UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ETSI Aeroespacial y Diseño Industrial

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería Aeroespacial y Diseño Industrial

Diseño e implementación de una planta de reutilizado de botellines de cerveza

Trabajo Fin de Grado

Grado en Ingeniería Electrónica Industrial y Automática

AUTOR/A: Ramos Jimenez, Jorge

Tutor/a: Martínez Turégano, Jaime

CURSO ACADÉMICO: 2023/2024

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ETSI Aeroespacial y Diseño Industrial

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela técnica superior de ingeniería del diseño

DISEÑO E IMPLEMENTACIÓN DE UNA LINEA DE REUTILIZACIÓN DE BOTELLINES DE CERVEZA

Trabajo Fin de Grado

Ingeniería electrónica industrial y automática

Autor/a: Ramos Jimenez, Jorge

Tutor/a: Martínez Turégano, Jaime

CURSO ACADÉMICO: 2023/2024

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ETSI Aeroespacial y Diseño Industrial

Índice

1. Memoria 1
   1.1 Objeto 1
   1.2 Estudio de necesidades 3
   1.3 Planteamiento de soluciones alternativas y justificación de la solución adoptada 4
      1.3.1 Elección de PLCs y HMIs 4
      1.3.2 Elección de robots 6
      1.3.3 Elección de cintas y sensores 7
   1.4 Descripción detallada de la solución adoptada 9
      1.4.1 GEMMA 10
      1.4.2 Zona 1 11
      1.4.3 Zona 2 14
      1.4.4 Zona 3 16
      1.4.5 Programación de los robots 18
   1.5 Justificación detallada de los elementos de la solución adoptada 19
      1.5.1 PLC 20
      1.5.2 Robot 23
      1.5.3 HMI 23
      1.5.4 Maquinaria de lavado y de llenado 24
      1.5.5 Dispositivos de entrada 26
2. Planos 29
3. Pliego de condiciones 32
   3.1 Objeto 32
   3.2 Condiciones de los materiales 32
   3.3 Condiciones de ejecución 33
   3.4 Pruebas y ajustes finales 33
4. Presupuesto 35
   4.1 Costes de materiales 35
   4.2 Coste de sofware 36
   4.3 Coste de la mano de obra 36
   4.4 Gastos totales 37

ANEXO I: MANUALES DE SOFTWARE 38
1. Studio 5000 39
1.1 Creación de un programa 39
1.2 Organización del proyecto 40
1.3 Variables 42
1.4 Rutinas y subrutinas 44
1.5 Elementos utilizados 46
2. RSLogix Emulate 5000 47
3. FactoryTalk View Studio 50
4. RobotStudio 55
ANEXO II: PROGRAMACIÓN DEL SOFTWARE 65
1. PLC 66
2. Robot 98
3. HMI 108
ANEXO III: ODS 125

1. Memoria

La automatización juega un papel fundamental en todas las áreas de producción industrial siendo muy relevante en la optimización de procesos, el ahorro en coste de producción y el ahorro energético. Sin embargo, la viabilidad económica de dicha automatización no siempre es garantizada. En el marco de este proyecto de reutilización de botellines de cerveza, se hace patente la necesidad de implementar un enfoque automatizado para cumplir con los requisitos de eficiencia en términos de tiempo, coste y precisión.

El principal objetivo del proyecto es aplicar los conocimientos adquiridos durante el estudio del grado en ingeniería electrónica industrial y automática con un proceso real dentro de la industria de la automatización, aplicando así los conocimientos teóricos aprendidos en clase en un contexto real. Este proyecto cuenta con varias operaciones que se dividirán en tres zonas. Esta división se realiza con el fin de gestionar de manera más eficiente las señales y, principalmente, facilitar futuras labores de mantenimiento y reparación. De esta manera, cualquier programador externo al proyecto podrá identificar de manera sencilla la fuente de cualquier fallo que pueda surgir.

Para alcanzar este propósito, se emplearán robots para el manejo de cargas, PLCs para el control integral de cada etapa del proceso y HMIs para la visualización y supervisión en tiempo real del estado operativo del sistema. Este enfoque permitirá garantizar una gestión eficaz y optimizada de la reutilización de los botellines, maximizando la eficiencia y minimizando los costes asociados al proceso.

1.1 Objeto

El objetico de este proyecto consiste en la automatización de una línea que reutiliza los botellines de vidrio de la cerveza, donde debemos controlar todas las fases de este proceso, la entrada de los botellines, el lavado, la supervisión de que el lavado sea correcto, el llenado, que debe hacerse con ciertas características especiales al tratarse de una bebida carbonatada y la pasteurización.

Se trabaja para que con esta línea de producción o una similar, podamos conseguir que las grandes empresas del sector y los demás eslabones de la cadena de suministros apuesten más por la reutilización del envase en el que se consume el producto final que la producción de nuevos envases, para los que se emplearan más materias primas, las cuales ya sabemos que son escasas.

Por tanto, para llegar a nuestro objetivo necesitamos: optimizar el tiempo de producción y reducir los costes del proceso de reutilización, donde lo enfocaremos principalmente a la reducción de personal, gracias a la automatización.

El proyecto se enfocará de la siguiente manera:

- Se va a dividir la línea de producción en 3 zonas, se hace por varios motivos, el primero es la organización, la línea debe estar estructurada de manera clara, para que en caso de fallo algún trabajador ajeno al proyecto o de nueva incorporación pueda de una manera rápida resolver cualquier fallo y la segunda, para que en caso de que por algún motivo ajeno al funcionamiento de la línea nos quedemos botellines durante un tiempo poder ahorrar energía, ya que se irán parando la mayoría de los componentes que no estén en uso según las zonas.
- Todo el control del proceso se realizará mediante un PLC el cual se programará mediante las herramientas gráficas vistas en el grado, GEMMA y GRAFCET, que nos ayudarán a estructurar de una manera eficiente, rápida e intuitiva el control de las señales del sistema.
- En la parte de control más externa, que son los modos de funcionamiento y que se hacen mediante la herramienta GEMMA, se utilizaran dos el modo automático, que es el único que tiene una función de producción real y el modo manual, que principalmente se utilizará para labores de testeo y puesta en marcha de la línea y que no debería ser utilizado por los trabajadores de la empresa.
- Se emplearán dos robots para las tareas en las que se requiere mover las cargas, el primer robot se empleará para despaletizar las cajas en las que vienen los botellines y llevarlos a la primera cinta de la línea, el segundo tendrá la tarea de extraer los botellines de estas cajas y pasarlos a otra cinta.
- Se utilizará un HMI con dos funciones, la visualización de los fallos, que principalmente se referirá a falta de producto o controles de temperatura y al uso de los modos que tenemos.

La elección específica de centrar el proyecto en el reutilizado de botellines se fundamenta principalmente en la alineación con los Objetivos de Desarrollo Sostenible (ODS) promovidos por la ONU, los cuales la Unión Europea busca implementar para fomentar un crecimiento económico que sea sostenible con el medio ambiente en el largo plazo. Este proyecto aborda principalmente el concepto de economía circular, en lo que respecta al envase que almacena el producto que se va a consumir. Al utilizar botellines de vidrio en lugar de plástico o metal, se permite su reutilización múltiple, siempre y cuando se someta al tratamiento adecuado para garantizar su viabilidad para un nuevo uso, lo que se alinea con el ODS 12, que promueve la producción y el consumo responsables.

Además del aspecto de la reutilización de los botellines, otro punto importante a considerar es que estos botellines no llevan etiquetado, sino que presentan el logotipo de la empresa distribuidora y la información pertinente impresa directamente en la forma del vidrio, al estilo de los botellines de cerveza 1925 de Alhambra. Esto conlleva múltiples beneficios, incluido el ahorro de energía y materias primas al no tener que pasar por el proceso de eliminación de etiquetas y pegamento, que requiere grandes cantidades de agua y genera desperdicio de papel y adhesivos durante el rellenado. Además, a largo plazo, esto resulta

económicamente beneficioso para la empresa distribuidora al reducir los gastos asociados a estos procesos.

1.2 Estudio de necesidades

Como se ha mencionado anteriormente el proyecto va a estar dividido en tres zonas de trabajo, para hacer una división de entradas, salidas, entradas de seguridad y programación. La división será de la siguiente manera:

- **Zona 1:** Los botellines llegan dentro de las cajas estándar utilizadas en hostelería, dispuestas en palés. Hemos establecido las medidas estándar de una caja y un palé, fijando un total de nueve cajas por palé, apiladas en tres niveles. En esta etapa, un robot extrae las cajas en grupos de tres, y posteriormente, otro robot saca los botellines individualmente de cada caja y los deposita en una cinta transportadora de acumulación.
- **Zona 2:** Los botellines avanzan desde la cinta acumuladora a una cinta transportadora diseñada para permitir el paso de los botellines en fila. Aquí, atraviesan una etapa de limpieza seguida de una etapa de secado. Posteriormente, se lleva a cabo un análisis mediante un sensor de metales y una cámara de visión artificial para asegurar que los botellines estén completamente limpios.
- **Zona 3:** Una vez limpios, los botellines son dirigidos a un carrusel de llenado. Después, pasan por una máquina coronadora para sellarlos herméticamente y ya para finalizar la cerveza es pasteurizada para garantizar su correcta conservación.

Entre estas 3 zonas contamos con los siguientes subprocesos:

- Despaletizado (Zona 1).
- Extracción (Zona 1).
- Lavado (Zona 2).
- Secado (Zona 2).
- Revisión de desperfectos (Zona 2).
- Extracción desperfectos (Zona 2).
- Llenado (Zona 3).
- Coronado (Zona 3).
- Pasteurización (Zona 3).

Para realizar todas estas tareas se necesitarán los siguientes componentes:

- Un PLC, será el que controle e integre a todos los demás componentes.
- Un HMI, lo utilizaremos con el fin de poder ver los fallos que sucedan en la línea y para parar y testear la línea.
- Dos robots, uno será el encargado de despaletizar las cajas con los botellines dentro y otro servirá para sacar los botellines de las cajas.
- Ocho sensores de proximidad para detectar los productos en varias zonas claves.

- Cintas transportadoras, las cuales deberemos de encargarlas a medida, tanto la longitud total de estas como el ancho, que varía según la zona.
- Tres sensores de puerta, ya que la zona de la línea de producción estará totalmente vallada por temas de seguridad, pero cada zona contará con una puerta de entrada.
- Tres setas de emergencia, para parar la línea en caso de alguna incidencia que sea percibida por el trabajador.
- Una máquina para el lavado de los botellines.
- Una máquina para el llenado de los botellines.
- Una máquina para el coronado de las cervezas.
- Dos hornos, aunque de distintas capacidades, uno para el sacado de los botellines y otro para el pasteurizado.
- Una cámara de visión artificial, para que nos pueda detectar si el lavado no ha sido correcto y queda algún objeto dentro del botellín o si el botellín tiene algún desperfecto.
- Dos actuadores neumáticos, uno utilizado para la retención de las cajas y otro para la expulsión de los botellines con desperfectos.

1.3 Planteamiento de soluciones alternativas y justificación de la solución adoptada

Los requisitos de la empresa contratante en cuanto a materiales se refieren es que se trabaje con PLCs de la marca Allen-bradley, pues esta nueva zona de reutilizado de botellines quiere que tenga la máxima similitud en cuanto a componentes que tiene el resto de la planta, que cuenta con distintas zonas de reutilizado y creación de botellines con su posterior llenado. Al trabajar con PLCs de esta marca se nos obliga a que el HMI deba ser también del mismo fabricante, por tener una mayor compatibilidad. Las cintas transportadoras deben ser iguales a las ya utilizadas en la planta y con el resto de los componentes tenemos una mayor flexibilidad.

1.3.1 Elección de PLCs y HMIs

A la hora de elegir un PLC debemos de tener en cuenta una serie de características, primero debemos saber el número de entradas y salidas que va a tener el sistema y si vamos a tener solo entradas y salidas digitales o también analógicas y el número, lo siguiente es saber la velocidad de procesamiento que tiene, si tenemos que automatizar un proceso muy crítico debemos tener una alta velocidad de procesamiento, la capacidad de memoria de programa y de datos, sobre todo si es un proceso complejo, tenemos que ver si necesitamos algún protocolo de comunicación para integrar al PLC con otros sistemas, el ambiente que pueda tener el entorno en el que opera el PLC y la flexibilidad con la que podemos añadir o eliminar módulos, todo esto a parte del coste del producto y de la seguridad que nos debe generar el fabricante.

Con todo esto y tras la búsqueda entre la gama de controladores que posee allen-bradley nos hemos quedado con 3 PLCs como posibles alternativas para nuestro proceso:

	1756-L83ES	5069-L320ER	5069-L380ER
Arquitectura	Modular	Modular	Modular
Memoria	10 MB	2 MB	10MB
Velocidad de procesamiento	1,20 ms/k	1.60 ms/k	0.98 ms/k
Capacidad de E/S	Escalable, miles de E/S	Escalable, hasta 31 módulos E/S	Escalable, hasta 300 módulos E/S
Conectividad	Ethernet/IP, ControlNet, DeviceNet, Modbus TCP/IP	Ethernet/IP	Ethernet/IP
Redundancia	Sí	No	Sí
Seguridad	Alta	Básica	Alta
Tipo de aplicación habitual	Grandes y complejas	Medianas	Medianas
Coste aproximado	7000€	4000€	5500€

Todos los elegidos son modulares por la posibilidad de que se pueda ampliar la zona de reutilizado y doblarla a modo espejo, teniendo así el doble de entradas y salidas y teniendo también una mayor carga de procesamiento. El PLC elegido es el 1756l-L83ES debido principalmente a su arquitectura, su memoria, es la más grande y nos da opción de ampliarla, la capacidad que tiene para añadir módulos E/S, aunque cualquiera de los otros dos nos serviría en este caso, las opciones de conexión entre otros elementos por número y por cantidad de protocolos de conexión, y esto teniendo en cuenta que tenemos dos robots y un HMI es fundamental, es una de las características que más pesa a la hora de tomar la decisión y por último la seguridad que nos proporciona este controlador. Por todo esto, aunque este controlador sea el más caro podemos proporcionar un servicio de mayor calidad y a prueba de fallos.

Para la elección del HMI debemos de considerar los siguientes factores: la compatibilidad con el PLC, el cual siendo de la misma marca no va a ser ningún problema, el tamaño de la pantalla, en nuestro caso no necesitamos una excesivamente grande, pues no tenemos una gran cantidad de objetos que visualizar en ella, la capacidad de memoria, la conectividad de red, las características de seguridad y operatividad y que pueda ofrecer una interfaz de usuario intuitiva y robusta para la operación y el monitoreo del sistema.

Con todo esto aquí tenemos nuestras tres alternativas escogidas entre la oferta de allen-bradley:

	PanelView Plus 7 Standard	PanelView Plus 7 Performance	PanelView 5310
Modelo	2711P-T15C22D8S	2711P-T15C22D9P	2713P-T12WD1
Pantalla	10 pulgadas	10 pulgadas	15 pulgadas
Memoria	512 MB de RAM y 512 MB memoria no volátil	4GB de RAM y 20 GB memoria no volátil	1GB de RAM y 1 GB memoria no volátil
Conectividad de red	Un puerto ethernet	Dos puertos ethernet, que soportan topologías de red DLR, lineal o estrella	Dos puertos ethernet, que soportan topologías de red DLR
Sistema operativo	Windows CE	Windows IoT	Personalizable
Precio	3500€	5350€	4700€

Principalmente por el factor de la conectividad y la memoria del HMI hemos elegido para el proyecto el PanelView 7 Performance ya que por un precio no mucho más elevado que el PanelView 5310 tenemos la posibilidad de integrar nuestro sistema con otros dispositivos y más topologías de red.

1.3.2 Elección de robots

A la hora de elegir los robots que realizaran las tareas de des paletizado y de extracción de las botellas de vidrio de las cajas tenemos que tener en cuenta diversos factores, la facilidad para la comunicación entre el robot y el PLC, el rango de movimiento que tiene el robot en todos los sentidos, los ejes que posee, la capacidad de carga que es capaz de levantar a la cual hay que tener en cuenta el gripper, la repetibilidad y algo que ya depende de la propia fabrica del robot que es la facilidad que nos proporcione su software para la programación y para la simulación del robot, aparte por supuesto de la fiabilidad, el coste y el servicio de atención al cliente que pueda tener.

Para el robot de des paletizado hemos de tener en cuenta que debe tener un amplio alcance, si debe retirar 3 filas de cajas y tiene que llegar a la última sabemos que deberá de moverse al menos 60cm mas la distancia a la que este colocado de la primera fila de cajas, por tanto, mínimo queremos que el alcance sea de dos metros para asegurarnos una buena movilidad y que el robot no llegue forzando ningún eje a ningún punto, pues la vida útil de este podría resentirse. En cuanto a la capacidad de carga, sabemos que las 3 cajas que debe levantar a la vez pesan entre 20kg y 25kg y el gripper unos 10kg, por lo que mínimo queremos que el robot tenga una capacidad de carga de 40kg.

Debido a la libertad con la que contamos de cara a la elección de los robots, se ha optado por elegir un robot de características similares entre marcas, las que son más utilizadas dentro de la industria de la automatización ABB, FANUC y KUKA:

	ABB IRB 4600-45	KUKA KR 60-3	FANUC M-710iC/50
Capacidad de carga	45kg	60kg	50kg
Alcance	2,05m	2,03m	2m
Repetibilidad	+/- 0,05mm	+/- 0,06mm	+/- 0.07mm
Controlador	IRC5	KRC4	R-30iB
Puntos fuertes	Alta velocidad y precisión	Versatilidad y robustez	Rigidez y estabilidad
Software	RobotStudio	Kuka Sim	CNC Guide y Roboguide
Coste	60000€	65000€	70000€

Hemos elegido el robot ABB IRB 4600-45 para la tarea de despaletizado por diversos motivos, el primero es que cumple con creces con todas características deseadas pero el factor determinante es el software de programación que utiliza ABB, que es muy intuitivo y manejable, además de que nos proporciona todas las herramientas necesarias para hacer simulaciones lo más próximas a la realidad, lo cual hace que a la hora del montaje haya mucho trabajo hecho de manera remota antes de llegar a la planta. Esto y unido al precio que sin duda es el más competitivo hace que sea nuestro robot despaletizador.

Para el segundo robot ya que la carga va a ser más o menos la misma, será un par de quilos menos al no tener que levantar las cajas, solo levanta las botellas, se ha decidido optar por la misma elección que el anterior, por varias razones, la primera es que para realizar una simulación y poder sincronizar los dos robots es mejor que sean de la misma marca y para futuras revisiones y reparaciones también, además de que para ponernos en contacto con los proveedores mucho mejor repetir productos que pedir para la misma aplicación maquinaria distinta.

1.3.3 Elección de cintas y sensores

Vamos a tratar de buscar alternativas para otros elementos utilizados como las cintas y los sensores de presencia, que son los otros componentes o bien más costosos o bien más empleados en el proceso.

Para la elección de las cintas transportadoras debemos tener en cuenta varios factores, que el material del que están hecho cumpla con las normativas de seguridad alimentaria, sea resistente al agua, habrá zonas como la de llenado o la de lavado que puedan mojarse más de lo habitual y que cuenten con un fácil mantenimiento.

	Dorner 3200 Series Modular Belt Conveyors	Intralox Series 900 Flat Top	FlexLink WLX Conveyor System
Marca	Dorner	Intralox	FlexLink
Material	Acero inoxidable	Plásticos aprobados en la industria alimentaria	Acero inoxidable
Seguridad	Protección contra fragmentos pequeños	Pocos puntos de atrapamiento	Pocos puntos de atrapamiento
Resistencia al agua	Sí	Sí	Sí
Modular	Sí	Sí	No
Coste	2500€/m	1500€/m	2000€/m

Ya que todas en cuanto características son bastante similares y queremos que sean modulares para poder tener un mejor mantenimiento o una adaptación más fácil en caso de que hiciera falta expandir alguno de los tramos, hemos elegido la cinta Intralox Series 9000, principalmente por su precio tan competitivo, ya que como necesitaremos unos 12m de cintas podemos ahorrar bastante y aun así ofrecer un servicio de calidad.

Para la elección de los sensores de presencia debemos tener en cuenta distintas cosas, lo primero la precisión y el alcance de detección, queremos que nos dé una señal digital apta para el PLC, también necesitamos que sea resistente en entornos húmedos y que tenga cierta resistencia contra impactos y suciedad en el ambiente.

Los tres sensores que hemos visto en el mercado y que cumplen con los requisitos planteados son los siguientes:

	UBE1000-18GM40A-SE2-V1	WTB4FP-213111	Q4XTBLAF300-Q8
Marca	Pepperl+Fuchs	Sick	Banner Engineering
Tipo	Ultrasónico	Fotoeléctrico	Láser
Rango de detección	1000mm	400mm	300mm
Salida	PNP/NPN conmutables	PNP	PNP/NPN seleccionable
Carcasa	Acero inoxidable	Plástico reforzado	Acero inoxidable
Coste	230€	200€	400€

El sensor elegido es el Pepperl+Fuchs UBE1000-18GM40A-SE2-V1 debido a su alta precisión y largo alcance de detección, su resistencia a entornos industriales complejos gracias a su carcasa de acero inoxidable, y su compatibilidad con salidas NPN/PNP.

1.4 Descripción detallada de la solución adoptada

La solución adoptada para la programación del PLC que controlara todos los movimientos, seguridades y maquinaria necesaria en el proceso ha sido hecho utilizando las herramientas que nos proporciona la guía grafica GEMMA y el GRAFCET.

GEMMA es una guía gráfica que permite presentar, de una forma sencilla y comprensible, los diferentes modos de marcha de una instalación de producción, así como las formas y condiciones para pasar de un modo a otro. Digamos que sería algo así como las macro etapas del GRAFCET principal, donde los movimientos que haya que hacer se hacen dentro de ellas. Tiene tres zonas: la zona A, donde se incluyen los procedimientos de parada, la zona F, donde van los procesos de funcionamiento y la zona D, donde van los procesos en defecto.

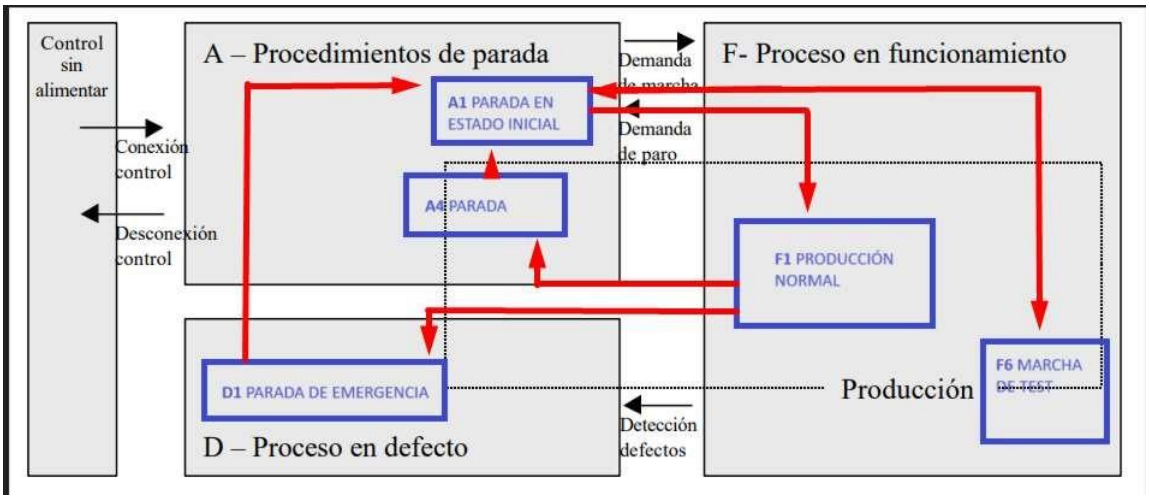


Fig 1: Funciones GEMMA utilizadas

Y el GRAFCET es un modelo de representación gráfica, de los sucesivos comportamientos de un sistema lógico, predefinido por sus entradas y salidas. Donde llevándonoslo a nuestra programación del PLC una etapa es activada por la anterior y las condiciones necesarias y realimentada por sí misma y la negada de la siguiente.

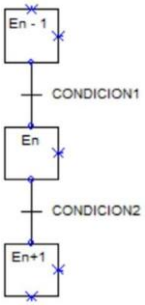


Fig 2: ejemplo de GRAFCET con 3 etapas

Fig 3: Ejemplo de programación del GRAFCET de la figx-1

1.4.1 GEMMA

Una vez hemos visto la lógica que hemos usado para la programación del proceso, vamos a ver como lo hemos implementado. Como hemos dicho la parte más externa de nuestro proceso es el GEMMA, donde hemos utilizado las macroetapas A1 de para en el estado de inicio, lo que sería la posición de partida, la F1, que será la de funcionamiento automático del proceso, la F6 para las marchas de test, donde podemos manejar uno a uno los distintos elementos, para la parada de emergencia, que puede ser por seta de emergencia o por apertura de las puertas del vallado se utiliza la macro D1, la A4 para una parada normal y la A6 para devolver los dispositivos que haga falta a un estado inicial después de los procesos de parada.

Este sería el resultado del GEMMA:

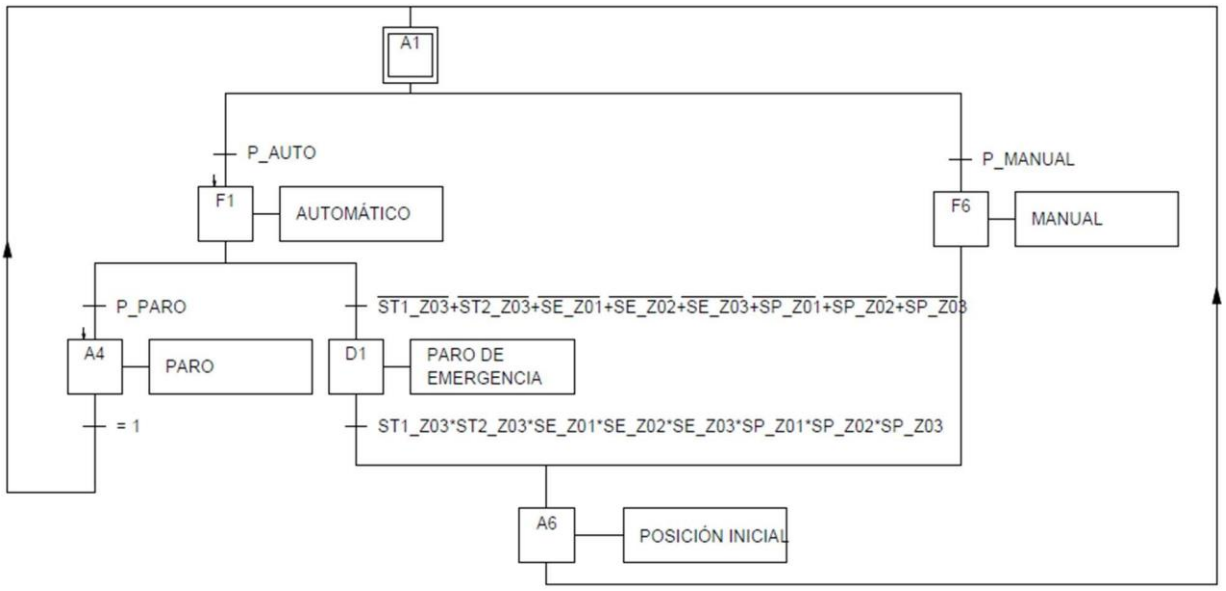


Fig 4: GEMMA utilizado

Tanto el pulsador P_AUTO como el pulsador P_MANUAL y el P_PARO, son botones no físicos que estarán en el HMI. En cuanto a las seguridades, los contactos que inician su nomenclatura por SE, se corresponden con las setas de emergencia, donde hay una por zona debido a las dimensiones de la planta y los que inician por SP, son los sensores de las puertas de cada zona, utilizamos lógica negada ya que estos contactos son siempre normalmente negados debido a cuestiones de seguridad.

En cuanto a los sensores de temperatura, pertenecen el primero a la parte del llenado, donde la cerveza se debe inyectar a unos 2-3ºC para que no salga espuma, y el segundo al horno de pasteurizado. Por tanto, como son procesos en los que ambos estaríamos perdiendo producto, en el primer caso por la espuma y en el segundo porque no estaríamos cumpliendo con las normas de producción alimentaria, se parara toda la estación de producción hasta que el fallo quede resuelto.

1.4.2 Zona 1

En la zona uno nos encontramos primero que todo con la parte donde se van dejando los pales, ya sea algún operario con una fenwick de uno en uno o ya sea a través de una serie de cintas de forma automatizada. Para estos pales hemos supuesto que las cajas son de botellines de un tercio de litro y que vienen 24 botellines en cada caja.

Cada caja tiene una medida de 400x260x300mm.

Fig 5: modelo de caja para las botellas

Y el botellín elegido será el siguiente:

Referencia	**577/3740**
Capacidad	33 cl.
Altura	238 mm
Diámetro	61 mm
Color	Topacio Oscuro
Peso	300 gr
Boca	Corona 26

Fig 6: modelo de botellín a reutilizar

En esta parte de la zona hay un robot que se dedica a coger las cajas en grupos de tres, esto es posible gracias a un gripper especial que cuenta con varias ventosas, que nos permite manejar el peso de las cajas de manera segura.

Apuntando a la última fila de cajas que coja el robot, ya que siempre será la misma, tendremos un sensor de proximidad, para que el robot repita el ciclo siempre que haya cajas y se resetee cuando deje de haberlas.

Una vez colocadas las cajas en la cinta transportadora, estas pasaran por un estrechamiento con barreras donde solo deben caber las cajas y llegaran hasta un sensor de proximidad que parara la cinta, pero previo a esto un actuador saldrá haciendo de tope para detener el avance de las cajas mientras la cinta continua activa unos segundos, tanto este último proceso como el estrechamiento de la cinta se hace para que el robot número dos, el que se dedica a coger los botellines de las 3 cajas tenga la mayor precisión posible a la hora de coger todas las botellas.

Llegado a este último punto, las botellas son sacadas por el robot dos y puestas en una cinta acumuladora, que contara con un tope inferior y otro superior, para saber cuándo tendremos botellines al principio, y cuando al final, ya que, si no hay botellines en la zona más estrecha, no tiene sentido que el resto de la línea siga funcionando y si hay botellines en la zona donde se dejan los botellines tampoco tiene sentido que se sigan acumulando ahí porque no caben más.

Una vez los botellines son extraídos y colocados en la cinta acumuladora se repite el proceso y las cajas continúan su camino hasta donde otro proceso o algún operario las vaya colocando donde sea necesario.

La disposición de esta zona seria la siguiente:

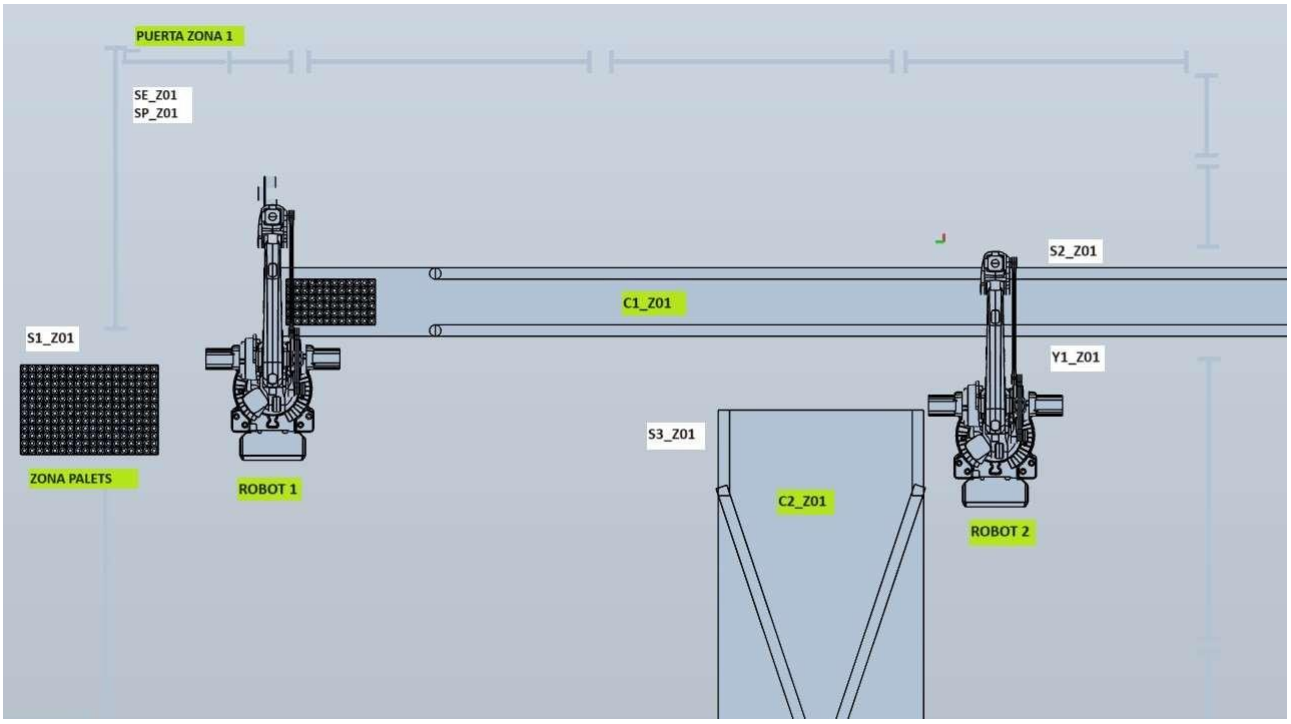


Fig 7: simulación zona 1 etiquetada

El grafcet de esta parte del proceso quedaría de la siguiente forma:

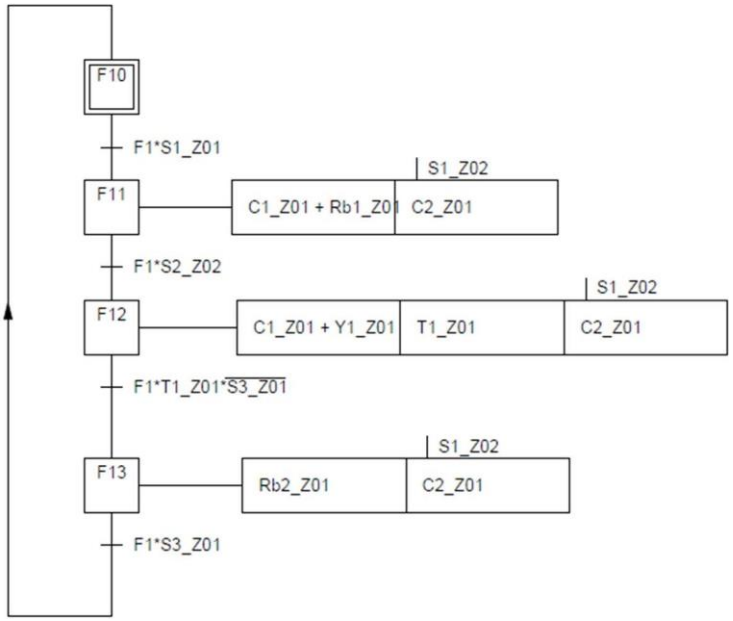


Fig 8: GRAFCET zona 1

ENTRADAS ZONA 1		
Nombre	**Descripción**	**Dirección**
SE_Z01	Seta de emergencia de la zona 1	Local:2:I:Data.0
SP_Z01	Sensor de puerta zona 1	Local:2:I:Data.1
S1_Z01	Sensor última fila palet	Local:3:I:Data.0
S2_Z01	Sensor cajas en posición delante de robot 2	Local:3:I:Data.1
S3_Z01	Detector límite de botellines cinta acumuladora	Local:3:I:Data.2

SALIDAS ZONA 1		
Nombre	**Descripción**	**Dirección**
C1_Z01	Cinta transportadora desde palet a robot extractor	Local:4:O.Data.0
C2_Z01	Cinta acumuladora de botellines	Local:4:O.Data.1
Y1_Z01	Actuador acumulador de cajas	Local:4:O.Data.2
Rb1_Z01	Señal para robot 1	Local:4:O.Data.3
Rb2_Z01	Señal para robot 2	Local:4:O.Data.4

1.4.3 Zona 2

La zona 2 comprende la zona de lavado, secado y expulsión de los botellines que no estén completamente limpios.

Los botellines entran desde la cinta acumuladora a una cinta donde solo cabe un botellín, de esta forma las maquinas empleadas más adelante podrán coger de forma más sencilla los botellines uno a uno. El primer proceso al que llegan los botellines a esta zona es a la parte de lavado, donde una máquina de tipo carrusel ira cogiendo botellas, dándoles la vuelta e inyectándoles agua con una solución jabonosa, tendrá que tener un momento de lavado y otro de aclarado para no dejar ningún jabón y/o desinfectante dentro, la maquina cuenta a la entrada y a la salida con dos ruedas dentadas giratorias por las cuales se pasan una a una las botellas que van entrando al carrusel y este las coge y las voltea para la inyección.

Una vez acabado el lavado pasan por un pequeño horno donde se secarán por completo y tras este secado son analizadas por una cámara de visión artificial y un detector de metales, para que no quede ningún desecho dentro. En caso de que alguno de estos dos sensores detecte un botellín sucio un actuador lo desechara a otra cinta.

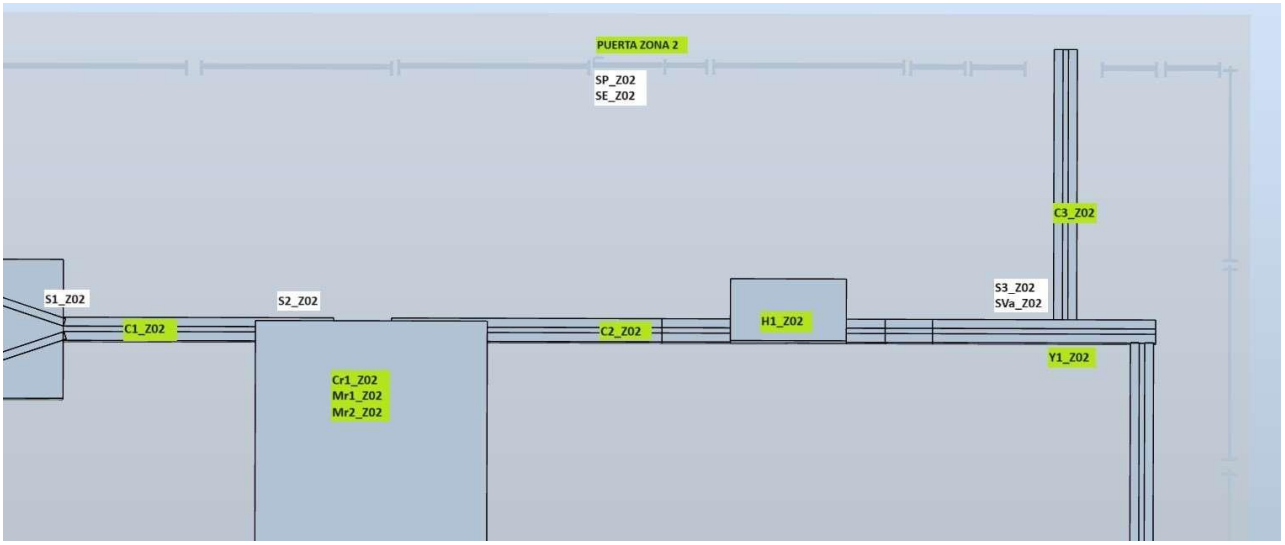


Fig 9: simulación zona 2 etiquetada

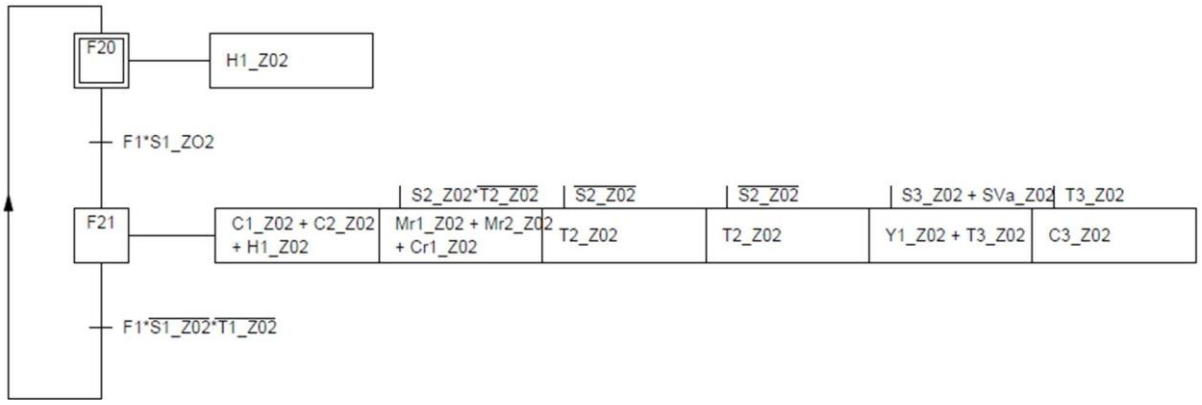


Fig 10: GRAFCET zona 2

ENTRADAS ZONA 2		
Nombre	**Descripción**	**Dirección**
SE_Z02	Seta de emergencia de la zona 2	Local:2:I:Data.2
SP_Z02	Sensor de puerta zona 2	Local:2:I:Data.3

S1_Z02	Sensor entrada cinta botella en fila	Local:3:I:Data.3
S2_Z02	Sensor entrada máquina de lavado	Local:3:I:Data.4
SVa_Z02	Entrada cámara de visión artificial	Local:3:I:Data.5

SALIDAS ZONA 2		
Nombre	**Descripción**	**Dirección**
C1_Z02	Cinta 1 de la zona 2, entrada a lavado	Local:4:O.Data.5
C2_Z02	Cinta 2 de la zona 2, salida de lavado	Local:4:O.Data.6
C3_Z02	Cinta de expulsión por defecto botella	Local:4:O.Data.7
Mr1_Z02	Motor rueda giratoria entrada carrusel	Local:4:O.Data.8
Mr2_Z02	Motor rueda giratoria salida carrusel	Local:4:O.Data.9
Cr1_Z02	Señal de entrada a carrusel	Local:4:O.Data.10
H1_Z02	Horno de secado	Local:4:O.Data.11
Y1_Z02	Actuador de expulsión botellín sucio	Local:4:O.Data.12

1.4.4 Zona 3

En la zona 3 y última, contamos con los últimos pasos para acabar con el producto final, tenemos la parte de llenado, la parte de coronado y la última parte que será la pasteurización de los botellines ya llenos y sellados.

Las botellas de vidrio que hayan pasado la criba anterior de la zona 2, llegaran por la primera cinta hasta las ruedas que colocan una a una los botellines en el carrusel de llenado, de la misma manera que se hacía en el carrusel de lavado, cuando ya han dado la vuelta completa y están llenas pasan por la segunda rueda que las coloca en la segunda cinta, donde van directas a pasar por la máquina de coronado, que conforme va detectando el sensor esta baja y coloca la chapa correspondiente. Con el producto ya finalizado solo falta pasar por el horno de pasteurizado, en el cual debemos de alcanzar unas temperaturas en torno a 70-75 ºC para eliminar las bacterias pertinentes y para cumplir así con las normas sanitarias.

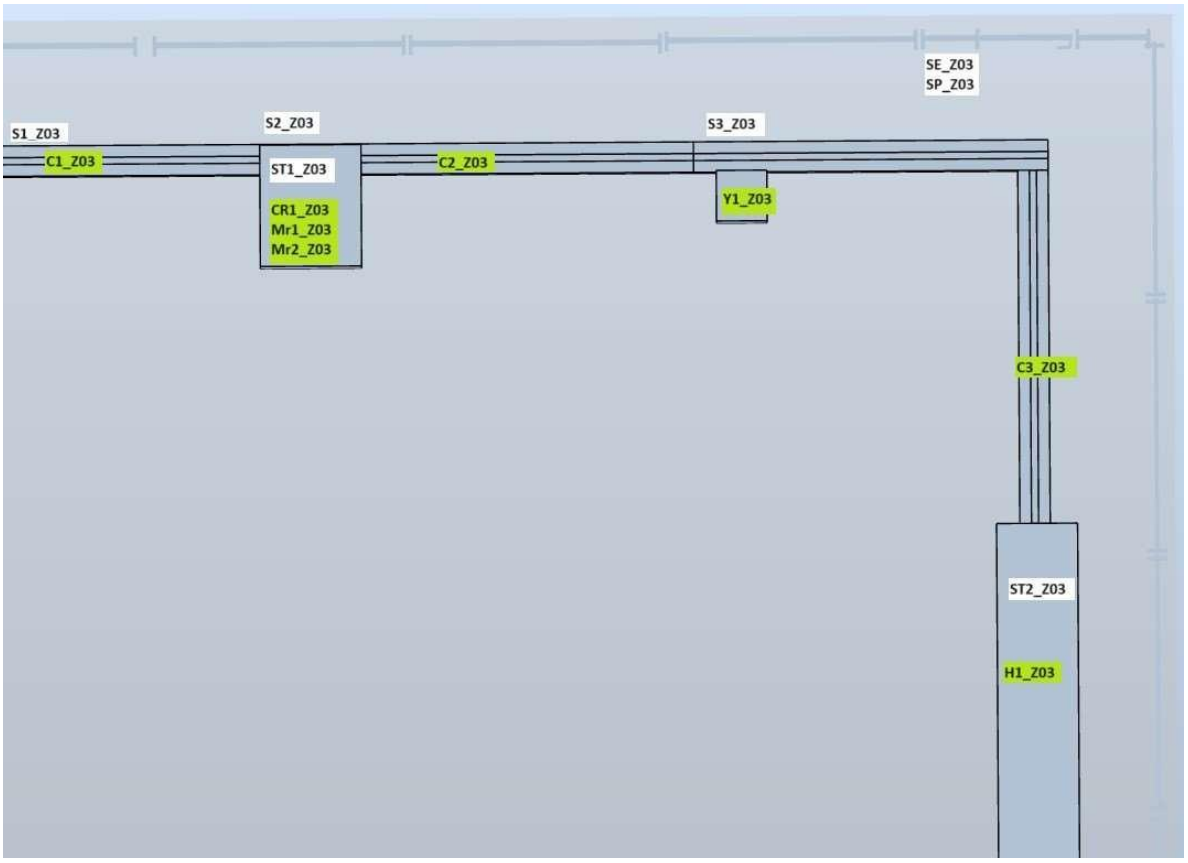


Fig 11: simulación zona 3 etiquetada

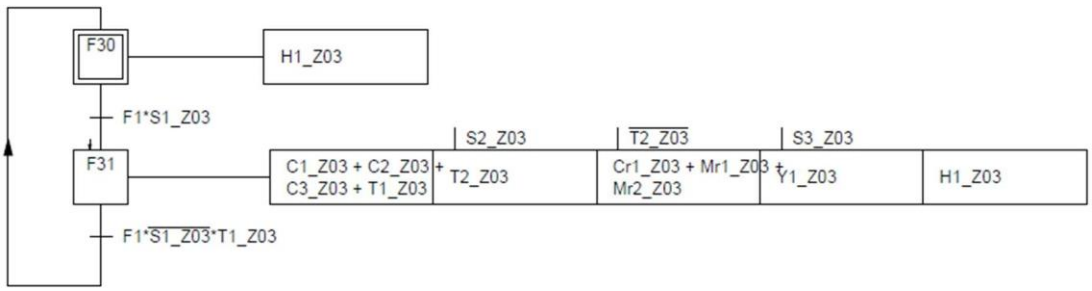


Fig 12: GRAFCET zona 2

ENTRADAS ZONA 3		
Nombre	**Descripción**	**Dirección**
SE_Z03	Seta de emergencia de la zona 3	Local:2:I:Data.4
SP_Z03	Sensor de puerta zona 3	Local:2:I:Data.5
S1_Z03	Sensor entrada a zona 3	Local:3:I:Data.6
S2_Z03	Sensor entrada máquina de llenado	Local:3:I:Data.7
S3_Z03	Sensor coronado	Local:3:I:Data.8
ST1_Z03	Sensor temperatura llenado	
ST2_Z03	Sensor llenado pasteurización	

SALIDAS ZONA 3		
Nombre	**Descripción**	**Dirección**
C1_Z03	Cinta 1 de la zona 3, entrada a llenado	Local:4:O.Data.13
C2_Z03	Cinta 2 de la zona 2, salida de llenado	Local:4:O.Data.14
C3_Z03	Cinta de entrada a horno pasteurización	Local:4:O.Data.15
Mr1_Z03	Motor rueda giratoria entrada carrusel	Local:5:O.Data.0
Mr2_Z03	Motor rueda giratoria salida carrusel	Local:5:O.Data.1
Cr1_Z03	Señal de entrada a carrusel	Local:5:O.Data.2
Y1_Z02	Actuador de coronado	Local:5:O.Data.3
H1_Z02	Horno de pasteurización	Local:5:O.Data.4

1.4.5 Programación de los robots

En cuanto a la programación de los robots, la hemos realizado mediante el sofware de ABB, RobotStudio, con este sofware lo que hemos hecho es recrear la planta con las medidas que tendrá la planta real, lo que conseguimos con esto es ahorrar todo el tiempo y el coste que gastaríamos en la planta cervecera tomando los puntos con los robots reales, aunque siempre en la vida real tendremos que realizar distintos ajustes.

La programación es sencilla, solo tenemos que grabar los puntos que queremos que haga el robot, dar las pausas necesarias para que se realicen las tareas de pick y de drop, a las cuales también tendremos que darles la señal a los grippers para que se realicen los agarres y en el caso del robot uno ir indicando las nuevas posiciones conforme se vayan retirando las cajas y se vaya vaciando el palet.

El robot numero uno, el que se encarga de coger las cajas y depositarlas en la cinta numero uno es el único que tiene lógica en su programación, ya que debemos de modificar la posición de las cajas tanto por fila como por columna, según se vaya vaciando el palet y queda de la siguiente manera:

```
PROC main()
    !Añada aquí su código
    VAR num Lenght_box := 400;
    VAR num High_box := 300;
    VAR num num1_box := 0;
    VAR num num2_box := 0;

    WHILE 1 DO

        WHILE PLC_OK = 1 DO
            WHILE num2_box < 3 DO
                WHILE num1_box < 3 DO
                    Path_10;
                    Pick:=[[16.284552209-Lenght_box*num1_box,946.749467076,902.705002952-High_box*num2_box],[0.499999999,-0.500000013,-0.499999969,-0.500000019],[0,1,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
                    Pick_Out:=[[16.284552209-Lenght_box*num1_box,946.749467076,1251.320587642-High_box*num2_box],[0.499999999,-0.500000013,-0.499999969,-0.500000019],[0,1,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
                    num1_box := num1_box + 1;
                ENDWHILE
                num2_box := num2_box + 1;
            ENDWHILE
            num1_box := 0;
            num2_box := 0;
        ENDWHILE

    ENDWHILE

ENDPROC
```

Fig 13: Lógica robot 1

Trata de un bucle en el cual en primera instancia atacamos a la primera caja, con la posición pick, una vez dejada en la cinta pasamos a la segunda caja, para lo que modificamos el punto pick y pick_out todo lo que mide una caja, una vez dejada se hace lo propio con la última, y tras esa primera fila se modifica también en el eje Z para bajar a la siguiente fila, una vez se vacié el palet se repite el bucle mientras la entrada PLC_OK que nos la da el sensor de palet detecte que hay caja.

El robot numero dos, el encargado de coger las botellas de vidrio de las cajas tan solo tendrá que repetir el bucle de la trayectoria de los puntos mientras le llegue la señal de PLC_OK, pero no tiene ningún tipo de lógica.

1.5 Justificación detallada de los elementos de la solución adoptada

La selección de los componentes para este proyecto se sustenta en dos criterios fundamentales: la disponibilidad de software de programación y la experiencia previa con marcas específicas. En este sentido, hemos optado por utilizar dispositivos PLC y HMI de la reconocida marca Allen-Bradley, respaldados por nuestra familiaridad con su programación y su fiabilidad en entornos industriales. Asimismo, hemos seleccionado robots de la marca ABB, aprovechando su reputación por su precisión y eficacia en aplicaciones industriales similares. Además, contamos con la ventaja adicional de disponer de software de simulación como RobotStudio, que nos proporciona una visión detallada y precisa del proyecto, facilitando así su desarrollo y optimización. Para los demás

componentes esenciales, como las cintas transportadoras, la máquina de lavado, la máquina de llenado, así como los sensores y dispositivos de seguridad como las setas de emergencia, se llevó a cabo un estudio exhaustivo del mercado.

1.5.1 PLC

Como bien hemos dicho antes hemos elegido para nuestro PLC productos de la marca Allen-bradley, hemos elegido en este caso un controlador modular, por los que somos nosotros los que elegimos componente a componente de nuestro PLC.

1.5.1.1 Chasis

Hemos elegido el chasis 1756-A10 que cuenta con 10 slots para conectar distintos módulos y a parte el de la batería. No vamos a ocupar por completo estos 10 huecos, pero si vamos a utilizar bastantes módulos, tenemos el controlador, el módulo de seguridad, el módulo de entradas de seguridad, el módulo de entradas, dos módulos de salidas y un módulo de entradas analógicas, por lo que en total utilizaremos un total de seis módulos dejando cuatro huecos libres para posibles futuras ampliaciones de la línea o futuras mejoras.

En cuanto a las especificaciones más técnicas que nos han hecho decantarnos por este producto es la alta velocidad de comunicación que nos proporciona su backplane, el conexionado de buses de comunicación entre módulos y la facilidad que tiene para elegir donde queremos situar cada componente, ya que cualquier componente es válido en cualquier ranura. Cuenta con unas medidas de 580x762x203 mm y un peso de 1.45 kg, con la batería que vamos a utilizar de 24 V en DC tenemos una corriente máxima en el backplane de 2.8 A por ranura y una disipación de potencia máxima de 5 W.

Fig 14: chasis PLC

1.5.1.2 batería

Para la Fuente de alimentación hemos elegido el modelo 176-PA75 donde el voltaje de entrada puede ser cualquier valor normal que encontremos en una toma en Europa o en los EEUU, es decir, podemos conectarla a a 120 V o a 240 V y a 50 Hz o a 60 Hz, por tanto,

es ideal tener componentes válidos para cualquiera de los dos mercados. Tiene una potencia de entrada máxima de 100 W y una potencia de salida máxima de 75 W y cuenta con una capacidad de corriente de 2.5 A a 24 V.

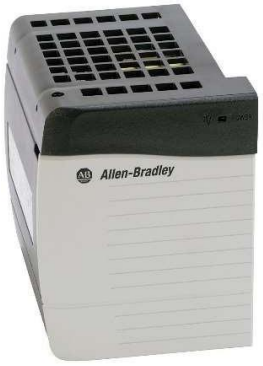


Fig 15: Fuente de alimentación PLC

1.5.1.3 Controlador

Aunque nosotros llamamos a estos componentes PLC modular y en este nombre incluimos todos los módulos, lo que es realmente el PLC en sí, es el controlador que es responsable de procesar la lógica de control definida en nuestro programa, supervisando las entradas y actualizando las salidas del sistema. Además, facilita la comunicación con otros dispositivos dentro del entorno industrial, como el HMI, a través de las redes de comunicación que posee. También asume la tarea detectar y gestionar errores o condiciones excepcionales, garantizando así el funcionamiento seguro y eficiente del proceso automatizado.

En el proyecto se utilizará el controlador 1756-L83ES, cuenta con una memoria de usuario de 10 MB ampliable hasta 2 GB, puede manejar un total de 128000 entradas y salidas, de las cuales solo 4000 pueden ser analógicas, tiene una potencia máxima de 6.2 W.

Fig 16: Controlador PLC

1.5.1.4 Módulos de seguridades

Para gestionar todas las entradas de seguridad, que son las setas de cada zona y los sensores de cerrado de las puertas del vallado, hemos elegido añadir un módulo especial de entradas de seguridad, que tienen una respuesta más rápida ante cualquier cambio. Para esto hemos elegido el módulo 1756-IB16S cuenta con 16 posibles entradas o salidas, pero nosotros solo utilizaremos 6 de ellas.

Fig 17: Módulo de E/S de seguridad

1.5.1.5 Módulos E/S

Los módulos de entrada y salida que se utilizaran para el proyecto son los siguientes, un módulo de entrada de 16 bits, pues contamos con un total de 10 entradas digitales de uso común, el módulo es el 1756-IV16, contaremos también con dos módulos 1756-OB16I, específicos para las salidas que contamos con un total de 21 salidas digitales. Y finalmente emplearemos un módulo especial para las entradas de los sensores de temperatura que ha de ser de entradas analógicas, el 1756-IF6I, que en este caso solo tendrá disponibles hasta 6 entradas.



Fig 18: Módulos de entradas digitales, salidas digitales y entradas analógicas

Para cada módulo es necesario utilizar un terminal para los pines, ya que todos vienen sin él, pero podemos utilizar el mismo para todos, usaremos el 1756-TBNH.

1.5.2 Robot

La elección del ABB IRB 4600-45/2.05 para la zona de reutilizado se justifica no solo por sus especificaciones técnicas, sino también por la capacidad de integración y comunicación que ofrece. Con su capacidad de carga de 45 kg, el robot puede manejar con seguridad y eficiencia las cajas de botellas vacías y el gripper, asegurando un manejo preciso y seguro de los botellines. El alcance de 2.05 metros permite cubrir adecuadamente el área de trabajo necesaria para recoger y posicionar las cajas desde los palets.

El uso de RobotStudio permite optimizar las operaciones antes de la implementación, reduciendo los tiempos de configuración y asegurando que el sistema funcione correctamente desde el primer día. La capacidad de comunicar eficientemente con PLCs y otros sistemas de automatización asegura una operación coordinada y sin interrupciones, lo que es crucial para una planta de reutilizado de botellines de cerveza donde la eficiencia y la precisión son esenciales. En términos de costo y beneficios operativos, el ABB IRB 4600-45/2.05 ofrece una excelente combinación de rendimiento y flexibilidad, haciendo de él la elección ideal para esta aplicación.

Fig 19: ABB IRB 4600-45/2.05

1.5.3 HMI

La elección de la pantalla HMI que nos proporcionara una lectura del estado de la planta en cada momento y la posibilidad de manejar ciertos elementos ha sido acotada por la elección

de un PLC de la marca Allen-bradley, que por compatibilidad y por usos de sofware el HMI será de la misma marca. Se ha optado por el PanelView 7 performance este HMI ofrece una pantalla táctil de alta resolución de 10 pulgadas, proporcionando una interfaz de usuario clara e intuitiva que facilita la operación y supervisión del sistema. Su capacidad de monitorización en tiempo real y control remoto permite a los operadores ajustar parámetros y controlar equipos. La compatibilidad con múltiples protocolos de comunicación industrial, incluidos Ethernet/IP, PROFINET y Modbus TCP/IP, asegura una integración fluida con PLCs y otros sistemas de automatización, garantizando una operación coordinada y eficiente de todos los componentes de la planta. Además, su robusto sistema de alarmas y notificaciones permite una respuesta rápida a cualquier problema o anomalía, minimizando el tiempo de inactividad y mejorando la productividad.

Fig 20: HMI PanelView 7 performance

1.5.4 Maquinaria de lavado y de llenado

Para las máquinas de lavado y llenado de las botellas, hemos elegido a la misma empresa para ambas, son de la compañía Frusso y su principio de funcionamiento es el mismo, en ambas vienen las botellas en fila de uno y una especie rueda dentada giratoria las va pasando una a una al carrusel una vez en este la maquina lo detecta y ya hace su función.

En el caso de la máquina de lavado, es una RIV40, tiene una capacidad de lavado de 120 bot/min y tiene un tamaño de 3000x3000x2300 mm y lo que hace es que cuando entran al carrusel las botellas, las voltea y mientras va girando el carrusel les va inyectando primero una solución de agua jabonosa con desinfectante y posteriormente agua para aclararlas una vez terminada la vuelta salen por la segunda rueda de la misma forma que han entrado y continua el proceso.

Fig 21: Maquina de lavado

En el caso de la máquina de llenado el principio y el funcionamiento es el mismo con la diferencia que las botellas no son volteadas, hemos elegido una maquina con una capacidad de 3000 l/h lo que nos da unas 150 bot/min teniendo en cuenta que son botellines de 1/3 l, las dimensiones son de 1000x1200x2000 mm y debemos tener en cuenta que el tanque de donde proviene la cerveza tiene que mantenerse a una temperatura de 2/3ºC para un llenado correcto de la cerveza.

Fig 22: Maquina de llenado

1.5.5 Dispositivos de entrada

Como entradas en el proyecto hemos utilizado siete sensores de proximidad, tres setas de emergencia, dos sensores de temperatura, un sensor detector de metales y una cámara de realidad virtual.

En cuanto a los sensores de proximidad se han elegido unos sensores ultrasónicos de la empresa pepperl+Fuchs, los UBE1000-18GM40A-SE2-V1, que funcionan a través del envío y recepción de impulsos ultrasónicos. Tiene un rango de detección amplio que va de los 15 mm a los 1000 mm, una tensión de trabajo de entre 10-30 V y una salida de 3 V de tipo PNP.

Fig 23: Sensor ultrasónico

Los sensores de temperatura elegidos son de tipo RTD pt100, válidos para liquitos y gases, por lo que emplearemos el mismo para el horno de pasteurizado y para el tanque de llenado de la cerveza, son de la marca jumo, con rangos de temperatura de -50 a 260 ºC.

Fig 24: Sensor PT100

Para las puertas hemos elegido sensores de tipo magnético de la marca ifm y el modelo MN508S - MN44008-AKOA/-H/US/8P, que nos proporciona dos contactos normalmente abiertos y uno normalmente cerrado, por lo que nosotros adecuaremos la tensión de los contactos a la de entrada al PLC.

Fig 25: Sensor magnético puerta

Para la cámara de visión artificial que nos ayudará a eliminar los botellines con defectos o mal lavados será la cámara de visión artificial XG-7000 de Bitmakers, es un sistema compacto y versátil diseñado para entornos industriales. Ofrece una combinación única de facilidad de configuración y flexibilidad, permitiendo ajustes tanto desde el controlador como desde un PC mediante el "Proceso 2Way". Con una amplia gama de cámaras disponibles, incluyendo modelos de área y de barrido lineal, la XG-7000 garantiza velocidades muy altas de procesamiento de imagen. Su controlador expandible de ultra alta velocidad permite la conexión de unidades de expansión de iluminación para un control simplificado. Gracias al procesamiento paralelo de ultra alta velocidad, esta cámara logra un rendimiento estable y rápido, respaldado por un diseño robusto y confiable sin ventilador y una unidad de disco de estado sólido (SSD). Además, nos proporciona uno o varios módulos de entradas y salidas.

La programación de la cámara se realiza mediante el software de Bitmakers, donde básicamente tenemos que nosotros mediante imágenes de muestra enseñarle cuales son las piezas que consideraremos defectuosas y cuáles son las que son buenas.

Fig 25: Camara VA

Bibliografía

https://cervezarios.com/articulos/pasteurizar-la-cerveza-pasteurizacion-cerveza/

https://estrellagalicia.es/amantes-cerveceros/tecnologia-para-envasar-la-cerveza/

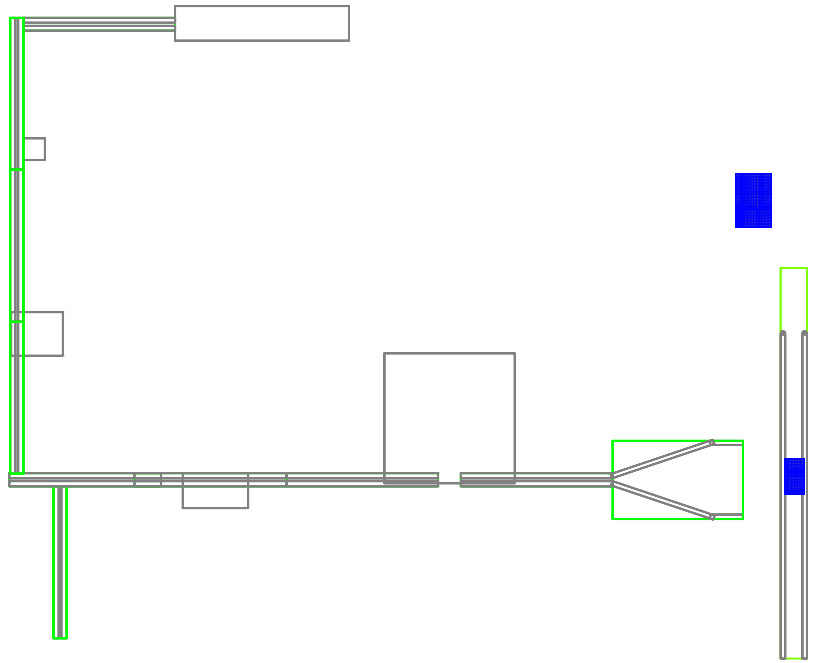
https://www.infoplc.net/

https://www.un.org/sustainabledevelopment/es/

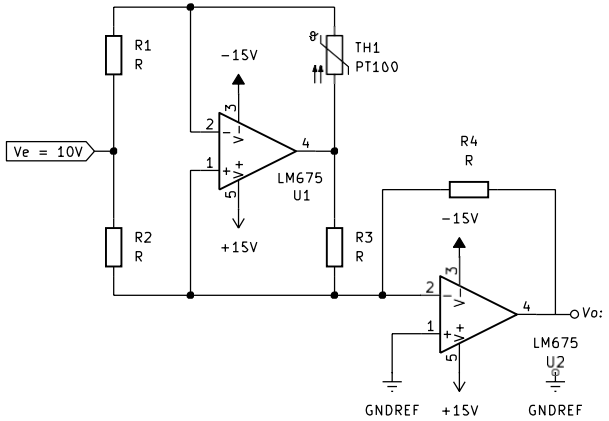
https://literature.rockwellautomation.com/idc/groups/literature/documents/rm/1756-rm003_-es-p.pdf

https://library.e.abb.com/public/6aeb483836740e11c1257b4b0052375b/3HAC032104-005_revE_es.pdf

2. Planos



PROYECTO: TITULAR: Emplazamiento: Referencia:		Fecha: xx/xx/xxxx Escala: 1:xx
Autor:	Plano: NOMBRE DEL PLANO	Plano Nº 00



$$Vo = \frac{R_1}{R} * Ve * \alpha * T$$

	Rango de temperatura de entrada	Ve	α	R	R1 (Normalizado 1%)
PT100 (1)	0 - 30 ºC	10V	0,003851 T^{-1}	100 Ω	866 Ω
PT100 (2)	0 - 100 ºC	10V	0,003851 T^{-1}	100 Ω	115 Ω

Universitat politècnica de València

Sheet: /
File: Esquema_PT100.kicad_sch

Title: Conexión PT100

Size: A4 | Date: 10/06/2024 | **Rev:**
KiCad E.D.A. 8.0.3 | Id: 1/1

3. Pliego de condiciones

En el presente pliego de condiciones se detallan las especificaciones de los materiales, las condiciones de ejecución, y las pruebas necesarias para garantizar la calidad y eficiencia del proyecto que nos demanda la empresa contratante. Este documento busca asegurar que todos los aspectos del proyecto cumplan con los más altos estándares de seguridad, eficiencia y sostenibilidad.

3.1 Objeto

El objeto del presente proyecto es la ampliación de una planta de embotellado de cerveza mediante la creación de una nueva zona dedicada a la reutilización de botellines de vidrio. Este proyecto incluye la instalación de equipos y sistemas automatizados para el manejo, lavado, inspección, llenado, coronado y pasteurización de los botellines, con el fin de optimizar la eficiencia del proceso y reducir los costes asociados. Cabe recordar el compromiso de la empresa contratante con el medio ambiente ya que no utilizan etiquetado en la botella, viene ya serigrafiado en el vidrio, por lo que se ahorraran varios procesos como el de eliminado del etiquetado antiguo y la puesta del nuevo, con lo que se ahorra tiempo, dinero, energía y materias primas.

La principal idea es conseguir que la planta sea lo más autónoma posible y que se prescinda al mínimo de la actividad humana, para poder tener la planta en funcionamiento las 24 horas del día y que sea lo más eficiente posible teniendo así el menor coste posible por cada botellín reutilizado y haciéndolo rentable para la empresa productora de cerveza.

3.2 Condiciones de los materiales

La empresa distribuidora cuenta siempre con los mismos estándares tanto de botellines como de las cajas en los que estos vienen, el botellín es el estándar 577/3740 y caben 24 botellines por caja, teniendo cada una de estas unas medidas de 400x260x300mm. Con estas medidas debemos ajustar tanto las cintas transportadoras como los grippers de los robots.

La empresa contratante quiere varios requisitos en los siguientes componentes que debemos utilizar:

- **Cintas Transportadoras**: Cintas de acero inoxidable, resistentes a la corrosión y adecuadas para el contacto con alimentos.
- **Robots de Manipulación**: Robots industriales con una precisión de +/- 0.1 mm.
- **PLCs (Controladores Lógicos Programables)**: PLCs de la marca Allen-Bradley.
- **Sensores de Proximidad**: Sensores capacitivos con una distancia de detección mínima de 10 mm, certificados para uso alimentario.
- **Cámaras de Visión Artificial**: Sistemas de inspección de alta resolución, capaces de detectar partículas de tamaño inferior a 0.5 mm y defectos estructurales en los botellines.

En cuanto a control de calidad se nos pide:

- **Certificaciones**: Todos los materiales y equipos deben cumplir con las normativas CE, ISO 9001 y las directrices de la FDA para uso alimentario.
- **Inspección de Materiales**: Inspección visual y pruebas aleatorias de resistencia mecánica y térmica en cada lote de botellines.
- **Documentación Técnica**: Provisión de certificados de conformidad, fichas técnicas y manuales de uso y mantenimiento para todos los componentes.
- **Pruebas de Aceptación**: Pruebas de aceptación en fábrica para los PLCs, HMIs, robots y sistemas de visión artificial.

3.3 Condiciones de ejecución

La instalación de los equipos se llevará a cabo siguiendo estrictamente los manuales proporcionados por los fabricantes y cumpliendo con todas las normativas de seguridad industrial. Esto incluye la correcta disposición de cada componente en su ubicación asignada dentro de la planta, asegurando su fijación y conexión adecuadas. La infraestructura eléctrica y mecánica debe estar preparada previamente para recibir los nuevos equipos, incluyendo la instalación de las cintas transportadoras, los robots de manipulación y los sensores de proximidad.

La integración de los sistemas es un paso un paso fundamental para que todo el proceso este perfectamente sincronizado, para ello, se desarrollará una arquitectura de control centralizada, donde los PLCs actúan como el cerebro del sistema, coordinando las señales de entrada y salida de todos los componentes. La programación de los PLCs y la configuración de las HMIs se realizará utilizando las herramientas GEMMA y GRAFCET, permitiendo una estructura de control eficiente y de fácil comprensión.

El cableado y las conexiones deben cumplir con los estándares de calidad y seguridad, utilizando materiales ignífugos y resistentes a la humedad. Todos los cables deben estar correctamente canalizados y etiquetados para facilitar futuras labores de mantenimiento y resolución de problemas. Además, se debe asegurar que las conexiones sean firmes y seguras para evitar fallos eléctricos o interrupciones en la comunicación de los dispositivos.

3.4 Pruebas y ajustes finales

Las pruebas iniciales son esenciales para verificar que todos los subsistemas funcionen correctamente de manera individual. Cada etapa del proceso, desde el despaletizado y lavado hasta el llenado, coronado y pasteurización, se probará en condiciones controladas para identificar y corregir cualquier problema antes de la integración total del sistema.

Una vez realizadas las pruebas iniciales, se procederá con los ajustes finos de los parámetros operativos y de control. Esto incluye la calibración de sensores, la optimización de los tiempos de ciclo y la verificación de la precisión de los robots de manipulación. Estos

ajustes aseguran que el sistema opere de manera óptima, maximizando la eficiencia y reduciendo los tiempos de inactividad.

Las pruebas de integración son la siguiente fase, donde se verifica el funcionamiento conjunto de todos los subsistemas bajo condiciones operativas reales. Se simularán diferentes escenarios de producción y posibles fallos para asegurar que el sistema puede manejar variaciones en la producción y responder adecuadamente a cualquier problema que surja. Este paso es crucial para garantizar que todas las partes del sistema estén correctamente sincronizadas y que la línea de producción pueda operar sin interrupciones.

La validación final se llevará a cabo en condiciones reales de producción, lo que implica poner en marcha el sistema completo y observar su desempeño durante un período prolongado. Esto asegurará que todos los componentes y sistemas funcionan como se espera y que los objetivos de eficiencia, calidad y seguridad se cumplen plenamente. Durante esta fase, se recopilarán datos operativos que se utilizarán para realizar cualquier ajuste adicional necesario.

4. Presupuesto

4.1 Costes de materiales

DESCRIPCIÓN	MARCA	REFERENCIA	Un.	PRECIO/Un.	TOTAL
Carrousel de lavado de botellas	Frusso	RIV40	1u	15.000€	15.000€
Carrousel de llenado de botellas	Frusso	LLRVS 30	1u	15.000€	15.000€
HMI	Allen-Bradley	2711P-T15C22D9P	1u	5.350€	5.350€
Robot ABB	Allen-Bradley	ABB IRB 4600-45/2.05	2u	60.000€	120.000€
Fuente de alimentación PLC	Allen-Bradley	176-PA75	1u	280€	280€
Backplane PLC	Allen-Bradley	1756-A10	1u	550€	550€
Controlador	Allen-Bradley	1756-L83ES	1u	7.000€	7.000€
Módulo de entradas digitales	Allen-Bradley	1756-IV16	1u	570€	570€
Módulo de salidas digitales	Allen-Bradley	1756-OB16I	2u	670€	1.340€
Módulo de entradas analógicas	Allen-Bradley	1756-IF6I	1u	1.200€	1.200€
Módulo de entradas de seguridad	Allen-Bradley	1756-IB16S	1u	1.200€	1.200€
Bloques de conexión	Allen-Bradley	1756-TBNH	5u	90€	360€
Sensor de ultrasonidos	pepperl+Fuchs,	UBE1000-18GM40A-SE2-V1	8u	230€	1.840€
Sensor de temperatura Pt100	Jumo	902020/10	2u	40€	80€
Setas de emergencia	Schneider	ZB4BS844	3u	27,29€	81,87€
Sensor puerta	ifm	MN508S - MN44008-AKOA/-H/US/8P	3u	120€	360€
Camara VA	Bitmakers	XG-7000	1u	10.000€	10.000€
Cintas transportadoras	Intralox	Series 900 Flat Top	12m	1500€/m	18.00€

El coste total en materiales asciende a 198.211,87€.

4.2 Coste de sofware

PROGRAMA	HORAS DE USO	PRECIO ANUAL	TOTAL
Studio 5000 Logix designer	50h	3.000€	74,70€
FactoryTalk View Machine Edition	20h	1.500€	14,94€
RobotStudio	50h	2.500€	49,80€
Autocad	25h	1.900€	23,66€

El cálculo de amortización de los softwares se hace teniendo en cuenta que en el presente año 2024 hay un total de 251 días de lunes a viernes que son laborables en los que se trabaja y por tanto se utilizan los sofwares 8 horas al dia y de ahí se extrae el coste por hora de cada sofware.

Así pues, el coste del proyecto en sofware es de 163,1€.

4.3 Coste de la mano de obra

PUESTO DEL TRABAJADOR	HORAS DE USO	PRECIO HORA	TOTAL
Ingeniero programador PLC y HMI	100h	33€/h	3.300€
Técnico programador de robots	80h	25€/h	2.000€
Ingeniero de proyectos	50h	33€/h	1.650€
Técnico montador	200h	25€/h	5.000€

Las horas de los trabajadores se tienen en cuenta tanto en oficina, llevando a cabo la programación de PLC, Robot y HMI entre los técnicos y el ingeniero programado y los planos de la planta tanto eléctricos como de construcción y la negociación con los proveedores es llevada a cabo por el ingeniero de proyectos. En cuanto a las horas dedicadas por los técnicos que se dedican al montaje de la practica la totalidad de sus horas trabajadas en industria.

El coste total de la mano de obra asciende a 11.950€.

4.4 Gastos totales

CONCEPTO	COSTE TOTAL
1. Materiales	198.211,87€
2. Sofware	163,1€
3. Mano de obra	11.950€
Presupuesto de ejecución	201.324,97€
Beneficio industrial (6%)	12.079,50€
Presupuesto de inversión	213.404.47€
I.V.A (21%)	44.814,94€
Presupuesto final	258.219,41€

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ETSI Aeroespacial y Diseño Industrial

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela técnica superior de ingeniería del diseño

ANEXO I: MANUALES DE SOFTWARE

Trabajo Fin de Grado

Ingeniería electrónica industrial y automática

1. Studio 5000

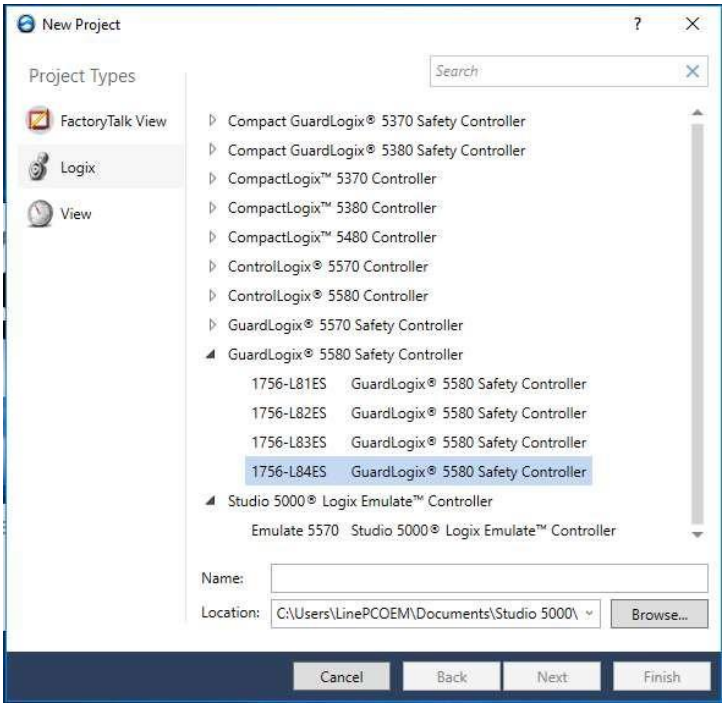
Para la programación del PLC hemos utilizado el sofware que proporciona la marca rockwell, de la que es el PLC que usaremos el Studio 5000, respecto a sus demás competidores como lo pueden ser siemens, con su software TIA portal o Omron, con su sofware CX-Programmer, no encontramos muchas diferencias en lo que a la interfaz y la forma de programar se refiere, donde si encontramos una gran ventaja es a la hora de realizar modificaciones en línea, ya que mientras la maquinaria está en marcha, nos podemos conectar de forma online al PLC que está controlando el proceso que queremos modificar y añadir o cambiar condiciones sin parar ningún proceso.

1.1 Creación de un programa

Para crear un proyecto desde cero es sencillo, solo tenemos que darle doble clic a nuestra aplicación de studio 5000 y se nos abrirá una ventana como la que tenemos a continuación:

En esta ventana podemos crear nuevos proyectos, abrir alguno que ya tengamos generado anteriormente, importar o abrir algún ejemplo.

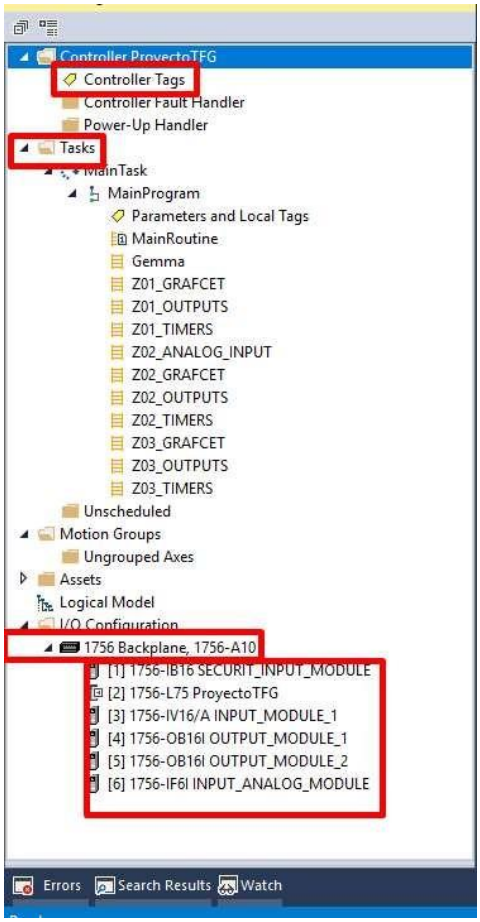
Una vez hemos creado el nuevo proyecto en la opción new Project tenemos que elegir el tipo de controlador que queremos utilizar tal y como aparece en la siguiente imagen:



Podemos elegir o bien un controlador real de los que aparecen en el catálogo de rockwell o bien elegir el que nos dice Studio 5000 logix emulate controller, con el que podemos hacer una simulación en nuestro ordenador para comprobar que todo funcione como queremos y que luego veremos cómo realizar.

1.2 Organización del proyecto

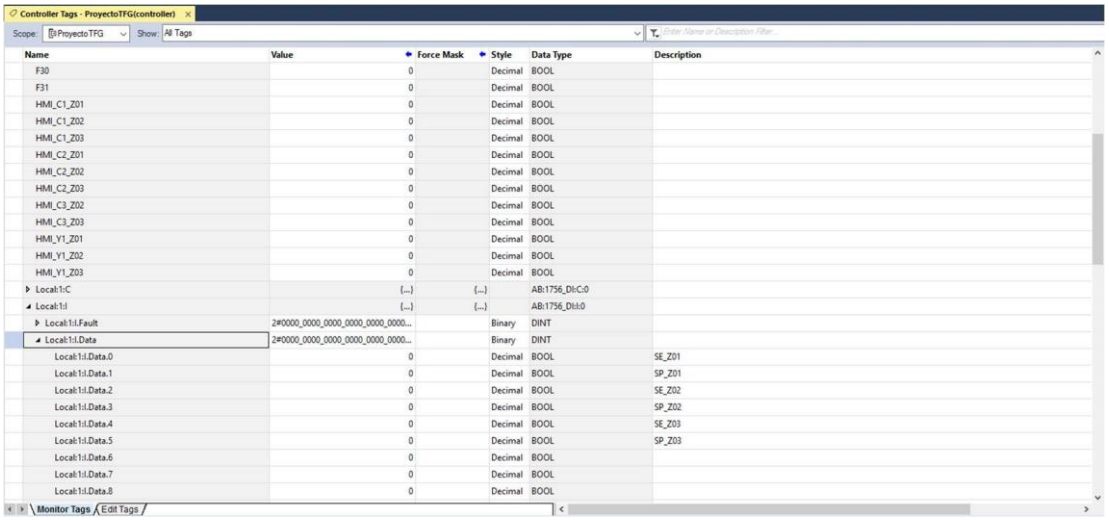
Una vez ya seleccionado el controlador y tenemos que fijarnos en el organizador del proyecto, que es el menú que nos aparece a la izquierda de la pantalla y que se ve de la siguiente manera:



Debemos quedarnos con las partes señaladas en rojo, que serán fundamentales para programar y entender nuestro proyecto:

- **Controller tags:** es un listado en el que se almacenan todas las variables que son comunes a todos los programas que almacena nuestro PLC, pueden ser físicas (entradas o salidas) o de programa, puede ser mas de uno, aunque nosotros en este caso solo tenemos el Main Program.
- **Tasks:** En esta carpeta se almacenan todos los programas que vayamos a desarrollar en nuestro controlador, como vemos dentro de cada programa aparece una nueva pestaña de parameters y local tags, que son exclusivos de ese programa en concreto, luego tenemos las rutinas y subrutinas del programa, donde debemos tener en cuenta que solo se ejecuta la MainRoutine.
- **I/O Configuration:** Aquí es donde debemos añadir los módulos que podemos utilizar, el backplane viene con el controlador que elijamos y luego los módulos de salida los elegimos según queramos, el numero previo al nombre de los dispositivos es su posición dentro del backplane.

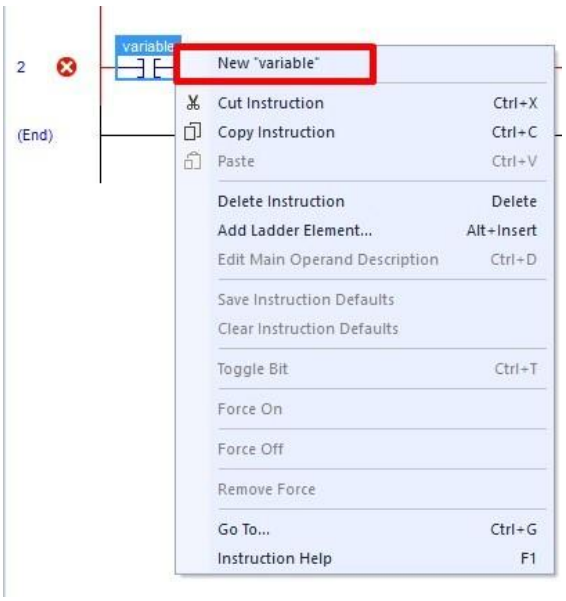
Dentro de la parte de los controller tags, que como hemos dicho es donde se almacenan todas las variables de uso global de nuestro controlador, tenemos la siguiente vista:



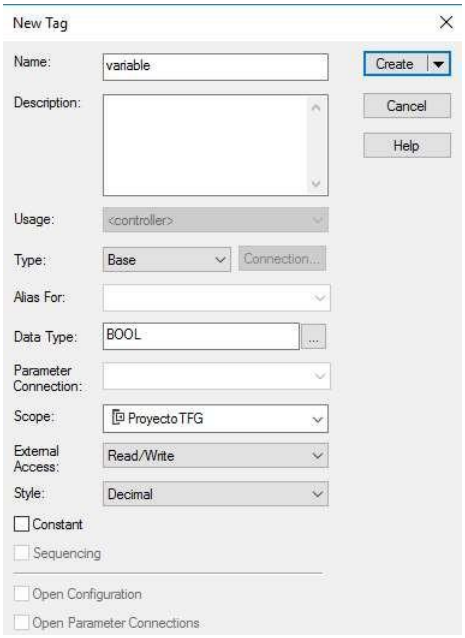
Tenemos todo el listado de las variables, con su nombre, su valor actual, que variara cuando estemos en simulación o conectados al PLC, el style, que es la base en la que se representa el valor, el data type, que nos dice el tipo de variable y la descripción donde ponemos lo que queramos que se lea para entender de forma más sencilla nuestro programa.

1.3 Variables

Para crear una variable lo hacemos de la siguiente manera:



El elemento que vayamos a utilizar, en este caso un contacto abierto, pero podría ser un timer, un mov, una salida, etc, lo referenciamos a una variable, para crearla y añadirla al controler tags tenemos que darle clic derecho y new, ahí nos aparecerá el siguiente dialogo:

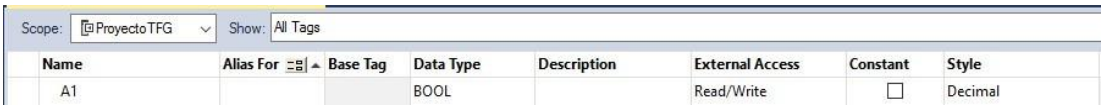


Aquí es donde rellenamos los datos que hemos mencionado antes:

- **Name:** el nombre que queramos para esa variable.
- **Description:** la descripción que queramos para la variable.
- **Type:** tenemos el tipo base, que son las marcas o tags creados por nuestro programa y usados por nuestro programa, los consumed, que son variables utilizadas por nuestro programa pero que vienen de otro PLC y los produced que son producidos por nuestro programa y consumido por otro.
- **Scope:** donde seleccionamos si queremos que sea una variable global del proyecto o especifica del programa en el que estamos.

Vamos a ver varios ejemplos de variables que hemos usado en el proyecto:

- **Variable no física:**



- **Variable física, entrada digital:**

Los tags que se generan por el modulo de entradas digitales son los siguientes:

◢ Local:1:I			AB:1756_DI:I:0		Read/Write	☐	
▷ Local:1:I.Fault			DINT		Read/Write		Binary
◢ Local:1:I.Data			DINT		Read/Write		Binary
Local:1:I.Data.0			BOOL	SE_Z01	Read/Write		Decimal
Local:1:I.Data.1			BOOL	SP_Z01	Read/Write		Decimal
Local:1:I.Data.2			BOOL	SE_Z02	Read/Write		Decimal
Local:1:I.Data.3			BOOL	SP_Z02	Read/Write		Decimal
Local:1:I.Data.4			BOOL	SE_Z03	Read/Write		Decimal
Local:1:I.Data.5			BOOL	SP_Z03	Read/Write		Decimal

Solo utilizamos los tags Local:1:I:Data, el numero 1 se refiere al slot que ocupa el modulo dentro del backplane, la I se refiere a inputs y el Data a las salidas físicas como tal.

- **Variable física, salida digital:**

▷ Local:5:C			AB:1756_DO:C:0		Read/Write	☐
▷ Local:5:I			AB:1756_DO:I:0		Read/Write	☐
▷ Local:5:O			AB:1756_DO:O:0		Read/Write	☐

En el caso de los módulos de salidas, utilizaremos los Local:5:O, que son los referidos a las salidas.

- **Variables timers:**

◢ T1_Z02			TIMER		Read/Write	☐	
▷ T1_Z02.PRE			DINT		Read/Write		Decimal
▷ T1_Z02.ACC			DINT		Read/Write		Decimal
T1_Z02.EN			BOOL		Read/Write		Decimal
T1_Z02.TT			BOOL		Read/Write		Decimal
T1_Z02.DN			BOOL		Read/Write		Decimal

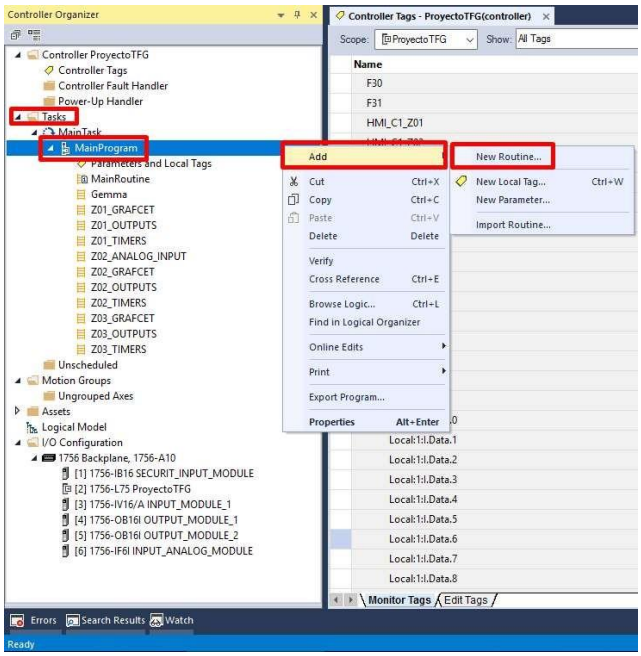
Los timers se crean con la variable tipo TIMER y tienen las siguientes marcas:

- **PRE:** donde se almacena la cantidad de milisegundos que debe esperar o actuar el temporizador, de tipo doble entero.
- **ACC:** donde se almacena la cuenta actual del temporizador, de tipo doble entero.
- **EN:** bit que nos habilita el temporizador.
- **TT:** bit que nos indica que el tiempo está en marcha.
- **DN:** bit que nos indica que el temporizador ya ha pasado del tiempo marcado.

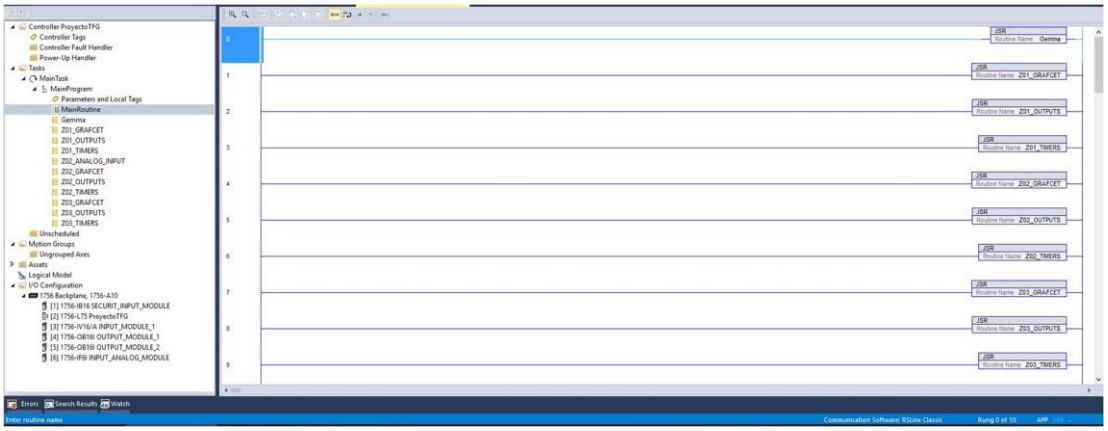
1.4 Rutinas y subrutinas

Como hemos visto ya en el organizador del programa tenemos dentro del programa una rutina llamada MainRoutine, que es la única que se ejecuta, todas las demás son subrutinas, divididas entre el grafcet, salidas, timers y entradas analógicas.

Para crear una subrutina el proceso es el siguiente:

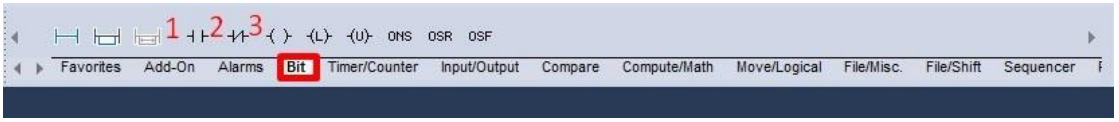


Y para que las subrutinas que hemos creado se ejecuten debemos incluirlas en el main de la siguiente manera con la función JSB (Jum to subroutine):



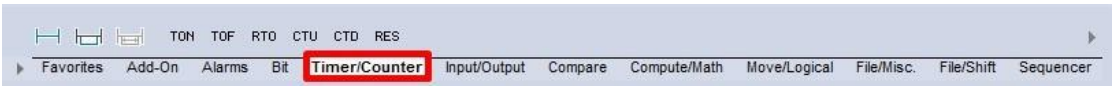
1.5 Elementos utilizados

- Elementos bit:



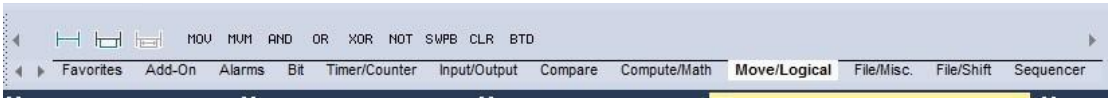
- ➔ El elemento numero 1 dentro del apartado de bit, es el contacto normalmente abierto, siempre ira en función del estado de la variable
- ➔ El elemento número 2, es el contacto negado nos enviará la señal negada que tenga la variable en ese momento.
- ➔ El elemento número 3, es la activación de la señal, el valor de la señal no se mantiene en el tiempo, es decir si mandamos un 1 y dejamos de mandarlo la señal estará a 0, para eso utilizamos los siguientes elementos latch y unlatch para salidas con enclavamiento.

• Elementos timer:



En este caso hemos utilizado los temporizadores a la conexión (TON), que se activaran cuando les llegue un 1 lógico durante el tiempo marcado y los temporizadores a la desconexión (TOF), que se activaran cuando les deje de llegar un 1 lógico y se habilitara su salida durante el tiempo marcado.

• Elementos move/Logical:

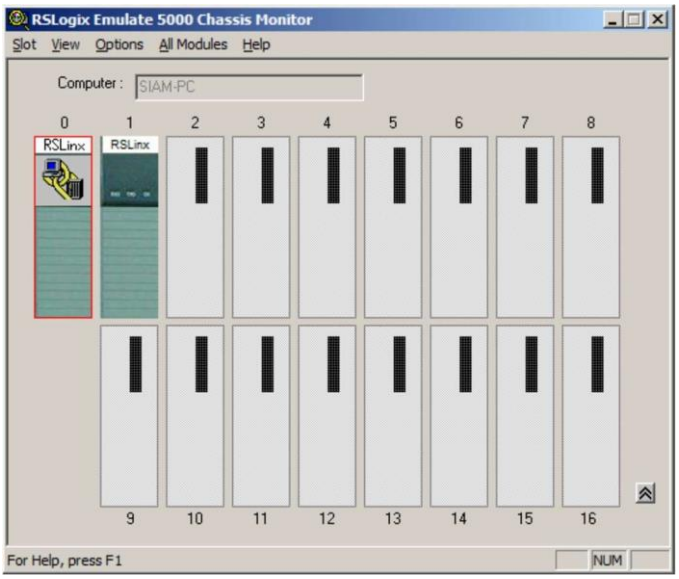


En este apartado tan solo hemos utilizado el operador Move, utilizado para mandar valores de un lugar a otro.

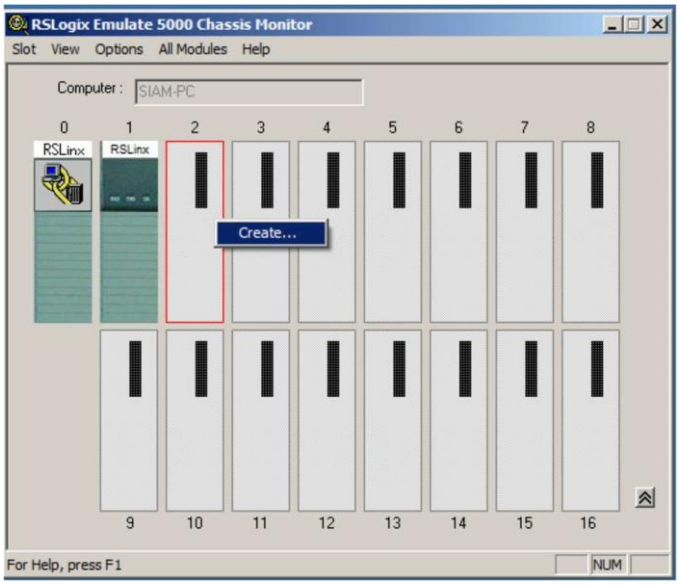
2. RSLogix Emulate 5000

RSLogix emulate 5000 es un programa que nos permite a través de un controlador virtual poder simular la programación y poder visualizar las entradas y salidas con las que estamos jugando.

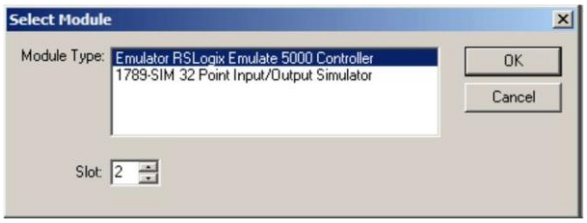
Para poder simular nuestro programa lo primero que debemos hacer es abrir el programa RSLogix emulate y crear nuestro PLC, debemos añadir tantos controladores y tantos módulos E/S como tengamos en nuestro programa de Studio 5000 y cada modulo debe ir en el slot correcto.



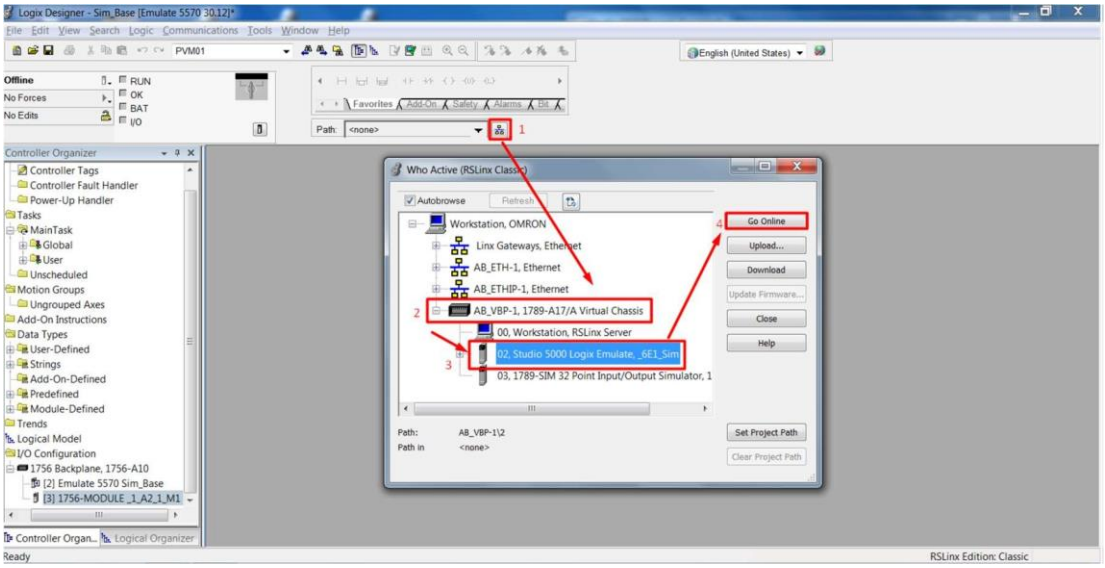
Una vez abierto el programa para añadir los distintos moduos solo debemos hacer clic derecho en el slot que queramos y nos saldrá la opción de crear.



Tenemos opciones un tanto limitadas, ya que en cuanto a módulos E/S se refiere solo podemos añadir del tipo 1789.

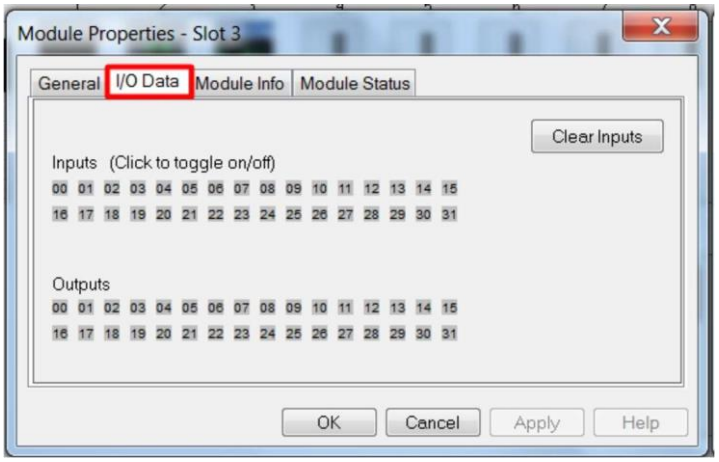


Para iniciar la simulación tendremos que irnos al studio 5000 sin cerrar el programa del emulador y seguir las siguientes instrucciones:



A partir de ahí solo tendremos que descargar el programa en el PLC y activar el modo run.

Posteriormente con el emulador podemos ver y controlar las E/S que hayamos creado.

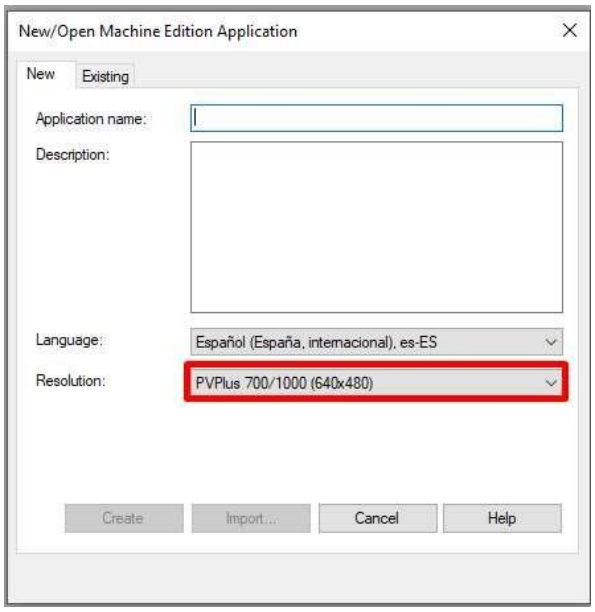


3. FactoryTalk View Studio

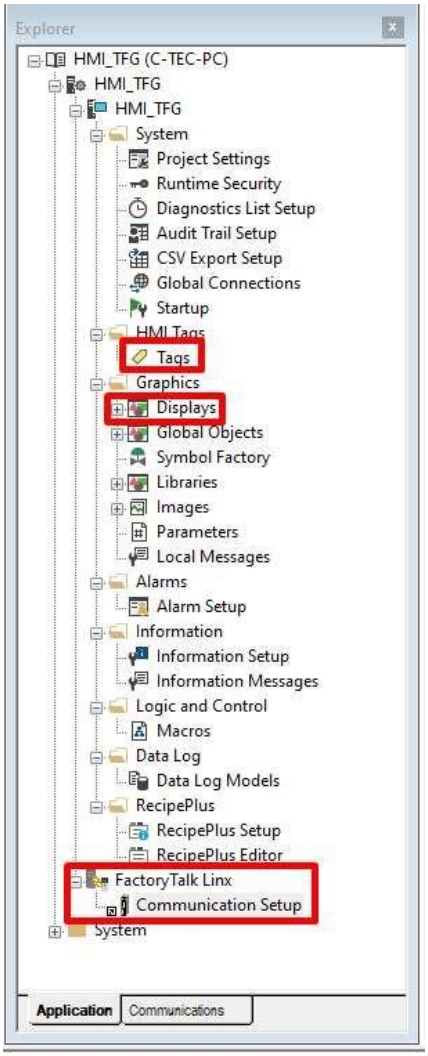
FactoryTalk View Studio es el entorno de diseño para FactoryTalk View que brinda las herramientas necesarias para desarrollar y probar aplicaciones de interfaz humano-máquina (HMI) distribuidas en red, estaciones de red y estaciones locales. En este entorno, encontrarás editores que te permitirán crear aplicaciones completas de manera eficiente.

Con FactoryTalk View Studio, puedes diseñar interfaces de usuario intuitivas y efectivas para controlar y monitorear sistemas industriales. Además de ofrecer la creación de aplicaciones, también incluye software de cliente y servidor para probar y validar tus diseños antes de su implementación en el entorno de producción.

Cuando abrimos la aplicación lo primero que nos pide es si queremos generar un nuevo programa o cargar uno ya existente y lo único que debemos seleccionar bien a la hora de crear uno nuevo es la resolución del HMI físico en el que queremos que vaya el programa.



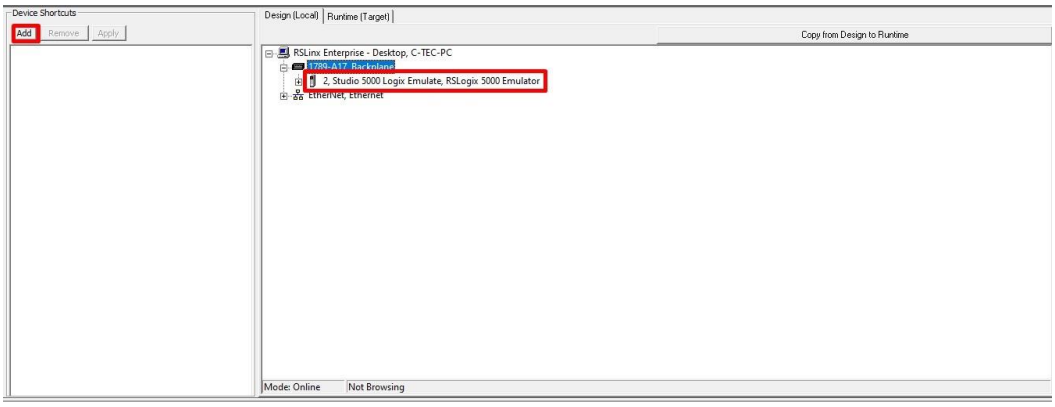
Igual que pasa con el Studio 5000 en factory talk view tambien tenemos un cuadro de dialogo que nos permite manejarnos por todas las herramientas de la aplicación.



Los menús más importantes son los de Tags, donde igual que en el PLC podemos ver todos los tags que hemos creado con las variables del HMI, el displays, donde podemos ver todas las distintas pantallas de navegación que hemos creado y el factory talk linx, para vincular nuestro PLC y nuestro HMI. Otra función interesante que nos aporta esta aplicación es que en el desplegable de Libraries contamos con muchas imágenes que podemos utilizar para hacer la recreación de nuestro proceso.

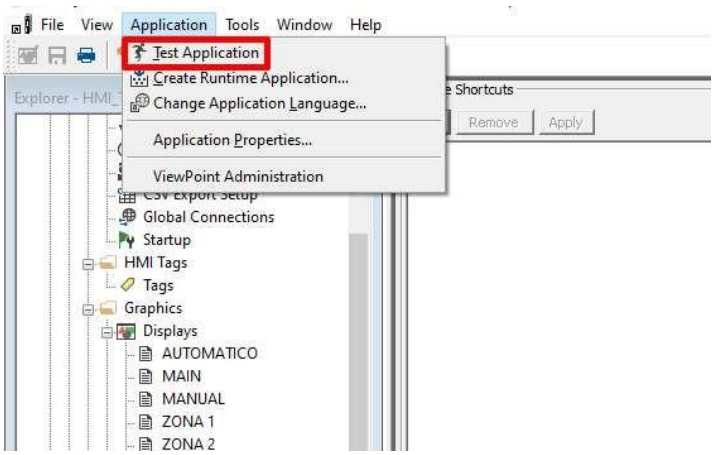
Aquí vemos como se muestra la zona de los displays, donde para añadir uno nuevo solo tenemos que dar clic derecho en el desplegable y añadir uno.

Para vincular el HMI con el PLC, como hemos dicho anteriormente tenemos que irnos al desplegable Factory talk linx → comunication set up, una vez en este menú veremos la siguiente imagen:

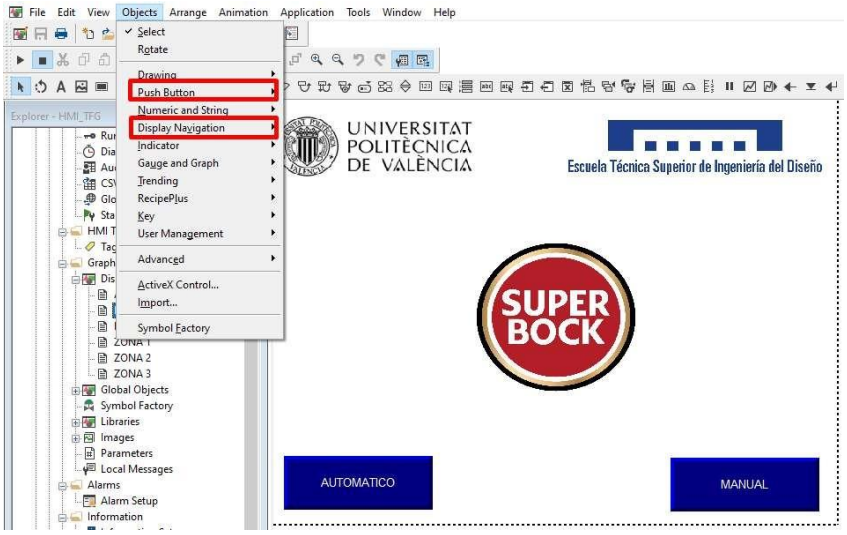


Seleccionamos nuestro PLC, en este caso el emulate y le damos a add, podemos guardarlo con el nombre que queramos y una vez realizada por primera vez la conexión ya se nos queda guardada.

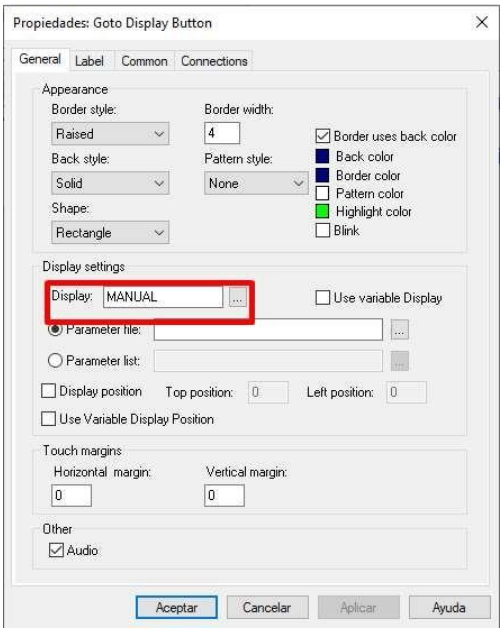
Con ambas aplicaciones ya conectadas para iniciar la simulación del HMI solo tenemos que ir a la barra superior izquierda, application → Test application.



Los elementos del HMI que hemos utilizado son principalmente botones, para utilizar botones como entradas en nuestro PLC, debemos seleccionar dentro de objects → pushbuttons y en nuestro caso momentary, es decir, que actuaran como pulsadores. Y los botnes para movernos por las distintas pantallas son los display navigation → go to.

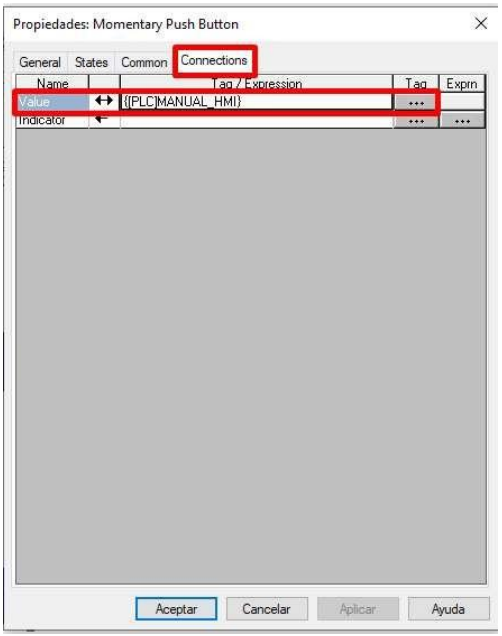


La configuración de los botones Goto, será la siguiente:



Seleccionamos en display a que pantalla queremos ir cuando pulsemos y en label añadimos el nombre que queremos que se muestre en el botón.

La configuración de los momentary push button será la siguiente:

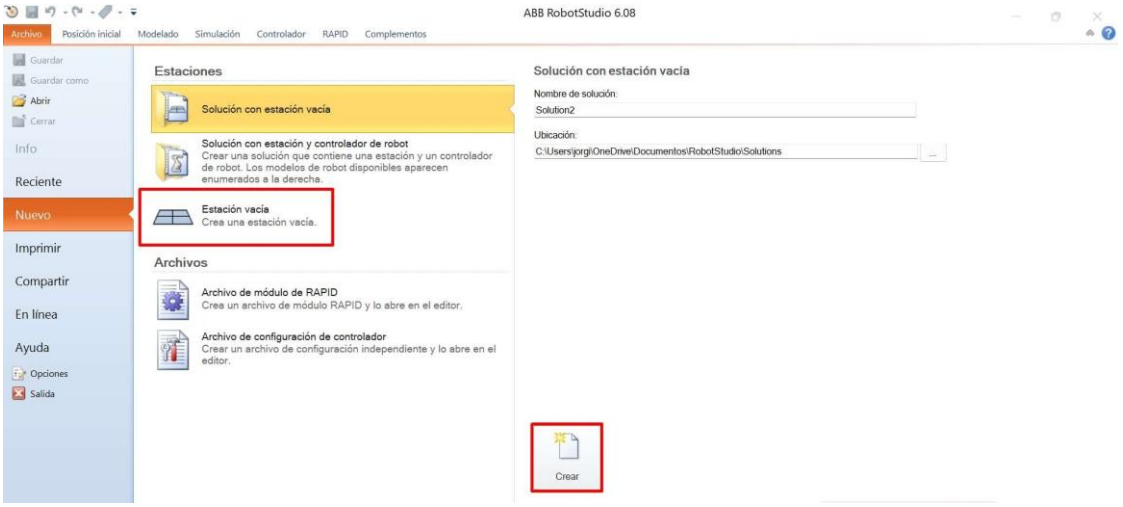


En value dentro de connections debemos referenciar la entrada a la que va conectada dentro del programa de nuestro PLC, primero poner el nombre que hemos utilizado para la conexión con nuestro PLC, en nuestro caso PLC y luego añadir el mismo nombre que tiene dicho tag en el programa de Studio 5000.

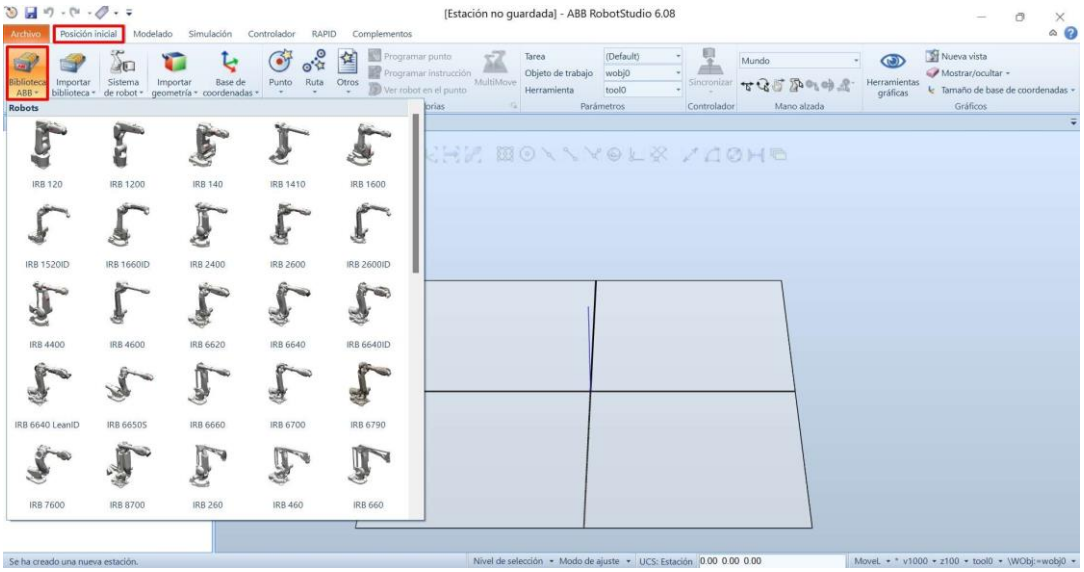
4. RobotStudio

El programa de RobotStudio es el software de ABB con el que se puede diseñar programar y recrear robots industriales y sus elementos dentro de una planta de producción. Nos permite recrear señales físicas y también dar movimientos al resto de elementos que compongan la zona donde va a desarrollar la actividad el robot, ya sea una cinta transportadora, el producto en si o cualquier elemento que pueda haber.

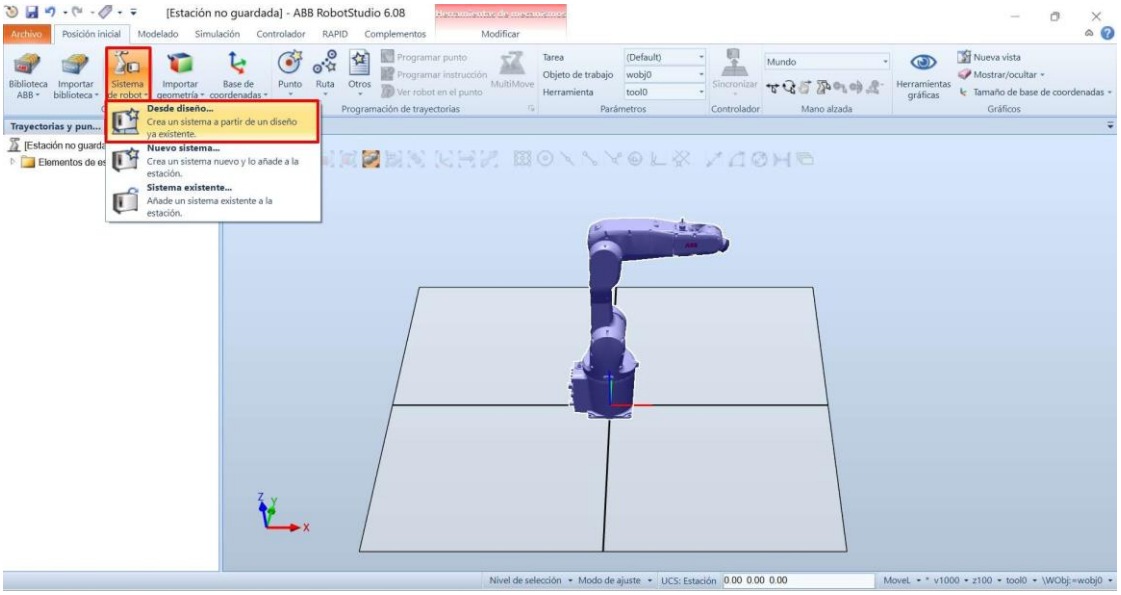
- Iniciar un programa



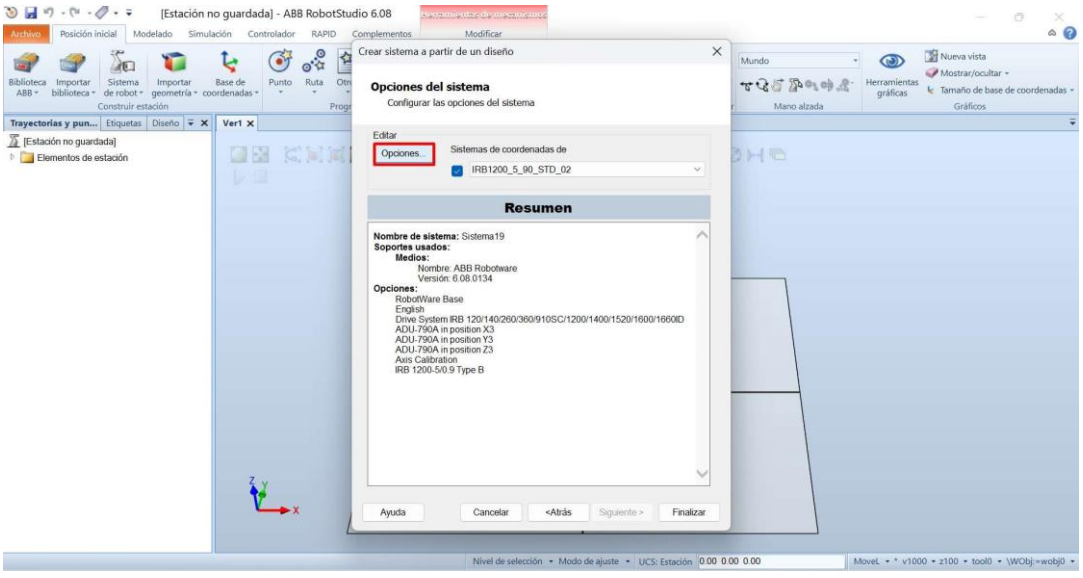
- Elección de robot



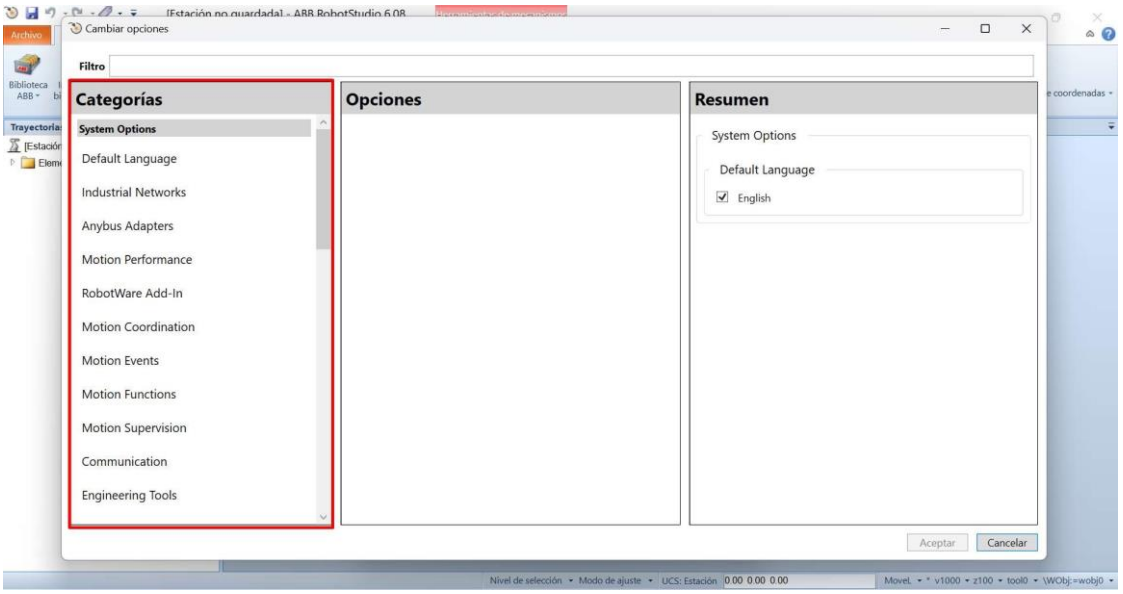
- Elección del sistema del robot



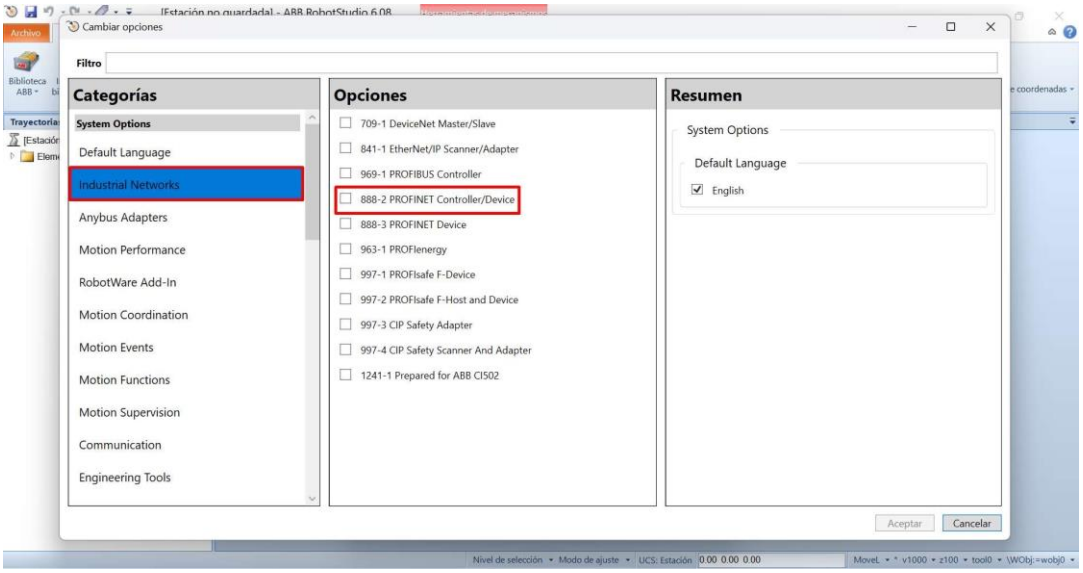
El sistema del robot es un paso fundamental para poder empezar a programar los movimientos del robot, lo elegimos con la opción “desde diseño”, en la que el programa ya nos da el controlador del robot que toca.



Una vez le damos a siguiente paso tenemos la opción de añadir distintas herramientas y periféricos:

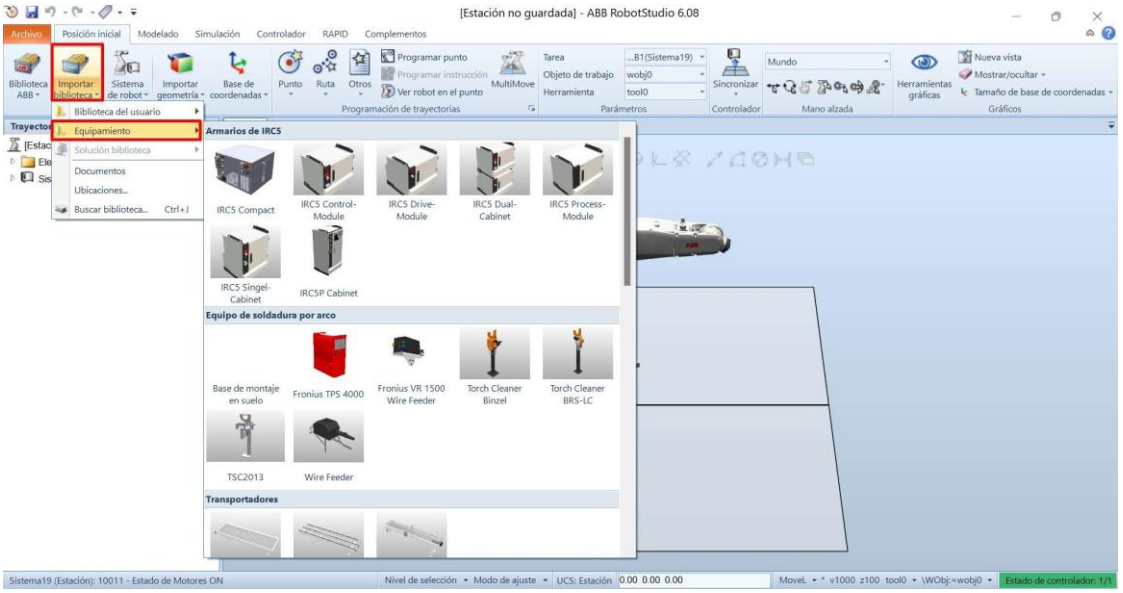


Para el proyecto hemos elegido siempre un periférico dentro del menú “industrial networks” la opción “888-2 PROFINET Controller/device” con la que podemos añadir distintas entradas y salidas para simular.

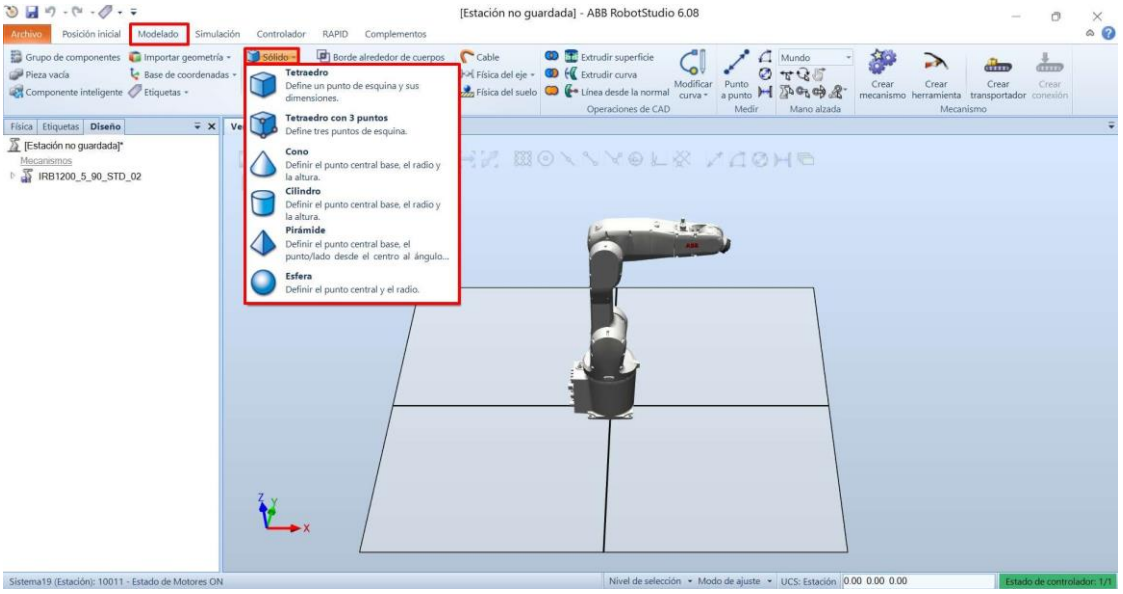


- Añadir componentes

ABB nos proporciona bibliotecas con herramientas o elementos ya creados para simular los distintos componentes de la estación que vayamos a simular. En el desplegable de "importar biblioteca >> equipamiento" podemos elegir desde los armarios de los controladores de los robots, hasta distintos equipos de soldadura, varias cintas transportadoras y algún gripper para utilizar con los robots.

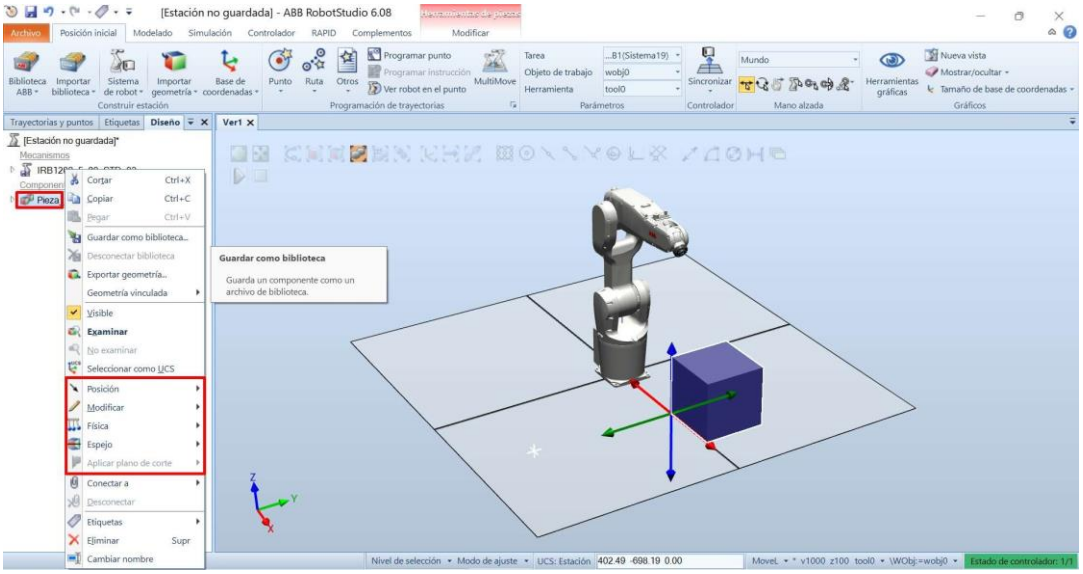


También podemos recrear nosotros cualquier objeto dentro del menú "modelado" en el desplegable "solido" y nos da distintas opciones para hacer formas y poder montar objetos que deseemos recrear.



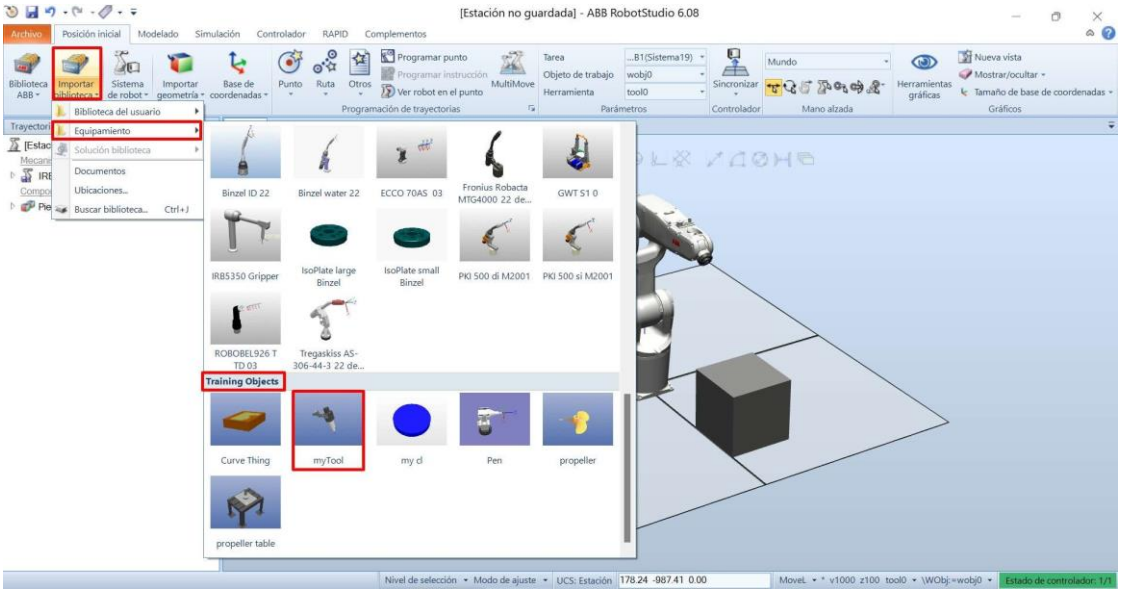
Una vez ya creado tenemos varias opciones para aplicar sobre el objeto que acabamos de crear pulsando clic en el botón derecho del ratón:

- Posición → para mover el objeto por toda la estación.
- Modificar → para modificar distintos aspectos visuales de la geometría.
- Física → para cambiar la condición física del objeto, si queremos que sea solido que se deforme o otras opciones que influirán en la simulación a la hora de que ese objeto se mueva o sea cogido por el robot.
- Espejo → para darle la vuelta la forma de la geometría.
- Conectar a → donde podemos unir distintos objetos creados o unirlo al robot o al gripper para simular que ha sido cogido.

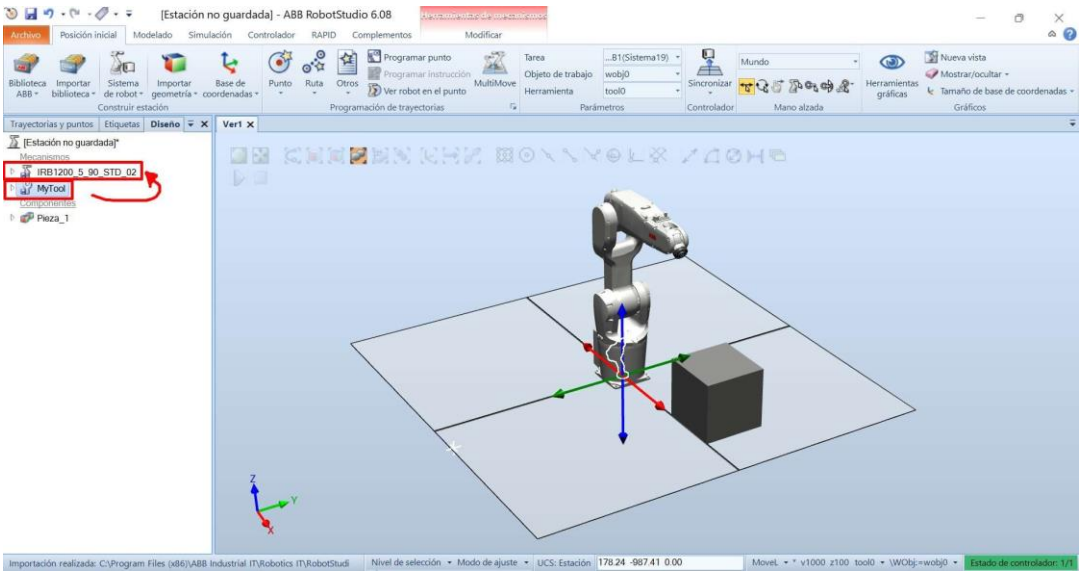


- Unión y elección del gripper

Tenemos como se ha comentado antes la opción de elegir un gripper ya creado dentro del equipamiento que nos viene dado por ABB, hay varias marcas que nos proporcionan la geometría ya creada y lista para usar en RobotStudio pero en caso de que no se puede hacer como hemos hecho en el proyecto, podemos montar nuestro propio gripper creando nosotros la forma de a geometría y le podemos dar incluso movimiento.



Para unir el gripper al robot lo único que tenemos que hacer es arrastrar la herramienta sobre el robot:



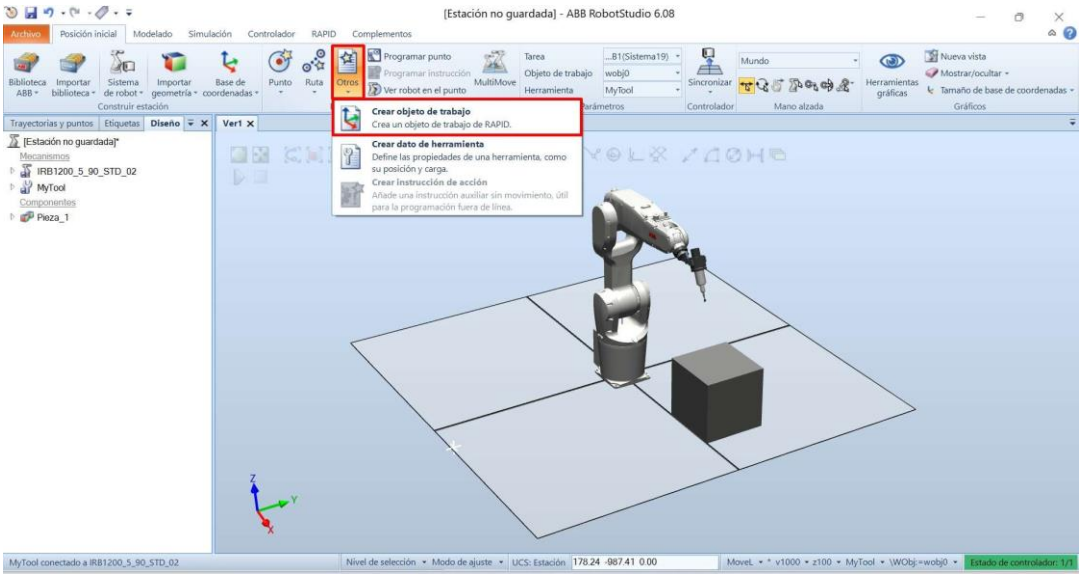
- Configuración de los movimientos del robot

Un WorkObject es un marco de referencia que facilita la programación y operación del robot. Define la posición y orientación de un objeto de trabajo en el espacio tridimensional y es crucial para mantener la precisión y consistencia de las operaciones del robot.

Si se mueve el objeto sobre el que se va a trabajar, como una cinta transportadora, una pieza para soldar, o cualquier otro componente, no es necesario reprogramar todos los puntos de movimiento del robot. Solo se necesita ajustar el WorkObject a la nueva posición del objeto, y todos los puntos de referencia se actualizarán automáticamente.

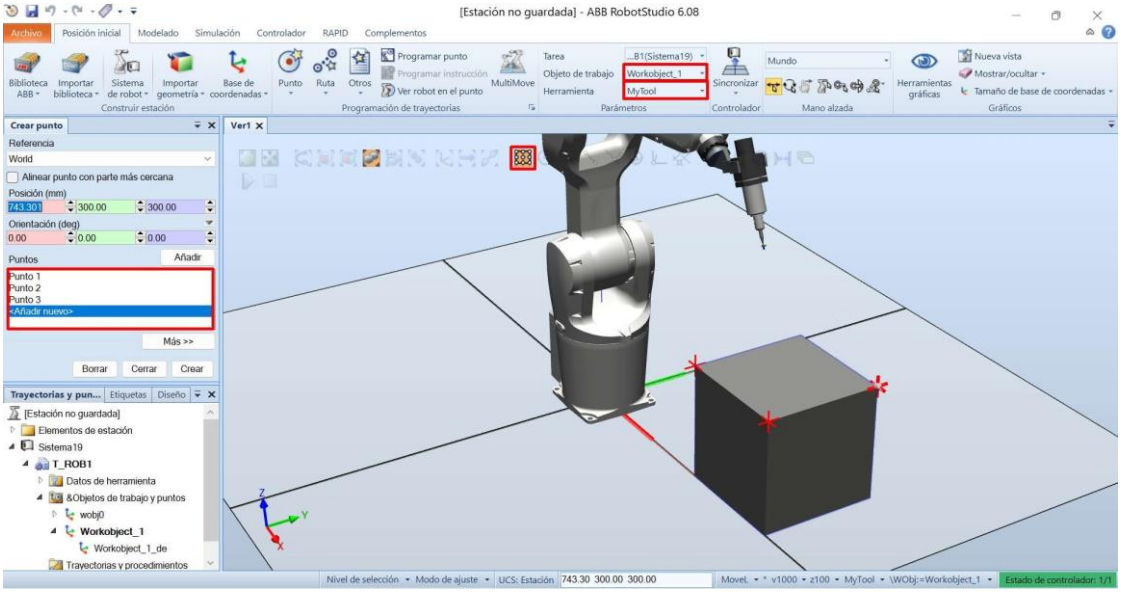
Procedimiento para Crear un WorkObject “otros >> crear objeto de trabajo” y configurar los siguientes pasos:

- Definición del Punto de Origen: Establece el punto de origen del WorkObject en la posición inicial del objeto de trabajo.
- Configuración de la Orientación: Define la orientación de los ejes X, Y, y Z del WorkObject para alinearse con la orientación del objeto de trabajo.
- Grabación de Puntos de Referencia: Una vez definido el WorkObject, se graban todos los puntos de movimiento del robot con respecto a este sistema de coordenadas.

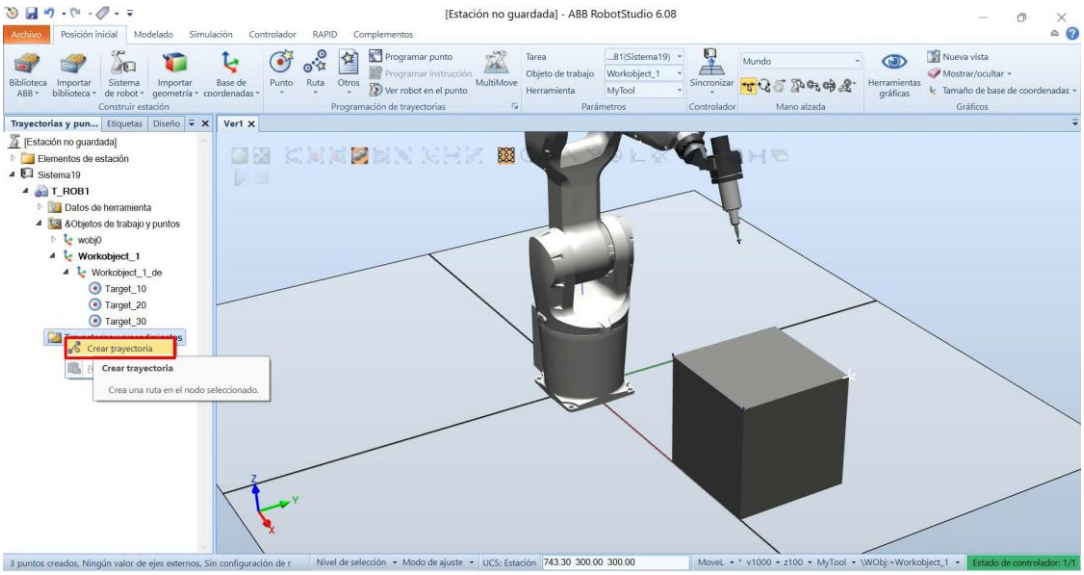


Y una vez el objeto de trabajo ya creado podemos pasar a elegir los puntos que van a conformar la trayectoria que queremos que siga el robot, siempre asegurándonos que trabajamos sobre nuestro WorkObject y con la herramienta que tiene.

Hay herramientas, como la marcada en rojo que nos permite colocar el punto justo sobre las esquinas y los centros de las geometrías para tener más precisión, pero si nos sabemos justo las coordenadas del punto también las podemos colocar.

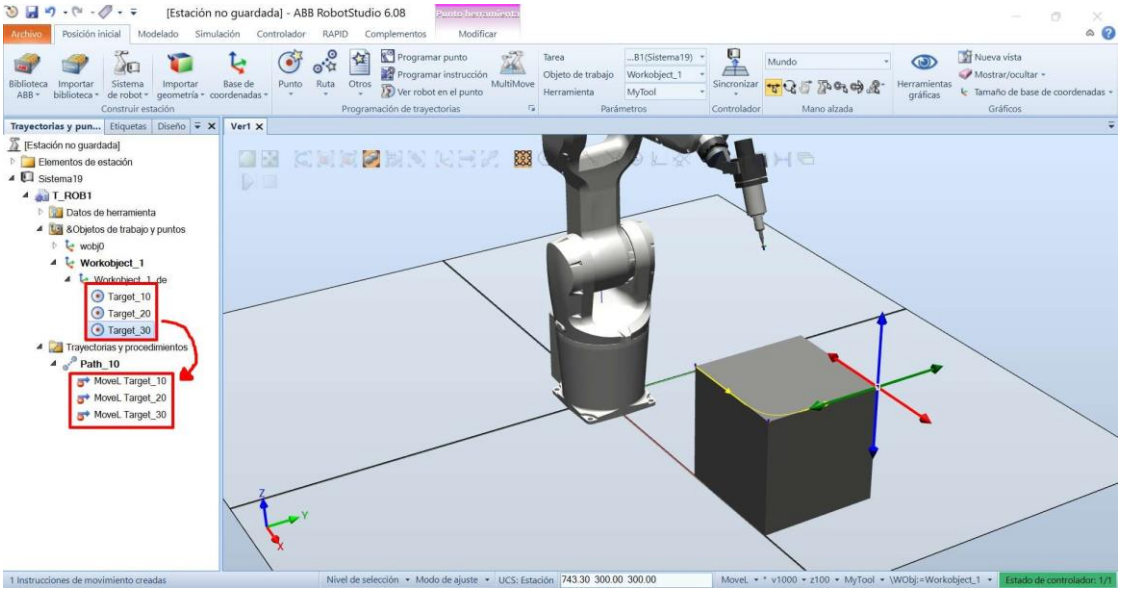


Una vez los puntos ya creados debemos de crear una trayectoria:



En esta trayectoria lo que haremos será arrastrar los puntos sobre los que queremos que se mueva robot y también el orden, se pueden repetir puntos dentro de la trayectoria, solo tenemos que arrastrarlos hacia ella.

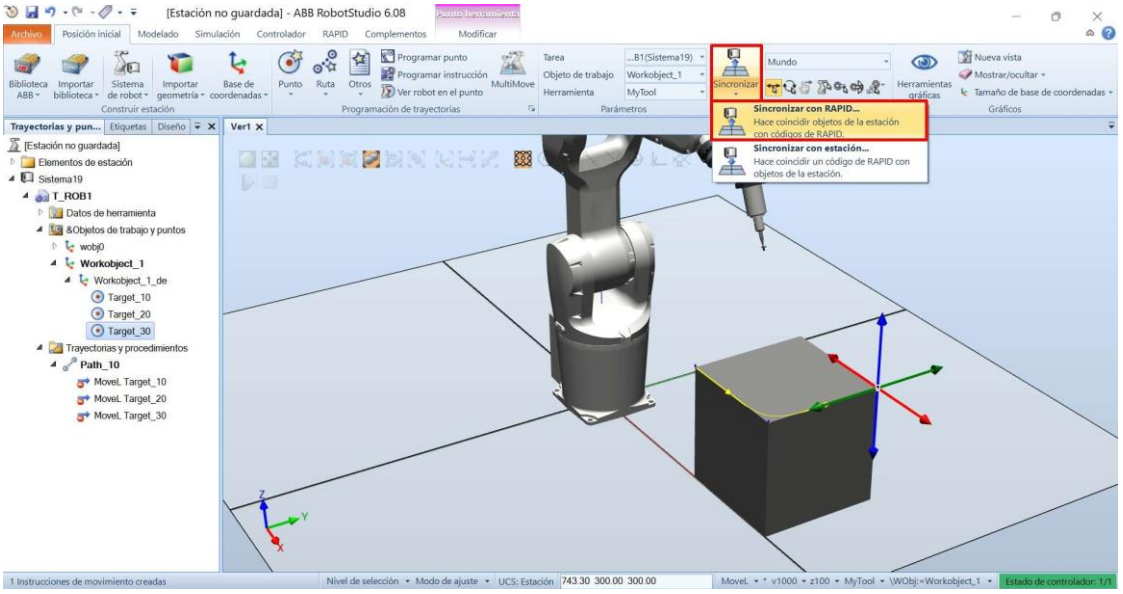
Podemos seleccionar si el movimiento es lineal con MoveL o un movimiento que puede o no ser lineal, pero es el más eficiente para el robot como es MoveJ. También podemos modificar velocidad y precisión al punto dándole clic derecho sobre el movimiento.



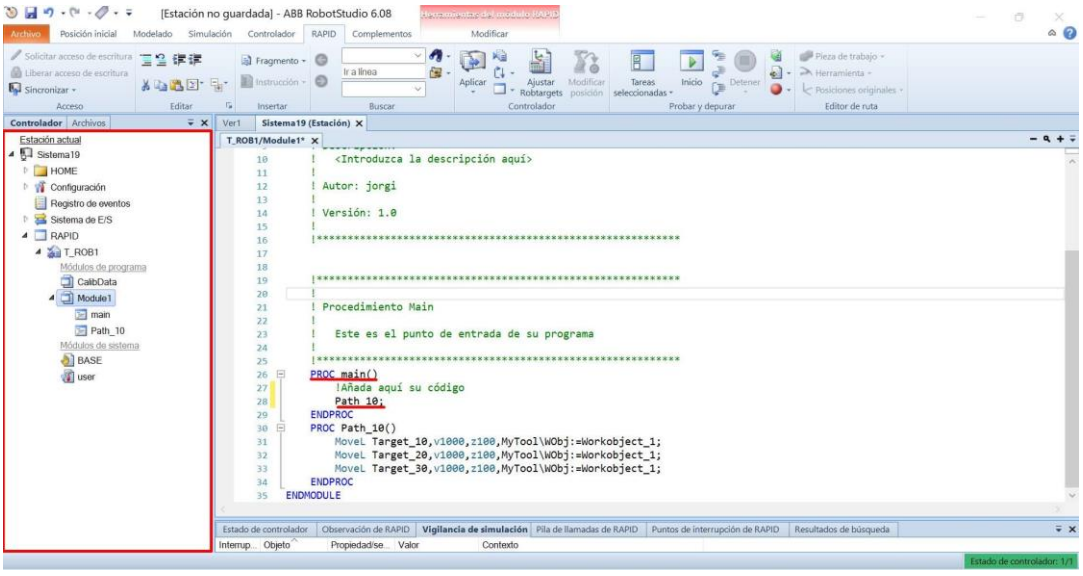
Una vez ya creada la trayectoria, si queremos aplicar alguna entrada o alguna salida o realizar modificaciones sobre los movimientos debemos hacerlo mediante el lenguaje de programación que nos proporciona ABB, RAPID.

RAPID es un lenguaje de programación al estilo C que nos permite crear estructuras de control, tipos de datos, modificar las instrucciones de movimiento, controlar las entradas y salidas y modificar datos.

Para emplearlo debemos de sincronizar la estación con RAPID de esta manera:



Una vez dentro este es el entorno de programación que nos ofrece, nuestro Path_10 es la trayectoria creada y para iniciar la simulación debemos de introducirla siempre dentro del main().



UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ETSI Aeroespacial y Diseño Industrial

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela técnica superior de ingeniería del diseño

ANEXO II: PROGRAMACIÓN DEL SOFTWARE

Trabajo Fin de Grado

Ingeniería electrónica industrial y automática

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela técnica superior de ingeniería del diseño

1. PLC

Trabajo Fin de Grado

Ingeniería electrónica industrial y automática

- **Controller TFG_Simulacion**
- **Controller Fault Handler**
- **Power-Up Handler**

Tasks

- **MainTask**
  - **MainProgram**
    - **MainRoutine**
    - **Gemma**
    - **Z01_GRAFCET**
    - **Z01_MESSAGE_STATES**
    - **Z01_OUTPUTS**
    - **Z01_TIMERS**
    - **Z02_ANALOG_INPUT**
    - **Z02_GRAFCET**
    - **Z02_MESSAGE_STATES**
    - **Z02_OUTPUTS**
    - **Z02_TIMERS**
    - **Z03_ANALOG_INPUT**
    - **Z03_GRAFCET**
    - **Z03_MESSAGE_STATES**
    - **Z03_OUTPUTS**
    - **Z03_TIMERS**
- **Unscheduled**

Motion Groups

- **Ungrouped Axes**

Add-On Instructions

Data Types

- **User-Defined**
- **Strings**
- **Add-On-Defined**
- **Module-Defined**
  - **AB:1756_MODULE:C:0**
  - **AB:1756_MODULE_DINT_4Bytes:O:0**
  - **AB:1756_MODULE_DINT_8Bytes:I:0**
  - **AB:1756_MODULE_DINT_8Bytes:O:0**

Trends

I/O Configuration

- **1756 Backplane, 1756-A10**
  - **[1] Emulate 5570 TFG_Simulacion**
  - **[2] 1756-MODULE _IN_OUT_MODULE_1**

Name	Value	Data Type	Scope
a	0	DINT	TFG_Simulacion
Constant	No		
External Access:	Read/Write		
*a - MainProgram/Z02_ANALOG_INPUT - *0(MOV), 1(CPT)*			
*a - MainProgram/Z03_ANALOG_INPUT - *0(MOV), 1(CPT)*			
A1	0	BOOL	TFG_Simulacion
Constant	No		
External Access:	Read/Write		
*A1 - MainProgram/Gemma - *0(OTE), 0(XIC), 1(XIC), 2(XIC), 4(XIO)*			
A1 - MainProgram/Z03_OUTPUTS - 1(XIC)			
A2	0	BOOL	TFG_Simulacion
Constant	No		
External Access:	Read/Write		
*A2 - MainProgram/Gemma - *4(OTE), 0(XIC), 1(XIO), 3(XIO), 4(XIC)*			
A6	0	BOOL	TFG_Simulacion
Constant	No		
External Access:	Read/Write		
A6 - MainProgram/Gemma - 2(XIO)			
AUTOMATICO_HMI	0	BOOL	TFG_Simulacion
Constant	No		
External Access:	Read/Write		
AUTOMATICO_HMI - MainProgram/Gemma - 1(XIC), 2(XIO)			
C1_Z01	0	BOOL	TFG_Simulacion
AliasFor:	Local:2:O.Data[0].0		
Base Tag:	Local:2:O.Data[0].0		
Constant	No		
External Access:	Read/Write		
C1_Z01 - MainProgram/Z01_GRAFCET - 0(XIC)			
*C1_Z01 - MainProgram/Z01_OUTPUTS - *1(OTE), 0(XIC)*			
C1_Z02	0	BOOL	TFG_Simulacion
AliasFor:	Local:2:O.Data[0].5		
Base Tag:	Local:2:O.Data[0].5		
Constant	No		
External Access:	Read/Write		
C1_Z02 - MainProgram/Z02_GRAFCET - 0(XIC)			
*C1_Z02 - MainProgram/Z02_OUTPUTS - *2(OTE), 0(XIC)*			
C1_Z03	0	BOOL	TFG_Simulacion
AliasFor:	Local:2:O.Data[0].13		
Base Tag:	Local:2:O.Data[0].13		
Constant	No		
External Access:	Read/Write		
C1_Z03 - MainProgram/Z03_GRAFCET - 0(XIC)			
*C1_Z03 - MainProgram/Z03_OUTPUTS - *2(OTE), 0(XIC)*			
C2_Z01	0	BOOL	TFG_Simulacion
AliasFor:	Local:2:O.Data[0].1		
Base Tag:	Local:2:O.Data[0].1		
Constant	No		
External Access:	Read/Write		
C2_Z01 - MainProgram/Z01_GRAFCET - 0(XIC)			
*C2_Z01 - MainProgram/Z01_OUTPUTS - *2(OTE), 0(XIC)*			
C2_Z02	0	BOOL	TFG_Simulacion
AliasFor:	Local:2:O.Data[0].6		
Base Tag:	Local:2:O.Data[0].6		
Constant	No		
External Access:	Read/Write		
C2_Z02 - MainProgram/Z02_GRAFCET - 0(XIC)			
*C2_Z02 - MainProgram/Z02_OUTPUTS - *3(OTE), 0(XIC)*			

C2_Z03 0 BOOL TFG_Simulacion
AliasFor: Local:2:O.Data[0].14
Base Tag: Local:2:O.Data[0].14
Constant No
External Access: Read/Write
C2_Z03 - MainProgram/Z03_GRAFCET - 0(XIC)
*C2_Z03 - MainProgram/Z03_OUTPUTS - *3(OTE), 0(XIC)*

C3_Z02 0 BOOL TFG_Simulacion
AliasFor: Local:2:O.Data[0].7
Base Tag: Local:2:O.Data[0].7
Constant No
External Access: Read/Write
C3_Z02 - MainProgram/Z02_GRAFCET - 0(XIC)
*C3_Z02 - MainProgram/Z02_OUTPUTS - *4(OTE), 0(XIC)*

C3_Z03 0 BOOL TFG_Simulacion
AliasFor: Local:2:O.Data[0].15
Base Tag: Local:2:O.Data[0].15
Constant No
External Access: Read/Write
C3_Z03 - MainProgram/Z03_GRAFCET - 0(XIC)
*C3_Z03 - MainProgram/Z03_OUTPUTS - *4(OTE), 0(XIC)*

Cr1_Z02 0 BOOL TFG_Simulacion
AliasFor: Local:2:O.Data[0].10
Base Tag: Local:2:O.Data[0].10
Constant No
External Access: Read/Write
Cr1_Z02 - MainProgram/Z02_GRAFCET - 0(XIC)
*Cr1_Z02 - MainProgram/Z02_OUTPUTS - *8(OTE), 0(XIC)*

Cr1_Z03 0 BOOL TFG_Simulacion
AliasFor: Local:2:O.Data[0].18
Base Tag: Local:2:O.Data[0].18
Constant No
External Access: Read/Write
Cr1_Z03 - MainProgram/Z03_GRAFCET - 0(XIC)
*Cr1_Z03 - MainProgram/Z03_OUTPUTS - *7(OTE), 0(XIC)*

D1 0 BOOL TFG_Simulacion
Constant No
External Access: Read/Write
*D1 - MainProgram/Gemma - *3(OTE), 0(XIC), 1(XIO), 3(XIC)*

DEGREES_T1_Z02 0 DINT TFG_Simulacion
Constant No
External Access: Read/Write
*DEGREES_T1_Z02 - MainProgram/Z02_ANALOG_INPUT - *1(CPT), 2(GRT), 2(LES), 2(LIM)*

F1 0 BOOL TFG_Simulacion
Constant No
External Access: Read/Write
*F1 - MainProgram/Gemma - *1(OTE), 0(XIO), 1(XIC), 2(XIC), 3(XIC), 4(XIC)*
*F1 - MainProgram/Z01_GRAFCET - *1(ONS), 1(XIC), 2(XIC), 3(XIC), 4(XIC)*
F1 - MainProgram/Z01_MESSAGE_STATES - 2(XIC)
*F1 - MainProgram/Z02_GRAFCET - *1(ONS), 1(XIC), 2(XIC)*
F1 - MainProgram/Z02_MESSAGE_STATES - 2(XIC)

F6 0 BOOL TFG_Simulacion
Constant No
External Access: Read/Write
*F6 - MainProgram/Gemma - *2(OTE), 0(XIO)*
F6 - MainProgram/Z01_OUTPUTS - 1(XIC), 2(XIC), 3(XIC)
F6 - MainProgram/Z02_OUTPUTS - 2(XIC), 3(XIC), 4(XIC), 5(XIC)
F6 - MainProgram/Z03_OUTPUTS - 2(XIC), 3(XIC), 4(XIC), 8(XIC)

F10 0 BOOL TFG_Simulacion

F10 (Continued)
Constant No
External Access: Read/Write
*F10 - MainProgram/Z01_GRAFCET - *1(OTE), 1(XIC), 2(XIC), 4(XIO)*
F10 - MainProgram/Z01_OUTPUTS - 2(XIC)
*F10 - MainProgram/Z03_GRAFCET - *1(ONS), 1(XIC), 2(XIC)*

F11 0 BOOL TFG_Simulacion
Constant No
External Access: Read/Write
*F11 - MainProgram/Z01_GRAFCET - *2(OTE), 1(XIO), 2(XIC), 3(XIC)*
F11 - MainProgram/Z01_OUTPUTS - 1(XIC), 2(XIC), 4(XIC)

F12 0 BOOL TFG_Simulacion
Constant No
External Access: Read/Write
*F12 - MainProgram/Z01_GRAFCET - *3(OTE), 2(XIO), 3(XIC), 4(XIC)*
F12 - MainProgram/Z01_OUTPUTS - 1(XIC), 2(XIC), 3(XIC)
F12 - MainProgram/Z01_TIMERS - 0(XIC)

F13 0 BOOL TFG_Simulacion
Constant No
External Access: Read/Write
*F13 - MainProgram/Z01_GRAFCET - *4(OTE), 1(XIC), 3(XIO), 4(XIC)*
F13 - MainProgram/Z01_OUTPUTS - 2(XIC), 5(XIC)

F20 0 BOOL TFG_Simulacion
Constant No
External Access: Read/Write
*F20 - MainProgram/Z02_GRAFCET - *1(OTE), 1(XIC), 2(XIC), 2(XIO)*
F20 - MainProgram/Z02_OUTPUTS - 1(XIC)

F21 0 BOOL TFG_Simulacion
Constant No
External Access: Read/Write
*F21 - MainProgram/Z02_GRAFCET - *2(OTE), 1(XIC), 1(XIO), 2(XIC)*
F21 - MainProgram/Z02_OUTPUTS - 1(XIC), 2(XIC), 3(XIC), 4(XIC), 5(XIC), 6(XIC), 7(XIC), 8(XIC)
F21 - MainProgram/Z02_TIMERS - 0(XIC), 1(XIC)

F30 0 BOOL TFG_Simulacion
Constant No
External Access: Read/Write
*F30 - MainProgram/Z03_GRAFCET - *1(OTE), 1(XIC), 2(XIC), 2(XIO)*
F30 - MainProgram/Z03_OUTPUTS - 1(XIC)

F31 0 BOOL TFG_Simulacion
Constant No
External Access: Read/Write
*F31 - MainProgram/Z03_GRAFCET - *2(OTE), 1(XIC), 1(XIO), 2(XIC)*
F31 - MainProgram/Z03_OUTPUTS - 1(XIC), 2(XIC), 3(XIC), 4(XIC), 5(XIC), 6(XIC), 7(XIC), 8(XIC)
F31 - MainProgram/Z03_TIMERS - 0(XIC)

H1_Z02 0 BOOL TFG_Simulacion
AliasFor: Local:2:O.Data[0].11
Base Tag: Local:2:O.Data[0].11
Constant No
External Access: Read/Write
H1_Z02 - MainProgram/Z02_GRAFCET - 0(XIC)
*H1_Z02 - MainProgram/Z02_OUTPUTS - *1(OTE), 0(XIC)*

H1_Z03 0 BOOL TFG_Simulacion
AliasFor: Local:2:O.Data[0].20
Base Tag: Local:2:O.Data[0].20
Constant No
External Access: Read/Write
H1_Z03 - MainProgram/Z03_GRAFCET - 0(XIC)
*H1_Z03 - MainProgram/Z03_OUTPUTS - *1(OTE), 0(XIC)*

HMI_C1_Z01 0 BOOL TFG_Simulacion
Constant No
External Access: Read/Write
HMI_C1_Z01 - MainProgram/Z01_OUTPUTS - 1(XIC)

HMI_C1_Z02 0 BOOL TFG_Simulacion
Constant No
External Access: Read/Write
HMI_C1_Z02 - MainProgram/Z02_OUTPUTS - 2(XIC)

HMI_C1_Z03 0 BOOL TFG_Simulacion
Constant No
External Access: Read/Write
HMI_C1_Z03 - MainProgram/Z03_OUTPUTS - 2(XIC)

HMI_C2_Z01 0 BOOL TFG_Simulacion
Constant No
External Access: Read/Write
HMI_C2_Z01 - MainProgram/Z01_OUTPUTS - 2(XIC)

HMI_C2_Z02 0 BOOL TFG_Simulacion
Constant No
External Access: Read/Write
HMI_C2_Z02 - MainProgram/Z02_OUTPUTS - 3(XIC)

HMI_C2_Z03 0 BOOL TFG_Simulacion
Constant No
External Access: Read/Write
HMI_C2_Z03 - MainProgram/Z03_OUTPUTS - 3(XIC)

HMI_C3_Z02 0 BOOL TFG_Simulacion
Constant No
External Access: Read/Write
HMI_C3_Z02 - MainProgram/Z02_OUTPUTS - 4(XIC)

HMI_C3_Z03 0 BOOL TFG_Simulacion
Constant No
External Access: Read/Write
HMI_C3_Z03 - MainProgram/Z03_OUTPUTS - 4(XIC)

HMI_Y1_Z01 0 BOOL TFG_Simulacion
Constant No
External Access: Read/Write
HMI_Y1_Z01 - MainProgram/Z01_OUTPUTS - 3(XIC)

HMI_Y1_Z02 0 BOOL TFG_Simulacion
Constant No
External Access: Read/Write
HMI_Y1_Z02 - MainProgram/Z02_OUTPUTS - 5(XIC)

HMI_Y1_Z03 0 BOOL TFG_Simulacion
Constant No
External Access: Read/Write
HMI_Y1_Z03 - MainProgram/Z03_OUTPUTS - 8(XIC)

Local:2:I AB:1756_MODULE_DINT_8Bytes:I:0 TFG_Simulacion
Constant No
External Access: Read/Write
Local:2:I.Data[0].0 0 BOOL
SE_Z01 - MainProgram/Gemma - 0(XIC), 3(XIO)
SE_Z01 - MainProgram/Z01_GRAFCET - 0(XIC)
SE_Z01 - MainProgram/Z01_MESSAGE_STATES - 0(XIC)
SE_Z01 - MainProgram/Z01_OUTPUTS - 0(XIC)
Local:2:I.Data[0].1 0 BOOL
SP_Z01 - MainProgram/Gemma - 0(XIC), 3(XIO)
SP_Z01 - MainProgram/Z01_GRAFCET - 0(XIC)
SP_Z01 - MainProgram/Z01_MESSAGE_STATES - 1(XIC)

Local:2:I (Continued)
SP_Z01 - MainProgram/Z01_OUTPUTS - 0(XIC)
Local:2:I.Data[0].2 0 BOOL
SE_Z02 - MainProgram/Gemma - 0(XIC), 3(XIO)
SE_Z02 - MainProgram/Z02_GRAFCET - 0(XIC)
SE_Z02 - MainProgram/Z02_MESSAGE_STATES - 0(XIC)
SE_Z02 - MainProgram/Z02_OUTPUTS - 0(XIC)
Local:2:I.Data[0].3 0 BOOL
SP_Z02 - MainProgram/Gemma - 0(XIC), 3(XIO)
SP_Z02 - MainProgram/Z02_GRAFCET - 0(XIC)
SP_Z02 - MainProgram/Z02_MESSAGE_STATES - 1(XIC)
SP_Z02 - MainProgram/Z02_OUTPUTS - 0(XIC)
Local:2:I.Data[0].4 0 BOOL
SE_Z03 - MainProgram/Gemma - 0(XIC), 3(XIO)
SE_Z03 - MainProgram/Z03_GRAFCET - 0(XIC)
SE_Z03 - MainProgram/Z03_MESSAGE_STATES - 0(XIC)
SE_Z03 - MainProgram/Z03_OUTPUTS - 0(XIC)
Local:2:I.Data[0].5 0 BOOL
SP_Z03 - MainProgram/Gemma - 0(XIC), 3(XIO)
SP_Z03 - MainProgram/Z03_GRAFCET - 0(XIC)
SP_Z03 - MainProgram/Z03_MESSAGE_STATES - 1(XIC)
SP_Z03 - MainProgram/Z03_OUTPUTS - 0(XIC)
Local:2:I.Data[1].0 0 BOOL
S1_Z01 - MainProgram/Z01_GRAFCET - 0(XIC), 2(XIC)
S1_Z01 - MainProgram/Z01_MESSAGE_STATES - 2(XIO)
S1_Z01 - MainProgram/Z01_OUTPUTS - 0(XIC), 2(XIC)
Local:2:I.Data[1].1 0 BOOL
S2_Z01 - MainProgram/Z01_GRAFCET - 0(XIC), 3(XIC)
S2_Z01 - MainProgram/Z01_OUTPUTS - 0(XIC)
Local:2:I.Data[1].2 0 BOOL
S3_Z01 - MainProgram/Z01_GRAFCET - 0(XIC), 1(XIC)
S3_Z01 - MainProgram/Z01_OUTPUTS - 0(XIC)
S3_Z01 - MainProgram/Z01_TIMERS - 0(XIO)
Local:2:I.Data[1].3 0 BOOL
S1_Z02 - MainProgram/Z02_GRAFCET - 0(XIC), 1(XIO), 2(XIC)
S1_Z02 - MainProgram/Z02_OUTPUTS - 0(XIC)
S1_Z02 - MainProgram/Z02_TIMERS - 0(XIO)
Local:2:I.Data[1].4 0 BOOL
S2_Z02 - MainProgram/Z02_GRAFCET - 0(XIC)
S2_Z02 - MainProgram/Z02_OUTPUTS - 0(XIC)
S2_Z02 - MainProgram/Z02_TIMERS - 0(XIO)
Local:2:I.Data[1].5 0 BOOL
S3_Z02 - MainProgram/Z02_GRAFCET - 0(XIC)
S3_Z02 - MainProgram/Z02_OUTPUTS - 0(XIC)
Local:2:I.Data[1].6 0 BOOL
SVa_Z02 - MainProgram/Z02_GRAFCET - 0(XIC)
SVa_Z02 - MainProgram/Z02_OUTPUTS - 0(XIC), 5(XIC)
SVa_Z02 - MainProgram/Z02_TIMERS - 1(XIC)
Local:2:I.Data[1].7 0 BOOL
S1_Z03 - MainProgram/Z03_GRAFCET - 0(XIC), 1(XIO), 2(XIC)
S1_Z03 - MainProgram/Z03_OUTPUTS - 0(XIC)
Local:2:I.Data[1].8 0 BOOL
S2_Z03 - MainProgram/Z03_GRAFCET - 0(XIC)
S2_Z03 - MainProgram/Z03_OUTPUTS - 0(XIC)
S2_Z03 - MainProgram/Z03_TIMERS - 0(XIO)
Local:2:I.Data[1].9 0 BOOL
S3_Z03 - MainProgram/Z03_GRAFCET - 0(XIC)
S3_Z03 - MainProgram/Z03_OUTPUTS - 0(XIC), 8(XIC)

Local:2:O AB:1756_MODULE_DINT_8Bytes:O:0
TFG_Simulacion

Constant No
External Access: Read/Write
Local:2:O.Data[0].0 0 BOOL
C1_Z01 - MainProgram/Z01_GRAFCET - 0(XIC)
*C1_Z01 - MainProgram/Z01_OUTPUTS - *1(OTE), 0(XIC)*
Local:2:O.Data[0].1 0 BOOL
C2_Z01 - MainProgram/Z01_GRAFCET - 0(XIC)

Local:2:O (Continued)
*C2_Z01 - MainProgram/Z01_OUTPUTS - *2(OTE), 0(XIC)*
Local:2:O.Data[0].2 0 BOOL
Y1_Z01 - MainProgram/Z01_GRAFCET - 0(XIC)
*Y1_Z01 - MainProgram/Z01_OUTPUTS - *3(OTE), 0(XIC)*
Local:2:O.Data[0].3 0 BOOL
Rb1_Z01 - MainProgram/Z01_GRAFCET - 0(XIC)
*Rb1_Z01 - MainProgram/Z01_OUTPUTS - *4(OTE), 0(XIC)*
Local:2:O.Data[0].4 0 BOOL
Rb2_Z01 - MainProgram/Z01_GRAFCET - 0(XIC)
*Rb2_Z01 - MainProgram/Z01_OUTPUTS - *5(OTE), 0(XIC)*
Local:2:O.Data[0].5 0 BOOL
C1_Z02 - MainProgram/Z02_GRAFCET - 0(XIC)
*C1_Z02 - MainProgram/Z02_OUTPUTS - *2(OTE), 0(XIC)*
Local:2:O.Data[0].6 0 BOOL
C2_Z02 - MainProgram/Z02_GRAFCET - 0(XIC)
*C2_Z02 - MainProgram/Z02_OUTPUTS - *3(OTE), 0(XIC)*
Local:2:O.Data[0].7 0 BOOL
C3_Z02 - MainProgram/Z02_GRAFCET - 0(XIC)
*C3_Z02 - MainProgram/Z02_OUTPUTS - *4(OTE), 0(XIC)*
Local:2:O.Data[0].8 0 BOOL
Mr1_Z02 - MainProgram/Z02_GRAFCET - 0(XIC)
*Mr1_Z02 - MainProgram/Z02_OUTPUTS - *6(OTE), 0(XIC)*
Local:2:O.Data[0].9 0 BOOL
Mr2_Z02 - MainProgram/Z02_GRAFCET - 0(XIC)
*Mr2_Z02 - MainProgram/Z02_OUTPUTS - *7(OTE), 0(XIC)*
Local:2:O.Data[0].10 0 BOOL
Cr1_Z02 - MainProgram/Z02_GRAFCET - 0(XIC)
*Cr1_Z02 - MainProgram/Z02_OUTPUTS - *8(OTE), 0(XIC)*
Local:2:O.Data[0].11 0 BOOL
H1_Z02 - MainProgram/Z02_GRAFCET - 0(XIC)
*H1_Z02 - MainProgram/Z02_OUTPUTS - *1(OTE), 0(XIC)*
Local:2:O.Data[0].12 0 BOOL
Y1_Z02 - MainProgram/Z02_GRAFCET - 0(XIC)
*Y1_Z02 - MainProgram/Z02_OUTPUTS - *5(OTE), 0(XIC)*
Local:2:O.Data[0].13 0 BOOL
C1_Z03 - MainProgram/Z03_GRAFCET - 0(XIC)
*C1_Z03 - MainProgram/Z03_OUTPUTS - *2(OTE), 0(XIC)*
Local:2:O.Data[0].14 0 BOOL
C2_Z03 - MainProgram/Z03_GRAFCET - 0(XIC)
*C2_Z03 - MainProgram/Z03_OUTPUTS - *3(OTE), 0(XIC)*
Local:2:O.Data[0].15 0 BOOL
C3_Z03 - MainProgram/Z03_GRAFCET - 0(XIC)
*C3_Z03 - MainProgram/Z03_OUTPUTS - *4(OTE), 0(XIC)*
Local:2:O.Data[0].16 0 BOOL
Mr1_Z03 - MainProgram/Z03_GRAFCET - 0(XIC)
*Mr1_Z03 - MainProgram/Z03_OUTPUTS - *5(OTE), 0(XIC)*
Local:2:O.Data[0].17 0 BOOL
Mr2_Z03 - MainProgram/Z03_GRAFCET - 0(XIC)
*Mr2_Z03 - MainProgram/Z03_OUTPUTS - *6(OTE), 0(XIC)*
Local:2:O.Data[0].18 0 BOOL
Cr1_Z03 - MainProgram/Z03_GRAFCET - 0(XIC)
*Cr1_Z03 - MainProgram/Z03_OUTPUTS - *7(OTE), 0(XIC)*
Local:2:O.Data[0].19 0 BOOL
Y1_Z03 - MainProgram/Z03_GRAFCET - 0(XIC)
*Y1_Z03 - MainProgram/Z03_OUTPUTS - *8(OTE), 0(XIC)*
Local:2:O.Data[0].20 0 BOOL
H1_Z03 - MainProgram/Z03_GRAFCET - 0(XIC)
*H1_Z03 - MainProgram/Z03_OUTPUTS - *1(OTE), 0(XIC)*

MANUAL_HMI 0 BOOL TFG_Simulacion
Constant No
External Access: Read/Write
MANUAL_HMI - MainProgram/Gemma - 1(XIO), 2(XIC)

MessageState DINT[2] TFG_Simulacion
Constant No
External Access: Read/Write

MessageState (Continued)

MessageState[0].0 0 BOOL
SISTEMA LISTO
*MessageState[0].0 - MainProgram/Gemma - *0(OTE)*

MessageState[0].1 0 BOOL
MODO AUTOMATICO
*MessageState[0].1 - MainProgram/Gemma - *1(OTE)*

MessageState[0].2 0 BOOL
MODO MANUAL
*MessageState[0].2 - MainProgram/Gemma - *2(OTE)*

MessageState[0].3 0 BOOL
FALLO EN EL SISTEMA
*MessageState[0].3 - MainProgram/Gemma - *3(OTE)*

MessageState[0].4 0 BOOL
PARO ACTIVADO
*MessageState[0].4 - MainProgram/Gemma - *4(OTE)*

MessageState[0].10 0 BOOL
PARO DE EMERGENCIA Z1 ACTIVADO
*MessageState[0].10 - MainProgram/Z01_MESSAGE_STATES - *0(OTE)*

MessageState[0].11 0 BOOL
PUERTA ABIERTA Z1
*MessageState[0].11 - MainProgram/Z01_MESSAGE_STATES - *1(OTE)*

MessageState[0].12 0 BOOL
PALET VACIO Z1
*MessageState[0].12 - MainProgram/Z01_MESSAGE_STATES - *2(OTE)*

MessageState[0].20 0 BOOL
PARO DE EMERGENCIA Z2 ACTIVADO
*MessageState[0].20 - MainProgram/Z02_MESSAGE_STATES - *0(OTE)*

MessageState[0].21 0 BOOL
PUERTA ABIERTA Z2
*MessageState[0].21 - MainProgram/Z02_MESSAGE_STATES - *1(OTE)*

MessageState[0].22 0 BOOL
BOTELLA DEFECTUOSA O SUCIA
*MessageState[0].22 - MainProgram/Z02_MESSAGE_STATES - *2(OTE)*

MessageState[0].23 0 BOOL
TEMPERATURA EN T1.Z2 OK
*MessageState[0].23 - MainProgram/Z02_MESSAGE_STATES - *3(OTE)*

MessageState[0].24 0 BOOL
TEMPERATURA EN T1.Z2 LOW
*MessageState[0].24 - MainProgram/Z02_MESSAGE_STATES - *4(OTE)*

MessageState[0].25 0 BOOL
TEMPERATURA EN T1.Z2 HIGH
*MessageState[0].25 - MainProgram/Z02_MESSAGE_STATES - *5(OTE)*

MessageState[1].0 0 BOOL
SETA DE EMERGENCIA Z3 ACTIVADA
*MessageState[1].0 - MainProgram/Z03_MESSAGE_STATES - *0(OTE)*

MessageState[1].1 0 BOOL
PUERTA ABIERTA Z3
*MessageState[1].1 - MainProgram/Z03_MESSAGE_STATES - *1(OTE)*

MessageState[1].2 0 BOOL
TEMPERATURA T1.Z3 OK
*MessageState[1].2 - MainProgram/Z03_MESSAGE_STATES - *2(OTE)*

MessageState[1].3 0 BOOL
TEMPERATURA T1.Z3 LOW
*MessageState[1].3 - MainProgram/Z03_MESSAGE_STATES - *3(OTE)*

MessageState[1].4 0 BOOL
TEMPERATURA T1.Z3 HIGH
*MessageState[1].4 - MainProgram/Z03_MESSAGE_STATES - *4(OTE)*

Mr1_Z02 0 BOOL TFG_Simulacion
AliasFor: Local:2:O.Data[0].8
Base Tag: Local:2:O.Data[0].8
Constant No
External Access: Read/Write
Mr1_Z02 - MainProgram/Z02_GRAFCET - 0(XIC)
*Mr1_Z02 - MainProgram/Z02_OUTPUTS - *6(OTE), 0(XIC)*

Mr1_Z03 0 BOOL TFG_Simulacion

Mr1_Z03 (Continued)
AliasFor: Local:2:O.Data[0].16
Base Tag: Local:2:O.Data[0].16
Constant No
External Access: Read/Write
Mr1_Z03 - MainProgram/Z03_GRAFCET - 0(XIC)
*Mr1_Z03 - MainProgram/Z03_OUTPUTS - *5(OTE), 0(XIC)*

Mr2_Z02 0 BOOL TFG_Simulacion
AliasFor: Local:2:O.Data[0].9
Base Tag: Local:2:O.Data[0].9
Constant No
External Access: Read/Write
Mr2_Z02 - MainProgram/Z02_GRAFCET - 0(XIC)
*Mr2_Z02 - MainProgram/Z02_OUTPUTS - *7(OTE), 0(XIC)*

Mr2_Z03 0 BOOL TFG_Simulacion
AliasFor: Local:2:O.Data[0].17
Base Tag: Local:2:O.Data[0].17
Constant No
External Access: Read/Write
Mr2_Z03 - MainProgram/Z03_GRAFCET - 0(XIC)
*Mr2_Z03 - MainProgram/Z03_OUTPUTS - *6(OTE), 0(XIC)*

PARO_HMI 0 BOOL TFG_Simulacion
Constant No
External Access: Read/Write
PARO_HMI - MainProgram/Gemma - 4(XIC)

R 0 DINT TFG_Simulacion
Constant No
External Access: Read/Write
*R - MainProgram/Z02_ANALOG_INPUT - *0(MOV), 1(CPT)*
*R - MainProgram/Z03_ANALOG_INPUT - *0(MOV), 1(CPT)*

R1_Z02 0 DINT TFG_Simulacion
Constant No
External Access: Read/Write
*R1_Z02 - MainProgram/Z02_ANALOG_INPUT - *0(MOV), 1(CPT)*
R1_Z02 - MainProgram/Z03_ANALOG_INPUT - 1(CPT)

Rb1_Z01 0 BOOL TFG_Simulacion
AliasFor: Local:2:O.Data[0].3
Base Tag: Local:2:O.Data[0].3
Constant No
External Access: Read/Write
Rb1_Z01 - MainProgram/Z01_GRAFCET - 0(XIC)
*Rb1_Z01 - MainProgram/Z01_OUTPUTS - *4(OTE), 0(XIC)*

Rb2_Z01 0 BOOL TFG_Simulacion
AliasFor: Local:2:O.Data[0].4
Base Tag: Local:2:O.Data[0].4
Constant No
External Access: Read/Write
Rb2_Z01 - MainProgram/Z01_GRAFCET - 0(XIC)
*Rb2_Z01 - MainProgram/Z01_OUTPUTS - *5(OTE), 0(XIC)*

S1_Z01 0 BOOL TFG_Simulacion
AliasFor: Local:2:I.Data[1].0
Base Tag: Local:2:I.Data[1].0
Constant No
External Access: Read/Write
S1_Z01 - MainProgram/Z01_GRAFCET - 0(XIC), 2(XIC)
S1_Z01 - MainProgram/Z01_MESSAGE_STATES - 2(XIO)
S1_Z01 - MainProgram/Z01_OUTPUTS - 0(XIC), 2(XIC)

S1_Z02 0 BOOL TFG_Simulacion
AliasFor: Local:2:I.Data[1].3

S1_Z02 (Continued)
Base Tag: Local:2:I.Data[1].3
Constant No
External Access: Read/Write
S1_Z02 - MainProgram/Z02_GRAFCET - 0(XIC), 1(XIO), 2(XIC)
S1_Z02 - MainProgram/Z02_OUTPUTS - 0(XIC)
S1_Z02 - MainProgram/Z02_TIMERS - 0(XIO)

S1_Z03 0 BOOL TFG_Simulacion
AliasFor: Local:2:I.Data[1].7
Base Tag: Local:2:I.Data[1].7
Constant No
External Access: Read/Write
S1_Z03 - MainProgram/Z03_GRAFCET - 0(XIC), 1(XIO), 2(XIC)
S1_Z03 - MainProgram/Z03_OUTPUTS - 0(XIC)

S2_Z01 0 BOOL TFG_Simulacion
AliasFor: Local:2:I.Data[1].1
Base Tag: Local:2:I.Data[1].1
Constant No
External Access: Read/Write
S2_Z01 - MainProgram/Z01_GRAFCET - 0(XIC), 3(XIC)
S2_Z01 - MainProgram/Z01_OUTPUTS - 0(XIC)

S2_Z02 0 BOOL TFG_Simulacion
AliasFor: Local:2:I.Data[1].4
Base Tag: Local:2:I.Data[1].4
Constant No
External Access: Read/Write
S2_Z02 - MainProgram/Z02_GRAFCET - 0(XIC)
S2_Z02 - MainProgram/Z02_OUTPUTS - 0(XIC)
S2_Z02 - MainProgram/Z02_TIMERS - 0(XIO)

S2_Z03 0 BOOL TFG_Simulacion
AliasFor: Local:2:I.Data[1].8
Base Tag: Local:2:I.Data[1].8
Constant No
External Access: Read/Write
S2_Z03 - MainProgram/Z03_GRAFCET - 0(XIC)
S2_Z03 - MainProgram/Z03_OUTPUTS - 0(XIC)
S2_Z03 - MainProgram/Z03_TIMERS - 0(XIO)

S3_Z01 0 BOOL TFG_Simulacion
AliasFor: Local:2:I.Data[1].2
Base Tag: Local:2:I.Data[1].2
Constant No
External Access: Read/Write
S3_Z01 - MainProgram/Z01_GRAFCET - 0(XIC), 1(XIC)
S3_Z01 - MainProgram/Z01_OUTPUTS - 0(XIC)
S3_Z01 - MainProgram/Z01_TIMERS - 0(XIO)

S3_Z02 0 BOOL TFG_Simulacion
AliasFor: Local:2:I.Data[1].5
Base Tag: Local:2:I.Data[1].5
Constant No
External Access: Read/Write
S3_Z02 - MainProgram/Z02_GRAFCET - 0(XIC)
S3_Z02 - MainProgram/Z02_OUTPUTS - 0(XIC)

S3_Z03 0 BOOL TFG_Simulacion
AliasFor: Local:2:I.Data[1].9
Base Tag: Local:2:I.Data[1].9
Constant No
External Access: Read/Write
S3_Z03 - MainProgram/Z03_GRAFCET - 0(XIC)
S3_Z03 - MainProgram/Z03_OUTPUTS - 0(XIC), 8(XIC)

SE_Z01 0 BOOL TFG_Simulacion

SE_Z01 (Continued)
AliasFor: Local:2:I.Data[0].0
Base Tag: Local:2:I.Data[0].0
Constant No
External Access: Read/Write
SE_Z01 - MainProgram/Gemma - 0(XIC), 3(XIO)
SE_Z01 - MainProgram/Z01_GRAFCET - 0(XIC)
SE_Z01 - MainProgram/Z01_MESSAGE_STATES - 0(XIC)
SE_Z01 - MainProgram/Z01_OUTPUTS - 0(XIC)

SE_Z02 0 BOOL TFG_Simulacion
AliasFor: Local:2:I.Data[0].2
Base Tag: Local:2:I.Data[0].2
Constant No
External Access: Read/Write
SE_Z02 - MainProgram/Gemma - 0(XIC), 3(XIO)
SE_Z02 - MainProgram/Z02_GRAFCET - 0(XIC)
SE_Z02 - MainProgram/Z02_MESSAGE_STATES - 0(XIC)
SE_Z02 - MainProgram/Z02_OUTPUTS - 0(XIC)

SE_Z03 0 BOOL TFG_Simulacion
AliasFor: Local:2:I.Data[0].4
Base Tag: Local:2:I.Data[0].4
Constant No
External Access: Read/Write
SE_Z03 - MainProgram/Gemma - 0(XIC), 3(XIO)
SE_Z03 - MainProgram/Z03_GRAFCET - 0(XIC)
SE_Z03 - MainProgram/Z03_MESSAGE_STATES - 0(XIC)
SE_Z03 - MainProgram/Z03_OUTPUTS - 0(XIC)

SP_Z01 0 BOOL TFG_Simulacion
AliasFor: Local:2:I.Data[0].1
Base Tag: Local:2:I.Data[0].1
Constant No
External Access: Read/Write
SP_Z01 - MainProgram/Gemma - 0(XIC), 3(XIO)
SP_Z01 - MainProgram/Z01_GRAFCET - 0(XIC)
SP_Z01 - MainProgram/Z01_MESSAGE_STATES - 1(XIC)
SP_Z01 - MainProgram/Z01_OUTPUTS - 0(XIC)

SP_Z02 0 BOOL TFG_Simulacion
AliasFor: Local:2:I.Data[0].3
Base Tag: Local:2:I.Data[0].3
Constant No
External Access: Read/Write
SP_Z02 - MainProgram/Gemma - 0(XIC), 3(XIO)
SP_Z02 - MainProgram/Z02_GRAFCET - 0(XIC)
SP_Z02 - MainProgram/Z02_MESSAGE_STATES - 1(XIC)
SP_Z02 - MainProgram/Z02_OUTPUTS - 0(XIC)

SP_Z03 0 BOOL TFG_Simulacion
AliasFor: Local:2:I.Data[0].5
Base Tag: Local:2:I.Data[0].5
Constant No
External Access: Read/Write
SP_Z03 - MainProgram/Gemma - 0(XIC), 3(XIO)
SP_Z03 - MainProgram/Z03_GRAFCET - 0(XIC)
SP_Z03 - MainProgram/Z03_MESSAGE_STATES - 1(XIC)
SP_Z03 - MainProgram/Z03_OUTPUTS - 0(XIC)

ST1_Z02_HIGH 0 BOOL TFG_Simulacion
Constant No
External Access: Read/Write
*ST1_Z02_HIGH - MainProgram/Z02_ANALOG_INPUT - *2(OTE)*
ST1_Z02_HIGH - MainProgram/Z02_GRAFCET - 0(XIC)
ST1_Z02_HIGH - MainProgram/Z02_MESSAGE_STATES - 5(XIC)
ST1_Z02_HIGH - MainProgram/Z02_OUTPUTS - 0(XIC)

ST1_Z02_LOW 0 BOOL TFG_Simulacion
Constant No
External Access: Read/Write
*ST1_Z02_LOW - MainProgram/Z02_ANALOG_INPUT - *2(OTE)*
ST1_Z02_LOW - MainProgram/Z02_GRAFCET - 0(XIC)
ST1_Z02_LOW - MainProgram/Z02_MESSAGE_STATES - 4(XIC)
ST1_Z02_LOW - MainProgram/Z02_OUTPUTS - 0(XIC)

ST1_Z03_HIGH 0 BOOL TFG_Simulacion
Constant No
External Access: Read/Write
*ST1_Z03_HIGH - MainProgram/Z03_ANALOG_INPUT - *2(OTE)*
ST1_Z03_HIGH - MainProgram/Z03_GRAFCET - 0(XIC)
ST1_Z03_HIGH - MainProgram/Z03_MESSAGE_STATES - 4(XIC)
ST1_Z03_HIGH - MainProgram/Z03_OUTPUTS - 0(XIC)

ST1_Z03_LOW 0 BOOL TFG_Simulacion
Constant No
External Access: Read/Write
*ST1_Z03_LOW - MainProgram/Z03_ANALOG_INPUT - *2(OTE)*
ST1_Z03_LOW - MainProgram/Z03_GRAFCET - 0(XIC)
ST1_Z03_LOW - MainProgram/Z03_MESSAGE_STATES - 3(XIC)
ST1_Z03_LOW - MainProgram/Z03_OUTPUTS - 0(XIC)

SVa_Z02 0 BOOL TFG_Simulacion
AliasFor: Local:2:I.Data[1].6
Base Tag: Local:2:I.Data[1].6
Constant No
External Access: Read/Write
SVa_Z02 - MainProgram/Z02_GRAFCET - 0(XIC)
SVa_Z02 - MainProgram/Z02_OUTPUTS - 0(XIC), 5(XIC)
SVa_Z02 - MainProgram/Z02_TIMERS - 1(XIC)

T1_Z02 TIMER TFG_Simulacion
Constant No
External Access: Read/Write
*T1_Z02 - MainProgram/Z02_TIMERS - *0(TON), *0(TON)*
T1_Z02.DN 0 BOOL
T1_Z02.DN - MainProgram/Z02_GRAFCET - 1(XIO)
T1_Z02.DN - MainProgram/Z02_OUTPUTS - 6(XIO), 7(XIO), 8(XIO)

T1_Z03 TIMER TFG_Simulacion
Constant No
External Access: Read/Write
*T1_Z03 - MainProgram/Z03_TIMERS - *0(TON)*
T1_Z03.DN 0 BOOL
T1_Z03.DN - MainProgram/Z03_GRAFCET - 1(XIC)

T1_ZO1 TIMER TFG_Simulacion
Constant No
External Access: Read/Write
*T1_ZO1 - MainProgram/Z01_TIMERS - *0(TON)*
T1_ZO1.DN 0 BOOL
T1_ZO1.DN - MainProgram/Z01_GRAFCET - 4(XIC)

T2_Z03 TIMER TFG_Simulacion
Constant No
External Access: Read/Write
T2_Z03.DN 0 BOOL
T2_Z03.DN - MainProgram/Z03_OUTPUTS - 5(XIO), 6(XIO), 7(XIO)

T3_Z02 TIMER TFG_Simulacion
Constant No
External Access: Read/Write
*T3_Z02 - MainProgram/Z02_TIMERS - *1(TOF)*
T3_Z02.DN 0 BOOL
T3_Z02.DN - MainProgram/Z02_MESSAGE_STATES - 2(XIO)
T3_Z02.DN - MainProgram/Z02_OUTPUTS - 4(XIC)

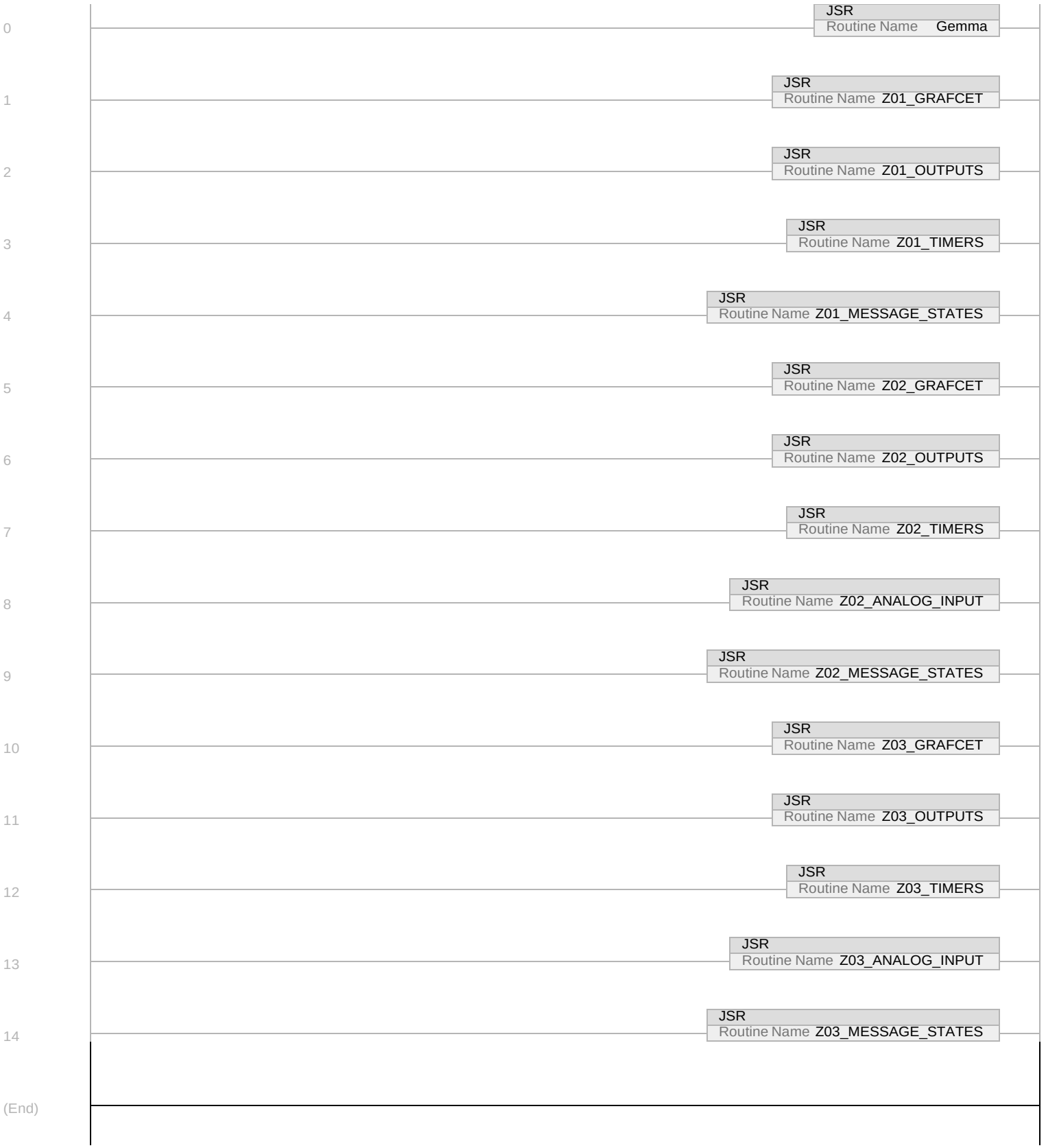
V0_T_Z03 0 DINT TFG_Simulacion
Constant No
External Access: Read/Write
*V0_T_Z03 - MainProgram/Z02_ANALOG_INPUT - *0(MOV), 1(CPT)*
*V0_T_Z03 - MainProgram/Z03_ANALOG_INPUT - *0(MOV), 1(CPT)*

Ve 0 DINT TFG_Simulacion
Constant No
External Access: Read/Write
*Ve - MainProgram/Z02_ANALOG_INPUT - *0(MOV), 1(CPT)*
*Ve - MainProgram/Z03_ANALOG_INPUT - *0(MOV), 1(CPT)*

Y1_Z01 0 BOOL TFG_Simulacion
AliasFor: Local:2:O.Data[0].2
Base Tag: Local:2:O.Data[0].2
Constant No
External Access: Read/Write
Y1_Z01 - MainProgram/Z01_GRAFCET - 0(XIC)
*Y1_Z01 - MainProgram/Z01_OUTPUTS - *3(OTE), 0(XIC)*

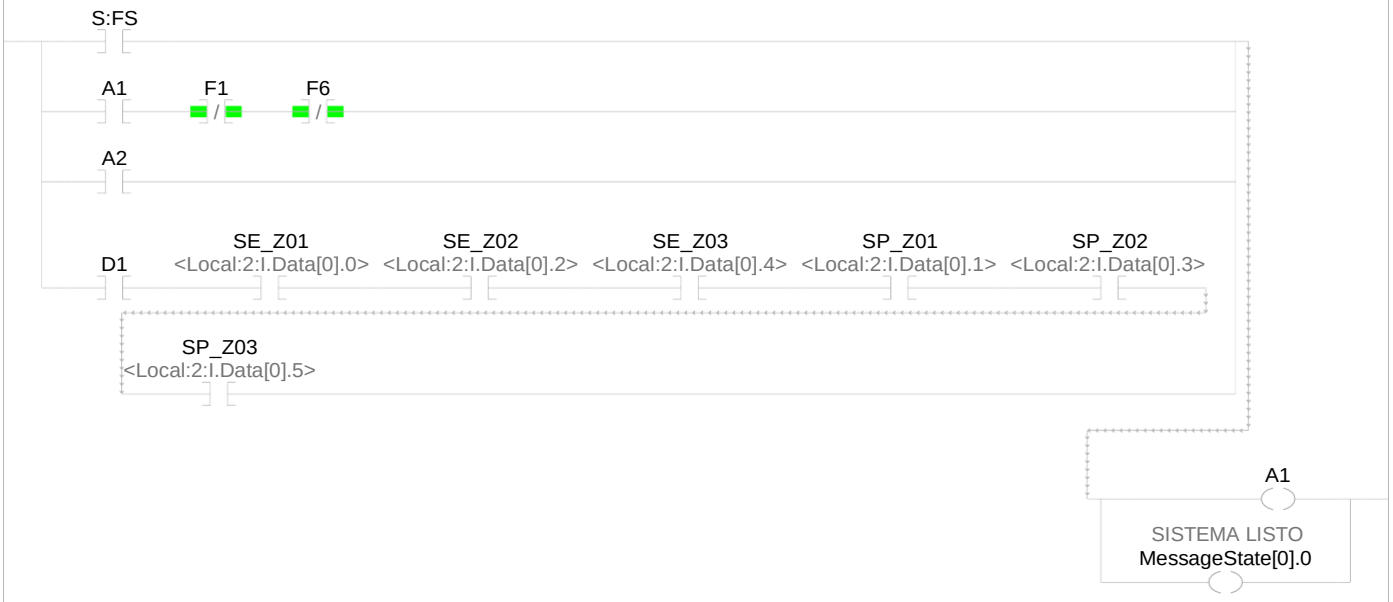
Y1_Z02 0 BOOL TFG_Simulacion
AliasFor: Local:2:O.Data[0].12
Base Tag: Local:2:O.Data[0].12
Constant No
External Access: Read/Write
Y1_Z02 - MainProgram/Z02_GRAFCET - 0(XIC)
*Y1_Z02 - MainProgram/Z02_OUTPUTS - *5(OTE), 0(XIC)*

Y1_Z03 0 BOOL TFG_Simulacion
AliasFor: Local:2:O.Data[0].19
Base Tag: Local:2:O.Data[0].19
Constant No
External Access: Read/Write
Y1_Z03 - MainProgram/Z03_GRAFCET - 0(XIC)
*Y1_Z03 - MainProgram/Z03_OUTPUTS - *8(OTE), 0(XIC)*


0
JSR
Routine Name Gemma
1
JSR
Routine Name Z01_GRAFCET
2
JSR
Routine Name Z01_OUTPUTS
3
JSR
Routine Name Z01_TIMERS
4
JSR
Routine Name Z01_MESSAGE_STATES
5
JSR
Routine Name Z02_GRAFCET
6
JSR
Routine Name Z02_OUTPUTS
7
JSR
Routine Name Z02_TIMERS
8
JSR
Routine Name Z02_ANALOG_INPUT
9
JSR
Routine Name Z02_MESSAGE_STATES
10
JSR
Routine Name Z03_GRAFCET
11
JSR
Routine Name Z03_OUTPUTS
12
JSR
Routine Name Z03_TIMERS
13
JSR
Routine Name Z03_ANALOG_INPUT
14
JSR
Routine Name Z03_MESSAGE_STATES
(End)

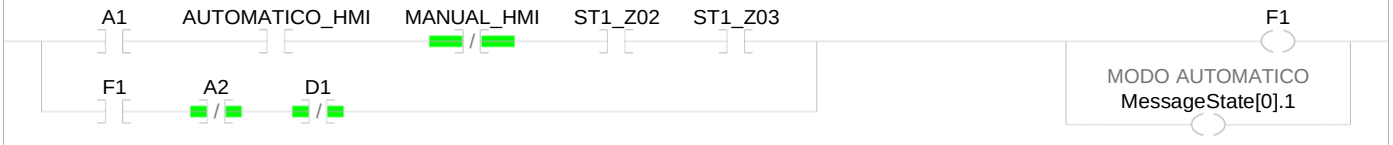
MACRO A1
LISTO PARA EMPEZAR A FUNCIONAR

0



MACRO F1
MODO DE FUNCIONAMIETNO AUTOMATICO

1

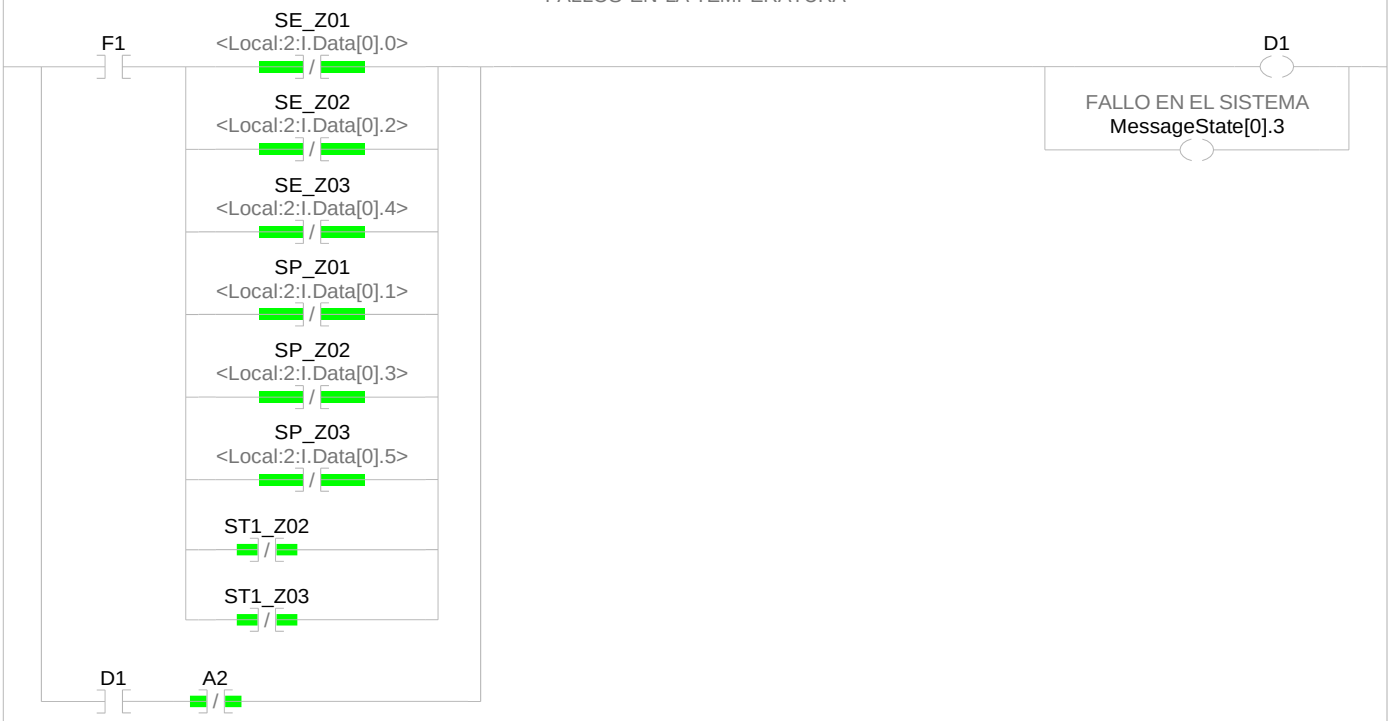


MACRO F6
MODO DE FUNCIONAMIETNO MANUAL

2

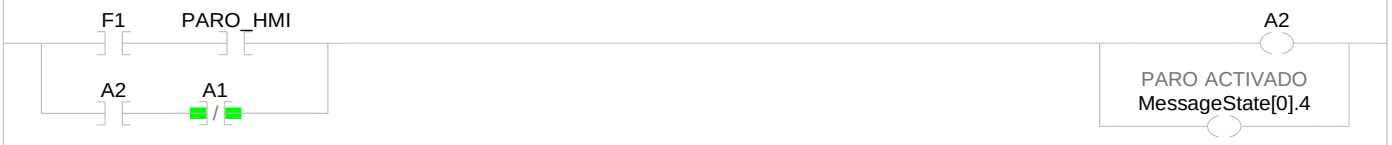
MACRO D1
ACTIVACIÓN DE PARO POR ELEMENTOS DE SEGURIDAD
O
FALLOS EN LA TEMPERATURA

3



MACRO A2
ACTIVACIÓN DE PARO NORMAL

4



(End)

INFORMATIVE RUNG

0
SE_Z01 <Local:2:I.Data[0].0> SP_Z01 <Local:2:I.Data[0].1> NOP
S1_Z01 <Local:2:I.Data[1].0> S2_Z01 <Local:2:I.Data[1].1> S3_Z01 <Local:2:I.Data[1].2>
Rb1_Z01 <Local:2:O.Data[0].3> Rb2_Z01 <Local:2:O.Data[0].4>
C1_Z01 <Local:2:O.Data[0].0> C2_Z01 <Local:2:O.Data[0].1> Y1_Z01 <Local:2:O.Data[0].2>

ETAPA 10
SE ACTIVA NADA MAS ENTRAR EN F1
Y DESPUES DE LA ETAPA 13 DONDE SE COLOCAN LOS BOTELLINES EN LA CINTA ACUMULADORA
Y SE REALIMENTA HASTA EMPEZAR LA ETAPA 11

1
F1 F13 S3_Z01 <Local:2:I.Data[1].2> F10
F10 F11 / F1 ONS

ETAPA 11
SI HAY SE;AL DE QUE HAY CAJAS LAS CINTAS Y EL ROBOT SE MUEVEN

2
F1 F10 S1_Z01 <Local:2:I.Data[1].0> F11
F11 F12 /

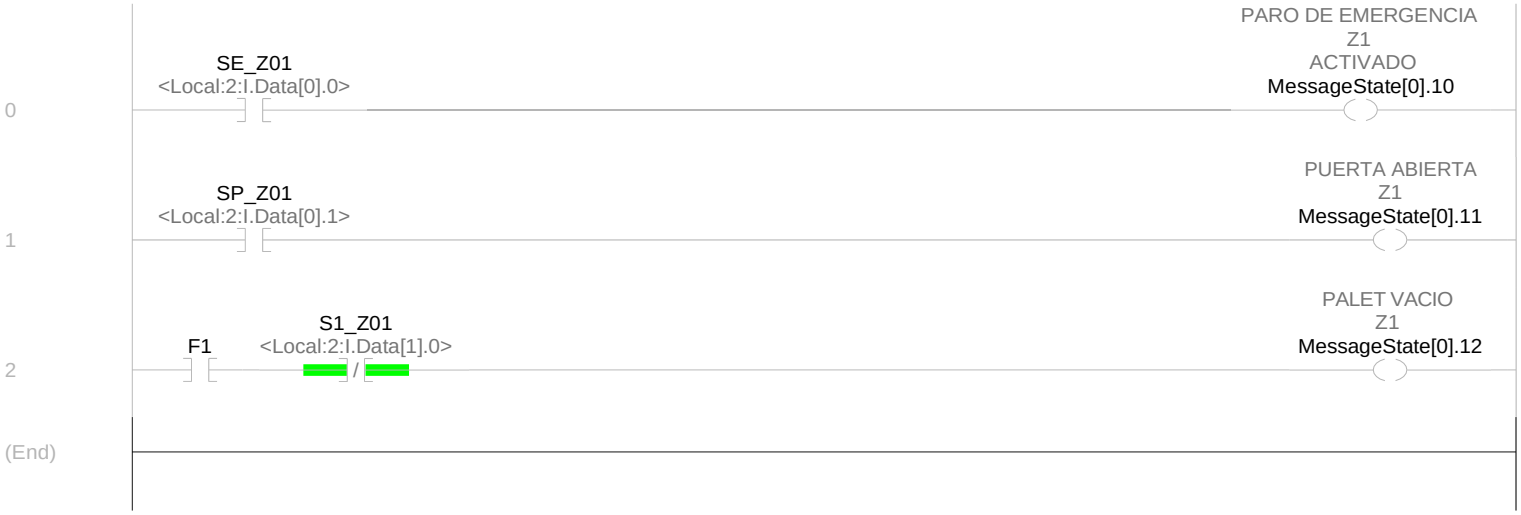
ETAPA 13
ACTUA EL ROBOT 2 Y COLOCA LOS BOTELLINES EN LA OTRA CINTA

3
F1 F11 S2_Z01 <Local:2:I.Data[1].1> F12
F12 F13 /

ETAPA 13
LA CINTA CUANTO LLEGA AL SENSOR CONTINUA 2 SEGUNDOS PARA ACUMULAR LAS CAJAS Y ASEGURARSE QUE VAN A LA MEDIDA DEL GRIPPER
PERO SE ASEGURA CON EL S3_Z02 QUE HAYA HUECO EN LA CINTA DE ACUMULACION

4
F1 F12 T1_ZO1.DN F13
F13 F10 /

(End)


0
SE_Z01
<Local:2:I.Data[0].0>
PARO DE EMERGENCIA Z1 ACTIVADO
MessageState[0].10
1
SP_Z01
<Local:2:I.Data[0].1>
PUERTA ABIERTA Z1
MessageState[0].11
2
F1
S1_Z01
<Local:2:I.Data[1].0>
/
PALET VACIO Z1
MessageState[0].12
(End)

INFORMATIVE RUNG

0

SE_Z01 <Local:2:I.Data[0].0> SP_Z01 <Local:2:I.Data[0].1> NOP

S1_Z01 <Local:2:I.Data[1].0> S2_Z01 <Local:2:I.Data[1].1> S3_Z01 <Local:2:I.Data[1].2>

Rb1_Z01 <Local:2:O.Data[0].3> Rb2_Z01 <Local:2:O.Data[0].4>

C1_Z01 <Local:2:O.Data[0].0> C2_Z01 <Local:2:O.Data[0].1> Y1_Z01 <Local:2:O.Data[0].2>

CINTA 1

1

F11

F12

HMI_C1_Z01 F6

C1_Z01 <Local:2:O.Data[0].0>

CINTA 2

2

F10 S1_Z01 <Local:2:I.Data[1].0>

F11

F12

F13

HMI_C2_Z01 F6

C2_Z01 <Local:2:O.Data[0].1>

ACTUADOR DE RETENCI'ON

3

F12

HMI_Y1_Z01 F6

Y1_Z01 <Local:2:O.Data[0].2>

SENAL A ROBOT 1

4

F11

Rb1_Z01 <Local:2:O.Data[0].3>

SENAL A ROBOT 2

5

F13

Rb2_Z01 <Local:2:O.Data[0].4>

(End)


0
F12
S3_Z01
<Local:2:I.Data[1].2>
TON
EN
DN
(End)

PASAMOS LOS VALORES DE LAS VARIABLES UTILIZADAS EN EL ACONDICIONAMIENTO DE ENTRADA DE LA SEÑAL DEL SENSOR DE TEMPERATURA

0

MOV
Source SIMUL_INPUT_TCERVEZA 0
Dest V0_T_Z03 0

MOV
Source 100
Dest R 0

MOV
Source 866
Dest R1_Z02 0

MOV
Source 10
Dest Ve 0

MOV
Source 0.003851
Dest a 0

1

CPT
Dest DEGREES_T1_Z02 0
Expression (V0_T_Z03*R)/(Ve*R1_Z02*a)

2

LIM
Low Limit 2
Test DEGREES_T1_Z02 0
High Limit 3

ST1_Z02

LES
Source A DEGREES_T1_Z02 0
Source B 2

ST1_Z02_LOW

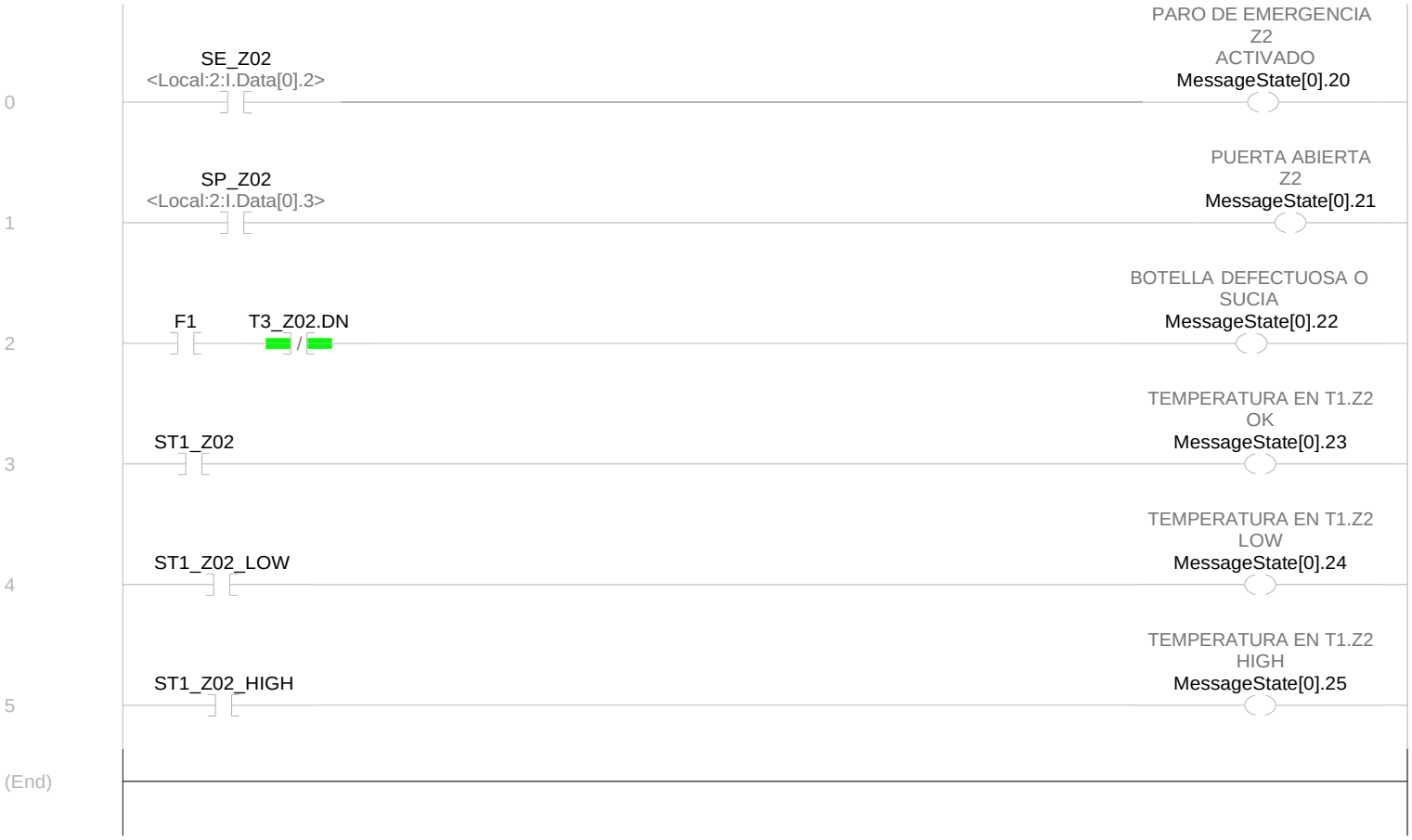
GRT
Source A DEGREES_T1_Z02 0
Source B 3

ST1_Z02_HIGH

(End)

General

Type:	Ladder Diagram	Number of Rungs:	3
In Program:	MainProgram		


SE_Z02
<Local:2:I.Data[0].2>
PARO DE EMERGENCIA Z2 ACTIVADO
MessageState[0].20
0
SP_Z02
<Local:2:I.Data[0].3>
PUERTA ABIERTA Z2
MessageState[0].21
1
F1
T3_Z02.DN
BOTELLA DEFECTUOSA O SUCIA
MessageState[0].22
2
ST1_Z02
TEMPERATURA EN T1.Z2 OK
MessageState[0].23
3
ST1_Z02_LOW
TEMPERATURA EN T1.Z2 LOW
MessageState[0].24
4
ST1_Z02_HIGH
TEMPERATURA EN T1.Z2 HIGH
MessageState[0].25
5
(End)

INFORMATIVE RUNG

0

SE_Z02 <Local:2:I.Data[0].2> SP_Z02 <Local:2:I.Data[0].3> NOP

S1_Z02 <Local:2:I.Data[1].3> S2_Z02 <Local:2:I.Data[1].4> S3_Z02 <Local:2:I.Data[1].5> SVa_Z02 <Local:2:I.Data[1].6>

C1_Z02 <Local:2:O.Data[0].5> C2_Z02 <Local:2:O.Data[0].6> C3_Z02 <Local:2:O.Data[0].7> Y1_Z02 <Local:2:O.Data[0].12>

Mr1_Z02 <Local:2:O.Data[0].8> Mr2_Z02 <Local:2:O.Data[0].9> Cr1_Z02 <Local:2:O.Data[0].10> H1_Z02 <Local:2:O.Data[0].11>

ST1_Z02 ST1_Z02_LOW ST1_Z02_HIGH

ENCENDIDO DE HORNO, DURANTE TODA LA SECUENCIA

1

F20
F21
H1_Z02 <Local:2:O.Data[0].11>

CINTA 1

2

F21
HMI_C1_Z02 F6
C1_Z02 <Local:2:O.Data[0].5>

CINTA 2

3

F21
HMI_C2_Z02 F6
C2_Z02 <Local:2:O.Data[0].6>

CINTA 3

4

F21 T3_Z02.DN
HMI_C3_Z02 F6
C3_Z02 <Local:2:O.Data[0].7>

ACTUADOR DE EXPULSION DE BOTELLA

5

F21 SVa_Z02 <Local:2:I.Data[1].6>
HMI_Y1_Z02 F6
Y1_Z02 <Local:2:O.Data[0].12>

RUEDA DE ENTRADA A CARRUSEL DE LAVADO

6

F21 T1_Z02.DN
Mr1_Z02 <Local:2:O.Data[0].8>

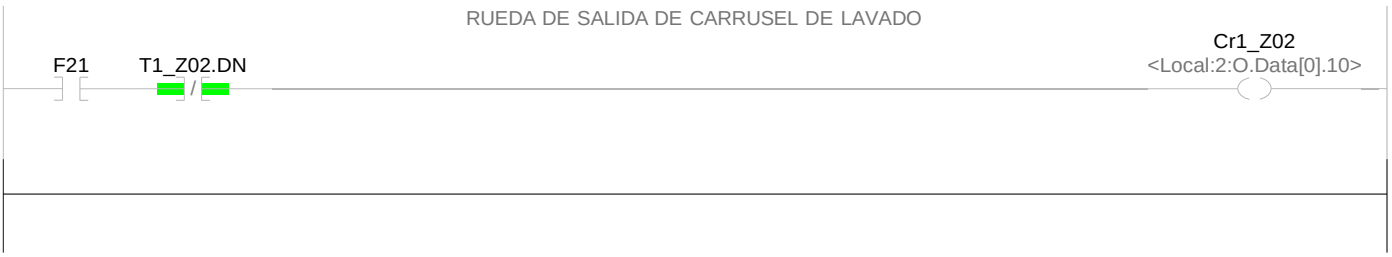
RUEDA DE SALIDA DE CARRUSEL DE LAVADO

7

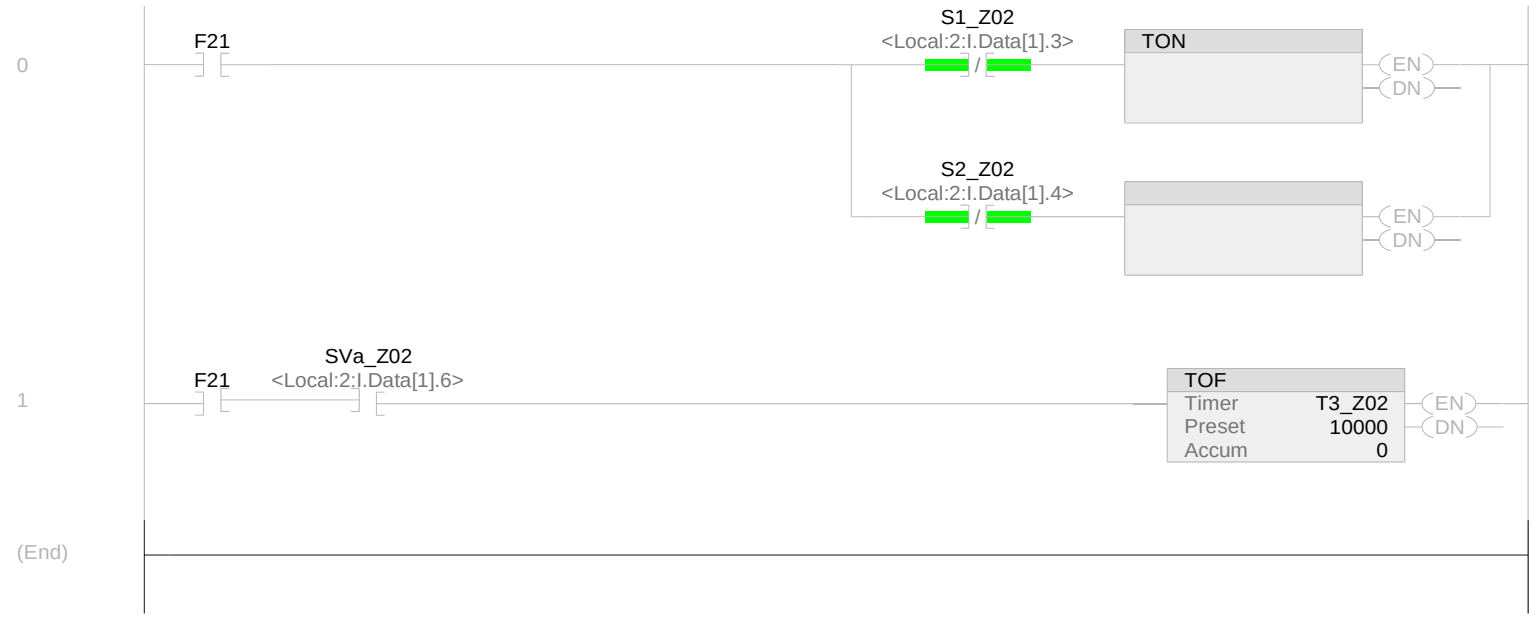
F21 T1_Z02.DN
Mr2_Z02 <Local:2:O.Data[0].9>

RUEDA DE SALIDA DE CARRUSEL DE LAVADO

8



(End)


0
F21
S1_Z02
<Local:2:I.Data[1].3>
TON
EN
DN
S2_Z02
<Local:2:I.Data[1].4>
EN
DN
1
F21
SVa_Z02
<Local:2:I.Data[1].6>
TOF
Timer T3_Z02
Preset 10000
Accum 0
EN
DN
(End)

PASAMOS LOS VALORES DE LAS VARIABLES UTILIZADAS EN EL ACONDICIONAMIENTO DE ENTRADA DE LA SEÑAL DEL SENSOR DE TEMPERATURA

0

MOV
Source SIMUL_INPUT_THORNO 0
Dest V0_T_Z03 0

MOV
Source 100
Dest R 0

MOV
Source 866
Dest R1_Z03 0

MOV
Source 10
Dest Ve 0

MOV
Source 0.003851
Dest a 0

1

CPT
Dest DEGREES_T1_Z03 0
Expression (V0_T_Z03*R)/(Ve*R1_Z02*a)

2

LIM
Low Limit 85
Test DEGREES_T1_Z03 0
High Limit 95

ST1_Z03

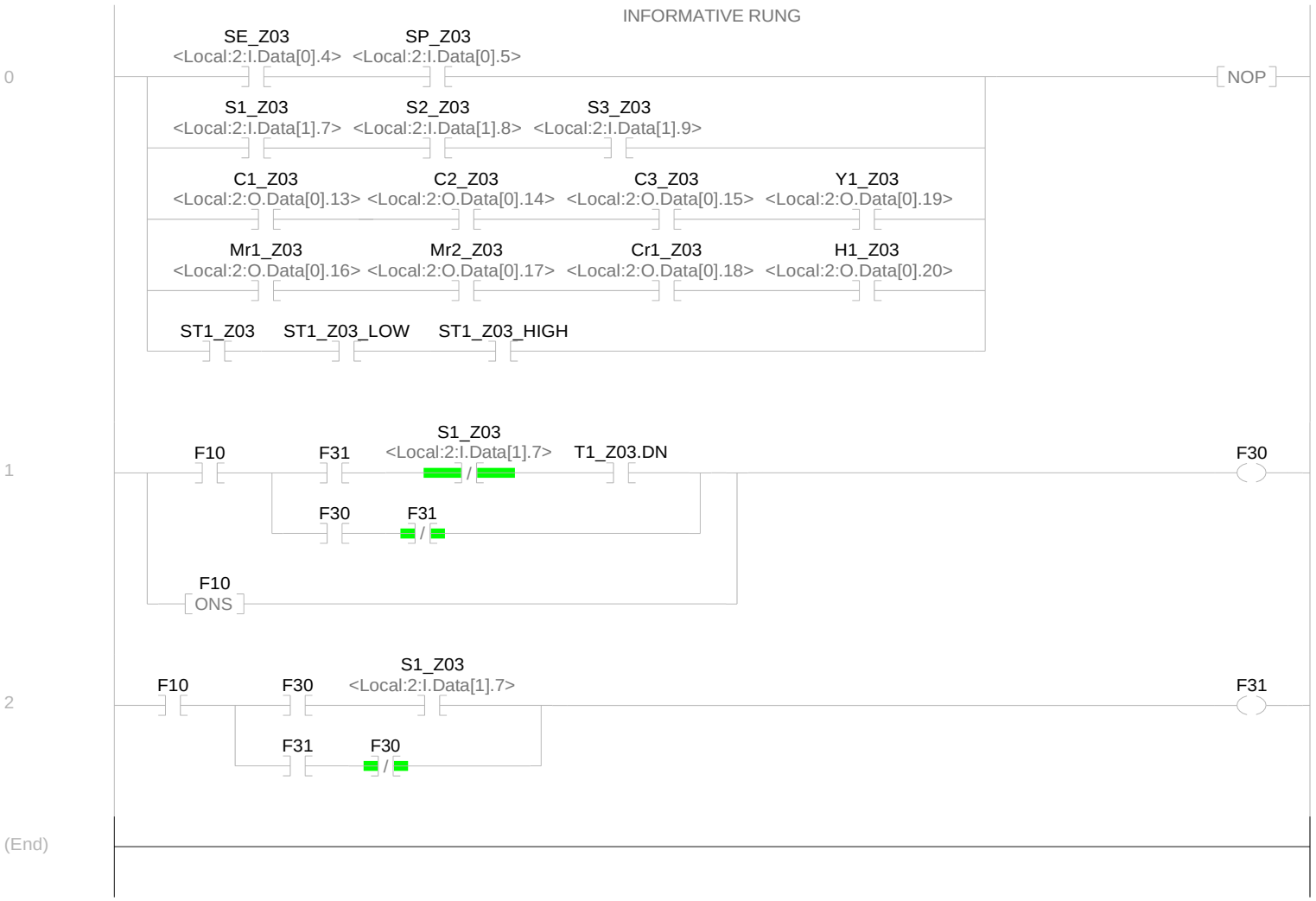
LES
Source A DEGREES_T1_Z03 0
Source B 85

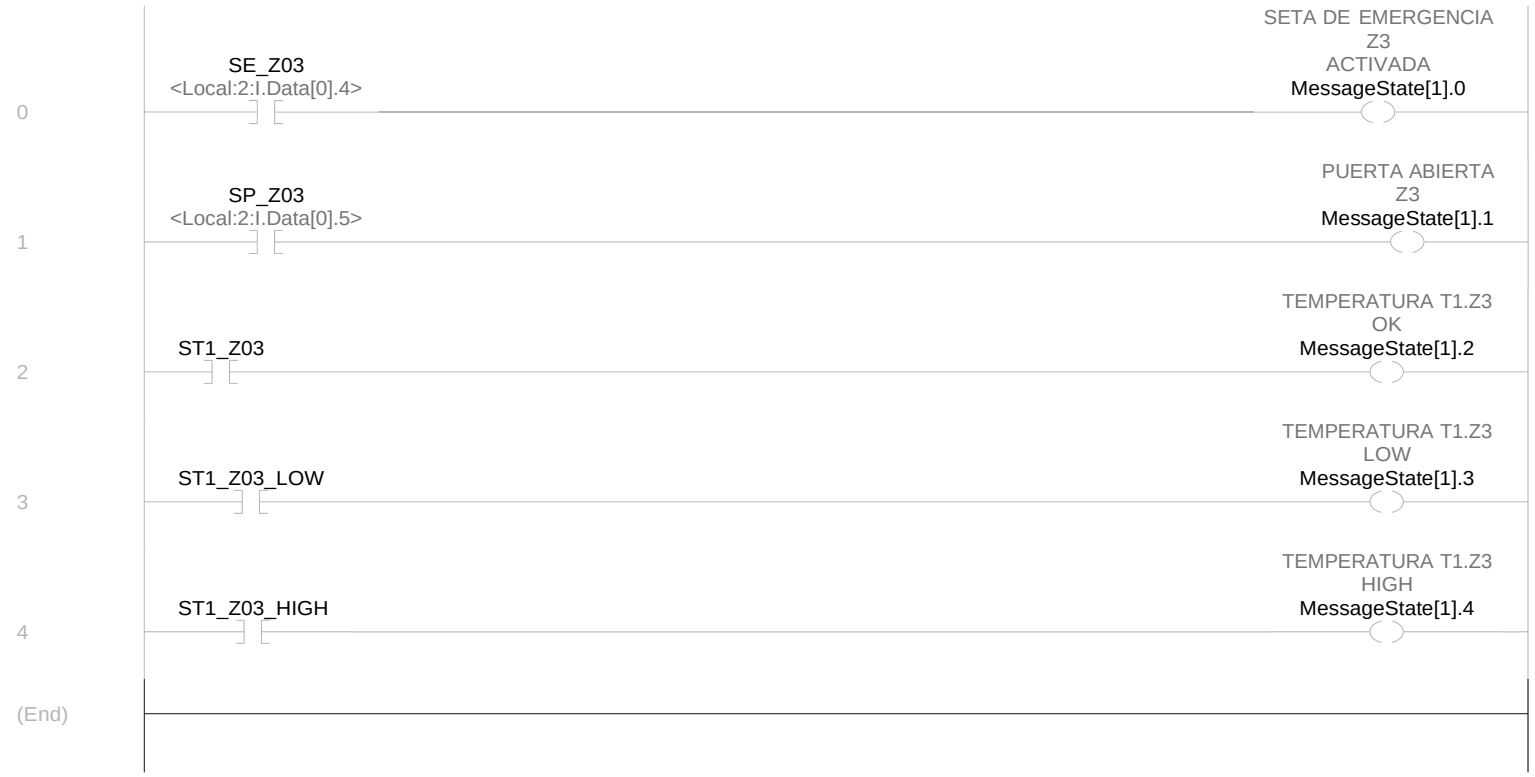
ST1_Z03_LOW

GRT
Source A DEGREES_T1_Z03 0
Source B 95

ST1_Z03_HIGH

(End)


INFORMATIVE RUNG
0
SE_Z03
<Local:2:I.Data[0].4>
SP_Z03
<Local:2:I.Data[0].5>
NOP
S1_Z03
<Local:2:I.Data[1].7>
S2_Z03
<Local:2:I.Data[1].8>
S3_Z03
<Local:2:I.Data[1].9>
C1_Z03
<Local:2:O.Data[0].13>
C2_Z03
<Local:2:O.Data[0].14>
C3_Z03
<Local:2:O.Data[0].15>
Y1_Z03
<Local:2:O.Data[0].19>
Mr1_Z03
<Local:2:O.Data[0].16>
Mr2_Z03
<Local:2:O.Data[0].17>
Cr1_Z03
<Local:2:O.Data[0].18>
H1_Z03
<Local:2:O.Data[0].20>
ST1_Z03
ST1_Z03_LOW
ST1_Z03_HIGH
1
F10
F31
S1_Z03
<Local:2:I.Data[1].7>
T1_Z03.DN
F30
F30
F31
F10
ONS
2
F10
F30
S1_Z03
<Local:2:I.Data[1].7>
F31
F31
F30
(End)


SETA DE EMERGENCIA Z3 ACTIVADA
SE_Z03
<Local:2:I.Data[0].4>
MessageState[1].0
0
PUERTA ABIERTA Z3
SP_Z03
<Local:2:I.Data[0].5>
MessageState[1].1
1
TEMPERATURA T1.Z3 OK
ST1_Z03
MessageState[1].2
2
TEMPERATURA T1.Z3 LOW
ST1_Z03_LOW
MessageState[1].3
3
TEMPERATURA T1.Z3 HIGH
ST1_Z03_HIGH
MessageState[1].4
4
(End)

INFORMATIVE RUNG

0

SE_Z03 <Local:2:I.Data[0].4>
SP_Z03 <Local:2:I.Data[0].5>
NOP
S1_Z03 <Local:2:I.Data[1].7>
S2_Z03 <Local:2:I.Data[1].8>
S3_Z03 <Local:2:I.Data[1].9>
C1_Z03 <Local:2:O.Data[0].13>
C2_Z03 <Local:2:O.Data[0].14>
C3_Z03 <Local:2:O.Data[0].15>
Y1_Z03 <Local:2:O.Data[0].19>
Mr1_Z03 <Local:2:O.Data[0].16>
Mr2_Z03 <Local:2:O.Data[0].17>
Cr1_Z03 <Local:2:O.Data[0].18>
H1_Z03 <Local:2:O.Data[0].20>
ST1_Z03
ST1_Z03_LOW
ST1_Z03_HIGH

HORNO SIEMPRE ENCENDIDO SALVO PARO O MANUAL

1

F30
F31
A1
H1_Z03 <Local:2:O.Data[0].20>

CINTA1

2

F31
HMI_C1_Z03
F6
C1_Z03 <Local:2:O.Data[0].13>

CINTA 2

3

F31
HMI_C2_Z03
F6
C2_Z03 <Local:2:O.Data[0].14>

CINTA 3

4

F31
HMI_C3_Z03
F6
C3_Z03 <Local:2:O.Data[0].15>

RODILLO 1

5

F31
T2_Z03.DN
Mr1_Z03 <Local:2:O.Data[0].16>

RODILLO 2

6

F31
T2_Z03.DN
Mr2_Z03 <Local:2:O.Data[0].17>

CARROUSEL LLENADO

7

F31
T2_Z03.DN
Cr1_Z03 <Local:2:O.Data[0].18>

CORONADO

8

(End)


S2_Z03
F31
<Local:2:I.Data[1].8>
TON
EN
DN
0
(End)

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ETSI Aeroespacial y Diseño Industrial

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela técnica superior de ingeniería del diseño

2. Robot

Trabajo Fin de Grado

Ingeniería electrónica industrial y automática

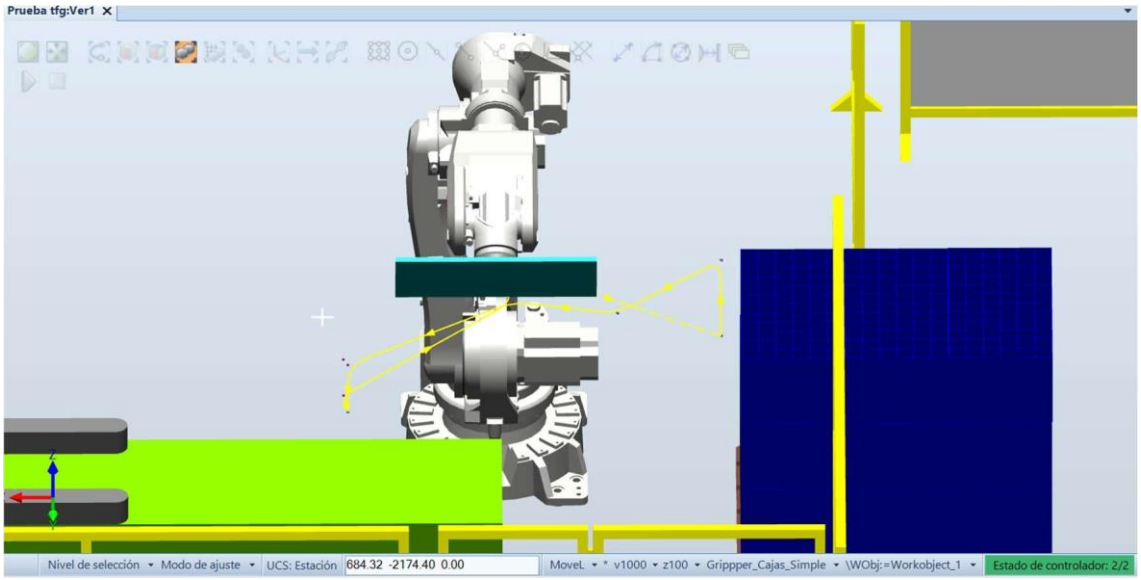
2.1 Robot 1: Despaletizado

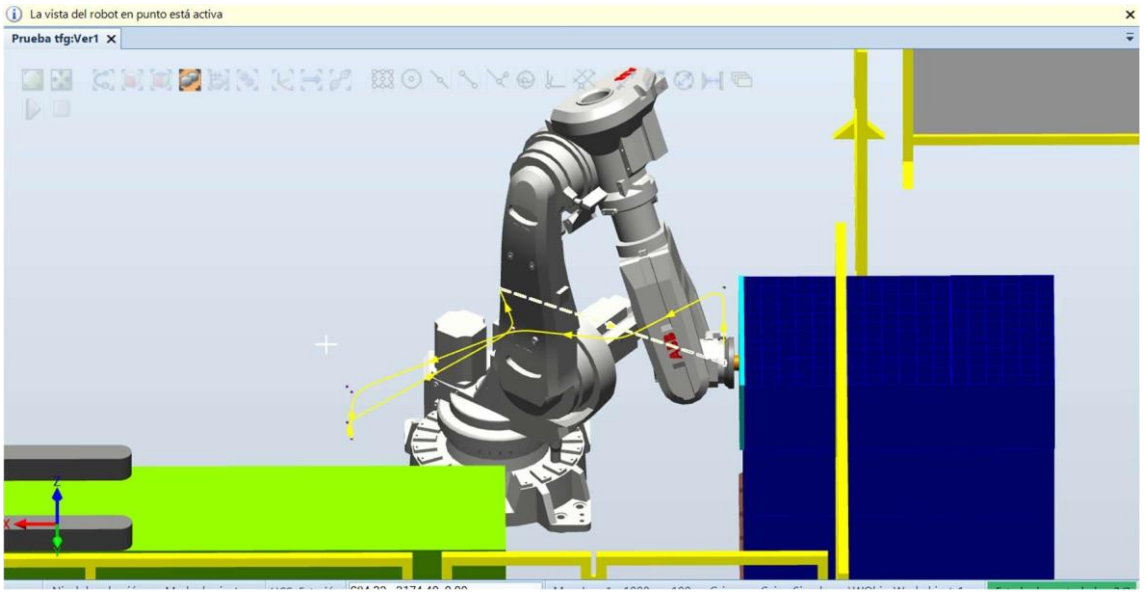
2.1.1 Código de RAPID

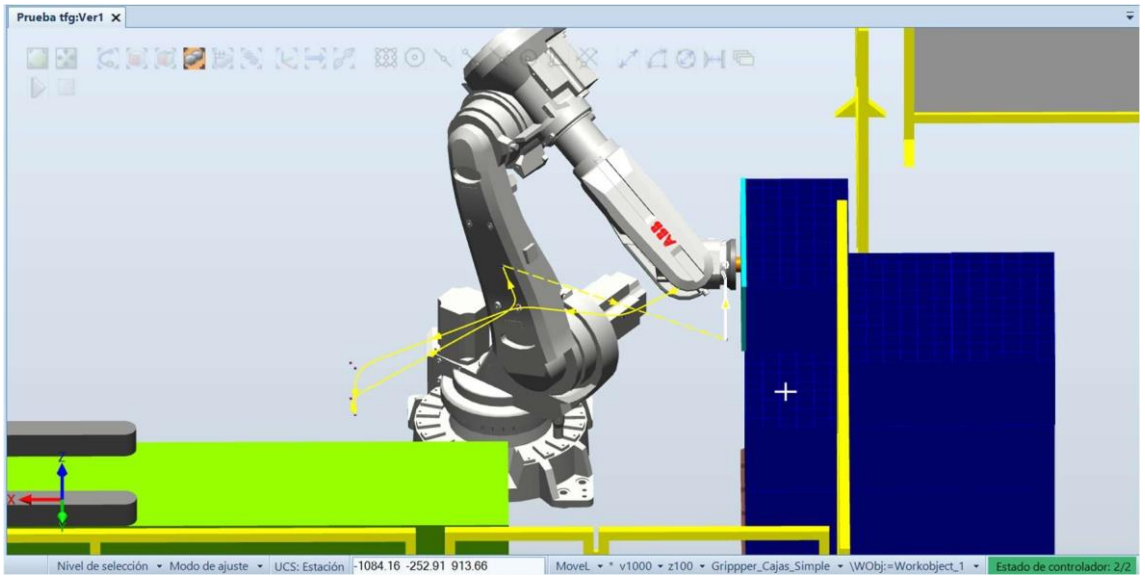
```
MODULE Module1
    CONST robtarget Home:=[[823.531856847,-135.801502315,1775],[0.5,0,0.866025404,0],[-1,0,-
1,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget Pick:=[[16.284552209,946.749467076,902.705002952],[0.499999999,-
0.500000013,-0.499999969,-0.500000019],[0,1,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget Out_zone_1:=[[215.119235679,629.028006139,1464.49963772],[0.499999999,-
0.500000013,-0.499999969,-0.500000019],[0,1,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget
Out_zone_2:=[[558.585701769,524.185200144,1464.49963772],[0.00000001,0.70710679,-
0.000000035,0.707106773],[0,1,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget Out_zone_3:=[[637.018685017,136.249211632,1464.49963772],[-
0.00000001,0.70710679,-
0.000000035,0.707106773],[0,1,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget Out_zone_4:=[[572.189704702,-
517.707185242,1158.696016335],[0.00000001,0.70710679,-
0.000000035,0.707106773],[0,1,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget Out_zone_5:=[[642.688634241,-517.707185242,1039.188178916],[-
0.00000001,0.70710679,-
0.000000035,0.707106773],[0,1,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget Place:=[[680.396454257,-
500.352405695,987.305473798],[0.00000001,0.70710679,-
0.000000035,0.707106773],[0,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
        CONST robtarget
Pick_Out_2:=[[390.546288027,543.114473997,1251.32094764],[0.347697323,-0.615716308,-
0.347697337,-0.6157163],[0,1,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
        CONST robtarget In_Zone_1:=[[680.396454257,-
500.352405695,1204.212231875],[0.00000001,0.70710679,-
0.000000035,0.707106773],[0,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
        CONST robtarget Pick_Out:=[[16.284552209,946.749467076,1251.320587642],[0.499999999,-
0.500000013,-0.499999969,-0.500000019],[0,1,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
    !***********************************************************
    !
    ! Módulo:  Module1
    !
    ! Descripción:
    !  <Introduzca la descripción aquí>
    !
    ! Autor: jorge
    !
    ! Versión: 1.0
    !
    !***********************************************************


    !***********************************************************
    !
    ! Procedimiento Main
    !
    !  Este es el punto de entrada de su programa
    !
    !***********************************************************
    PROC main()

        VAR num Lenght_box := 400;
        VAR num High_box := 300;
```

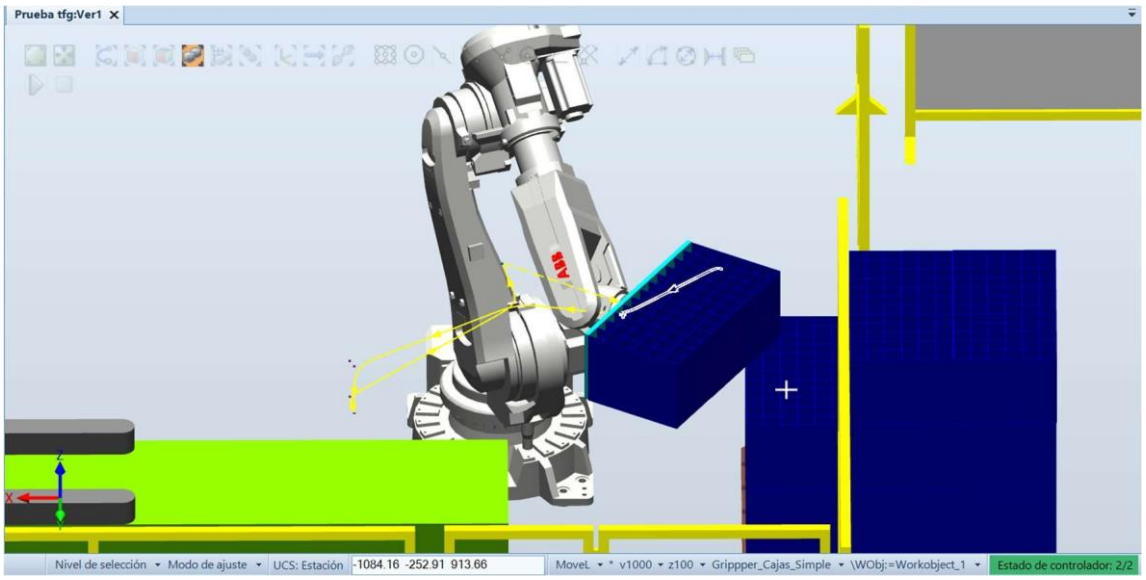
```
    VAR num num1_box := 0;
    VAR num num2_box := 0;

    WHILE 1 DO

      WHILE PLC_OK = 1 DO
        WHILE num2_box < 3 DO
          WHILE num1_box < 3 DO
            Path_10;
            Pick:=[[16.284552209-Lenght_box*num1_box,946.749467076,902.705002952-
High_box*num2_box],[0.499999999,-0.500000013,-0.499999969,-
0.500000019],[0,1,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
            Pick_Out:=[[16.284552209-Lenght_box*num1_box,946.749467076,1251.320587642-
High_box*num2_box],[0.499999999,-0.500000013,-0.499999969,-
0.500000019],[0,1,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
            num1_box := num1_box + 1;
          ENDWHILE
          num2_box := num2_box + 1;
        ENDWHILE
        num1_box := 0;
        num2_box := 0;
      ENDWHILE

    ENDWHILE

  ENDPROC
  PROC Path_10()
    MoveL Home,v1000,z100,Grippper_Cajas_Simple\WObj:=Workobject_1;
    MoveJ Pick,v500,fine,Grippper_Cajas_Simple\WObj:=Workobject_1;
    WaitTime 2;
    SetDO Gripper_Close 1;
    WaitTime 2;
    MoveL Pick_Out,v1000,z100,Grippper_Cajas_Simple\WObj:=Workobject_1;
    MoveL Pick_Out_2,v1000,z100,Grippper_Cajas_Simple\WObj:=Workobject_1;
    MoveL Out_zone_3,v1000,z100,Grippper_Cajas_Simple\WObj:=Workobject_1;
    MoveL In_Zone_1,v1000,z100,Grippper_Cajas_Simple\WObj:=Workobject_1;
    MoveL Place,v500,fine,Grippper_Cajas_Simple\WObj:=Workobject_1;
    WaitTime 2;
    SetDO Gripper_Close 0;
    WaitTime 2;
    MoveL Out_zone_5,v1000,z100,Grippper_Cajas_Simple\WObj:=Workobject_1;
    MoveL Out_zone_3,v1000,z100,Grippper_Cajas_Simple\WObj:=Workobject_1;
    MoveL Home,v1000,z100,Grippper_Cajas_Simple\WObj:=Workobject_1;
  ENDPROC
ENDMODULE
```

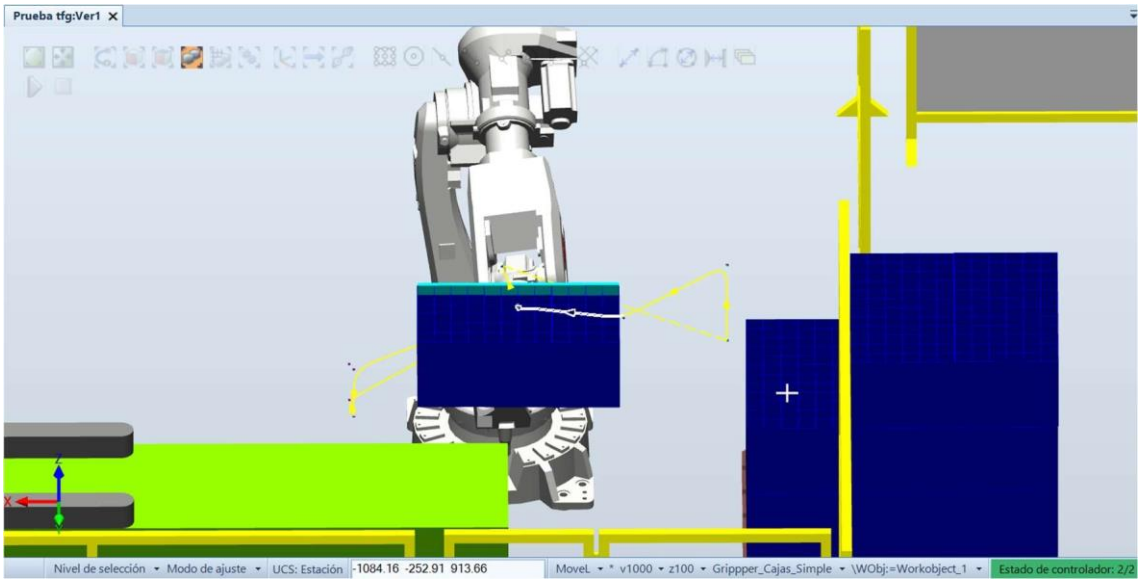
2.1.2 Secuencia de pasos

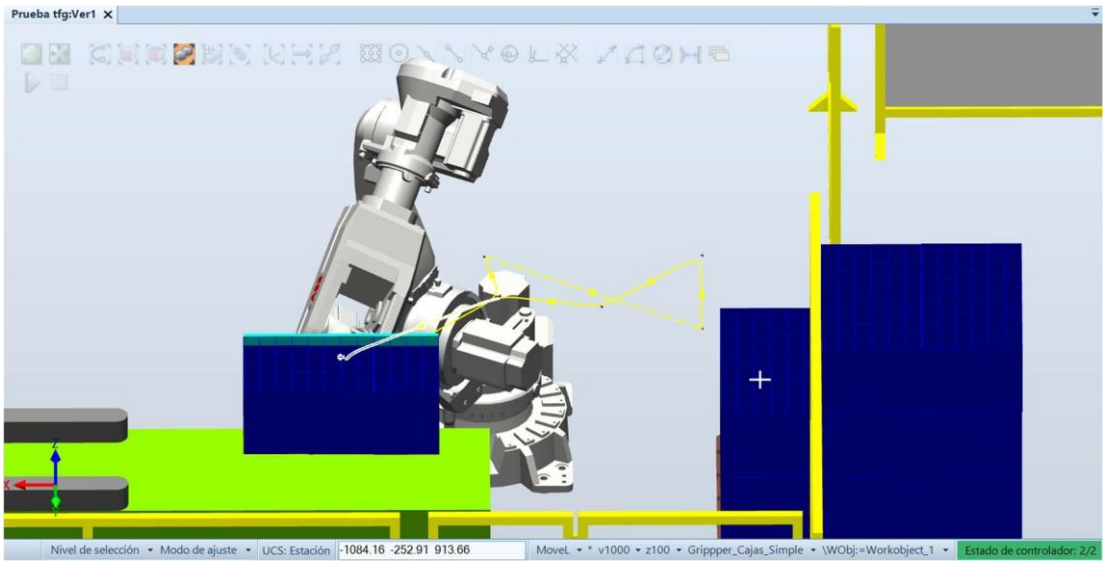


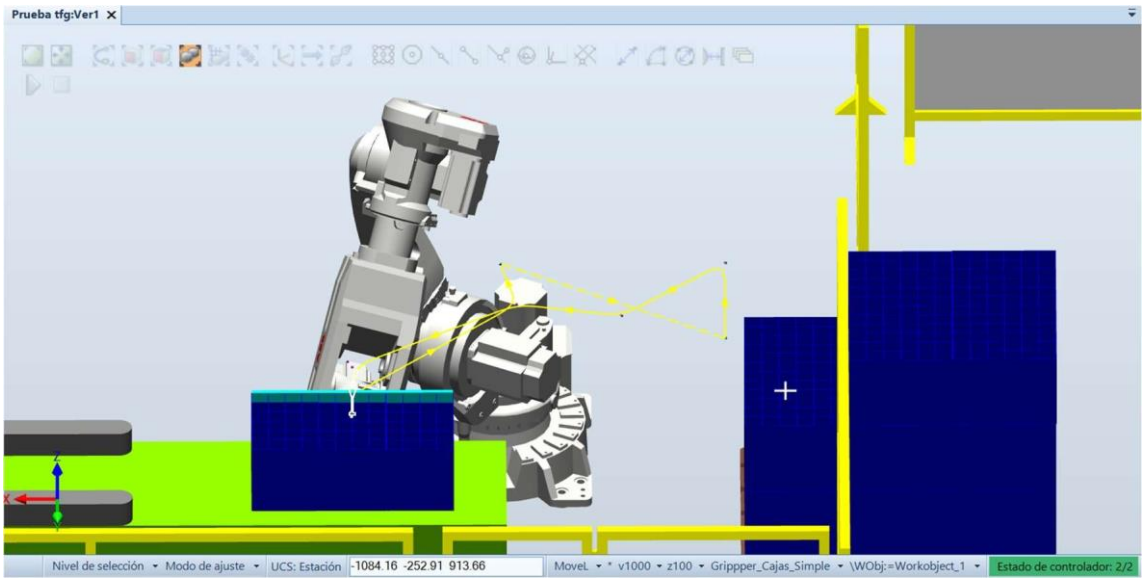


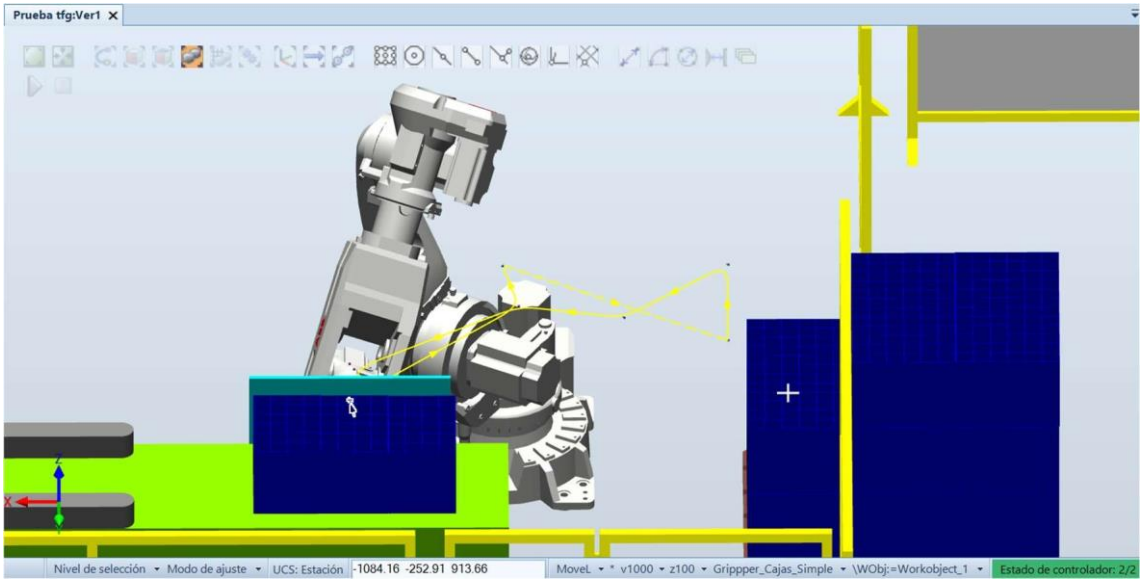


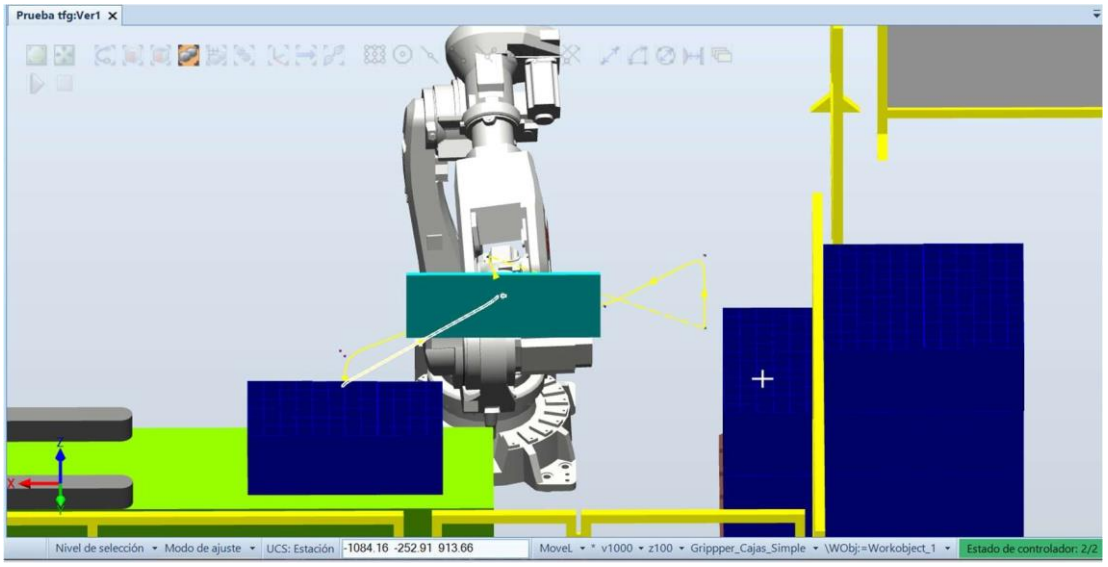

Prueba tfg:Ver1
Nivel de selección
Modo de ajuste
UCS: Estación
-1084.16 -252.91 913.66
MoveL * v1000 z100 Grippper_Cajas_Simple \WObj:=Workobject_1
Estado de controlador: 2/2

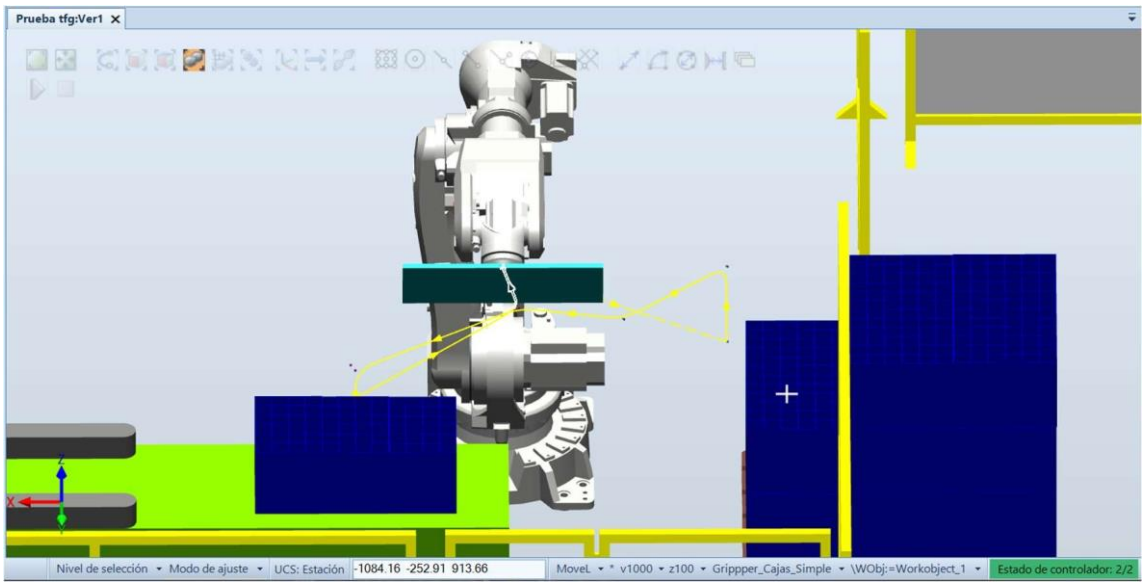

Prueba tfg:Ver1
Nivel de selección
Modo de ajuste
UCS: Estación
-1084.16 -252.91 913.66
MoveL * v1000 z100 Grippper_Cajas_Simple \WObj:=Workobject_1
Estado de controlador: 2/2


Prueba tfg:Ver1
Nivel de selección
Modo de ajuste
UCS: Estación
-1084.16 -252.91 913.66
MoveL * v1000 z100 Grippper_Cajas_Simple \WObj:=Workobject_1
Estado de controlador: 2/2


Prueba tfg:Ver1
Nivel de selección
Modo de ajuste
UCS: Estación
-1084.16 -252.91 913.66
MoveL * v1000 z100 Grippper_Cajas_Simple \WObj:=Workobject_1
Estado de controlador: 2/2


Prueba tfg:Ver1
Nivel de selección
Modo de ajuste
UCS: Estación
-1084.16 -252.91 913.66
MoveL * v1000 z100 Grippper_Cajas_Simple \WObj:=Workobject_1
Estado de controlador: 2/2


Prueba tfg:Ver1
Nivel de selección
Modo de ajuste
UCS: Estación
-1084.16 -252.91 913.66
MoveL * v1000 z100 Grippper_Cajas_Simple \WObj:=Workobject_1
Estado de controlador: 2/2



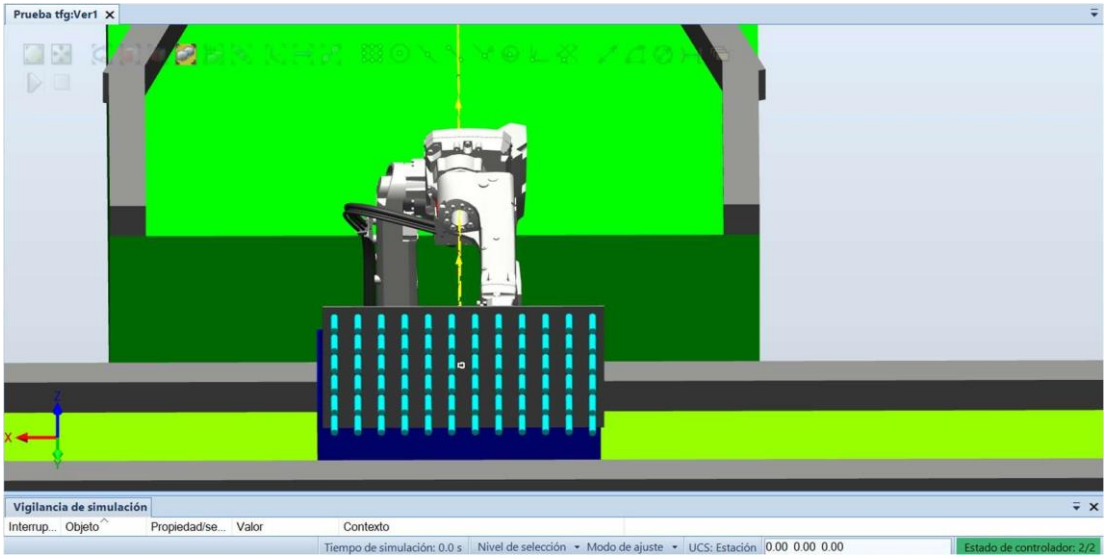
2.2 Robot 2: Extracción de botellas

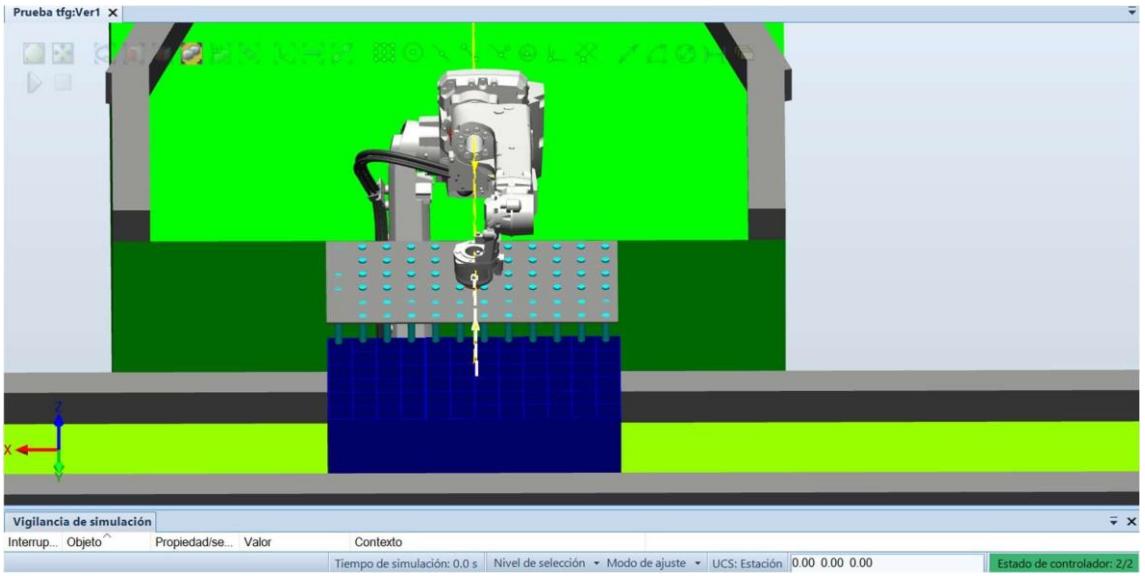
2.2.1 Codigo de RAPID

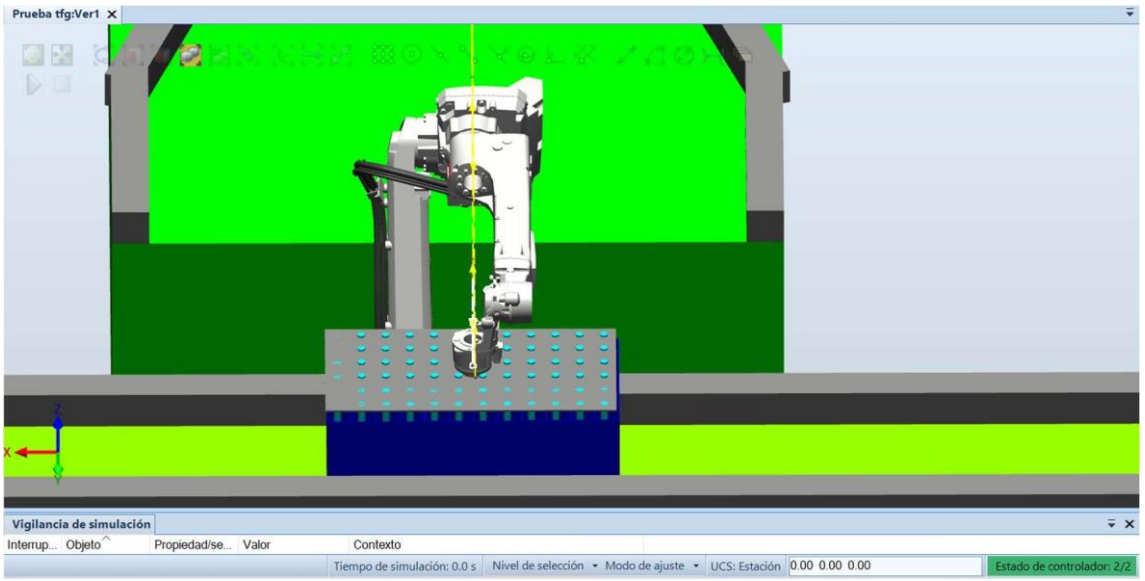
```
MODULE Module1
    CONST robtarget
Home:=[[1288,0,1495],[0.5,0.5,0.5,0.5],[0,0,1,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget In_Out_Zone_Pick:=[[732.137245126,-3.57387097,1446.874266219],[-
0.000000013,0.707106838,-0.707106724,-0.000000013],[-1,0,-
2,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget Pick:=[[732.137190641,-3.573871227,1172.443099875],[0,-
0.707106781,0.707106781,0],[0,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget Out_Zone_Pick_2:=[[732.137190641,-3.573871227,1511.936765795],[0,-
0.707106781,0.707106781,0],[0,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget In_Zone_Place:=[[-727.170318016,-0.000063571,1531.297132447],[0.008029866,-
0.705333412,-0.708785191,0.007990762],[2,-1,-2,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget Place:=[[-727.170318016,-0.000063571,1147.016423261],[0.008029866,-
0.705333412,-0.708785191,0.007990762],[2,-1,-2,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
    CONST robtarget In_Out_Zone_Place:=[[-727.170318016,-
0.000063571,1531.297132447],[0.008029866,-0.705333412,-0.708785191,0.007990762],[2,-1,-
2,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
    !***********************************************************
    !
    ! Módulo:  Module1
    !
    ! Descripción:
    !  <Introduzca la descripción aquí>
    !
    ! Autor: jorgi
    !
    ! Versión: 1.0
    !
    !***********************************************************


    !***********************************************************
    !
    ! Procedimiento Main
    !
    !  Este es el punto de entrada de su programa
```

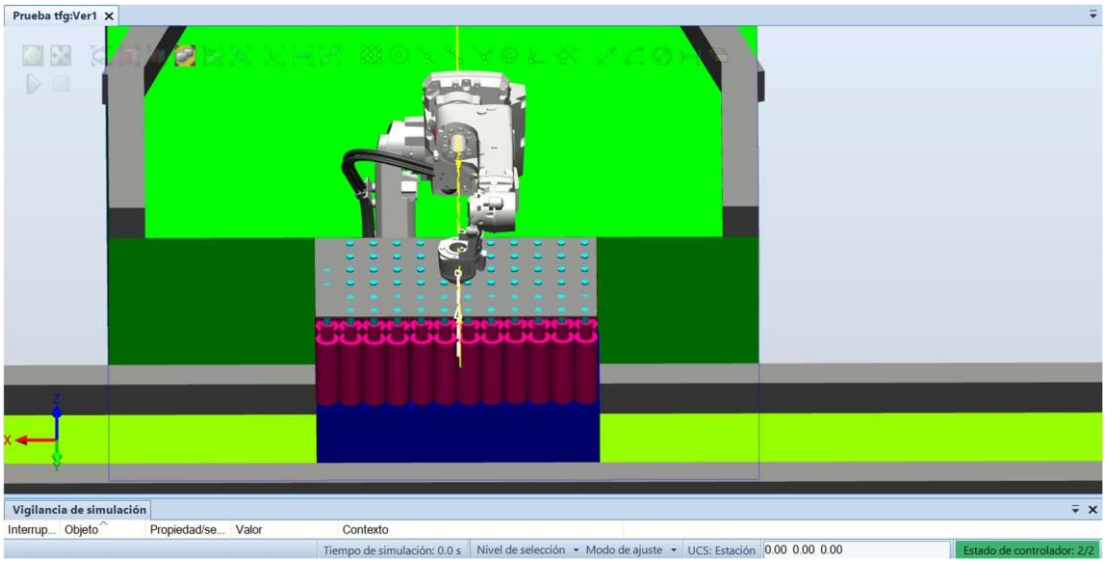
```
!
!*************************************************************
PROC main()
    !Añada aquí su código
    WHILE 1 DO
        WHILE PLC_OK2 = 1 DO
            Path_10;
        ENDWHILE
    ENDWHILE
ENDPROC
PROC Path_10()
    MoveJ Home,v1000,z100,MyNewTool\WObj:=Workobject_1;
    MoveJ In_Out_Zone_Pick,v1000,z100,MyNewTool\WObj:=Workobject_1;
    MoveJ Pick,v1000,z100,MyNewTool\WObj:=Workobject_1;
    WaitTime 2;
    SetDO Gripper2 1;
    WaitTime 2;
    MoveJ In_Out_Zone_Pick,v1000,z100,MyNewTool\WObj:=Workobject_1;
    MoveJ Out_Zone_Pick_2,v1000,z100,MyNewTool\WObj:=Workobject_1;
    MoveJ In_Out_Zone_Place,v1000,z100,MyNewTool\WObj:=Workobject_1;
    MoveL Place,v1000,z100,MyNewTool\WObj:=Workobject_1;
    WaitTime 2;
    SetDO Gripper2 0;
    WaitTime 2;
    MoveL In_Out_Zone_Place,v1000,z100,MyNewTool\WObj:=Workobject_1;
    MoveJ Home,v1000,z100,MyNewTool\WObj:=Workobject_1;
ENDPROC
ENDMODULE
```

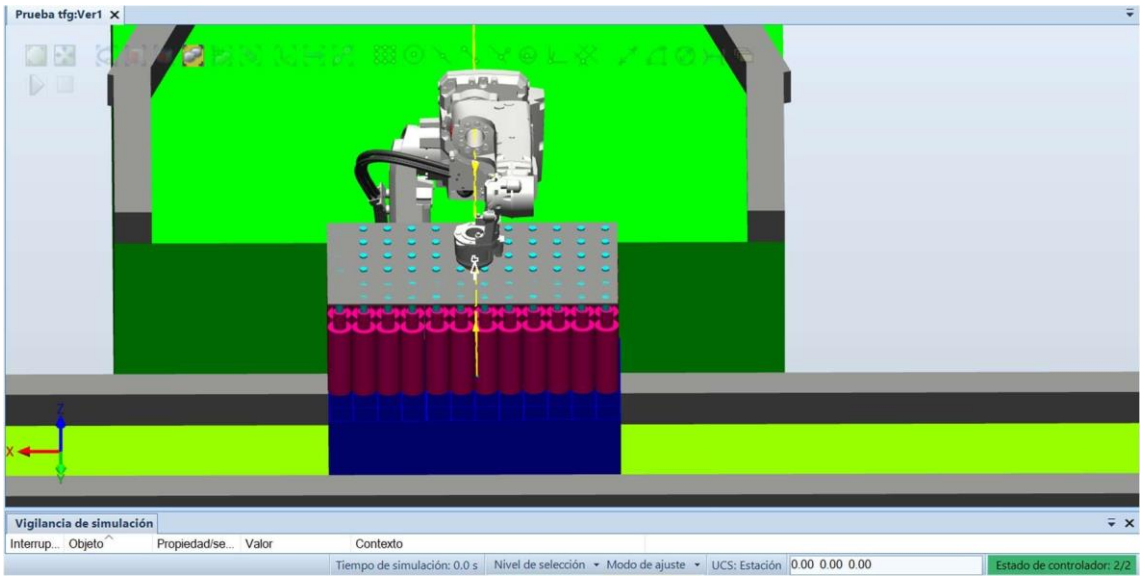
2.2.2 Secuencia de pasos



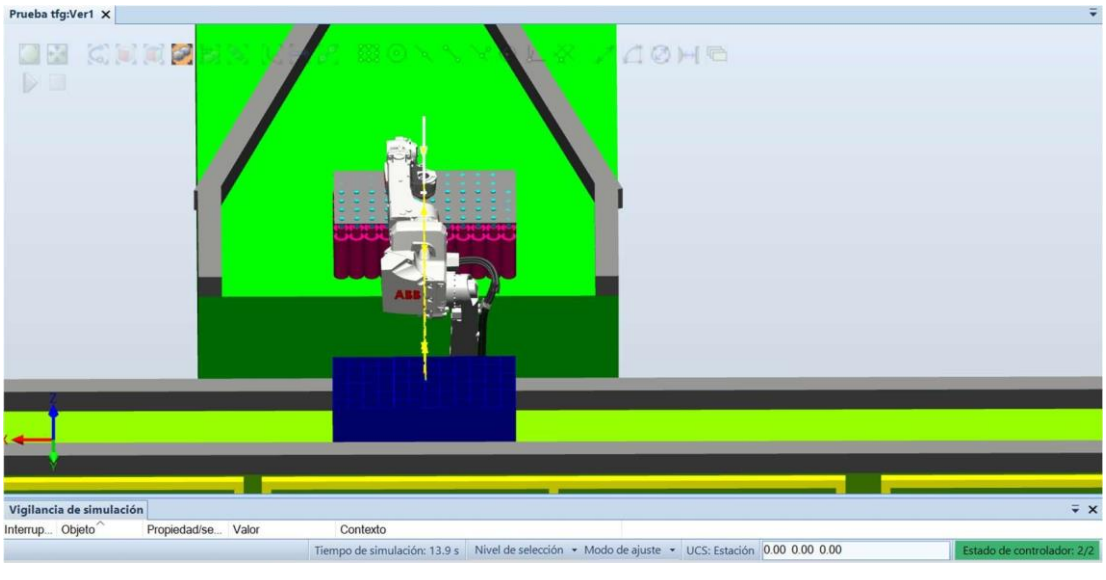

Prueba tfg:Ver1
Vigilancia de simulación
Interrup... Objeto Propiedad/se... Valor Contexto
Tiempo de simulación: 0.0 s Nivel de selección Modo de ajuste UCS: Estación 0.00 0.00 0.00 Estado de controlador: 2/2

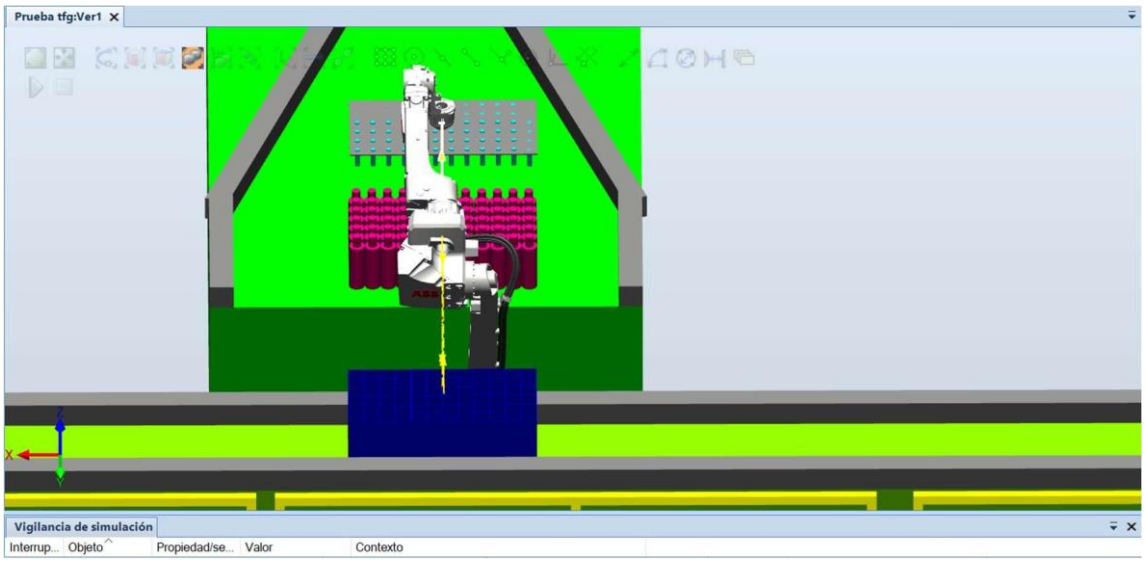

Prueba tfg:Ver1
Vigilancia de simulación
Interrup... Objeto Propiedad/se... Valor Contexto
Tiempo de simulación: 0.0 s Nivel de selección Modo de ajuste UCS: Estación 0.00 0.00 0.00 Estado de controlador: 2/2

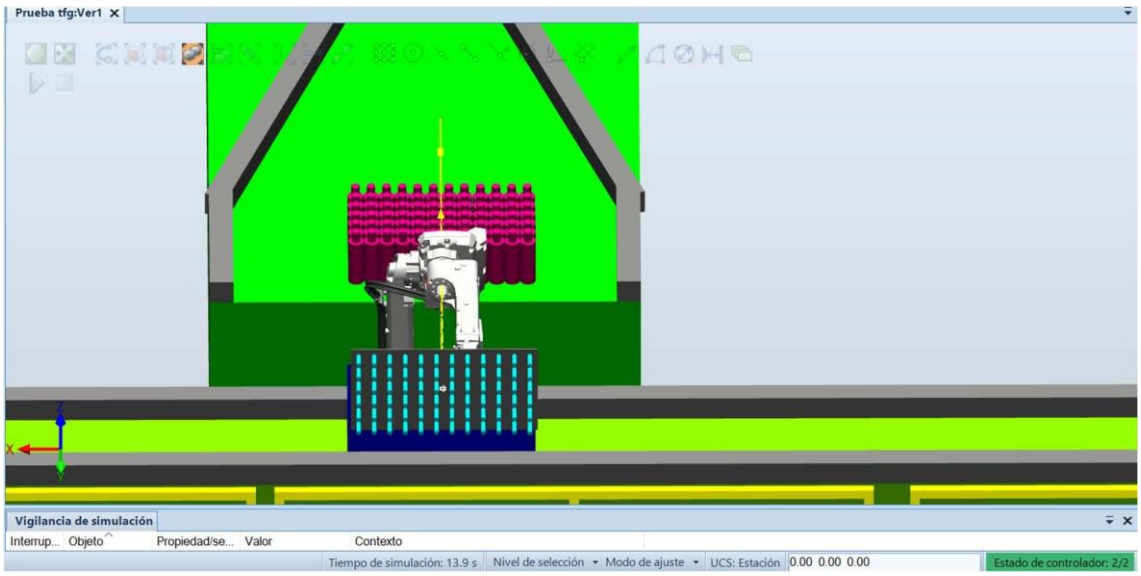

Prueba tfg:Ver1
Vigilancia de simulación
Interrup... Objeto Propiedad/se... Valor Contexto
Tiempo de simulación: 0.0 s Nivel de selección Modo de ajuste UCS: Estación 0.00 0.00 0.00 Estado de controlador: 2/2


Prueba tfg:Ver1
Vigilancia de simulación
Interrup... Objeto Propiedad/se... Valor Contexto
Tiempo de simulación: 0.0 s
Nivel de selección
Modo de ajuste
UCS: Estación
0.00 0.00 0.00
Estado de controlador: 2/2


Prueba tfg:Ver1
Vigilancia de simulación
Interrup... Objeto Propiedad/se... Valor Contexto
Tiempo de simulación: 13.9 s
Nivel de selección
Modo de ajuste
UCS: Estación
0.00 0.00 0.00
Estado de controlador: 2/2


Prueba tfg:Ver1
Vigilancia de simulación
Interrup... Objeto Propiedad/se... Valor Contexto
Tiempo de simulación: 13.9 s
Nivel de selección
Modo de ajuste
UCS: Estación
0.00 0.00 0.00
Estado de controlador: 2/2


Prueba tfg:Ver1
Vigilancia de simulación
Interrup...
Objeto
Propiedad/se...
Valor
Contexto


Prueba tfg:Ver1
Vigilancia de simulación
Interrup...
Objeto
Propiedad/se...
Valor
Contexto
Tiempo de simulación: 13.9 s
Nivel de selección
Modo de ajuste
UCS: Estación
0.00 0.00 0.00
Estado de controlador: 2/2

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ETSI Aeroespacial y Diseño Industrial

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela técnica superior de ingeniería del diseño

3. HMI

Trabajo Fin de Grado

Ingeniería electrónica industrial y automática

Escuela Técnica Superior de Ingeniería del Diseño

11/06/2024 14:11:18

AUTOMATICO

MANUAL

Escuela Técnica Superior de Ingeniería del Diseño

11/06/2024 14:12:13

START

STOP

Trigger Page

Trigger
Trigger Name : {[PLC]MessageState[0].10}
Trigger Type : Bit
Ack all value : 0
Trigger handshake :
Acknowledge :
Remote Ack :
Remote Ack handshake :

Trigger
Trigger Name : {[PLC]MessageState[0].20}
Trigger Type : Bit
Ack all value : 0
Trigger handshake :
Acknowledge :
Remote Ack :
Remote Ack handshake :

Trigger
Trigger Name : {[PLC]MessageState[1].0}
Trigger Type : Bit
Ack all value : 0
Trigger handshake :
Acknowledge :
Remote Ack :
Remote Ack handshake :

Trigger
Trigger Name : {[PLC]MessageState[0].11}
Trigger Type : Bit
Ack all value : 0
Trigger handshake :
Acknowledge :
Remote Ack :
Remote Ack handshake :

Trigger
Trigger Name : {[PLC]MessageState[0].21}
Trigger Type : Bit
Ack all value : 0
Trigger handshake :
Acknowledge :
Remote Ack :
Remote Ack handshake :

Trigger
Trigger Name : {[PLC]MessageState[1].11}
Trigger Type : Bit
Ack all value : 0
Trigger handshake :
Acknowledge :
Remote Ack :
Remote Ack handshake :

Trigger
Trigger Name : {[PLC]MessageState[0].12}
Trigger Type : Bit
Ack all value : 0
Trigger handshake :
Acknowledge :
Remote Ack :

Remote Ack handshake :

Trigger
Trigger Name : {[PLC]MessageState[0].22}
Trigger Type : Bit
Ack all value : 0
Trigger handshake :
Acknowledge :
Remote Ack :
Remote Ack handshake :

Trigger
Trigger Name : {[PLC]MessageState[0].23}
Trigger Type : Bit
Ack all value : 0
Trigger handshake :
Acknowledge :
Remote Ack :
Remote Ack handshake :

Trigger
Trigger Name : {[PLC]MessageState[0].24}
Trigger Type : Bit
Ack all value : 0
Trigger handshake :
Acknowledge :
Remote Ack :
Remote Ack handshake :

Trigger
Trigger Name : {[PLC]MessageState[0].25}
Trigger Type : Bit
Ack all value : 0
Trigger handshake :
Acknowledge :
Remote Ack :
Remote Ack handshake :

Trigger
Trigger Name : {[PLC]MessageState[1].2}
Trigger Type : Bit
Ack all value : 0
Trigger handshake :
Acknowledge :
Remote Ack :
Remote Ack handshake :

Trigger
Trigger Name : {[PLC]MessageState[1].3}
Trigger Type : Bit
Ack all value : 0
Trigger handshake :
Acknowledge :
Remote Ack :
Remote Ack handshake :

Trigger
Trigger Name : {[PLC]MessageState[1].4}
Trigger Type : Bit
Ack all value : 0
Trigger handshake :
Acknowledge :

Remote Ack :
Remote Ack handshake :

Message Page
Use Alarm Identifier : Off
Message
Trigger : {[PLC]MessageState[0].10}
Trigger Value : 1
Message : SETA EMERGENCIA ACTIVADA EN ZONA 1
Identifier : 1
Display : On
Audio : Off
Print : Off
Background : Red = 128, Green = 0, Blue = 0
Foreground : Red = 255, Green = 255, Blue = 255

Message
Trigger : {[PLC]MessageState[0].20}
Trigger Value : 1
Message : SETA EMERGENCIA ACTIVADA EN ZONA 2
Identifier : 2
Display : On
Audio : Off
Print : Off
Background : Red = 128, Green = 0, Blue = 0
Foreground : Red = 255, Green = 255, Blue = 255

Message
Trigger : {[PLC]MessageState[1].0}
Trigger Value : 1
Message : SETA EMERGENCIA ACTIVADA EN ZONA 3
Identifier : 3
Display : On
Audio : Off
Print : Off
Background : Red = 128, Green = 0, Blue = 0
Foreground : Red = 255, Green = 255, Blue = 255

Message
Trigger : {[PLC]MessageState[0].11}
Trigger Value : 1
Message : PUERTA ABIERTA EN ZONA 1
Identifier : 4
Display : On
Audio : Off
Print : Off
Background : Red = 128, Green = 0, Blue = 0
Foreground : Red = 255, Green = 255, Blue = 255

Message
Trigger : {[PLC]MessageState[0].21}
Trigger Value : 1
Message : PUERTA ABIERTA EN ZONA 2
Identifier : 5
Display : On
Audio : Off
Print : Off
Background : Red = 128, Green = 0, Blue = 0
Foreground : Red = 255, Green = 255, Blue = 255

Message

Trigger : {[PLC]MessageState[1].11}
Trigger Value : 1
Message : PUERTA ABIERTA EN ZONA 3
Identifier : 6
Display : On
Audio : Off
Print : Off
Background : Red = 128, Green = 0, Blue = 0
Foreground : Red = 255, Green = 255, Blue = 255

Message
Trigger : {[PLC]MessageState[0].12}
Trigger Value : 1
Message : FALTAN CAJAS EN LA ZONA DE DESPALETIZADO
Identifier : 7
Display : On
Audio : Off
Print : Off
Background : Red = 128, Green = 0, Blue = 0
Foreground : Red = 255, Green = 255, Blue = 255

Message
Trigger : {[PLC]MessageState[0].22}
Trigger Value : 1
Message : BOTELLA DEFECTUOSA O SUCIA
Identifier : 8
Display : On
Audio : Off
Print : Off
Background : Red = 128, Green = 0, Blue = 0
Foreground : Red = 255, Green = 255, Blue = 255

Message
Trigger : {[PLC]MessageState[0].23}
Trigger Value : 1
Message : TEMPERATURA DEL TANQUE OPTIMA
Identifier : 10
Display : On
Audio : Off
Print : Off
Background : Red = 128, Green = 0, Blue = 0
Foreground : Red = 255, Green = 255, Blue = 255

Message
Trigger : {[PLC]MessageState[0].24}
Trigger Value : 1
Message : TEMPERATURA DEL TANQUE INFERIOR
Identifier : 11
Display : On
Audio : Off
Print : Off
Background : Red = 128, Green = 0, Blue = 0
Foreground : Red = 255, Green = 255, Blue = 255

Message
Trigger : {[PLC]MessageState[0].25}
Trigger Value : 1
Message : TEMPERATURA DEL TANQUE SUPERIOR
Identifier : 12
Display : On
Audio : Off
Print : Off

Background : Red = 128, Green = 0, Blue = 0
Foreground : Red = 255, Green = 255, Blue = 255

Message
Trigger : {[PLC]MessageState[1].2}
Trigger Value : 1
Message : TEMPERATURA DEL HORNO OPTIMA
Identifier : 13
Display : On
Audio : Off
Print : Off
Background : Red = 128, Green = 0, Blue = 0
Foreground : Red = 255, Green = 255, Blue = 255

Message
Trigger : {[PLC]MessageState[1].3}
Trigger Value : 1
Message : TEMPERATURA DEL HORNO INFERIOR
Identifier : 14
Display : On
Audio : Off
Print : Off
Background : Red = 128, Green = 0, Blue = 0
Foreground : Red = 255, Green = 255, Blue = 255

Message
Trigger : {[PLC]MessageState[1].4}
Trigger Value : 1
Message : TEMPERATURA DEL HORNO SUPERIOR
Identifier : 15
Display : On
Audio : Off
Print : Off
Background : Red = 128, Green = 0, Blue = 0
Foreground : Red = 255, Green = 255, Blue = 255

Advanced Page
Display
Alarm window : [ALARM]
History
History size : 128
Time intervals
Notification hold time : 250ms
Reset hold time : 250ms
Acknowledge notification : operator & remote
Control
Silence :
Remote silence :
Remote global ack :
Global ack handshake :

Escuela Técnica Superior de Ingeniería del Diseño

11/06/2024 14:12:28

ZONA 1

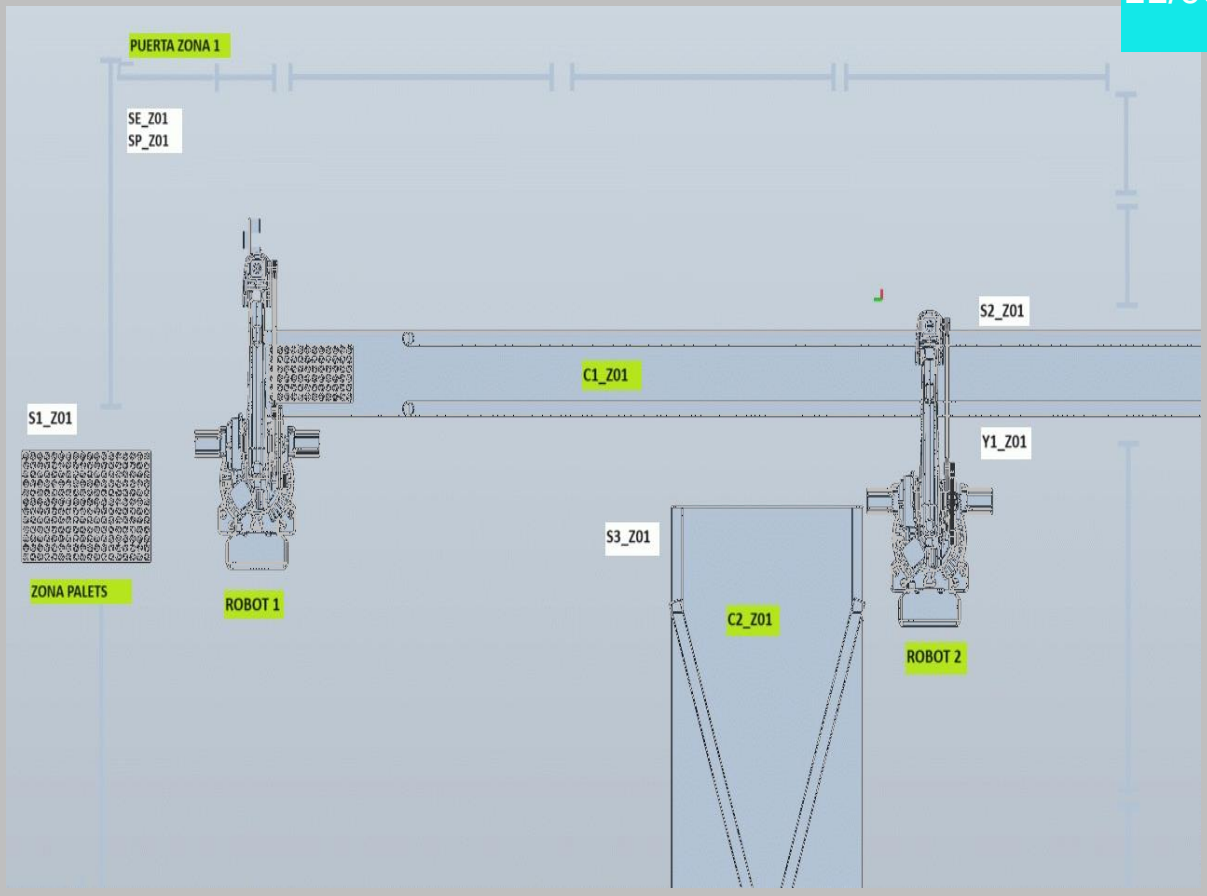
ZONA 3

ZONA 2

START

STOP

11/06/2024 14:12:42



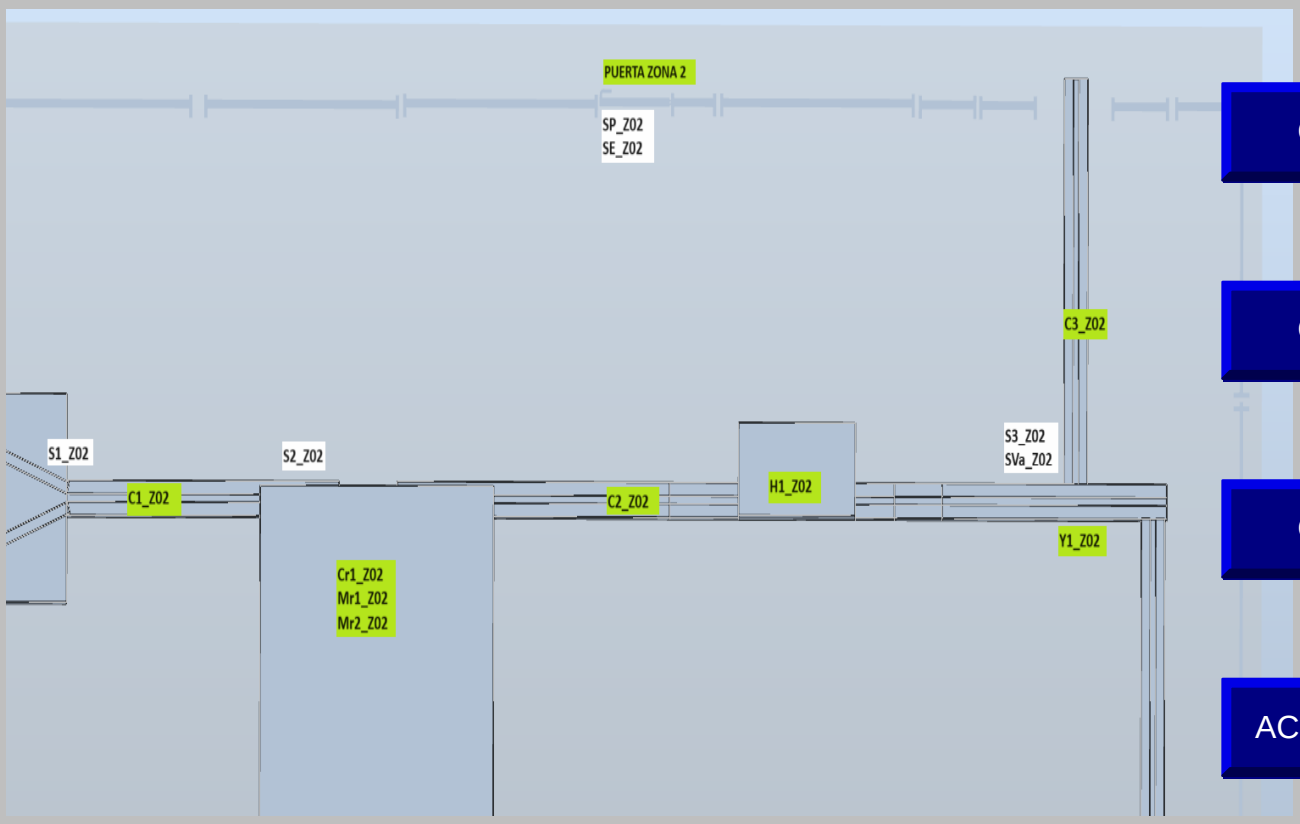
CINTA 1

CINTA 2

ACTUADOR 1

VOLVER

11/06/2024 14:12:55



CINTA 1

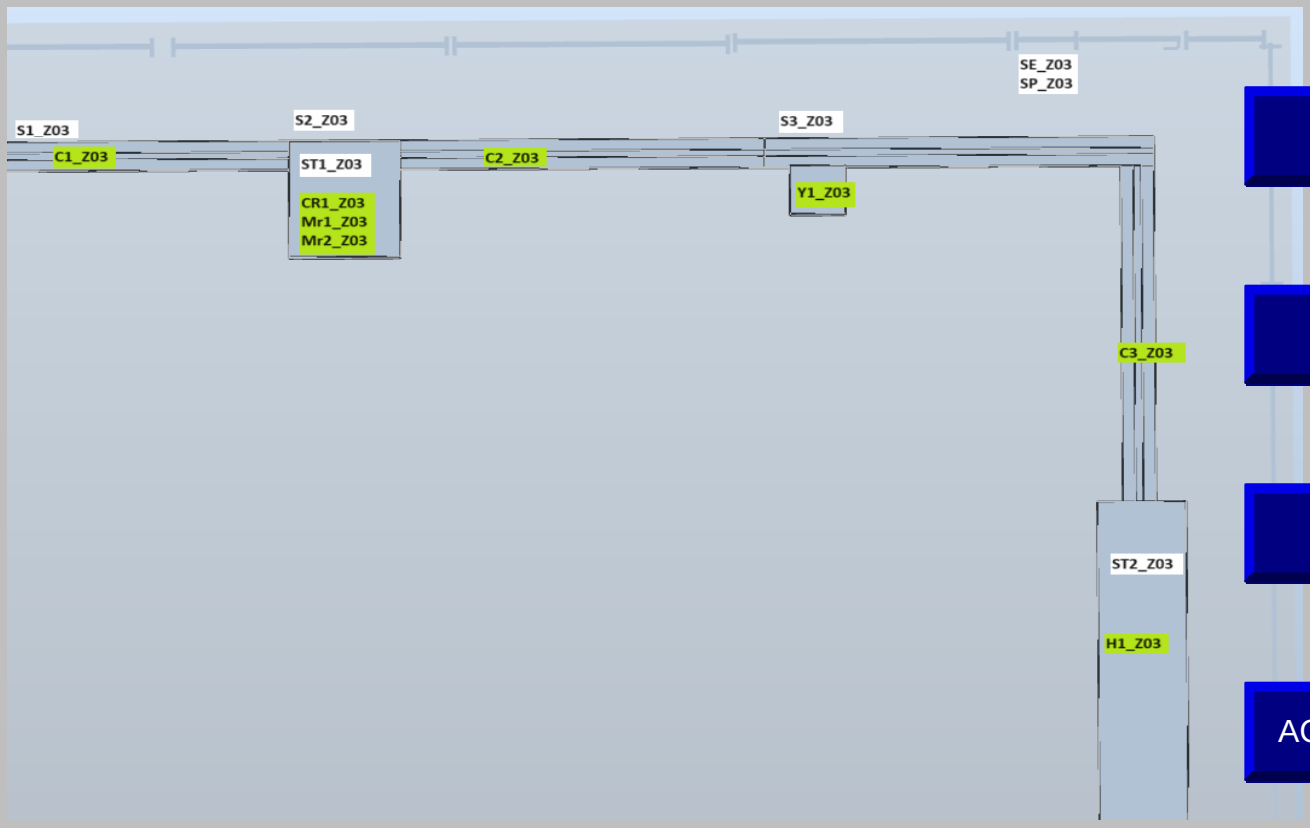
CINTA 2

CINTA 3

ACTUADOR 1

VOLVER

12/06/2024 7:13:04



CINTA 1

CINTA 2

CINTA 3

ACTUADOR 1

VOLVER

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela técnica superior de ingeniería del diseño

ANEXO III: ODS

Trabajo Fin de Grado

Ingeniería electrónica industrial y automática

ANEXO I. RELACIÓN DEL TRABAJO CON LOS OBJETIVOS DE DESARROLLO SOSTENIBLE DE LA AGENDA 2030

Anexo al Trabajo de Fin de Grado y Trabajo de Fin de Máster: Relación del trabajo con los Objetivos de Desarrollo Sostenible de la agenda 2030.

Grado de relación del trabajo con los Objetivos de Desarrollo Sostenible (ODS).

Objetivos de Desarrollo Sostenibles	**Alto**	Medio	**Bajo**	**No Procede**
ODS 1. **Fin de la pobreza.**				X
ODS 2. **Hambre cero.**				X
ODS 3. **Salud y bienestar.**			X	
ODS 4. **Educación de calidad.**				X
ODS 5. **Igualdad de género.**				X
ODS 6. **Agua limpia y saneamiento.**				X
ODS 7. **Energía asequible y no contaminante.**				X
ODS 8. **Trabajo decente y crecimiento económico.**		X		
ODS 9. **Industria, innovación e infraestructuras.**	X			
ODS 10. **Reducción de las desigualdades**.				X
ODS 11. **Ciudades y comunidades sostenibles**.			X	
ODS 12. **Producción y consumo responsables.**	X			
ODS 13. **Acción por el clima.**			X	
ODS 14. **Vida submarina.**				X
ODS 15. **Vida de ecosistemas terrestres.**				X
ODS 16. **Paz, justicia e instituciones sólidas.**				X
ODS 17. **Alianzas para lograr objetivos.**				X

Descripción de la alineación del TFG/TFM con los ODS con un grado de relación más alto.

***Utilice tantas páginas como sea necesario.

Editor: Secretaria General / UPV · D.L.: V-5092-2006 · ISSN: 1887-2298 · www.upv.es/secgen

Anexo al Trabajo de Fin de Grado y Trabajo de Fin de Máster: Relación del trabajo con los Objetivos de Desarrollo Sostenible de la agenda 2030. (Numere la página)

El presente proyecto, aunque en sintonía con el horizonte verde que planea la Unión Europea y, por tanto, España, no se alinea específicamente con ningún ODS de relación social. Sin embargo, esto no implica que no tenga un impacto positivo en la sociedad. Aunque es un proyecto principalmente industrial, sí contribuye a varios ODS clave de manera significativa.

En primer lugar, este proyecto promueve el trabajo decente y el crecimiento económico (ODS 8). Los puestos generados serán de alto valor añadido y con tareas que minimizan el desgaste físico, lo que mejora la calidad del empleo. Además, se garantiza la igualdad de oportunidades, asegurando un trato equitativo tanto para hombres como para mujeres y fomentando un equipo de ingenieros e ingenieras equilibrado. Esto es especialmente importante en un sector donde la presencia femenina, aunque en aumento, aún es menor de lo deseado.

El proyecto también se alinea directamente con los ODS 9 y 12, correspondientes a industria, innovación e infraestructuras y a producción y consumo responsable, respectivamente. La planta cervecera implementará un sistema de embotellado basado en la reutilización de botellines de vidrio, eliminando la necesidad de etiquetado tradicional. Toda la información requerida, incluida la marca, estará serigrafiada en el botellín, evitando el uso de papel, plástico y pegamento. Esto no solo reduce el uso de recursos y energía, sino que también minimiza los residuos y simplifica el proceso de reciclaje.

Al promover la reutilización en lugar de la creación de nuevos envases, el proyecto reduce la demanda de materias primas, disminuye la generación de residuos y evita el uso intensivo de energía en la fabricación de nuevos botellines. Este enfoque no solo es económicamente eficiente sino que también es ecológicamente responsable, contribuyendo a un ciclo de producción más sostenible y alineado con los principios de la economía circular.

Por tanto, aunque el proyecto no aborda directamente los ODS de índole social, su contribución a los ODS 8, 9 y 12 demuestra un compromiso con la creación de empleo de calidad, la innovación industrial y la sostenibilidad ambiental. Estos esfuerzos colectivos ayudan a construir una industria más sostenible y responsable, en armonía con los objetivos globales de desarrollo sostenible.

Editor: Secretaria General / UPV · D.L.: V-5092-2006 · ISSN: 1887-2298 · www.upv.es/secgen