaning	Explanation	Action by Operator
RED	Emergency	Hazardous condition	Immediate action to deal with hazardous condition (i.e., switching off the machine supply, being alert to the hazardous condition and staying clear of the machine)
YELLOW	Abnormal	Abnormal condition Impending critical condition	Monitoring and intervention (i.e., by re-establishing the intended function)
BLUE	Mandatory	Indication of a condition that requires action by the operator	Mandatory action
GREEN	Normal	Normal condition	Optional
WHITE	Neutral	Other condition, may be used whenever doubt exists about the application of the other colors	Monitoring

Indicator lights and displays shall be selected and installed in such manner as to be visible from the average position of the operator, in case that additional emphasis is required possible to provide flashing mode to the indicator lights and displays, according to the following purposes:

- Attract attention
- Request immediate action
- Indicate discrepancy between the command and the actual state
- Indicate a change in process (flashing during transition)

It is recommended that higher frequency flashing lights or display be used for higher priority information.[36]

For the rest of the components and accessories for the ICS's, there is a great variety of brands such as Siemens®, Phoenix Contact®, and so forth. In this case, the brand that will be used is Wago®.

When a robot picks up a part, the robot must stably grasp one part by avoiding the collision between the robot and the environment.[37] At present, the picking problem is solved with different technologies. Handling is not a problem if parts have fixed forms and fixed mechanical properties, which facilitates their manipulation with vacuum suction pads or specifically designed clamps.

Although there are several commercial options of end effectors for robots of different operation methods, dimensions and costs, thanks to the iterative process of design of assembly stations for cars with LEGO® bricks, the final effector operation method is a concept of which He has enough knowledge.

Because the natural way of removing the bricks is applying pressure between the upper and lower faces between 2 cubes, this way is the most appropriate way for the end effector to take the pieces from the slides to the assembly area without requiring any external force during the journey. The Figure 2-26 shows this procedure explained in more detail. The problem arises when releasing the piece because a force similar to that exerted for the union of the bricks is required. Currently, this is solved with a pneumatic cylinder that applies a constant force in the center of the piece to be able to release it; this implies the integration of a pneumatic installation to satisfy this need, adding a degree of complexity to the assembly station.

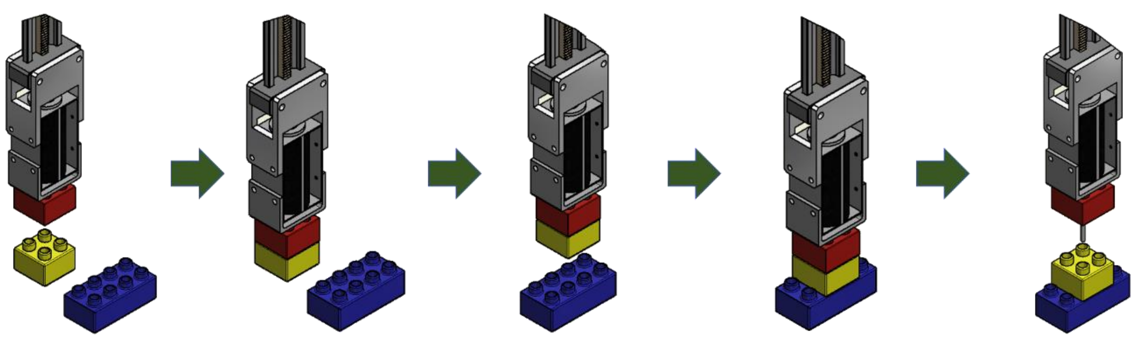


Figure 2-26 Bricks Assembly Procedure

It is for the preceding that the replacement of the pneumatic cylinder is proposed by a component with a linear movement whose power source is electrical so that the entire pneumatic system could be eliminated. The selection process was made by other students working on one of the other assembly stations as an after work for their machine. The model of the solenoid and its mounting brackets were proposed as shown in Table 2-6.

Table 2-6 Solenoid Technical Specifications

Description	Pull magnet, Pressure solenoid with spring return
Voltage	12 / 24 VDC
Weight	317 g
Stroke	25-30 mm
Max Power	80 Watt
Max Force Stroke	56 N/mm

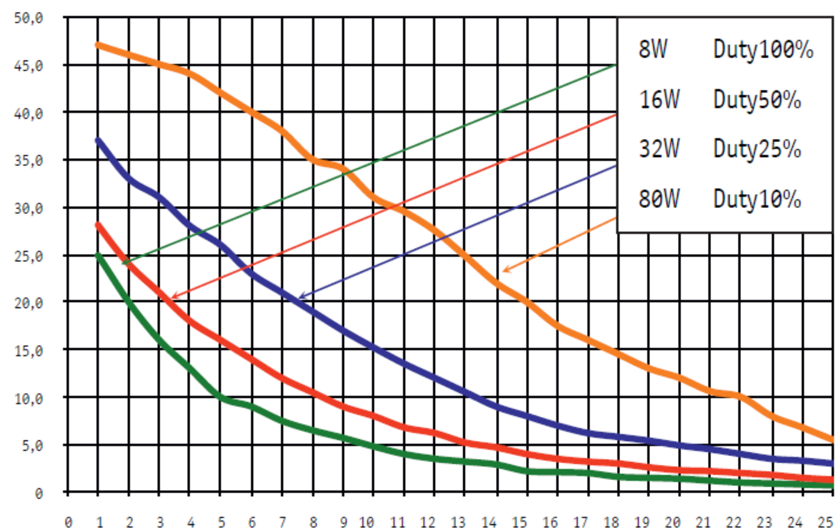


Figure 2-27 Solenoid's Force vs Stroke Curves
Technical Data extracted from manufacturer's datasheet

Assembly Controller

Although in the market of components for ICS's is quite competitive and all brands offer solutions that meet the requirements of machines such as the one described here. For this project the possibility of using components from brands such as Siemens® or Beckhoff® was considered due to the extensive experience with which the work team counts; favoring in no small extent the implementation of components of the Beckhoff brand. Because to date, it is one of the leading brands to focus their developments based on the IEC 61131-3 standard together with the 3S-Smart Software Solutions® company and its platform of development CoDeSys®.

Despite all these favorable characteristics for the implementation of any of these brands, another option that satisfies the characteristics that could be required. Wago®, like Beckhoff®, has solutions for automation applications that are quite flexible, working under the PLCOpen approach indicated by the IEC61131-3 standard, as well as having several years of experience in the development of all the peripheral components necessary for ICS's. At affordable prices.

On this project, the possibility of sponsorship was presented by Wago®, who was very accessible to participate in the development of a project of these dimensions, because like the FH Aachen share the approach oriented to the industry 4.0.

The controller must have a port to be able to communicate with the ERP system and be able to share the appropriate information since the server is mounted in the cloud and transfers the information to the other stations through the internet, it would be more appropriate for the PLC has at least one Ethernet port.

For each stepper motor, it is necessary to use a module that controls the speed and position, as well as a module that can interpret the signals of the position sensors that are integrated into each joint as feedback.

It is also necessary to include digital input and output modules to be able to control the signaling elements or receive the signals from the limit sensors of each stroke of the robot.

In addition to the modules for controlling stepper motors and the modules for reading the signals from the encoders of each of the motors, the PLC must include a master module for the AS-i safety protocol; this module will be responsible for configuring the protocol and assign addresses to the security slaves. Putting all these requirements for the PLC together, the configuration of the PLC described is shown in Figure 2-28.

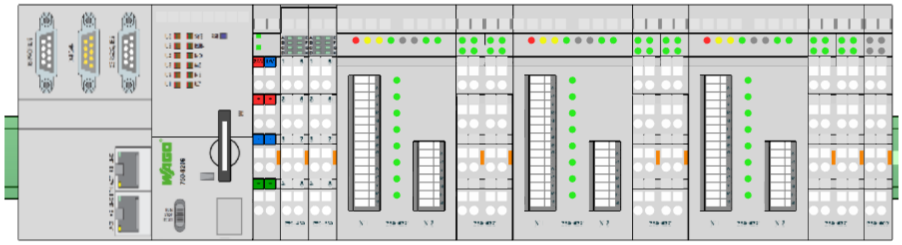


Figure 2-28 Set-up of the 3DOF Robot's PLC

In Table 2-7, the order in which the controller modules are distributed in the PLC Fieldbus is shown, the position or slot 0 (from left to right) being the same controller and the last module would be the bus terminator.

Table 2-7 Modules of the 3DOF Robot's PLC

Slot	Model	Type
0	750-8206	PFC 200 Controller
1	750-655	AS-I Master
2	750-430	8 Channel Digital Inputs
3	750-530	8 Channel Digital Outputs
4	750-672	70VDC Stepper Motor Driver
5	750-637	Incremental Encoder Module
6	750-672	70VDC Stepper Motor Driver
7	750-637	Incremental Encoder Module
8	750-672	70VDC Stepper Motor Driver
9	750-637	Incremental Encoder Module
10	750-600	Fieldbus End Cap

Supply Controller

The second robot, whose task will be to supply bricks to the assembly station, can be considered a replica of the concept of the first robot. Because the components will be the same brand and with very similar characteristics; Although the configuration of the robot is different, the motors used to move each of its joints have the same control technique. In such a way that the components that will be used for the first robot could be used for the second; without the need to make any modification or adaptation to the controller's configuration.

So, for the second robot, the configuration of the controller with all the necessary modules to control all the articulations of the robot is as shown in Figure 2-29.

Figure 2-29 Set-up of the 5DOF Robot's PLC

In Table 2-8, the order in which the controller modules are distributed in the PLC Fieldbus is shown, the position or slot 0 (from left to right) being the same controller and the last module would be the bus terminator.

Table 2-8 Modules for the 5DOF Robot's Controller

Slot	Model	Type
0	750-8206	PFC 200 Controller
1	750-430	8 Channel Digital Inputs
2	750-530	8 Channel Digital Outputs
3	750-672	70VDC Stepper Motor Driver
4	750-637	Incremental Encoder Module
5	750-672	70VDC Stepper Motor Driver
6	750-637	Incremental Encoder Module
7	750-672	70VDC Stepper Motor Driver
8	750-637	Incremental Encoder Module
9	750-672	70VDC Stepper Motor Driver
10	750-637	Incremental Encoder Module
11	750-672	70VDC Stepper Motor Driver
12	750-637	Incremental Encoder Module
13	750-600	Fieldbus End Cap

Safety Components

When we are designing a machine, an aspect to be considered is to implement safety systems to overwatch the possible risk conditions that could appear during the operation mode of the machine. Unfortunately, nowadays there a broad set of safety-related standards that could make expensive and complicated for machine builders to implement these standards correctly, nevertheless they are still responsible for their products and related safety aspects.

In the modern machinery, is more and more often seen a clear separation between the safety part and the functional part, understanding functional part as the main routine or program that the machine follows during normal operating conditions. This separation most of the times results in the safety aspects being included at the end, and not integrated into the whole system philosophy since the beginning, and poorly tested.

The necessary standard requirements of a safety application for machine builders within all applicable safety standards are:

- Distinction between safety and non-safety functionalities
- Use of applicable programming languages and languages subsets
- Use of validated software blocks
- Use of applicable programming guidelines
- Use of recognized error-reducing measures for the lifecycle of the safety software

For all these aspects the standard IEC 61131 within the PLCOpen Safety section provides a generic software architecture model (Figure 2-30) to describe the typical implementation of the safety function blocks, and their interaction with the functional application also differentiates between both, Safety and Functional, applications within the machinery control system.[38]

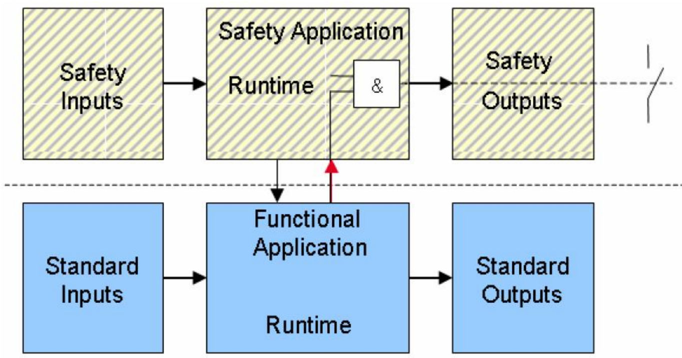


Figure 2-30 Generic Software Architecture Model for Machinery Control Systems[38]

Regarding the hardware for the safety system, there is an open standard Fieldbus that is applied by most of the leading manufacturers of electrical components for industrial automation, including Siemens®, ABB®, Wago®, and so on. This standard protocol is called Actuator/Sensor Interface or AS-interface.

Abbreviated to AS-i, is a connection system for the lowest process level in automation systems. AS-Interface provides the option of transferring standard data and safety-oriented data on the same cable, allowing, for example, the emergency stop cabling of a machine or system to be implemented over AS-i.

Several main characteristics commonly distinguish the AS-interface:

- Is optimized for connecting Digital and Analog sensors and actuators.
- The wiring is straightforward and cost-effective, and because it enables to do tree-like topologies, it has high flexibility.
- The maximum reaction time of an AS-i master device is 5ms with up to 31 nodes.
- The Nodes on the AS-i bus can be either Sensors/Actuators with AS-i already implemented or Modules to which Sensors/Actuators can be connected
- Up to 124 Actuators/Sensors can be operated via AS-i cable

However, the most well-known characteristic of the AS-interface is that the AS-i cable is used for data exchange between slaves and master meanwhile supplies power for the sensors and the actuators. [39]

The essential components of an AS-i network are AS-i power unit, AS-i cable, AS-i master, a Safety Monitor and AS-i Slaves (Modules or Sensors/actuators with AS-i integrated); but if the network requires it, other devices such as Diagnostic units, Repeaters, Extenders and Extension Plugs can be added. A generic example of an AS-I network topology is shown in Figure 2-31.

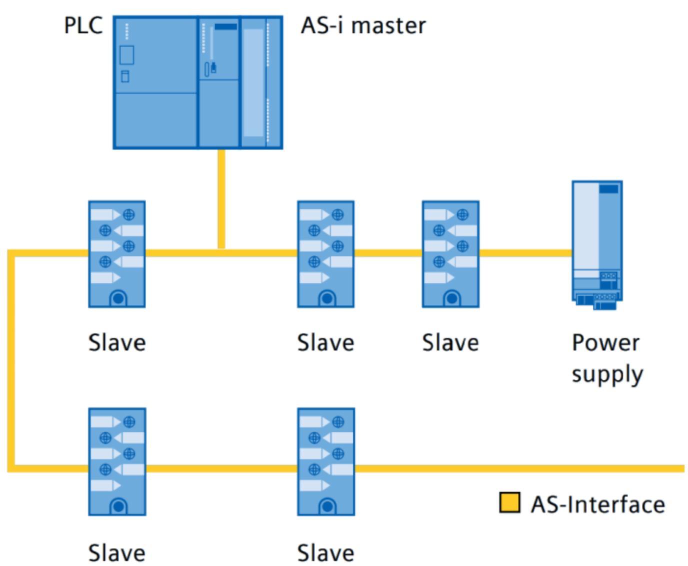


Figure 2-31 Basic Topology of an AS-i Network[39]

Safety relays are widely used to maintain safe working conditions, they react to changes outside of standard operating parameters, thus informing control devices to remove potentially hazardous to machinery and personnel.

In Figure 2-32, can be seen the general concept of the security system that will be implemented in the assembly station. In this system, the PLC of the assembly robot will be the one that works as a master of the AS-i protocol. It also requires a Safety supervisor who is in charge of monitoring the signals of all safety signals connected to each installed slave. Also, this supervisor will be in charge of activating or deactivating a safety relay that cuts off the power supply of all the actuators of the machine. That is, this safety relay will cut the power supply of the drivers for each of the motors of both robots, as well as the feeding of the solenoids that are used as final effectors.

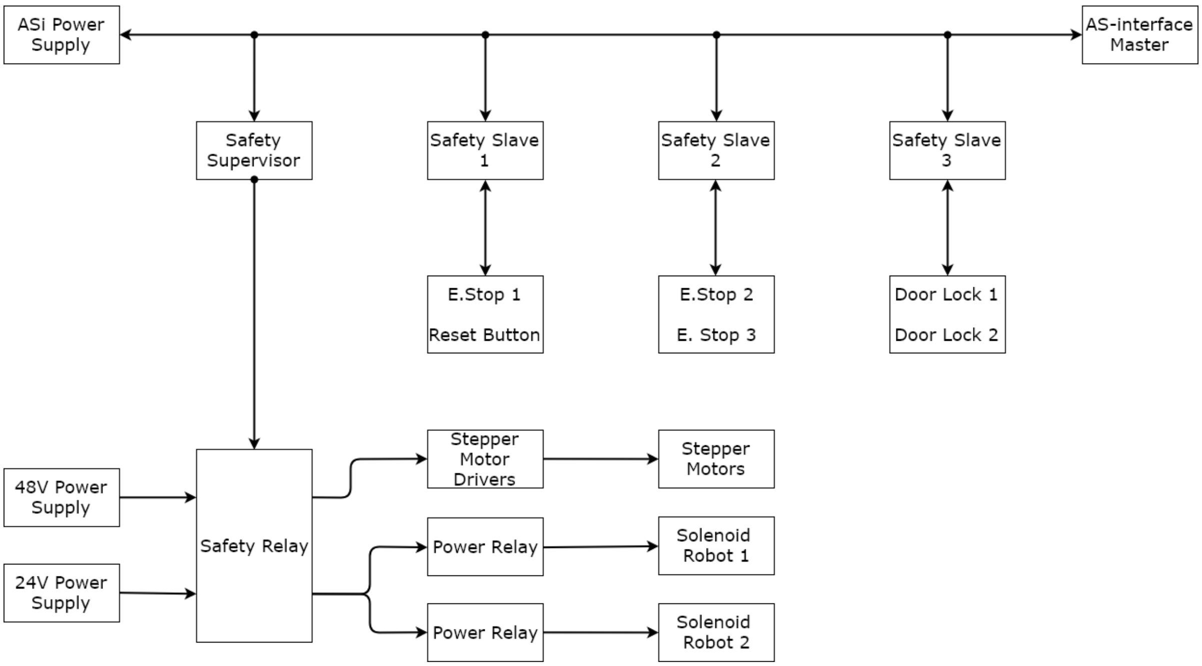


Figure 2-32 General Concept for the Safety System

In the case of this machine, the following signals for the security system are considered:

- Three Emergency stop buttons placed on different sides of the machine near the area where the operator will be living with the machine.
- One Reset button which will be located in the door of the lower section of the machine.
- Two Security door switches, which will be placed in each of the doors that block access to both the assembly section and where the electrical panel is located.

Taking all the above into consideration, we have that the electrical concept of the whole machine is as shown in the block diagram of Figure 2-33, where can be seen the electrical dependence of all electrical components of the machine. From the top, the connection to the electrical network *(220VAC @ 50 HZ)* as mentioned above, followed by a general switch of the whole machine.

Subsequent are the power supplies that will provide the different levels of Direct Current Voltage for the various components as required.

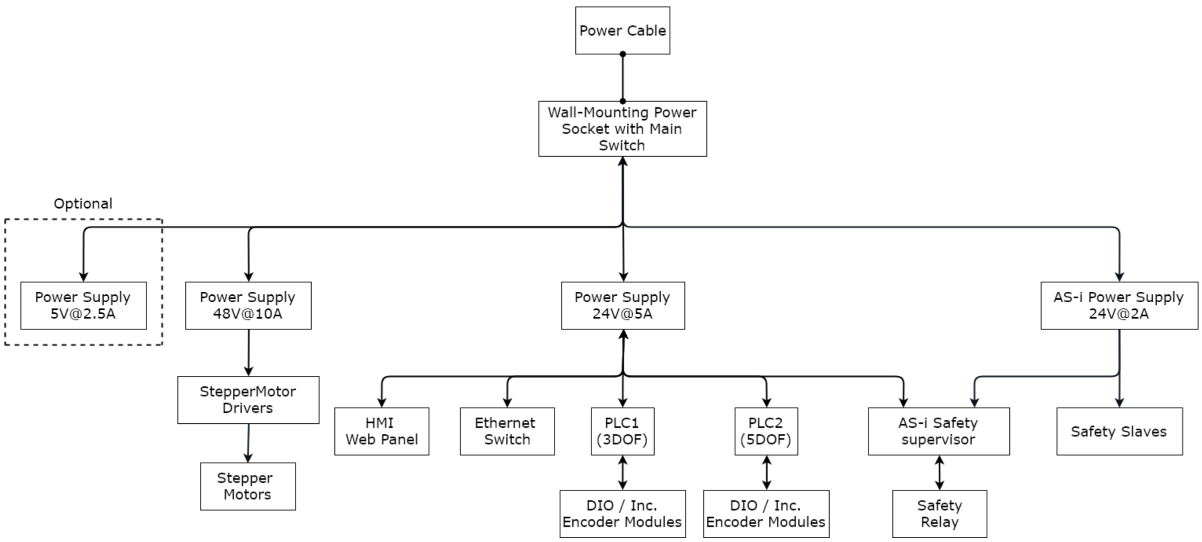


Figure 2-33 Block Diagram of the Electrical Wiring of the Controller

For the general concept, Figure 2-34 shows a 3D model of the electrical panel and its components that will be installed in the machine. The panel will be placed in the lower section of the structure, and the necessary connections to the robots, lamps, and buttons will be hidden through the edges of this section. The panel will be divided into two parts: the rear panel that covers almost entirely the space of the lower cabinet and the front panel that will be of a lower height, but with the same width as the rear panel.

Figure 2-34 3D Model of the Electrical Panel for the Assembly Station

Finally, the touchscreen will be mounted on an extendable arm on one side of the upper section; this will allow the operator to adjust the position of the screen ergonomically and the buttons and lamps will be mounted on the door of the lower section of the machine as shown in Figure 2-35.

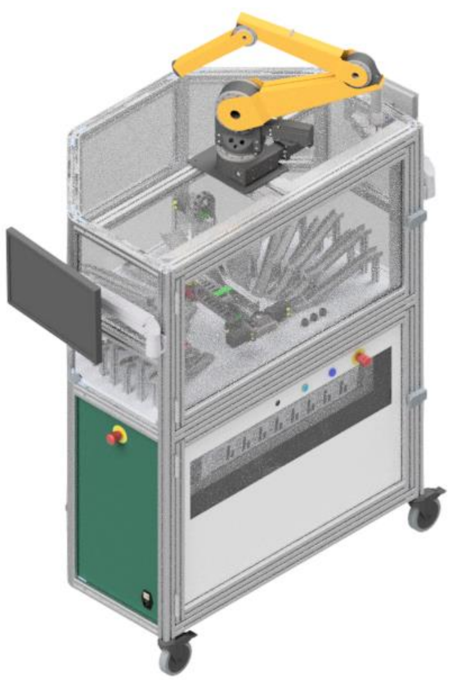


Figure 2-35 Isometric View of the Assembly Station

Software Architecture

Main Logic

As mentioned in previous chapters, the main logic of the assembly station can be separated into several blocks. These blocks will be small parts of the code that will be focused on executing specific routines.

These routines will be governed by a general state machine of the entire system. This state machine will be running continuously within the assembly PLC; this is because it is the PLC with the least number of axes to control and is the one who must perform the most crucial task that is the assembly of the requested product.

The state machine that will be implemented can be seen in Figure 2-36. This state machine works in four main sections which are described below:

- **Initialization**: These are the first states, in which the system verifies that all its components are enabled and ready to operate.
- **Selection of Operation Mode**: a state in which the machine waits for the indication by the operator's pate to determine which are the following states that must be executed
- **Execution**: As mentioned in the general concept of the system, there are several operating modes, each of these operating modes will be executed in different sections of the state machine.
- **Error**: The system is continuously checking fault conditions, so if an emergency condition occurs the state machine changes to an error state, when restoring the state machine returns to the initial state.

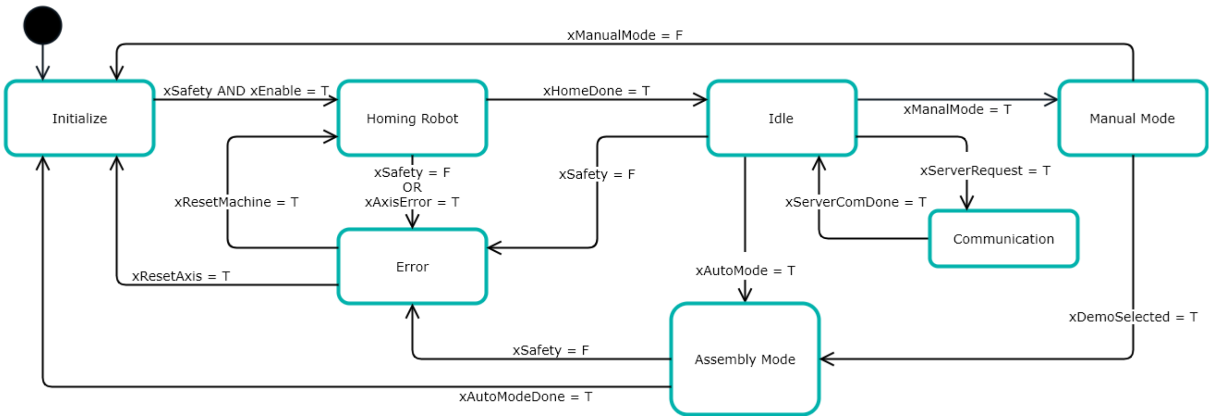


Figure 2-36 Main State Machine Diagram

The general state machine will be executed in the main program of the PLC, in each of the states other routines will be called. In turn, these routines will share data between them so that the primary task can be carried out, the communication between them will be through small functions that will function as interfaces whose task will be to convert and prepare the necessary data types between each of them. Working in this way the communication between the blocks allows a certain level of flexibility in the logic so that in the future -if required- this logic can be modified, and new modules can be integrated. Figure 2-37 shows a diagram of classes of how these subroutines that make up the main logic of the PLC are related.

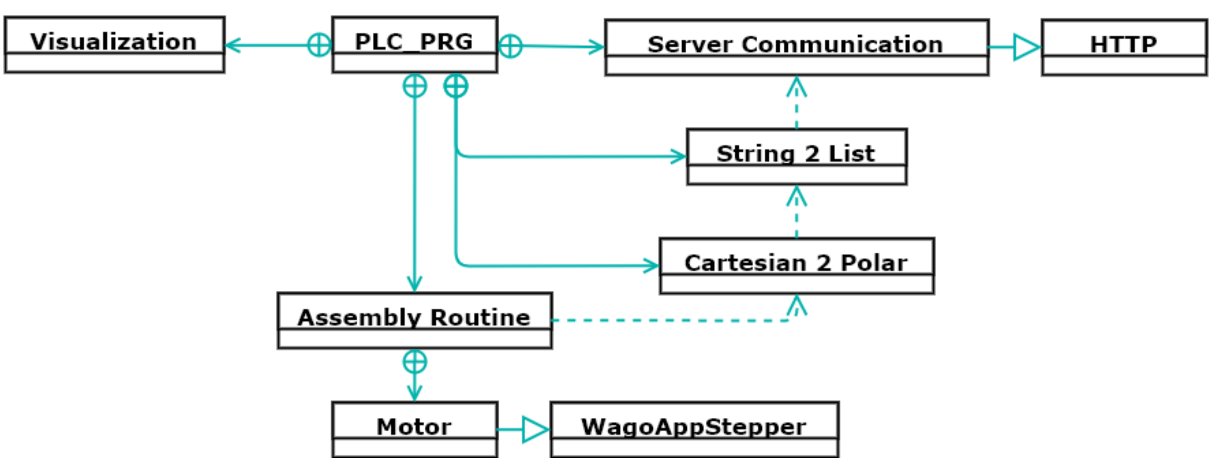


Figure 2-37 Simplified Main Class Diagram

In the diagram above, are shown the four main blocks of the main logic of the system:

- **Visualization** concerning the HMI
- **PLC_PRG** would be the primary function where the state machine is located
- **Server Communication** will be the block in charge of the communication
- **Assembly Routine** and Motor that refers to motion control

In addition to these blocks other blocks that act as interfaces between the aforementioned main blocks can be observed, their operation will be described later.

Communication

Web Communications follow a standard two-stage protocol: the client sends a request expressing an information need, and the server responds with the requested data.[40] As it has already been mentioned, the orders that the assembly station must carry out will be defined by the ERP system; this system is mounted on a computer that functions as a server. That is why it is necessary to establish a protocol through which the plant and the server can share information. Currently, the other assembly stations communicate over the Internet using the HTTP protocol.

From this protocol, is possible to use already defined methods like *GET* and *PUT* to obtain the necessary information from the Server and respond with a status of the Assembly Station. Figure 2-38 shows the URL that the plant must use to request an order to the server.

Figure 2-38 URL format for request to the Server

Concerning the message that the server sends when the station makes a request, it is received in JSON format. This format is commonly used in server-browser communication as it is a free format that is structured as human-readable. Figure 2-39 shows an example of the message that the server issues once a request has been made. In this message contains detailed information on the configuration of the product to be manufactured: The type of brick, its official serial number defined by LEGO® and the coordinates of the brick concerning the assembly.

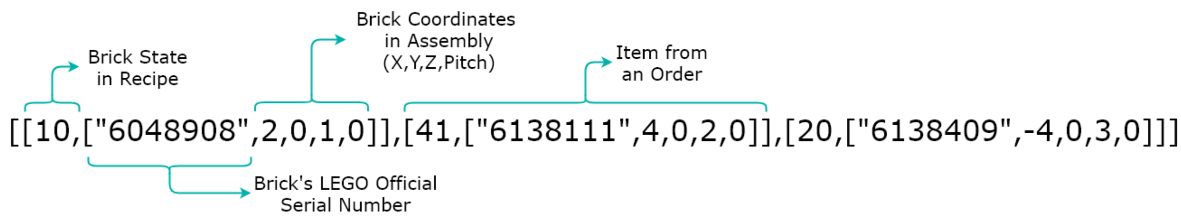


Figure 2-39 Raw JSON format of a Car Order

One of the data contained in the message is the state of the brick; this data is related to the assembly process of each of the different recipes that can be manufactured, in such a way that the same piece can have different states. With these states for each brick, the ERP indicates to the stations the order of the brick concerning the assembly sequence of the recipe. Thus, the value of this state together with the coordinates makes it easier for the station to determine the location of the brick in the assembly.

Figure 2-40 specifies the different possible states of the bricks assigned according to their position in the recipe. Each position of the assembly sequence, indicated by *10*, *20*, *30*, and so forth; also has variations due to the selectable colors in each brick of the recipe.

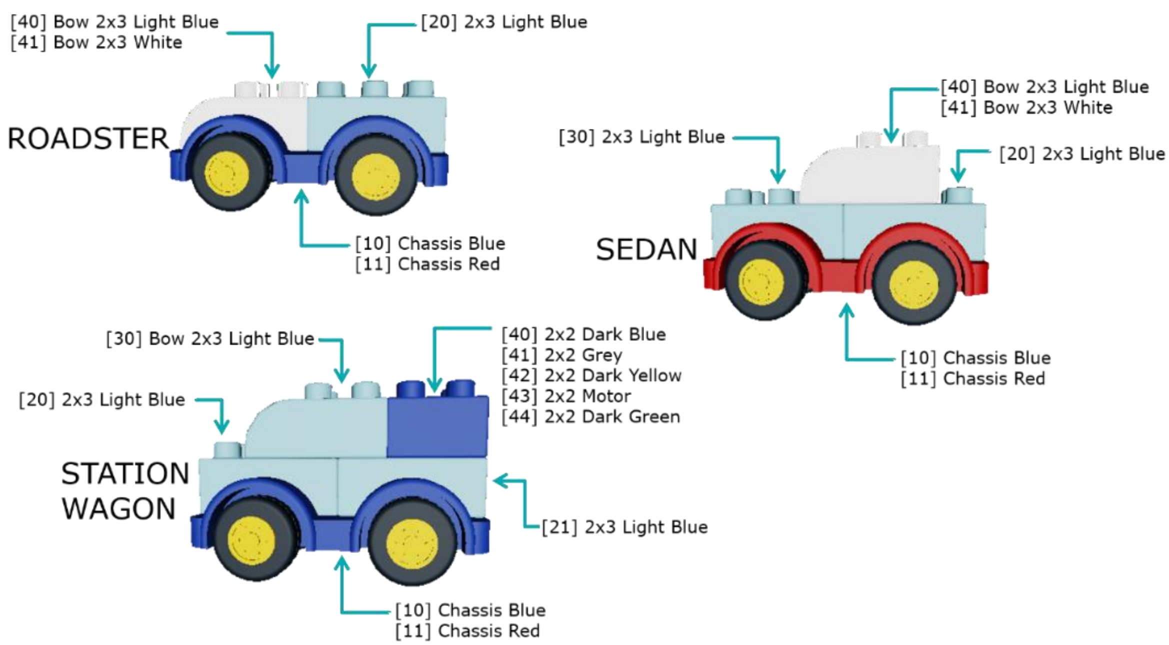


Figure 2-40 Bricks States according to each Recipe

On the other hand, the machine must publish to the server the information of the state in which the machine is located, but this response to the server has not been implemented in standardized form to all assembly stations. For this, the machine must use the *PUT* method of the HTTP protocol.

This subroutine will be executed within the states of initiation of the state machine, so it can be confirmed in the first instance if there is communication with the server and if it does not exist, the station can select an adequate mode of operation. In such a way that at the start of each execution cycle of the main routine, a state of the machine is issued and a new command is received from the server. In Figure 2-41 can be seen the Flowchart of this subroutine.

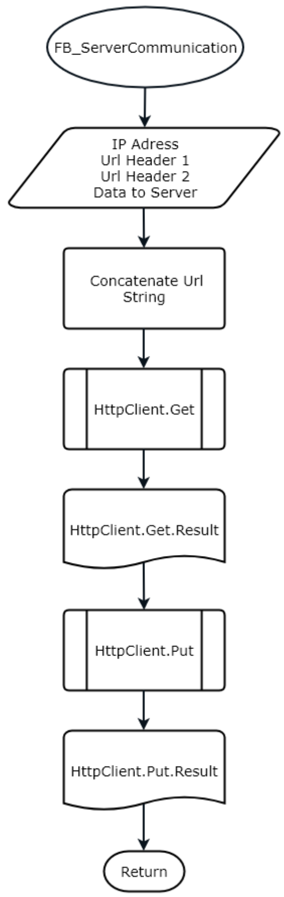


Figure 2-41 Server Communication Flowchart

Assembly

The Robot's pick-and-place action is parametrized with an object to be picked and a location at which the object is to be placed. The action is defined as a sequence of poses relative to the object (pre-grasp, grasp, and lift poses) followed by poses relative to the target location (transfer, lower, and drop poses). [41]

Taking this into consideration, the Figure 2-42 describes the characteristic flow diagram of the assembly routine. This function will be responsible for indicating assigning the corresponding values to the positions and speeds that will have to be taken into consideration for the functions of Motion control; based on the list of objects that are received and conditioned by the server.

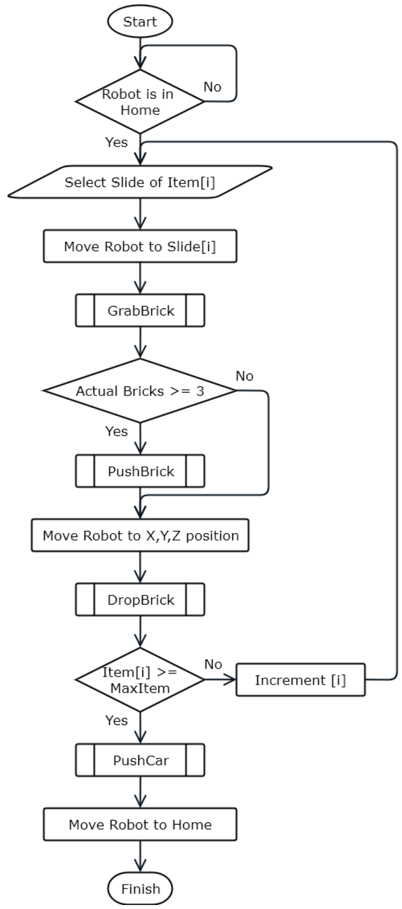


Figure 2-42 Flowchart of the Assembly Subroutine

Motion Control

Motion control is one of the most required tasks in automation. Nowadays, it is quite common to find many projects related to this topic, so much so that it is possible to classify them in the following way:

- CNC application
- Pick & Place application
- Tripod robots and palletizing robots
- Labeling/cam application

That is why the PLCOpen® organization has worked on developing groups of generic function blocks that can be used for any application in which the control of rotating machines will be necessary.

The implementation of these standardized functions is done through a state machine for each axis; this state machine is described in Figure 2-43.

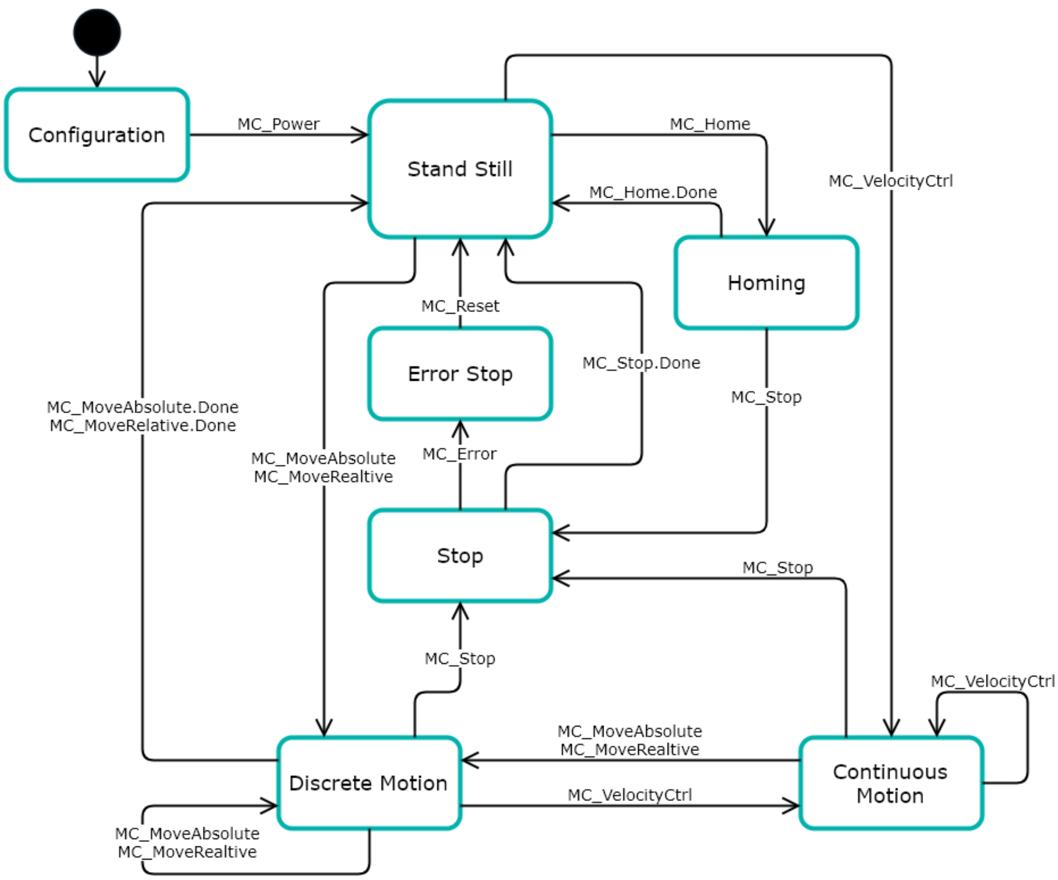


Figure 2-43 State Diagram for Motion Control of an Axis

A typical routine would start loading the configuration parameters and the motor disabled. In this state, the power can be switched on by transferring the axis to the state *StandStill*. From there the *Homing* state can be reached, which after normal completion returns to *StandStill*. From here the axis can also be transferred to either *Discrete Motion* or *Continuous Motion*.

Issuing a single axis move command will bring the axis back to either *Discrete* or *Continuous Motion*. Via the state *Stop* one can return to *StandStill*. *ErrorStop* is a state the axis is transferred to in case of an error appears. With *Reset* command is possible to return to *StandStill*, from which the machine can be moved to an operational state again.[28]

Based on this state machine, an axis responds to a continual changeable set value. This value will be generated by another Function block from the PLC program, in the case of this project will come out of the Assembly function block. Each axis will consist of a single stepper motor which is connected to digital outputs and will react to pulses. Also, each axis will have parametrizable properties which are usually related to the description of the required behavior during its operation.

Safety Logic

Due to the AS-i protocol is ruled by international standards, all hardware and software elements are included in these same standards. Therefore, the security logic that must be implemented in this project must be defined under this same principle. In this way, the security supervisor that will be installed on the machine must be configured with the software of the brand that is used.

The primary task of the security logic is to continuously indicate the status of the OSSDs concerning the configuration of the signals that the security buttons and sensors emit.

The entire sequence operates in principle with a safety circuit in which all the EDMs are connected in series, and the activation of the OSSDs is possible when all these signals are closed. The security logic can be divided into three simple steps:

1. Monitoring safety stop signals: All the signals of the switching elements are connected by an AND-type logical operator, in this way, there will only be a power state when all these signals are active.

2. Monitor reset or restart signals: After all the switching signals are active, they are connected to the reset elements using an OR-type logical operator. The reset the state of the OSSDs can be made with one or more reset signals, but it is not necessary that all signals be activated at the same time.
3. Activate or deactivate the output signals: Once all the conditions of the previous steps have been fulfilled, it can be said that all the signals are activated, now the different outputs that have been configured will be activated.

This sequence of the safety logic can be observed in better detail in Figure 2-44, where a generic example of a function block diagram for the safety logic is implemented.

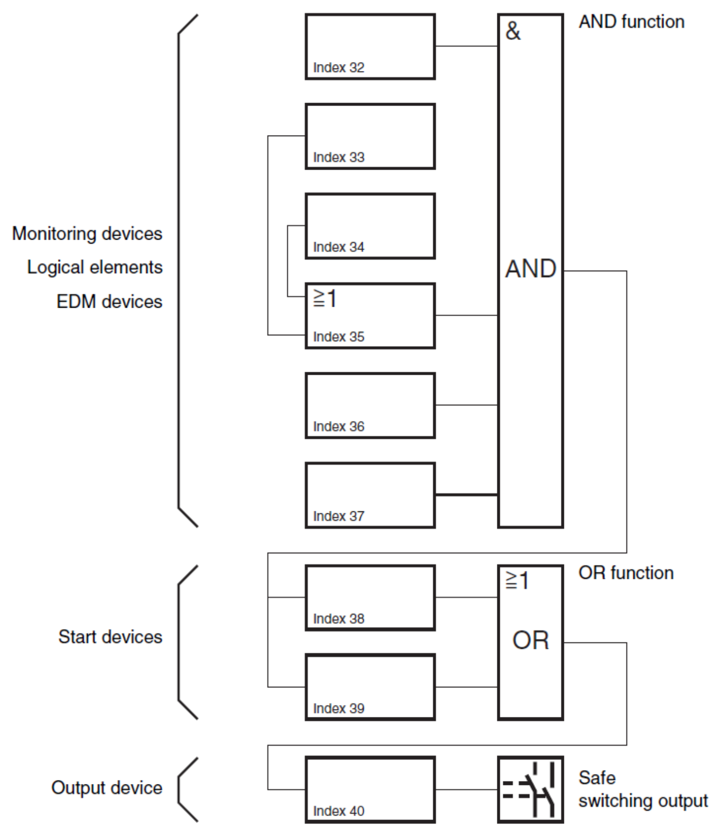


Figure 2-44 Generic Function Block for Safety Logic implementations[42]

Human-Machine Interface

Humans can interpret more information more quickly if it is graphically represented [43] That is why HMIs are an essential factor in the automation world, they play an essential role at the machine level because they provide an interface between the system and the operator. The HMI provides an intuitive visualization of the process providing necessary data for analyzing the behavior of the system. [44]

It can be defined as an HMI from simple line displays with some push buttons for operating the machine to highly complex control centers for machine tools.[45] Nowadays the evolution of IT technologies like the Internet are now defining what HMI systems should look like, within Internet web pages can be displayed in standard web browsers and screenshots are usually displayed in HTML, XML, and BMP format.

One technique of the application of HMIs via the Internet is the web-visualizations are XML format graphical representation of the project variables, which is loaded to the controller together with Java-Applet and can be displayed via TCP/IP on a browser. This visualization allows inputs to the PLC program in online mode virtual elements contained in this representation.[46]

A good HMI should always have a proper page that shows progressively details of the information displayed. Usually, the first page of an HMI is an Overview screen, the pages after the show more detailed information of the process where the interaction with the process occurs. [47] The hierarchy enables the operator to navigate from an overview to details in a logical manner. In Figure 2-45, is shown the hierarchy diagram of the visualizations that should be implemented in the HMI of the system.

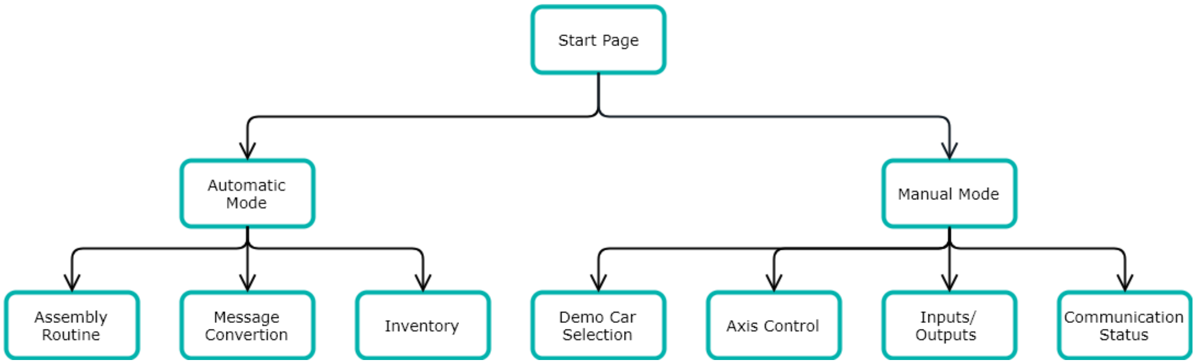


Figure 2-45 HMI's Hierarchy of Pages

When designing an HMI, it is essential to consider several aspects to be able to guarantee that the information that will be shown to the operator is necessary and adequate at all times. So he can make the most appropriate decision while interacting with the system, especially if said HMI contains elements graphics and analog indicators of the machine data. That is why in Table 2-9 a comparison is made of aspects that help to discern between an HMI that is highly effective against a deficient one.

Table 2-9 Comparison between Effective and Poor HMIs[43]

Poor HMIs	**Effective HMIs**
Representation of raw data as numbers (temperatures, pressures, and so forth)	Depiction of process status and values as information, not numbers
No Trends	Layout consistent with operator's model of process
Flashing or Spinning graphics	Key performance indicators as trends
Bright colors, 3D shadows	No gratuitous information
Color coding of piping and active elements contents	Gray backgrounds, low contrast
Measurement units in large, bright text	Limited use of color (as alarming)
Lots of crossing lines	Consistent visual and color coding
Alarm related colors for non-alarm related elements	Gray process lines
Inconsistent colors	Measurement units in low contrast lettering

For this project a Web panel is proposed to work as the HMI for the operator, these web panels are typically configured as a Web browser which directly connects to the controller with their web server via LAN connection. Because the project uses Wago® PLCs, it is advisable to use a web panel of the same brand. The panel that is proposed to be used is from the e!DISPLAY 7300T family model 762-3003, which is shown in Figure 2-46.

Figure 2-46 Wago® Web Panel[48]

This panel has a TFT LCD resistive touch screen with LED backlight and can recognize sweeps and scrolling with the touch at a single point. Because it is a touchscreen, it is necessary to exert a certain level of pressure during its handling. In Table 2-10, the technical specifications of the selected model are listed.

Table 2-10 Technical Data of Web Panel

Parameter	
Display Type	TFT, Wide Viewing Angle
Screen size(diagonal)	10.1" (257mm)
Aspect	16:9
Display colors	16 million colors
Graphics Resolution	1280x800 pixels
Contrast Ratio	800:1
Viewing angle horizontal/vertical	85° / 85°
Brightness	Max 800 cd/m2, settable
Processor	ARM® Cortex™ A8, 32 Bit, 600 MHz
Operating System	Linux 3.6.11
Web Visualization	e!RUNTIME®

3. Results

Mechanical Implementation

Structure

As for the structural components that comprise the assembly station, the slides that store the bricks for the assembly were made of a 2mm thickness aluminum metal sheet. All of them were manufactured and were ready to be installed in the structure of the machine. This structure could not be received in the time comprising this document, due to the delivery times of the aluminum profile components requested. The advantage that the structure has been designed using extruded aluminum profile is the simplicity of the assembly of all its components which will allow the structure to be assembled entirely in a short period.

In the Figure 3-1 are shown the different types of slides that will be used in the machine with the sizes of pieces that will contain each of them.

Figure 3-1 Manufactured Slides for the Assembly Station

First, a slide of the sample was made to determine if the angle of inclination and the friction of the materials were suitable for the application, it was observed that when the amount of bricks in the slides is less than three the bricks do not manage to slide until the tray of departure. In the Figure 3-2, this phenomenon can be observed.

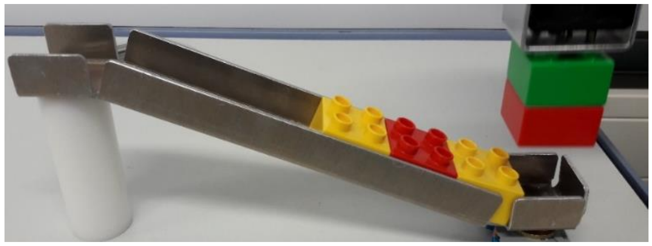


Figure 3-2 Clogged Bricks in Slide

It was also found that a way to solve this condition of the slides. Which would not require modifications to the structure or assembly of the slides is by software. This solution contemplates adding a series of movements that the robot must make when taking a brick from the slide as can be seen in the series of Figure 3-3.

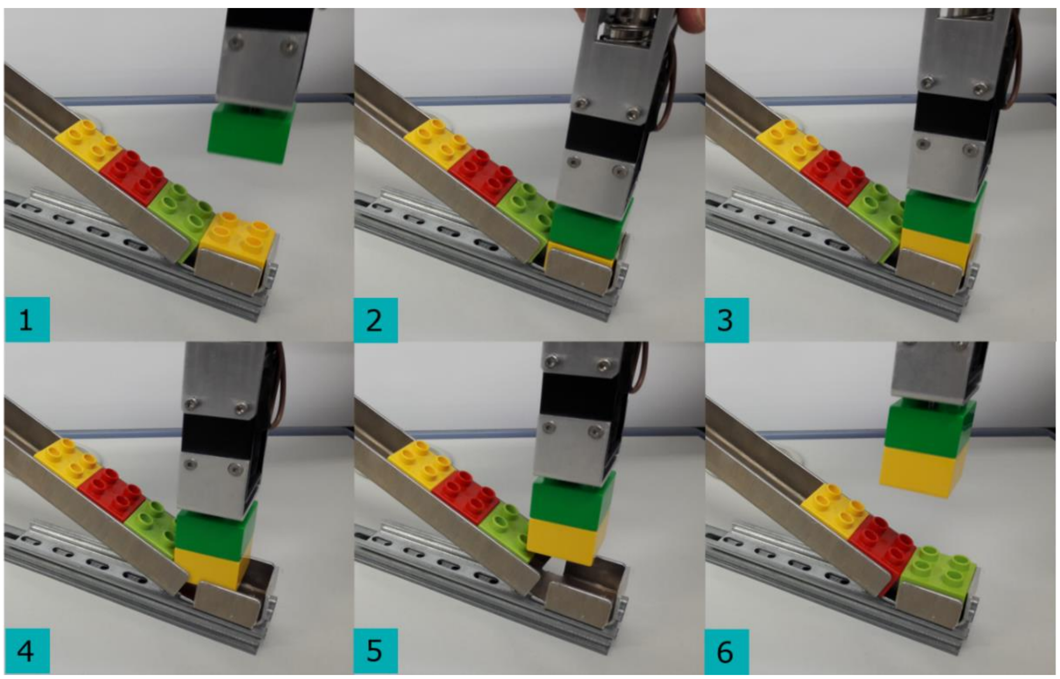


Figure 3-3 Release Routine of the Robot

This routine is divided into the following six steps:

1. Position of the end effector on the piece
2. Take the piece
3. Raise the piece to half the height of the brick
4. Move the piece against the slide half the distance of a brick
5. Raise the piece to the level of trajectory towards the assembly area
6. Correct arrangement of the remaining pieces

Steps 1 and 2 are part of the assembly routine when executed usually; Steps 3 and 4 are the steps that are added when the amount of bricks in the slide is less than four bricks; Steps 5 and 6 are also part of the assembly routine normally executed.

Assembly Mechanism

Because the assembly robot had to meet the requirement of being low cost, the option that was chosen was to manually configure the axes that integrate it, using articulations and individual linear guides instead of resorting to a completely integrated solution. For this reason, the use of IGUS brand components was considered appropriate.

To make the union of all the joints and actuators, it was necessary to design the links that would make up the robot. These links are built mainly by stretches of extruded aluminum profile and pieces made of a folded aluminum sheet. Figure 3-4 shows the articulations of the robot before being assembled.

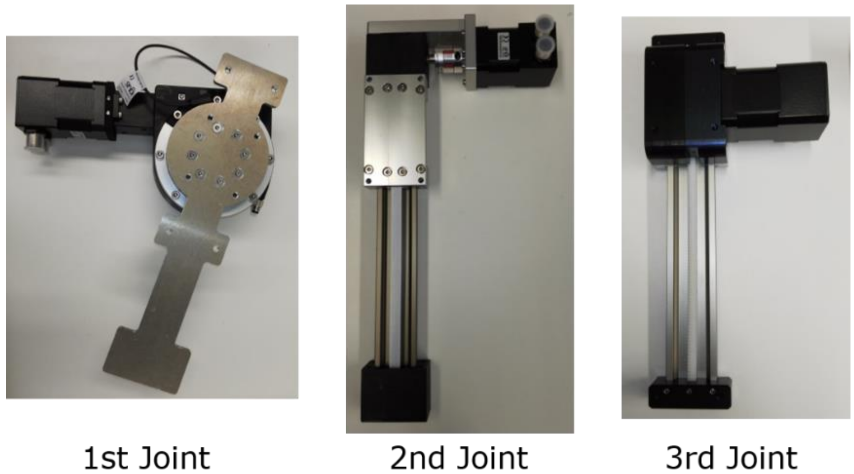


Figure 3-4 Joints of the Assembly Robot

Figure 3-5 shows the final assembly of the assembly robot with the end effector mounted. Each of the joints has a stepper motor and a previously assembled encoder that came with the actuator. Due to the weight of the end effector, the vertical linear guide must be in its position of most significant extension; this implies that, during its operation, the motor must continuously be energized to maintain the desired position concerning the Z-axis.

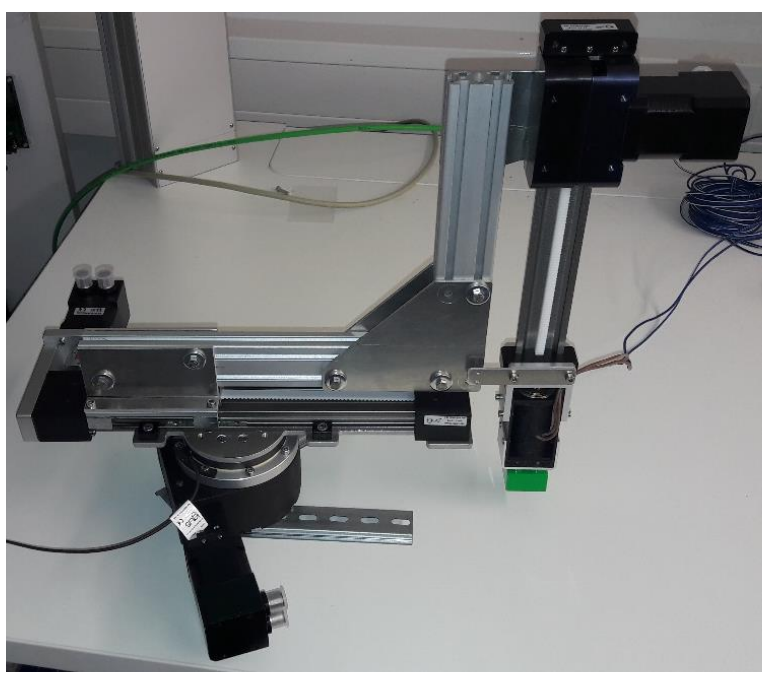


Figure 3-5 Assembly Robot

Because the structure was not available, the robot could not be assembled to test its workspace and calibrate the parameters of its motors.

It should be mentioned that the actuators themselves have a mechanical play that is almost nil or considerably negligible. However, when the complete robot is assembled, and all its joints are extended to the furthest point of its working space; in the end effector, a mechanical play of about 0.5 ° can be seen concerning the base joint. Concerning the vertical axis, the end effector also has a mechanical play due to the weight of the end effector and the dimensions of the linear guide that supports it. However, these perturbations to the position of the end effector do not turn out to be critical to a large extent. So, clamping and releasing the bricks is not affected.

Supply Mechanism

The robot to supply the bricks to the slides in the machine will not be implemented at this stage of project development. This was because this robot was not considered in the initial scope of the project. Although the machine is suitable for the integration of said robot, the considerations that were made are based on the characteristics of a robot that IGUS® offers already configured. A more detailed review of the mechanical characteristics of this product line would have to be done to determine if they meet the requirements of this application. Thus. The detailed characterization of this is pending for the second stage of development of this machine.

However, the electrical installation and the structure are designed to support a solution of this range of products.

Electrical Implementation

General Components

The first step was to make the electrical diagrams; these were developed using the software EPLAN 2.7 in its student version, in the Figure 3-6 can be seen the section referring to general components of the electrical diagrams. The full version of the electrical diagrams can be reviewed in the appendix section.

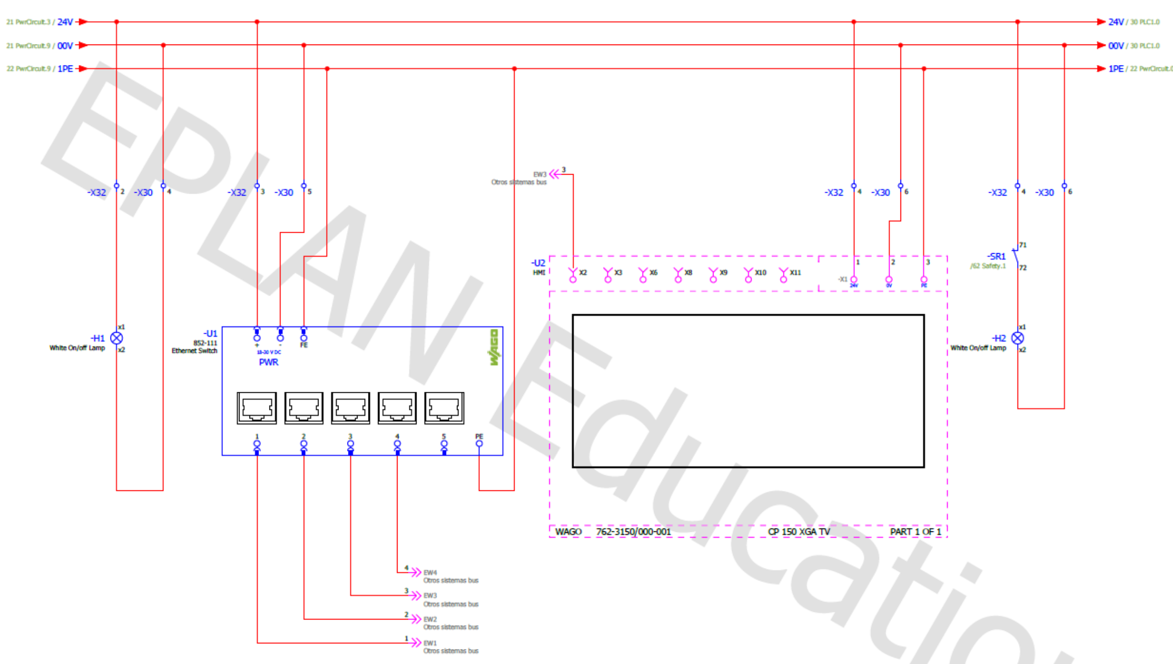


Figure 3-6 Electrical Schematic of General Components

As for the construction, the entire panel was assembled on a stainless-steel plate where all the electrical components were mounted as shown in Figure 3-7. The distribution of the components is based on the concept proposed in the previous chapters. Regarding space and distribution of components, the panel is enabled to add components in the future if necessary, as well as the possibility of adding digital signals to the PLC if required.

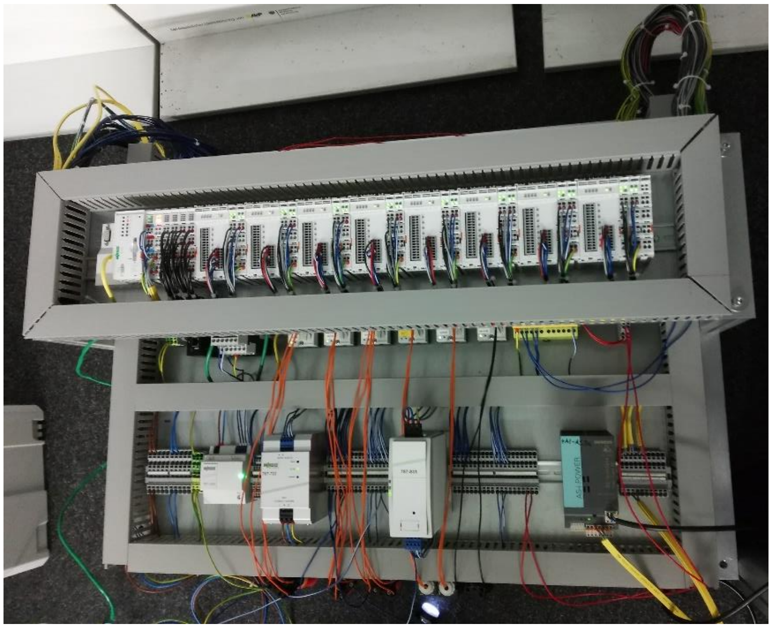


Figure 3-7 Assembly of the Electrical Panel

For wiring the panel, it was done in such a way that the result was a board that could be used to execute logic tests without the need to have all the mechanical components operable; the board was enabled to be installed inside the machine without having to make many steps during the assembly of the structure. For the wiring, it was carried out under the indications of IEC 60446: 2007 and the standard used by the FH Aachen for the development of electrical panels. Table 3-1 lists the colors used, their characteristics and their application.

Table 3-1 Color-Coding applied to the Electrical Cabinet

Color	**Diameter in mm^2**	**Description**
Light Blue	1.5	220VAC Line
Brown		220VAC Neutral
Green-Yellow		Protected Ground
Dark Blue	0.75	VDC V+
Dark Blue-White		VDC V-
Black		Digital Signal Positive Reference 1
Gray		Digital Signal Negative Reference 1
Red		Digital Signal Positive Reference 2
White		Digital Signal Negative Reference 2
Orange		Safety devices
Yellow (Light Blue/Brown)	2 x 1.5	AS-I Interface

For the feeding of all the components that comprise the ICS. It was necessary to integrate four voltage sources, which provide the different voltage levels of direct current, from 5V for the ignition, 24V for the controllers and general components and 48V which is the voltage required for the actuators of the system as can be seen in Figure 3-8. Also, in Table 3-2 are shown the implemented models and their technical characteristics.

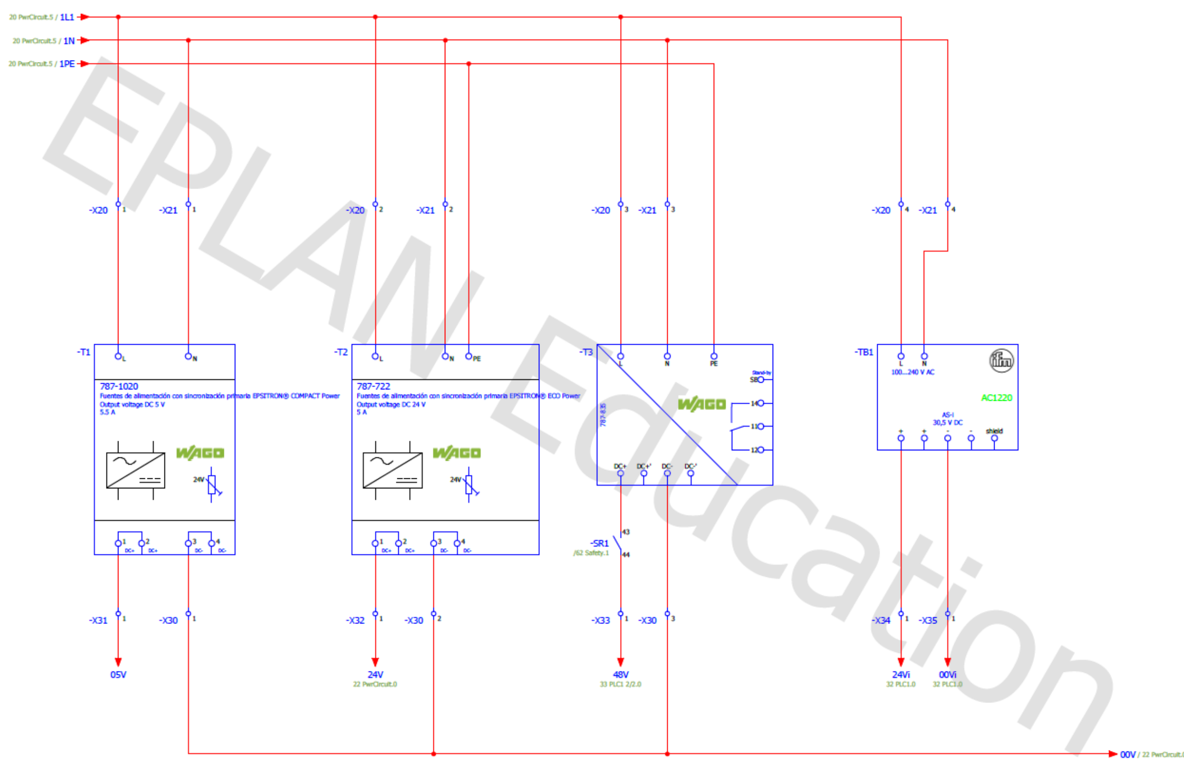


Figure 3-8 Electrical Wiring Diagram for Power Supply Devices

Table 3-2 Power Supplies Specifications

				
Brand	Wago	Wago	Wago	Siemens
Model	787-1020	787-722	787-835	3RX9-502-0BA00
Input Current	0.48 Amp	1.6 Amp	4 Amp	1.5 Amp
Power	12 W	100 W	480 W	345 W
Output Voltage	5 VDC	24 VDC	48 VDC	24 VDC AS-i
Output Current	2.5 Amp	4.2 Amp	10 Amp	5 Amp

In the machine, a LAN was installed through which; the PLC connects to the server where the ERP is mounted, also the touchscreen is connected by Ethernet with the PLC to make the deployment of the operator interface. To make all these connections, a passive ethernet switch was installed, considering the communication with the second PLC and leaving an Ethernet port available to connect a computer for maintenance purposes. The topology of the LAN can be seen in Figure 3-9.

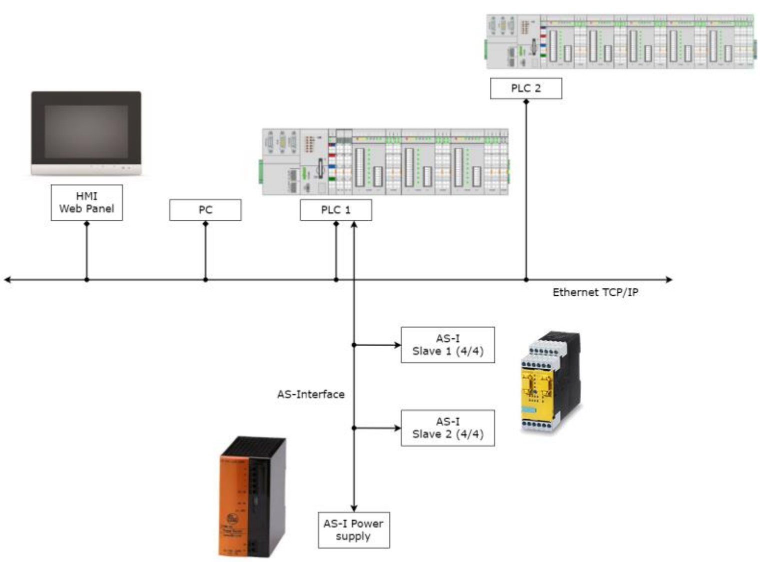


Figure 3-9 Network Topology of the Assembly Station

The devices that are connected to the LAN, as well as the IP addresses assigned to them, can be seen in Table 3-3. These IP addresses are fixed to avoid configuration problems and data traffic in the network. The addresses that refer to the components of the assembly station these addresses were assigned based on the availability in the laboratory where the electrical panel configuration tests were performed.

Table 3-3 Ethernet Network connected devices and IP Addresses

Device	IP Address
Server	149.201.219.203
Web Panel	149.201.219.242
PLC1	149.201.219.243
PLC2	149.201.219.245

Assembly Controller

Due to issues of component delivery, it was not possible to receive both controllers in time, limiting the complete assembly of the electrical components. However, one of the essential characteristics of PLC's is the excellent modularity they have regarding hardware management, that is, they are highly scalable by merely adding modules in the same Fieldbus to a maximum of 64 modules without the electrical conditions of all of them are affected. That is why it is easy to integrate all the modules of both PLC's with a single driver without complications. From the point of view, this integration is a bit more laborious since it is necessary to verify that the configurations of the controllers are similar or compatible with aspects such as addressing of variables and communication ports differ between PLC's. For this reason, it was decided to make a temporary installation of all modules connected to a single controller as shown in Figure 3-10.

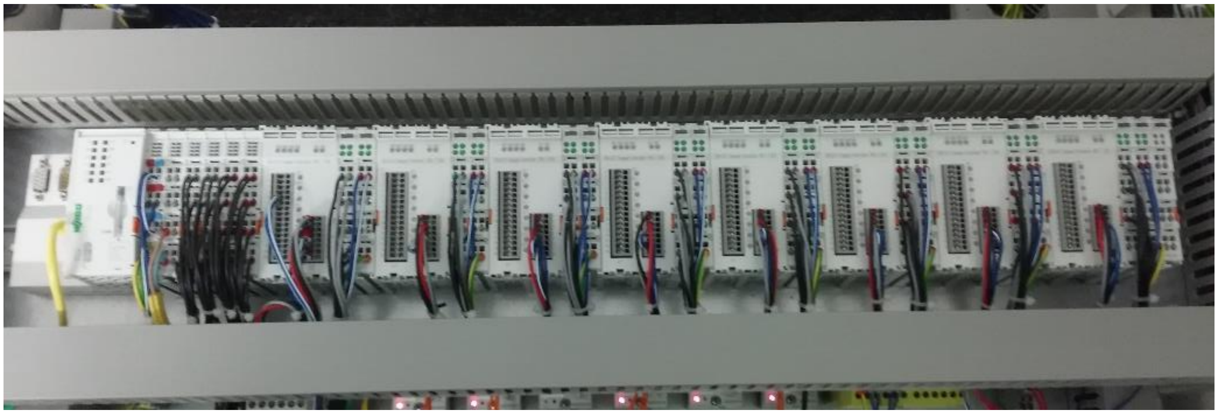


Figure 3-10 Assembly of PLC's Setup

Regarding the control of stepper motor drivers, it was observed that due to its internal configuration it was necessary to connect external signals to each of the modules to activate or deactivate functions or operating modes for the motors. For this, some of the signals from the digital output modules had to be assigned. In this way, the signals required by the module are activated via the PLC program. It would be of great benefit to the performance of the system to be able to identify if it is possible to modify said flags or signals of the drivers using some value in their control registers. Allowing the system to have these free digital signals available for other purposes.

The modules for the cables were incorporated into the configuration; the corresponding wiring was made to leave said modules enabled once the cables were received together with the robots that were to be installed.

Supply Controller

The second PLC was not possible to obtain it at the time of the realization of the project, that is why all the modules were integrated into the PLC assembly as already mentioned. However, it is possible to incorporate this PLC in the future if it is received.

Safety Components

Taking the concept of the proposed security system, we used the SISTEMA® software (version 2.0.7) created by the Institute for Occupational Safety and Health of the German Social Accident Insurance (IFA). To assess the reliability of the security system based on the components that make it up, concerning the ISO 13849-1 standard.

For this evaluation we used the electronic components that generate some signal that is related to the safety circuit, these components are those described in the concept section.

Once the electrical components were introduced, it was necessary to enter the category (based on the standard EN 954-1) to which they belong, in Figure 3-11 the models used, their level of performance and their classification according to the standard.

Status	Name	...	PL	PL-Software	PFHD [1/h]	...	...	...	Category	Requirements of the category
✔ SB	EStop Button 1		c	n.a.	1.7E-6	...	...	...	1	fulfilled
✔ SB	EStop Button 2		c	n.a.	1.7E-6	...	...	...	1	fulfilled
✔ SB	EStop Button 3		c	n.a.	1.7E-6	...	...	...	1	fulfilled
✔ SB	DoorLock 1		c	n.a.	1.7E-6	...	...	...	1	fulfilled
✔ SB	DoorLock 2		c	n.a.	1.7E-6	...	...	...	1	fulfilled
✔ SB	Safety Supervisor		e	e	3.2E-8	...	...	...	*not relevant*	*not relevant*
✔ SB	Safety Relay		e	n.a.	3.2E-8	...	...	...	*not relevant*	*not relevant*

Figure 3-11 Safety Devices and their Safety Specifications

Once all the components with the necessary data have been defined, the software tells us the maximum Performance Level (PL) that the system can have, as well as the PFHD probability that the system will fail. In Figure 3-12 is shown the maximum performance level of protection of the system.

Figure 3-12 Definition of the Performance Level of the Safety System

This level of performance is for low-risk conditions due to the type of sensor elements that were used, since they are standard components and fall into a category 1. On the other hand, the safety supervisor and the safety relay, are certified devices for Safety systems allowing the level of performance to rise.

While these results are acceptable, these characteristics and performance level satisfy the requirements of the components found inside the assembly station and do not consider the safety requirements that the second robot demands once it is installed.

For the implementation of the safety system in the assembly station, a Wago® brand module connected to the PLC that functions as a master of the AS-i protocol was used. The configuration of this module was made using the software of the same brand Wago® I/O Check®, accessing the configuration of the module as shown in Figure 3-13.

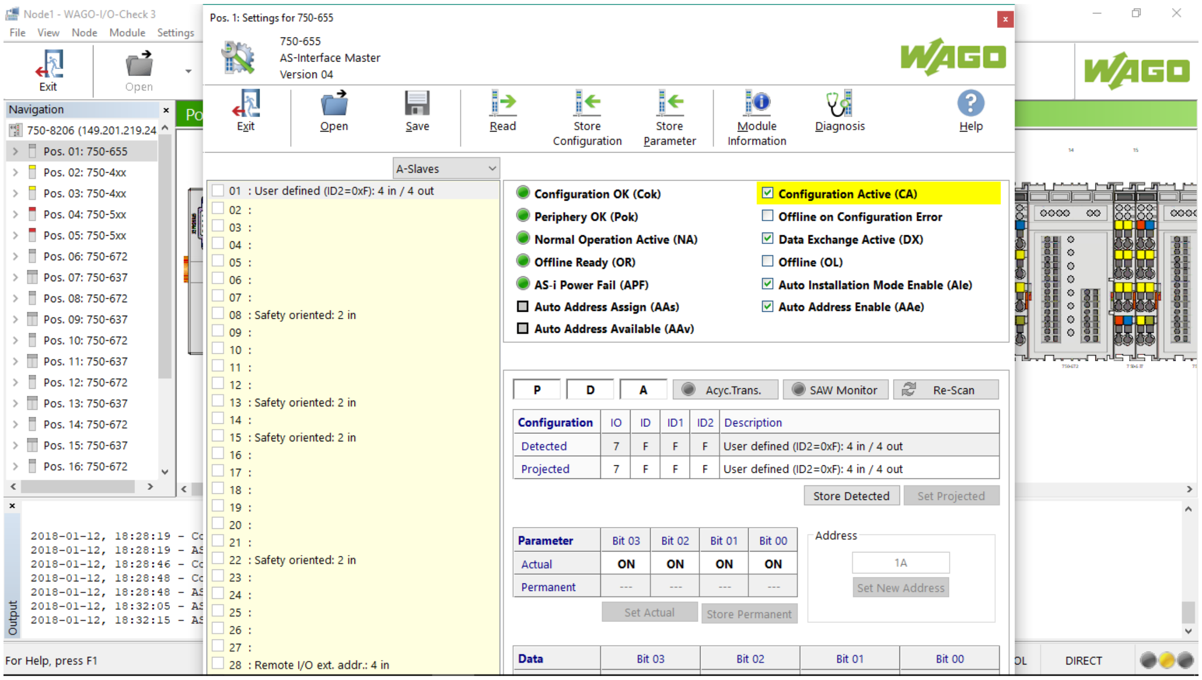


Figure 3-13 Configuration of the AS-i Master with Wago® I/O Check®

It should be mentioned that, during the start-up of the master module, it is necessary to make a short circuit with a normally open button between terminals M1 and M2, as shown in Figure 3-14. To put the module in automatic detection mode of slave devices that are active in the network. This signal must be sent to the module in a single display when a modification to the PLC program is made, or in case any extension or modification is made to the number of devices that are connected to the network.

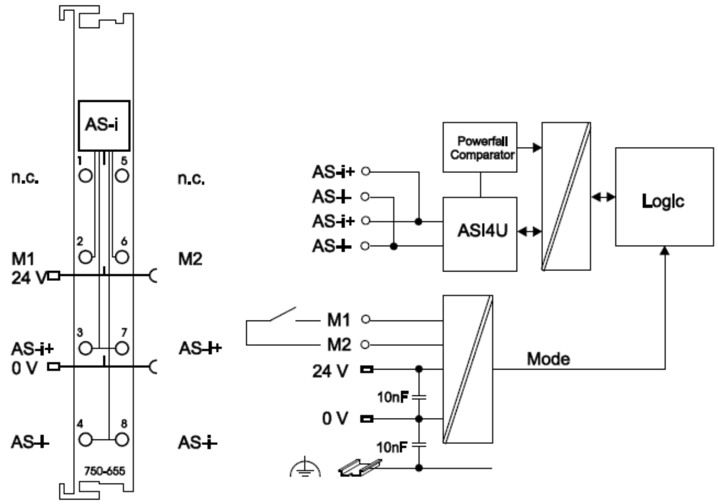


Figure 3-14 Schematic Diagram of AS-i Master Module[49]

Although with this tool it is possible to direct or commission slaves that are mounted on the network, the slaves that were used were not configured, because they came with an already defined address and to change this address an instrument of the same brand is required. It is not available in the laboratory. However, these previously assigned addresses do not affect the performance of the devices or their integration into the network designed for the machine. In Table 3-4 all slaves integrated into the safety net are listed.

Table 3-4 Safety Slaves and Addresses

Address	Device	Model	Input/Output Signals
01	Safety Supervisor	3RK1205-0BQ00-0AA3	Safety Relay
08	Safety Slave		Emergency Stop 1
13	Safety Slave		Emergency Stop 2
15	Safety Slave		Emergency Stop 3
22	Safety Slave		Door Lock 1
			Door Lock 2
28	Standard Slave	3RK1205-0CQ00-0AA3	Reset Button

During the integration of the security system, there was a need to add more slaves to the network that had been considered, due to the connection required by said slaves when safety devices such as emergency stop buttons or safety switches are connected. Door, since for this type of devices it is necessary to have redundant signals, therefore, for each emergency stop button or door switch, a dedicated slave is necessary.

The safety supervisor or EDM used is from Siemens®, model 3RK1105-1BE04-2CA0, which has two outputs that can be configured as independent or simultaneous activation. In the case of the machine in question, only one of the outputs was used to activate or deactivate a safety relay.

The safety relay that was implemented from PILZ®, model 20005-3FR-07 of the PNOZ X10.1 line and has the following characteristics:

- Relay outputs: 6 safety contacts (N/O) and 4 auxiliary contacts (N/C), positive-guided.
- Connections for emergency stop button, safety gate limit switch and reset button.
- Power indicator
- Status Indicators.
- Feedback control loop for monitoring external contactors/relays.

From all this, we can see the complete safety installation in Figure 3-15.

Figure 3-15 Safety Components Mounted in the Electrical Panel

Software Implementation

Main Logic

All the control logic was developed in the IDE platform of the same PLC brand -Wago® e! Cockpit®-, although these controllers can be programmed from the IDE of CoDeSys® itself since both brands work under the ICE 61133 standard. IDE own brand presents a workspace very similar to the software interfaces developed by Microsoft; this makes the first approach with the tool flow more naturally. However, it still has all the features and power of compilation offered by the CoDeSys® IDE.

The language used was structured text that is validated by the IEC 61133-3 standard, it also allows to encode complex expressions with conditional functions (IF-ELSE, FOR, and so forth). It is also quite intuitive for programmers with prior knowledge of C or C ++ and being a high-level language has a full clarity while programming, that facilitates the identification of errors. In addition to this, more and more companies are beginning to integrate this language into their development platforms.

The primary state machine that defines the behavior of the assembly station is running in the main routine of the program in the PLC. Figure 3-16 shows a capture of the code that is related to this primary state machine.

```
CASE iRobotState OF
    eRobotState.Initialize:
        iOperationMode:= eOperationMode.IDLE;
        iLEDState := eLEDState.Standby;
        /// Execute Initialization routines
        IF xSafetyOk THEN [1 lines]
        END_IF
    eRobotState.HomingRobot:
        // Execute Homing Robot
        IF xHomingRobotDone THEN [1 lines]
        END_IF
        IF NOT xSafetyOk THEN [1 lines]
        END_IF
    eRobotState.IDLE:
        IF xServerConnected AND NOT xServerDone THEN [1 li
        END_IF
        xServerDone := FALSE;
        IF xManualMode THEN [1 lines]
        ELSIF xAssemblyMode THEN [1 lines]
        END_IF
    eRobotState.Communication:
        // Execute HTTP Client
        xServerDone := TRUE;
        iRobotState := eRobotState.IDLE;
    eRobotState.Manual:
        iOperationMode := eOperationMode.Manual;
        // Execute Manual Mode
        IF NOT xManualMode THEN [1 lines]
        END_IF
    eRobotState.Assembly:
        iOperationMode := eOperationMode.Assembly;
        iLEDState := eLEDState.AssemblyOn;
        // Execute Assembly Mode
        IF xAssemblyDone THEN [1 lines]
        END_IF
        IF NOT xSafetyOk THEN [1 lines]
        END_IF
    eRobotState.Error :
        iLEDState := eLEDState.Error;
        IF xAssemblyBusy THEN [1 lines]
        END_IF
        xManualMode := FALSE;
        xAssemblyMode := FALSE;
        IF xResetMachine THEN [1 lines]
        END_IF
END_CASE
```

Figure 3-16 Main State Machine Code Capture

In the main routine, in addition to executing the general state machine of the logic, as mentioned in the concept section, it has other functions that are being executed at the beginning of each execution cycle; these functions, among other things, assign initial values or establish the preconfigured configuration for some variables.

In this block, some variables are also declared; whose intention is to function as a link between the values that return some function blocks and will be used as an input variable for another function block. In the Figure 3-17, the class diagram for the main function block is observed.

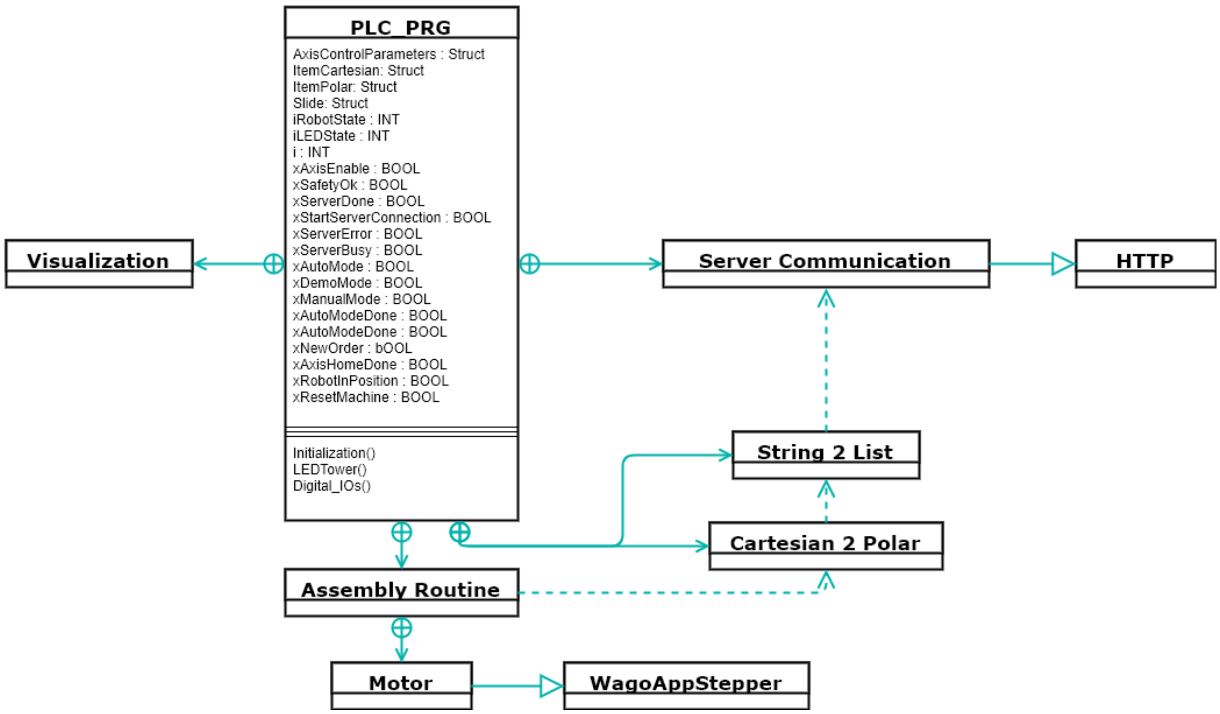


Figure 3-17 Main Routine Class Diagram

Within the functions that are executed within the initial cycle is the initialization function block, which is intended to prepare the initial values and declare variables that will be used by other function blocks, such is the case of the configuration parameters for each of the axes that must be controlled. Other parameters that are declared in the initialization function are the coordinates of each of the trays of the slides. Some of the variables that are modified in this function block related to the Axis to be controlled are described in Table 3-5.

Table 3-5 typAxisSettings Structure Components

Variable Name	Type	Description
Struct typAxisSettings		
iFreqDiv	INT	Frequency Prescaller for the stepper motor
iAccDiv	INT	Acceleration Prescaller for the stepper motor
dwSetupSpeed	DWORD	Predefined value for Axis speed
dwSetupAccel	DWORD	Predefined value for Axis Acceleration
byReferenceMode	BYTE	
byReferenceOffset	BYTE	
dwAccelStopFast	DWORD	Predefined value for Axis Fast Deceleration
iAccelMode	INT	Value for the Acceleration curve selector
dwAccelRampUp	DWORD	
dwAccelRampDown	DWORD	
iTargetPosition	INT	Target position for the stepper motor

Another function that is executed is one of digital inputs and outputs; it is this function block that instantiates each one of the addresses of these modules. This function is intended to be an interface between the physical signals and the different variables that can modify their value.

Another function that runs in the primary function cycle is the *LEDTower* function which defines the color configuration for the signal tower that is installed at the top of the assembly station. This color configuration is based on the different states in which the machine can be found while it is in operation, and the Table 3-6 describes in more detail the list of color combinations that were defined for the signal tower. Not all the cases on the list were implemented, but they are defined if they were to be required or expanded as necessary.

Table 3-6 Table of States for the LED tower

State	Green	Yellow	Red	Meaning
0	On	Off	Off	Machine in standby
1	Flash	Off	Off	Assembly Process running
2	On	On	Off	No Server Communication
3	Flash	Flash	Off	Spare
10	Off	On	Off	Supply Process running
11	Off	Flash	Off	Spare
12	Off	On	On	Error
13	Off	Flash	Flash	Warning
20	Off	Off	On	Emergency Stop
21	Off	Off	Flash	Spare
22	On	Off	On	Spare
23	Flash	Off	Flash	Spare
30	Off	Off	Off	Machine Off
31	On	On	On	Spare
32	Flash	Flash	Flash	Spare

Communication

For communication with the server, the CoDeSys® library called *HttpClient* was used, which is available for use in the Wago® e!Cockpit® IDE. This library includes the predefined HTTP methods in a single function block.

With this library, a Function Block was created that defines two function blocks of the library which execute the *Get* and *Put* methods respectively. Figure 3-18 shows the class diagram of this function implemented in the PLC program.

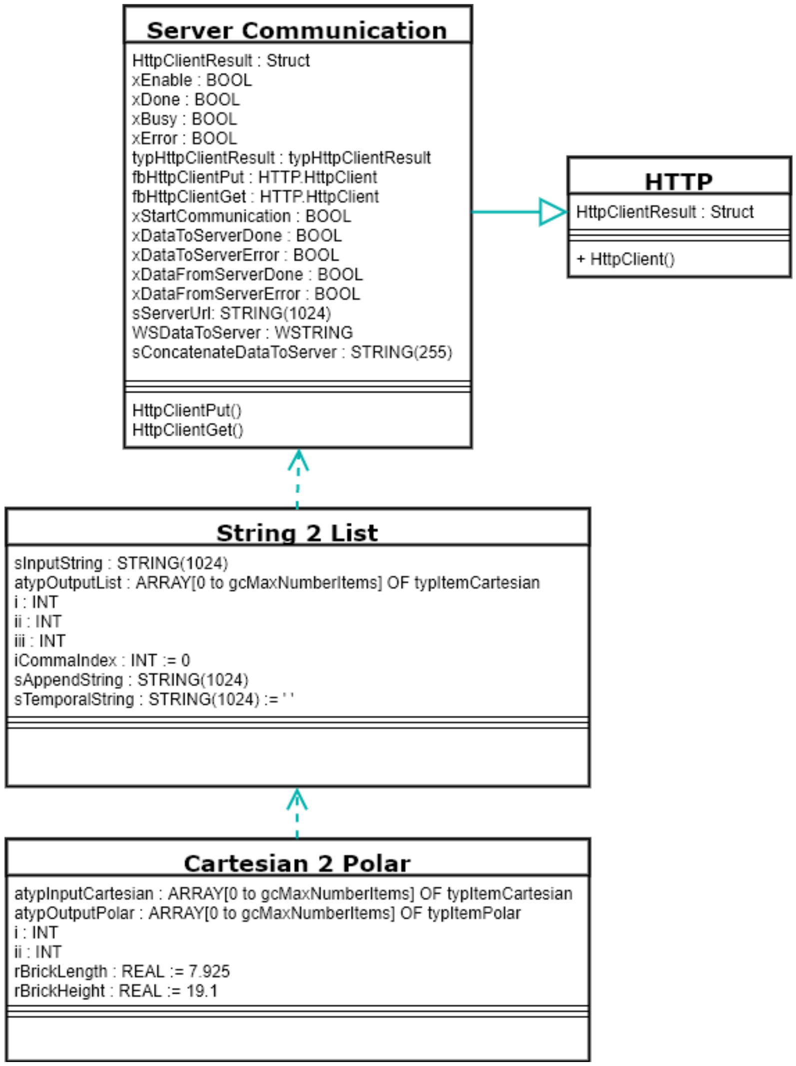


Figure 3-18 Server Communication Class Diagram

The implementation of this class is shown in Figure 3-19, the code of this figure is related to the flowchart explained in the concept section.

```
sServerUrl := CONCAT(GLOBALCONS.gcsServerUrlHeader1,GLOBALVAR.sServerIpAdress);
sServerUrl := CONCAT(sServerUrl, GLOBALCONS.gcsServerUrlHeader2);
sConcatenateDataToServer := CONCAT(GLOBALCONS.gcsDataToServerHeader1, GLOBALVAR.sMachineStatus);
sConcatenateDataToServer := CONCAT(sConcatenateDataToServer, GLOBALCONS.gcsDataToServerHeader2);
wsDataToServer := STRING_TO_WSTRING(sConcatenateDataToServer);

fbHttpClientPut ( sUrl := sServerUrl, eRequestType := HTTP.REQUEST_TYPE.PUT, eContentType := HTTP.CONTENT_TYPE.APPLICATION_JSON,
                pwsPostValue := ADR(wsDataToServer), udiTimeOut := GLOBALVAR.udiCommunicationTimeOut, xExecute := xEnable,
                xDone => xDataToServerDone, xError => xDataToServerError);
fbHttpClientGet ( sUrl := sServerUrl, eRequestType := HTTP.REQUEST_TYPE.GET, eContentType := HTTP.CONTENT_TYPE.APPLICATION_JSON,
                udiTimeOut := GLOBALVAR.udiCommunicationTimeOut, xExecute := xEnable,
                xDone => xDataFromServerDone, xError => xDataFromServerError);
typHttpClientResult.diContentLengthPut := fbHttpClientPut.httpResult.diContentLength;
typHttpClientResult.diContentLengthGet := fbHttpClientGet.httpResult.diContentLength;
typHttpClientResult.sContentPut := fbHttpClientPut.httpResult.sContent;
typHttpClientResult.sContentGet := fbHttpclientGet.httpResult.sContent;
IF xDataToServerDone AND xDataFromServerDone THEN
    xDone := TRUE;
    xBusy := FALSE;
ELSE
    xDone := FALSE;
    xBusy := TRUE;
END_IF
IF xDataToServerError OR xDataFromServerError THEN
    xError := TRUE;
ELSE
    xError := FALSE;
END_IF
```

Figure 3-19 Server Communication Code Capture

Due to the format of the message that is received from the server and the structure of the information contained within it, it is convenient to implement a function block as an interface between the communication routine and the rest of the program routines. A suitable form in which the information of the message can be used is in the form of a list, in this way it is convenient to assign a list of steps that the robot must follow to assemble the product.

In this function, a string variable was taken as an input value, and each type of data that the message contains is extracted. In Figure 3-20 can be seen the flow chart of the interface that performs to process this variable and return a data table. Afterwards, in Figure 3-21 is shown a capture of the code where this flowchart is implemented.

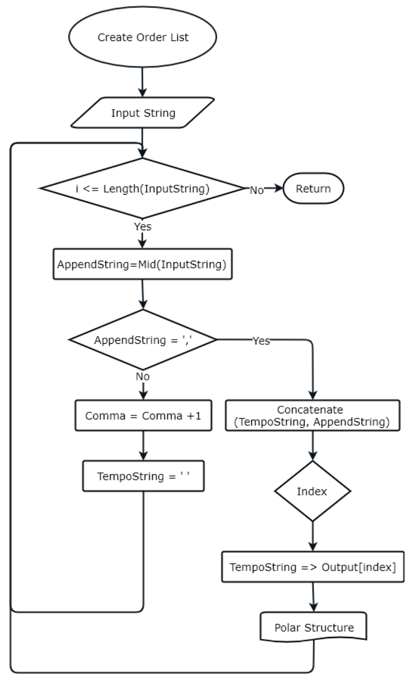


Figure 3-20 Order Message Conversion Flowchart

```
FUNCTION_BLOCK FB_String2ListOrder
VAR_INPUT
    sInputString : STRING(1024);            // Message to convert into a List
END_VAR
VAR_OUTPUT
    atypOutputList : ARRAY [0..GLOBALCONS.gciMaxNumberItems] OF typItemCartesian; // List of Bricks from Order Received
END_VAR
VAR
    i : INT;
    ii : INT;
    iii : INT;
    iCommaIndex : INT := 0;
    sAppendString : STRING(1024);
    sTemporalstring : STRING(1024) := '';

sTemporalString := '';

FOR i := 0 TO LEN(STR := sInputString) DO
    sAppendString := MID(STR := sInputString, LEN := 1, POS := i);
    IF NOT (sAppendString = '[' OR sAppendString = '"') THEN
        IF sAppendString = ',' THEN
            iCommaIndex := iCommaIndex + 1;
            sTemporalString := '';
        ELSE
            sTemporalString := CONCAT(sTemporalString, sAppendString);
            CASE iCommaIndex OF
                0: atypOutputList[ii].iItemId := STRING_TO_INT (sTemporalString);
                1: atypOutputList[ii].sItemName := sTemporalString;
                2: atypOutputList[ii].rCoordX := STRING_TO_REAL (sTemporalString);
                3: atypOutputList[ii].rCoordY := STRING_TO_REAL (sTemporalString);
                4: atypOutputList[ii].rCoordZ := STRING_TO_REAL (sTemporalString);
                5: atypOutputList[ii].rPitch := STRING_TO_REAL (sTemporalString);
                6: iCommaIndex := 0; ii := ii + 1;
            END_CASE
        END_IF
    END_IF
END_FOR
```

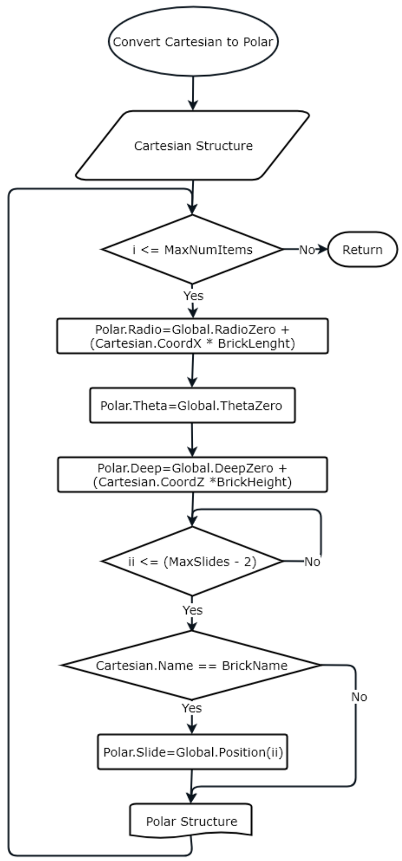
Figure 3-21 Message Conversion Code Capture

The list generated by this function is composed of data of structure type, which is described in Table 3-7. It is important to mention that due to the topology of the assembly robot the pieces whose *Pitch value is greater than 0*, cannot be assembled in this station. This information must be considered by the ERP system for the assignment of tasks to this plant. However, within this interface, this verification should be implemented and inform the server that there is an error regarding the specifications of the order received.

Table 3-7 typItemCartesian Structure Components

Variable Name	**Size**	**Description**
Struct typItemCartesian		
iItemId	INT	Item type identifier number
sItemName	STRING	LEGO's Official Serial Number of Brick
rCoordX	REAL	Car's coordinate frame X position
rCoordY	REAL	Car's coordinate frame Y position
rCoordZ	REAL	Car's coordinate frame Z position
rPitch	REAL	Rotation in the Z axis

After obtaining the list of elements that make up an order, it is necessary to convert the coordinate system because the system used by the server is of the Cartesian type and currently the system that is used by the robot is of the Polar type. For this, another interface was implemented whose task is to transform the coordinate systems of the pieces within the assembly. Figure 3-22 shows the flow diagram and the implemented code of this interface.




```
FUNCTION_BLOCK FB_Cartesian2PolarOrder
VAR_INPUT
    atypInputCartesian : ARRAY [0..GLOBALCONS.gciMaxNumberItems] OF typItemCartesian;
END_VAR
VAR_OUTPUT
    atypOutputPolar : ARRAY [0..GLOBALCONS.gciMaxNumberItems] OF typItemPolar;
END_VAR
VAR
    i : INT;
    ii : INT;
END_VAR
VAR CONSTANT
    rBrickLength : REAL := 7.925; // 1/4 of leght of a 2x2 brick in mm
    rBrickHeight : REAL := 19.1; // height of a brick in mm
END_VAR

FOR i := 0 TO  GLOBALCONS.gciMaxNumberItems DO
    atypOutputPolar[i].rCoordRadio := GLOBALCONS.gcrCoordRadioZero + (atypInputCartesian[i].rCoordX * rBrickLength);
    atypOutputPolar[i].rCoordTetha := GLOBALCONS.gcrCoordTethaZero;
    atypOutputPolar[i].rCoordDeep := GLOBALCONS.gcrCoordDeepZero +(atypInputCartesian[i].rCoordZ * rBrickHeight);
    FOR ii := 0 TO (GLOBALCONS.gciMaxSlides - 2) DO
        IF atypInputCartesian[i].sItemName = GLOBALVAR.atypPositions[ii].sBrickName THEN
            atypOutputPolar[i].typSlide := GLOBALVAR.atypPositions[ii];
        END_IF
    END_FOR
END_FOR
```

Figure 3-22 Cartesian to Polar Flowchart and Conversion Code Capture

For this function block, the input parameter is a list of Cartesian structures and is converted into another list of the same length whose elements are of the structure type as described in Table 3-8. This list contains the form of appropriate data so that its elements are processed by the assembly routine.

Table 3-8 typItemPolar Structure Components

Variable Name	**Size**	**Description**
Struct typItemPolar		
rCoordTetha	REAL	Global Polar Coordinate Frame Axis 1 in degrees
rCoordRadio	REAL	Global Polar Coordinate Frame Axis 2 in mm
rCoordDeep	REAL	Global Polar Coordinate Frame Axis 3 in mm
typSlide	typSlide	Designated Slide for this Brick

Once the items list is converted to polar coordinates, to each element will be assigned a new variable which is called *typSlide*. This variable is related to the slide where this kind of brick is stored. This structure is shown more detailed in Table 3-9.

Table 3-9 typSlide Structure Components

Variable Name	**Type**	**Description**
Struct typSlide		
iSlideId	INT	Number of Slide from 0 to 23
sBrickName	STRING	Official LEGO's serial number
iBrickMax	INT	Max number of Brick in the Slide
iBrickActual	INT	Actual amount of Bricks remaining on Slide
rCoordTetha	REAL	Global Polar Coordinate Frame Axis 1 in degrees
rCoordRadio	REAL	Global Polar Coordinate Frame Axis 2 in mm
rCoordDeep	REAL	Global Polar Coordinate Frame Axis 3 in mm

Assembly

As mentioned in previous chapters, the assembly routine takes as input parameters the list of elements coming from the server after having been converted to the machine's coordinate system, after that the respective coordinates of the slides corresponding to the type of brick in turn. With this information, the assembly function block will mainly modify the position and speed values for each of the robot axes. Figure 3-23 shows the class diagram related to this function block.

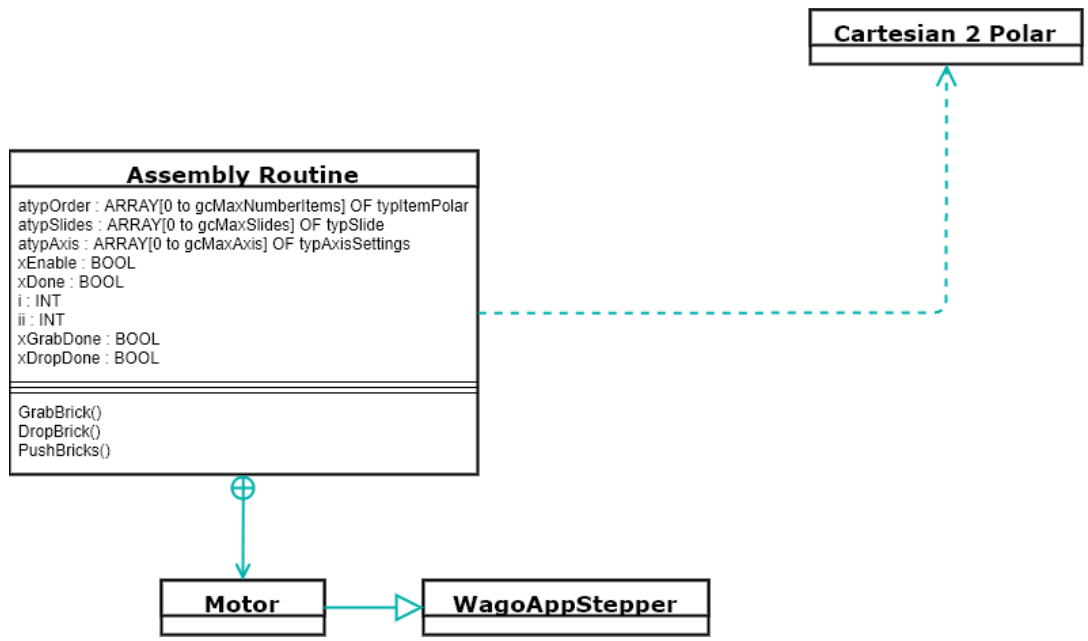


Figure 3-23 Assembly Class Diagram

This function block also calls other functions within the cycle of its execution cycle. These functions are *GrabBrick* and *DropBrick* which activate or deactivate the end effector of the robot depending on the position in which the robot is located.

Another relevant section of the assembly routine is the *PushBricks* function, which is executed after taking the piece from the material output tray. This function indicates to the robot to make a movement with which it pushes the bricks that are still stored in the slide, thus preventing the bricks from getting stuck in the slide.

In Figure 3-24 can be seen the code written for this function block of the assembly routine, in the upper part are displayed the public and private variables that are necessary to execute the task, in the lower part is to be written the logic of this function.

```
IF xEnable THEN
    FOR i := 0 TO GLOBALCONS.gciMaxNumberItems DO
        atypAxis[0].wAxisTargetPostion := REAL_TO_WORD(atypOrder[i].typSlide.rCoordTetha);
        atypAxis[1].wAxisTargetPostion := REAL_TO_WORD(atypOrder[i].typSlide.rCoordRadio);
        atypAxis[2].wAxisTargetPostion := REAL_TO_WORD(atypOrder[i].typSlide.rCoordDeep);
        // Move Robot
        xGrabDone := FU_GrabBrick();
        IF xGrabDone THEN
            atypOrder[i].typSlide.iBrickActual := atypOrder[i].typSlide.iBrickActual - 1;
            IF atypOrder[i].typSlide.iBrickActual <= 3 THEN
                //fbPushSlide();
            END_IF
        END_IF
        atypAxis[i].wAxisTargetPostion := REAL_TO_WORD(atypOrder[i].rCoordTetha);
        atypAxis[i].wAxisTargetPostion := REAL_TO_WORD(atypOrder[i].rCoordRadio);
        atypAxis[i].wAxisTargetPostion := REAL_TO_WORD(atypOrder[i].rCoordDeep);
        // MoveRobot
        xDropDone := FU_DropBrick();
    END_FOR
    // fbPushCar();
    xDone := TRUE;
END_IF
```

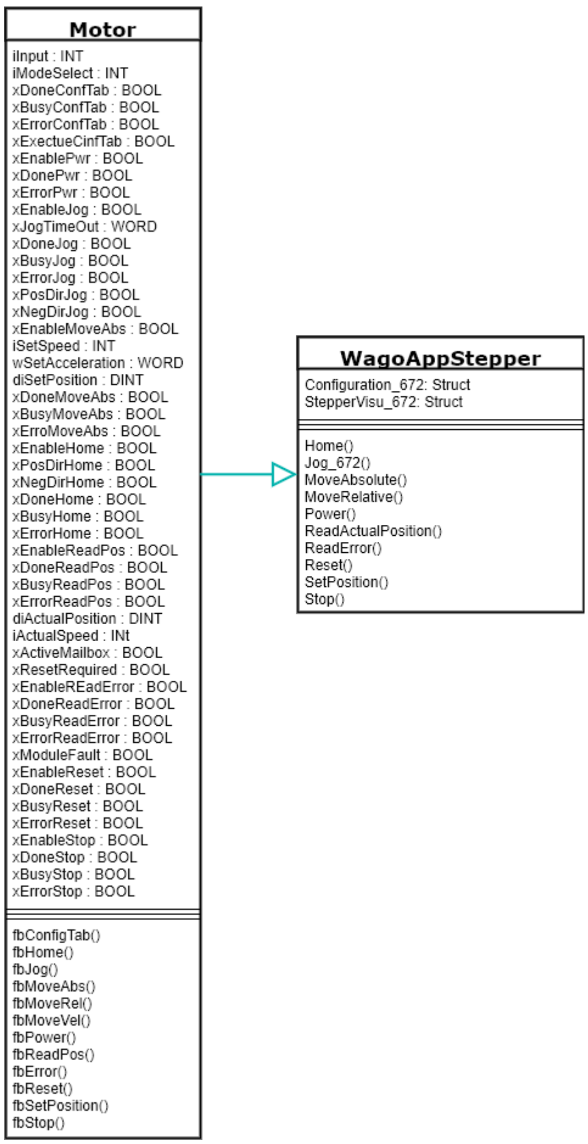
Figure 3-24 Code Capture of the Assembly Routine

Motion Control

For the motion control functions, Wago's library was used, which is an adaptation of the *Motion Control* library of OpenPLC to the different axis controllers that the brand offers for its PLC line. Among the tools included in this library are basic configuration blocks, which are intended for simple axis control applications that do not require a very sophisticated control sequence.

For the realization of module configuration tests and the start-up of an engine, these simple functions were used, to understand the behavior of these modules. From these tests performed to these functions, it was obtained that the speed of response of the engine has a considerable delay when a change is made in some parameter of the engine. For that reason, it was decided to use directly the predefined methods included in the library, which agree with those defined directly by the standard and not the simplified versions that Wago offers in the library.

In such a way that the implementation of these methods was done through a function block called *Motor*. The class diagram and the code of this function block can be seen in Figure 3-25.




```
CASE iInput OF
    eMotorState.Configuration: // Set Config Table
        IF xDoneConfTab THEN
            xExecuteConfTab := FALSE;
            iInput := eMotorState.Enable;
        ELSE
            xExecuteConfTab := TRUE;
        END_IF
        IF xErrorConfTab THEN
            xExecuteConfTab := FALSE;
            iInput := -1;
        END_IF
    eMotorState.Enable: // Power Axis
        IF xDonePwr THEN
            iInput := eMotorState.Idle;
        ELSE
            xEnablePower := TRUE;
        END_IF
        IF xErrorPwr THEN
            xEnablePower := FALSE;
            iInput := eMotorState.ReadError;
        END_IF
    eMotorState.Idle: // Select Operation Mode
        IF xHoming THEN
            iInput := eMotorState.Homing;
        ELSIF xJog THEN
            iInput := eMotorState.Jogging;
        ELSIF xMoveAbs THEN// Move Abs
            iInput := eMotorState.MoveAblsolute;
        ELSIF xMoveRel THEN// Move Rel
            iInput := eMotorState.MoveRealtive;
        END_IF
    eMotorState.Homing: // Set Homing
        IF xDoneHome THEN
            xEnableHome := FALSE;
            iInput := eMotorState.Stop;
        ELSE
            xEnableHome := TRUE;
        END_IF
        IF xErrorHome THEN
            xEnableHome := FALSE;
            iInput := eMotorState.ReadError;
        END_IF
    eMotorState.Jogging: // Jog Enable
        IF xDoneJog THEN
            xEnableJog := FALSE;
```

Figure 3-25 Motion Control Class Diagram and Code Implementation

Although these simplified functions of the library were not suitable for the required application, they are an excellent empirical tool to obtain the configuration of suitable parameters for the correct functioning of the axis.

The most used tools were the visualizations that are included in the library, which are used to test the operation of the engine. The first visualization that must be executed is the configuration visualization who verifies if there is a connection with the desired controller. The second tool that is used to make tests with the axis was the control display for the stepper motor. This function establishes a manual operation mode to perform the corresponding tests with the respective axis. Figure 3-26 shows both visualizations that are used to configure and control an axis manually.

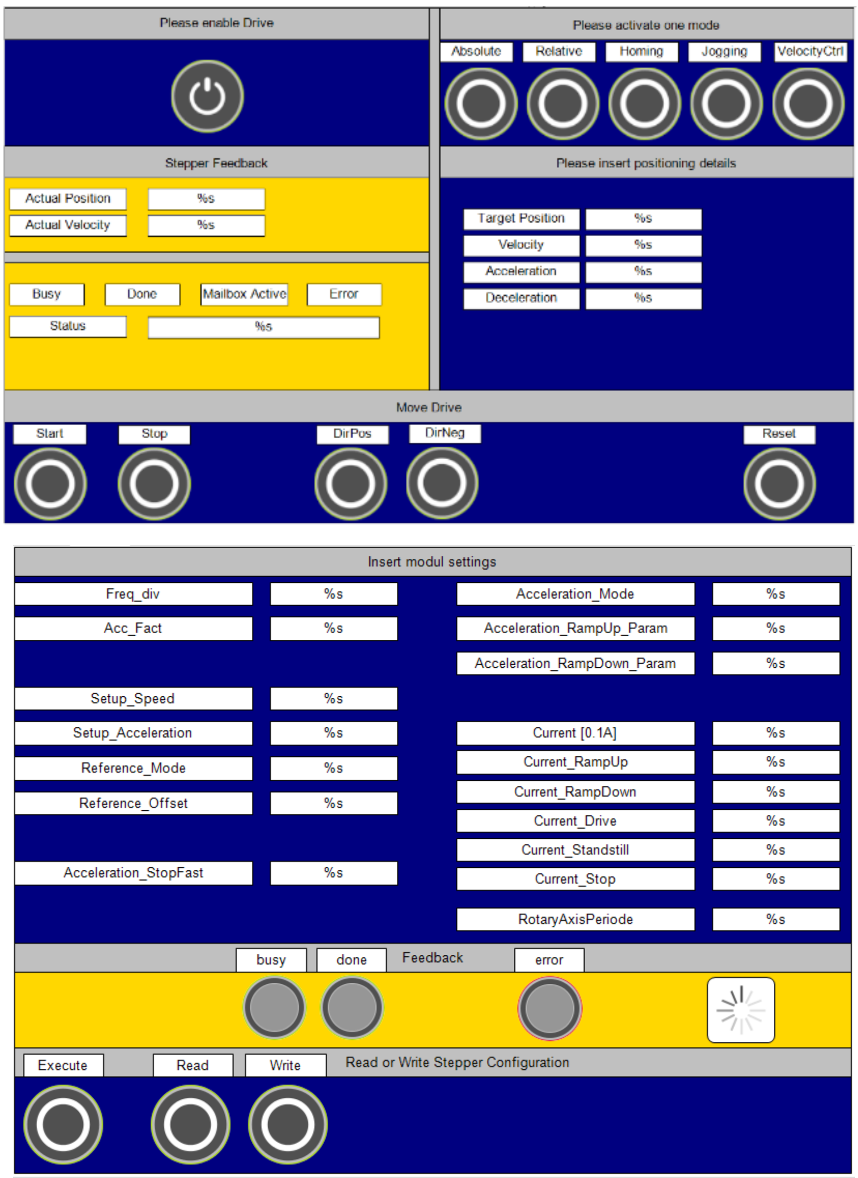


Figure 3-26 Stepper Control Visualization 672

These visualizations write the configuration parameters of the axis which are listed in Table 3-10.

Table 3-10 Parameters for Stepper Motor Driver

Parameter	Value
Acceleration Stop Fast	1000
Acceleration Ramp Up	10
Acceleration Ramp Down	10
Ramp Up Param	300
Ramp down Param	300
Acceleration Type	Constant
Deceleration Type	Constant
Frequency Div	200
Acceleration Factor	8
Motor Steps per Rev	200
Current	5 A
Current Ratio Stand Still	33%
Current Ratio Ramp Up	100%
Current Ratio Drive	50%
Current Ramp Down	90%
Current Ratio Stop	100%
Range Negative	-8388607
Range Positive	8388607
Axis Period	0
Speed	10
Speed Limit	25000
Drive Direction	Deactivated
Application Selector	Position Control
DC Link Voltage Min	15
DC Link Voltage Max	80
Homing / Jogging Setup Speed	100
Homing / Jogging acceleration	10
Reference Offset	0

These visualizations must be connected to the function blocks of the library through the program in the PLC. Therefore, in Figure 3-27 is shown a capture of the code that is needed to implement this visualization tools.

```
VAR
    visu : typStepperVisu_672;
    configuration_table : FbConfigurationTable;
    configuration_values : typConfiguration_672;
    control : FbStepperControlBasic;
    power : FbPower;
    move_absolute : FbMoveAbsolute;
    job: eMode;

x_enablestepper:=TRUE;
configuration_table( xExecute:=visu.executeCT, I_Port:=Steppercontroller, xRead := visu.readCT, xWriteFlash := visu.writeCT,
        pData := ADR(configuration_values),  xDone => visu.doneCT, xBusy => visu.busyCT, xError => visu.errorCT,
        oStatus => visu.statusCT, eErrorCodes => visu.BasicErrorCodes);

job := FuModeSelect (xMoveAbsoluteRequest:=visu.MoveAbs, xMoveRelativeRequest:= visu.MoveRel, xHomingRequest := visu.Homing,
        xJoggingRequest:=visu.Jog, xVelocityCtrlRequest:=visu.Velocity, xDrivePrgRequest:=FALSE, xTorqueEdge:= FALSE,
        xTorqueLevel:=FALSE, xSetPosition:= FALSE);

control( xEnable := visu.BasicEnable, I_Port:= Steppercontroller, eJobType:= job, xDirPos := visu.BasicPos,
        xDirNeg:=visu.BasicNeg, iSpeed:=visu.BasicSpeed, wAcceleration := visu.BasicAcceleration,
        wDeceleration := visu.BasicDeceleration, diPosition := visu.BasicRefPosition, xStop:=visu.BasicStop,
        xJobFinished => visu.BasicDone, xBusy => visu.BasicBusy, xError =>visu.BasicError, oStatus => visu.BasicStatus1,
        eErrorCodes => visu.BasicErrorCodes, diActualPosition => visu.BasicActualPosition,
        iActualSpeed => visu.BasicActualSpeed, xMailboxActive => visu.BasicMailboxActive,
        xResetWarning => visu.BasicResetWarning, xTriggerStart := visu.BasicStart, xTriggerErrorReset := visu.BasicReset);
```

Figure 3-27 Basic Stepper Motor Control Code Capture

Because the functions of the library are practical tools to perform the calibration of the desired axis, a dynamic table was developed to be able to estimate the predefined values of each axis concerning the values of its pre-scalers and the speed and acceleration parameters in physical units.This table is based on the equations described in the module configuration manual for stepper motor control, and they vary according to the model of each module. An example of this selector of pre-scaler values can be seen in Table 3-11. Said table takes the different configurations of suggested pre-scaler values and delivers the velocity and acceleration values.

Table 3-11 Selection of Prescallers for the Stepper Motor Driver

Presc. Freq Div	Accel Fact	Number Poles Pairs	Motor Speed	FM	FP	am	Internal accel	Set Point Vel.	Set Point Accel
4	8	2	1800	30	15360	200	102400	768	51200
20								3840	256000
80								15360	1024000
200								38400	2560000
4	80							768	5120
20								3840	25600
80								15360	102400
200								38400	256000
4	800							768	512
20								3840	2560
80								15360	10240
200								38400	25600
4	8000							768	51.2
20								3840	256
80								15360	1024
200								38400	2560

For further information related to the configuration of the stepper motor driver, please refer to the user manual of the module in the Appendix section.

Safety Logic

As mentioned before, the security devices that were used are Siemens. Therefore, the software used to define and configure the security logic is ASIMON V3 of the same brand.

The security supervisor connects to the computer through an RS-232 serial adapter; once communication is established and the software has recognized the device, it is necessary to locate the slaves that are active and indicate what type of configuration (standard or security).

Since the identification and configuration of the hardware that is going to be used have been done. The Stop task that is wanted to be executed must be configured; for this, we drag a Stop block to the workspace, where a window will later appear where they are configured the available outputs of the selected EDM.

After having configured the stoppage task, the safety elements or signals are added, i.e., emergency stop buttons, safety curtains, door switches, and so forth. For each type of element, there is a block that can be included in the workspace. In the case of this configuration, three emergency-stop buttons and two door switches were occupied. When any of these blocks are added to the workspace, the software automatically adds the block of the AND logical operator and makes automatic wiring between the blocks of the signals with the logical operator, in turn, makes the same wiring between the logical operator and the Stop block.

When a signal block is added, when double-clicked on it, a window appears in which must be assigned an address corresponding to one of the available slaves in the network.

After adding the safety signals, the elements that restore or activate the state of the output signals are added. Here only a reset button was added. Therefore, it was only necessary to add a reset button block; these blocks were they add to the workspace in the same way as the safety blocks or the stop blocks, and also making the auto-wiring with the block of the stoppage task.

The configuration of the security task in the software can be seen in Figure 3-28. On the left side of the figure, there is a list of the components that can be added to the workspace, on the right side of the image the connection is observed between all the function blocks required for the previously designed topology.

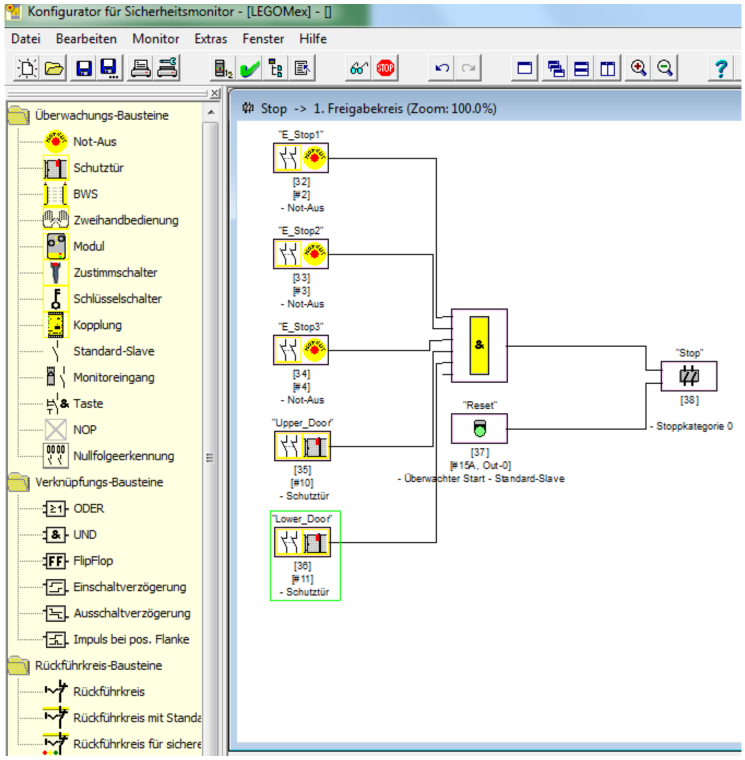


Figure 3-28 Configuration of Safety Task in ASIMON Software

When the configuration of the security task is downloaded to the EDM, the software has the functionality to communicate in online mode and observe the actual status of all the devices connected to the network, indicating with colors (green, yellow and red) each one of the blocks that are in operation. Figure 3-29 shows the on-line monitoring of the safety logic implemented when the system encounters all the emergency signals activated. This means that none of the emergency stop buttons has been pressed and that the doors they are closed, but the system is waiting for a reset signal (yellow) so that the stop task changes its status from off (red) to on (green).

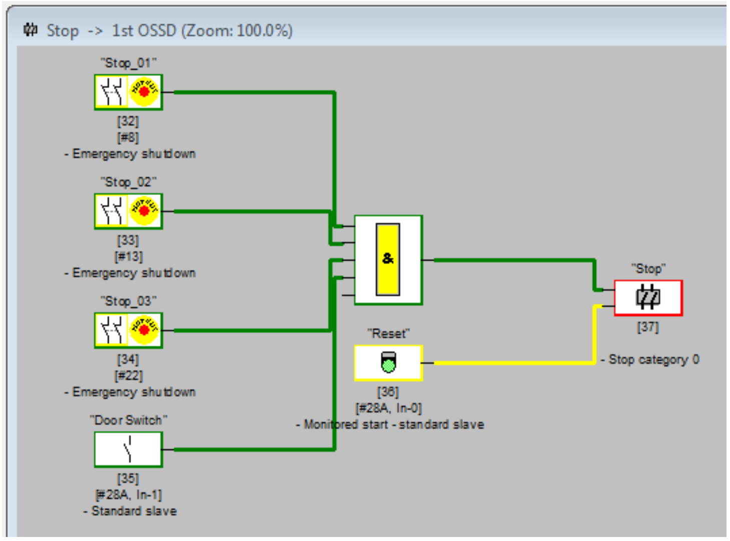


Figure 3-29 On-Line Monitoring of the Safety System Waiting for Reset Signal

In Figure 3-30, can be seen that the status of the whole system is green, this is because the security supervisor already received the *reset* signal. If any of the emergency stop buttons are pressed instantly, the route from that block to the stop block will change status to red.

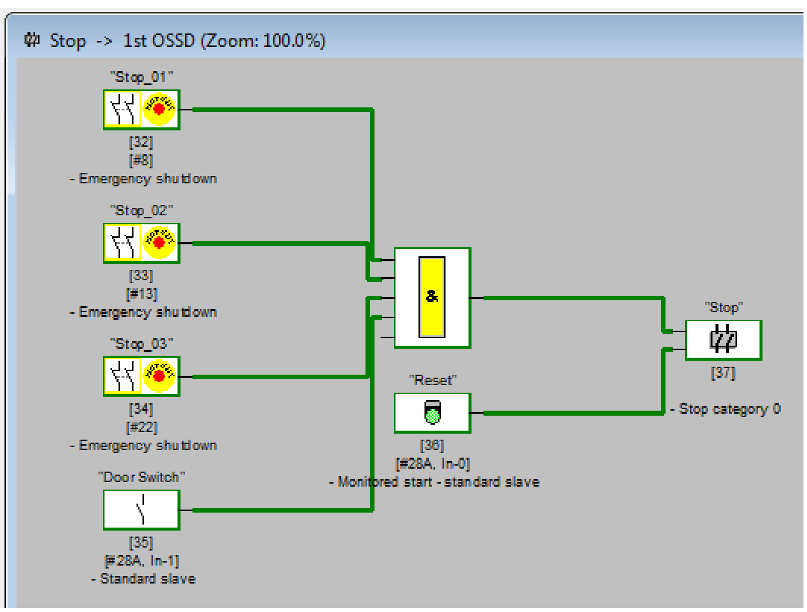
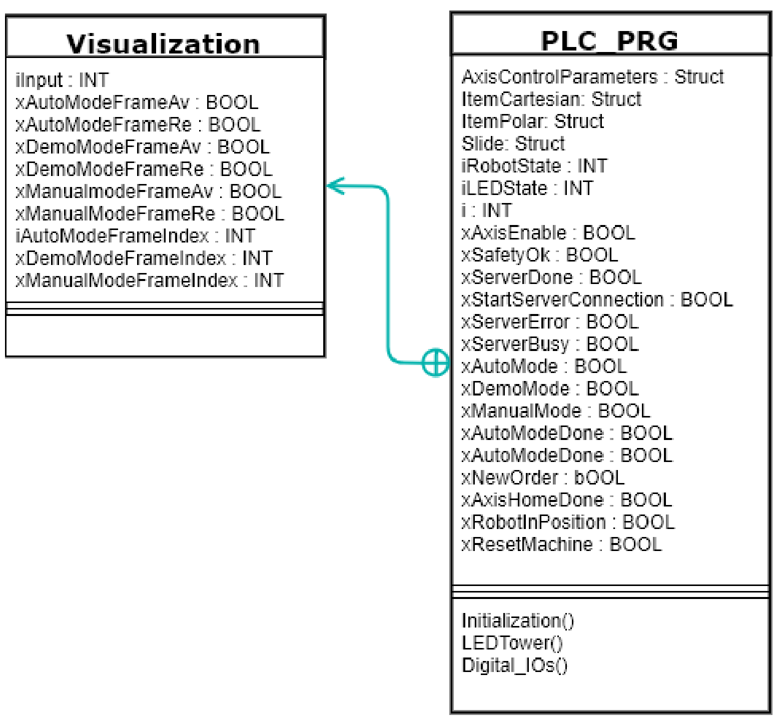


Figure 3-30 On-Line Monitoring of the Safety System Activated

Human-Machine Interface

For the visualization of the data of the PLC, the HMI was developed in the same IDE where the logic of the controller -e! Cockpit®- was developed. Because like CoDeSys®, it has a section for the development of screens where buttons and indicators can be integrated that show values of variables within the logic.

Once these screens are created in the IDE, they are compiled together with the logic of the PLC and downloaded to the internal memory of the controller. In such a way that it is possible to create a function block within the program which oversees managing and conditioning the variables that should be shown in the different screens that make up the web visualization. In Figure 3-31, the class diagram and the code corresponding to the function block of the visualization are observed. The primary purpose of this function block is to control the screens that must be displayed on the panel according to the state in which the machine is located.



```
CASE iInput OF
    eOperationMode.IDLE: (* Stand By *)                          // Display the Logos Window
    eOperationMode.Assembly: (* AutoMode     *)                      // Goes to Automatic Mode Window
        IF xAutoModeFrameAv THEN
            iAutoModeFrameIndex := iAutoModeFrameIndex + 1;
            IF iAutoModeFrameIndex > GLOBALCONS.gciMaxAutoFrameIndex THEN
                iAutoModeFrameIndex := 0;
            END_IF
            xAutoModeFrameAv := FALSE;
        ELSIF xAutoModeFrameRe THEN
            iAutoModeFrameIndex := iAutoModeFrameIndex - 1;
            IF iAutoModeFrameIndex < 0 THEN
                iAutoModeFrameIndex := GLOBALCONS.gciMaxAutoFrameIndex;
            END_IF
            xAutoModeFrameRe := FALSE;
        END_IF
        IF xDemoModeFrameAv THEN
            iDemoModeFrameIndex := iDemoModeFrameIndex + 1;
            IF iDemoModeFrameIndex > GLOBALCONS.gciMaxDemoFrameIndex THEN
                iDemoModeFrameIndex := 0;
            END_IF
            xDemoModeFrameAv := FALSE;
        ELSIF xDemoModeFrameRe THEN
            iDemoModeFrameIndex := iDemoModeFrameIndex - 1;
            IF iDemoModeFrameIndex < 0 THEN
                iDemoModeFrameIndex := GLOBALCONS.gciMaxDemoFrameIndex;
            END_IF
            xDemoModeFrameRe := FALSE;
        END_IF
    eOperationMode.Manual: (*   ManualMode   *)
        IF xManualModeFrameAv THEN
            iManualModeFrameIndex := iManualModeFrameIndex + 1;
            IF iManualModeFrameIndex > GLOBALCONS.gciMaxManualFrameIndex THEN
                iManualModeFrameIndex := 0;
            END_IF
            xManualModeFrameAv := FALSE;
        ELSIF xManualModeFrameRe THEN
            iManualModeFrameIndex := iManualModeFrameIndex - 1;
            IF iManualModeFrameIndex < 0 THEN
                iManualModeFrameIndex := GLOBALCONS.gciMaxManualFrameIndex;
            END_IF
            xManualModeFrameRe := FALSE;
        END_IF
END_CASE
```

Figure 3-31 Visualization Function Block Code Capture

All input values that the operator must enter are made in a tactile way, either a digital signal (buttons) or analog values, for the latter the operator must touch the value that you want to modify, and a pop-up keyboard will appear with which you can enter the desired value. Concerning the hierarchy of screens described above, the different screens that were implemented according to the diagram described are presented below. Starting from the main screen in Figure 3-32 and then other screens that shows the detail of the different operating modes of the machine in Figure 3-33.

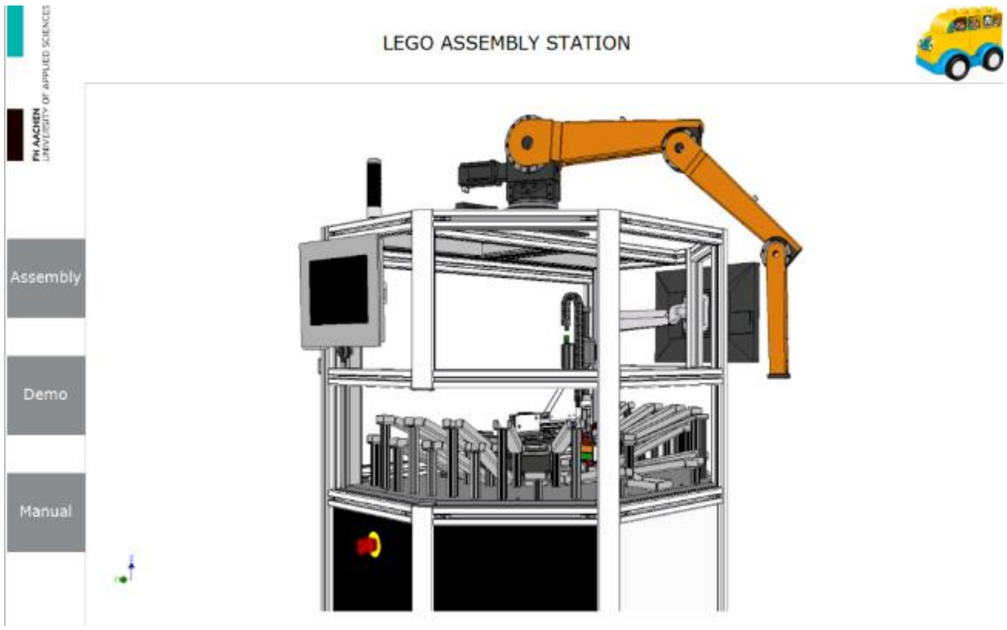


Figure 3-32 Main Page of the Web Visualization

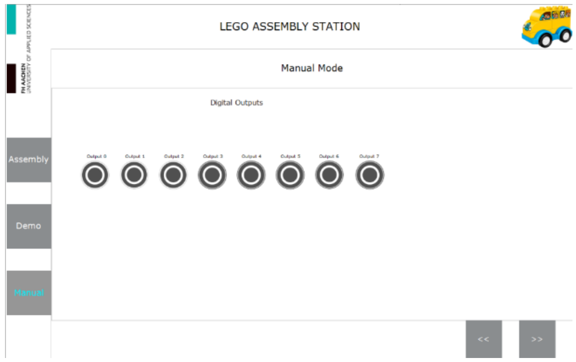


Figure 3-33 Subpages of the Web Visualization

Thanks to the panel having a proper resolution of 1280x800 pixels, it was possible to include good quality visual elements within the HMI design, which makes the interface comfortable for the user and the objective of the station and its operability is understandable. In Figure 3-34 is shown the main screen displayed on the panel.

Figure 3-34 HMI Running in the Web Panel

The style of the screens is intended to match with the ghapich aspects of the corporate image of the school; this is because when the machine is exposed matches the visual style. All this combined with the recommendations so that the interface is efficient, and its navigation is intuitive.

4. Conclusions

The constant evolution of production processes trying to satisfy the continuous demand of the market for highly personalized products is a concept that will be present for quite some time, that is why it is essential that engineering students develop skills and an understanding of this phenomenon. Therefore, the development of this type of projects in which the learning process is accelerated.

On the other hand, the fact that this type of project has a didactic orientation, allows the demonstration of the concepts of industry 4.0 and flexible assembly cells to people linked to the area of technology and are not yet familiar with the subject.

The implementation of LEGO® bricks provides a clear and practical representation of a general assembly process, which can then be scaled to any area of application in all branches of the industry.

The work presented here is a part of the development of an assembly station that contributes to the application of the fundamental concepts of Industry 4.0. The integration of brand components such as Wago® or IGUS® allows, in turn, to demonstrate that it is possible to develop automation projects with a broader range of options that are economical and satisfy the demands of the industry.

We could conclude that the use of different tools, such as SysML combined with a well-defined methodology such as VDI 2206, contribute significantly to defining the scope and route that an engineering project should follow.

Unfortunately, for this document the complete development of the machine was not possible to complete it, however, the detailed design of the different elements will facilitate later the integration of the missing components. Due to most of the developed parts are designed for continuous development and improvement of the system.

5. Future work

The control of the 5DOF robot is recommended to be transferred to an independent controller. In this way the computational effort that the 3DOF robot controller must perform will be diminished, making the maintenance of each robot program more accessible and more transparent in the future. While it is true that the computational effort of the controller can encompass both robots; in case of making extensions or significant modifications to the whole assembly station, these will be easily implemented thanks to this decentralization of the control of the robots.

One critical aspect that should be considered during the start-up of the 5DOF robot is the safety system around the workspace of himself. Due to the configuration of the assembly station, this workspace exceeds the edges of the structure that covers the other components; it should be remembered that its function is to supply the slides with new parts when necessary. That is why it is recommended to install security curtains in the areas of the workspace where the interaction with an operator could be presented. This implementation would directly affect only the safety logic, without the need to modify the program within the Controllers, in case that said safety system is electronic. It can also be a system of fences or screens that delimit the access of operators, if so, the implementation would be much more straightforward, especially for exhibitions and showrooms.

Another important aspect that should be covered in the future work of this project is the implementation of sensors that allow knowing the state of the pieces that are on the slides. Allowing to know if the quantity of pieces is enough or if any piece is stopped and cannot reach the lowest part of the slide. For this, several options can be implemented; from the implementation of digital sensors in each of the slides to the use of vision sensors that can be mounted at a position from which they can obtain the information of all the slides at the same weather. Of course, this implementation must consider the financial costs due to the mechanical modifications to be made in the structure to be able to mount said sensors, as well as the price of the sensor itself. On the other hand, it is essential to include the hardware that will be responsible for processing said signals, that means, the extra modules that should be added to the PLC or even, an independent device that processes the sensor information and of the local network is communicated to the central controller.

Although the implementation of individual sensors for each of the slides as a feedback system would result in a faster solution, It is recommended the implementation of a system of vision as feedback. Due to the quality and quantity of information provided by this type of systems are higher than that of a digital sensor system, despite the higher computational effort that this entails.

6. Bibliography

[1] S. Sriram, M. E. Kuhl, B. K. Thorn, and A. L. Carrano, “A novel work-sharing protocol for U-shaped assembly lines,” in *Proceedings of the Winter Simulation Conference 2014*, 2014, pp. 2113–2123.

[2] C. Zhao, J. Li, and N. Huang, “Efficient Algorithms for Analysis and Improvement of Flexible Manufacturing Systems,” *IEEE Trans. Autom. Sci. Eng.*, vol. 13, no. 1, pp. 105–121, Jan. 2016.

[3] Y. Cohen, M. Faccio, F. G. Galizia, C. Mora, and F. Pilati, “Assembly system configuration through Industry 4.0 principles: the expected change in the actual paradigms,” *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 14958–14963, Jul. 2017.

[4] K. K. Abd, *Intelligent Scheduling of Robotic Flexible Assembly Cells*. Cham: Springer International Publishing, 2016.

[5] Abd K, Abhary K, and Marian R, “A Scheduling Framework for Robotic Flexible Assembly Cells,” *Aijstpme*, vol. 4, no. 1, pp. 31–38, 2011.

[6] Y. Maeda, H. Kikuchi, H. Izawa, H. Ogawa, M. Sugi, and T. Arai, “‘Plug & Produce’ functions for an easily reconfigurable robotic assembly cell,” *Assem. Autom.*, vol. 27, no. 3, pp. 253–260, 2007.

[7] K. Zhou, T. Liu, and L. Zhou, “Industry 4.0: Towards future industrial opportunities and challenges,” *2015 12th Int. Conf. Fuzzy Syst. Knowl. Discov. FSKD 2015*, pp. 2147–2152, 2016.

[8] H. Meissner, R. Ilsen, and J. C. Aurich, “Analysis of Control Architectures in the Context of Industry 4.0,” *Procedia CIRP*, vol. 62, pp. 165–169, Jan. 2017.

[9] Y. Uygun and M. Ilie, “Autonomous Manufacturing-related Procurement in the Era of Industry 4.0,” in *Digitalisierung im Einkauf*, Wiesbaden: Springer Fachmedien Wiesbaden, 2018, pp. 81–97.

[10] Chi-Yi Tsai, Ching-Chang Wong, Chia-Jun Yu, Chih-Cheng Liu, and Tsung-Yen Liu, “A Hybrid Switched Reactive-Based Visual Servo Control of 5-DOF Robot Manipulators for Pick-and-Place Tasks,” *IEEE Syst. J.*, vol. 9, no. 1, pp. 119–130, Mar. 2015.

[11] G. Humbert, X. Brun, M.-T. Pham, M. Guillemot, and D. Noterman, “Development of a methodology to improve the performance of multi-robot pick & place applications: From simulation to experimentation,” in *2016 IEEE International Conference on Industrial Technology (ICIT)*, 2016, pp. 1960–1965.

[12] Wen-Bo Li, Guang-Zhong Cao, Xiao-Qin Guo, and Su-Dan Huang, “Development of a

4-DOF SCARA robot with 3R1P for pick-and-place tasks," in *2015 6th International Conference on Power Electronics Systems and Applications (PESA)*, 2015, pp. 1–5.

[13] I. Adept Technology, "AdeptOne Robot with SmartController User's Guide ade pt." Adept Technology Inc., 2005.

[14] J.-C. Latombe, *Robot Motion Planning*. Springer US, 1991.

[15] L. Wouters, S. Creff, E. E. Bella, and A. Koudri, "Collaborative systems engineering: Issues & challenges," in *2017 IEEE 21st International Conference on Computer Supported Cooperative Work in Design (CSCWD)*, 2017, pp. 486–491.

[16] eMarketer, "• Digital buyers worldwide 2021 | Statistic." [Online]. Available: https://www.statista.com/statistics/251666/number-of-digital-buyers-worldwide/. [Accessed: 03-Jan-2018].

[17] M. Naumann and M. Fechter, "Robots as enablers for changeability in assembly applications," in *15. Internationales Stuttgarter Symposium* , Springer Vieweg, Wiesbaden, 2015, pp. 1155–1171.

[18] P. David, "New Safety Certifications To Standardise Collaborative Robots," *Auto Tech Rev.*, vol. 5, no. 4, pp. 14–15, Apr. 2016.

[19] P. Barattini, C. Morand, and N. M. Robertson, "A proposed gesture set for the control of industrial collaborative robots," in *2012 IEEE RO-MAN: The 21st IEEE International Symposium on Robot and Human Interactive Communication*, 2012, pp. 132–137.

[20] L. Kong *et al.*, "AdaSharing: Adaptive Data Sharing in Collaborative Robots," *IEEE Trans. Ind. Electron.*, vol. 64, no. 12, pp. 9569–9579, Dec. 2017.

[21] "RoboCup Soccer." [Online]. Available: http://www.robocup2014.org/?p=893. [Accessed: 03-Jan-2018].

[22] "Rethinking Robotics." [Online]. Available: http://rethinkingrobotics.com. [Accessed: 03-Jan-2018].

[23] A. Campeau-Lecours and C. Gosselin, "An anticipative kinematic limitation avoidance algorithm for collaborative robots: Two-dimensional case," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016, pp. 4232–4237.

[24] W.-B. Lee, S.-D. Lee, and J.-B. Song, "Design of a 6-DOF collaborative robot arm with counterbalance mechanisms," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 3696–3701.

[25] F. Biennier, J. Favrel, and J. P. Denat, "Collaborative autonomous control," in

Advances in Production Management Systems, Boston, MA: Springer US, 1998, pp. 91–102.

[26] A. N. Panfir, A. Covaci, C.-C. Postelnicu, and G. Mogan, “Control Interfaces for a Collaborative System Using LabView Package,” in *DoCEIS 2012: Technological Innovation for Value Creation*, Springer, Berlin, Heidelberg, 2012, pp. 33–40.

[27] S. Sabanovic, “Socializing co-robots: Incorporating users and situated interaction in the design of collaborative robots,” in *2014 International Conference on Collaboration Technologies and Systems (CTS)*, 2014, pp. 113–114.

[28] PLCOpen, “Part 1 - Basics & Part 2 - Extensions.” [Online]. Available: http://www.plcopen.org/pages/tc2_motion_control/part_1_2/index.htm. [Accessed: 31-Jan-2018].

[29] The Lego Group, “The LEGO Group - A short presentation,” Denmark, p. 23, 2017.

[30] G. K. Christiansen, “Toy Building Brick,” US 3005282 A, 1961.

[31] Wikipedia, “2_duplo_lego_bricks.jpg (900×900).” [Online]. Available: https://upload.wikimedia.org/wikipedia/commons/0/0f/2_duplo_lego_bricks.jpg. [Accessed: 28-Dec-2017].

[32] “LEGO® DUPLO® Product Safety - Articles - DUPLO LEGO.com.” [Online]. Available: https://www.lego.com/en-us/duplo/articles/safety-dd0dcb8e9e124807872a4b5d4e44ebc9?ignorereferer=true. [Accessed: 28-Dec-2017].

[33] J. Gausemeier and S. Moehringer, “NEW GUIDELINE VDI 2206 - A FLEXIBLE PROCEDURE MODEL FOR THE DESIGN OF MECHATRONIC SYSTEMS,” *DS 31 Proc. ICED 03, 14th Int. Conf. Eng. Des. Stock.*, 2003.

[34] S. Friedenthal, A. Moore, and R. Steiner, “Getting Started with SysML,” in *A Practical Guide to SysML*, 2nd Editio., Morgan Kaufmann, 2012, pp. 29–49.

[35] “CYLINDIRICAL ROBOTS - Robotpark ACADEMY.” [Online]. Available: http://www.robotpark.com/academy/all-types-of-robots/stationary-robots/cylindirical-robots/. [Accessed: 06-Feb-2018].

[36] European Committee for Electrotechnical Standardization, *Safety of machinery – Electrical equipment of machines Part 1: General requirements*. 2006, pp. 1–126.

[37] K. Harada, T. Tsuji, K. Kikuchi, K. Nagata, H. Onda, and Y. Kawai, “Base position planning for dual-arm mobile manipulators performing a sequence of pick-and-place tasks,” in *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, 2015, pp. 194–201.

[38] PLCOpen, “TC5 - Safety - Introduction.” [Online]. Available:

http://www.plcopen.org/pages/tc5_safety/introduction/. [Accessed: 04-Feb-2018].

[39] Siemens AG, "AS-Interface – the clever standard," Germany, 2013.

[40] T. Turc, "Using WEB Services in SCADA Applications," *Procedia Technol.*, vol. 19, no. April, pp. 584–590, Jan. 2015.

[41] J. Baraglia, M. Cakmak, Y. Nagai, R. Rao, and M. Asada, "Initiative in robot assistance during collaborative task execution," in *2016 11th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, 2016, pp. 67–74.

[42] R. Brinkmann, *AS-i Safety Monitor Configuration Software*, 11th ed. Bonn: Moeller GmbH, 2001.

[43] P. Gruhn, "Human Machine Interface (HMI) Design : The Good , The Bad , and The Ugly (and what makes them so)," *66th Annu. Instrum. Symp. Process Ind.*, pp. 1–10, 2011.

[44] T. Turc, "SCADA Systems Management Based on WEB Services," *Procedia Econ. Financ.*, vol. 32, no. 15, pp. 464–470, Jan. 2015.

[45] K. Wucherer, "HMI, The Window to the Manufacturing and Process Industry," *IFAC Proc. Vol.*, vol. 34, no. 16, pp. 101–108, Sep. 2001.

[46] 3S Smart Software Solutions GmbH, *The CoDeSys Visualization*. Kempten: 3S - Smart Sotfware Solutions GmbH, 2003.

[47] A. Hossain and T. Zaman, "AC 2012-3605 : HMI DESIGN : AN ANALYSIS OF A GOOD DISPLAY FOR SEAMLESS INTEGRATION BETWEEN USER UNDERSTANDING AND HMI Design : An Analysis of a Good Display for Seamless Integration Between User Understanding and Automatic Controls Abstract : In process a," *Am. Soc. Eng. Educ.*, p. 14, 2012.

[48] Wago Kontakttechnik and C. Kg, "762-3003, e!DISPLAY 7300T, Web Panel," 2016th ed., vol. 49, no. 0, WAGO Kontakttechnik, Ed. 2016, pp. 9–10.

[49] Wago Kontakttechnik, "753-655; WAGO AS-Interface Master," vol. 49, no. 0, Wago Kontakttechnik, Ed. Minden: WAGO Kontakttechnik GmbH, 2016, p. 32423.

7. Appendix

Due to the nature of the documents described in this section, they are contained on a CD-ROM along with this document, and organized as follows:

Appendix A: Complete electrical schematics in PDF format and the EPLAN project file. Within the folder “Electrical Schematics.”

Appendix B: Datasheets and User manuals of the electrical components used in the project. Within the folder “Components Documentation.”

Appendix C: Detailed diagrams and flowcharts in XML and PNG format for the project. Within the folder “Diagrams”.

Appendix D: Source code for the PLC as a e!Cockpit project with external libraries. Within the folder “PLC Software”.

Appendix E: Source code for the Safety System as ASIMON V3 project file and the project file for the SISTEMA risk analysis. Within the folder “Safety Configuration”.

Appendix F: Digital version of this Document in PDF format. within the folder “Thesis”.