UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

UNIVERSITAT POLITECNICA DE VALENCIA
ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA DEL DISEÑO

CONTROL Y VISUALIZACIÓN DE UN ALMACÉN AUTOMÁTICO

TRABAJO FIN DE GRADO

Autor:

Parrilla Amaro, Pedro

Tutor:

Puche Panadero, Rubén

Cotutor:

Sapena Bañó, Ángel

Marzo, 2019

ÍNDICE GENERAL

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

UNIVERSITAT POLITECNICA DE VALENCIA
ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA DEL DISEÑO

CONTROL Y VISUALIZACIÓN DE UN ALMACÉN AUTOMÁTICO

1. MEMORIA

ÍNDICE DE LA MEMORIA

1.1 Objetivo del proyecto

El objetivo del trabajo fin de grado es realizar el control, la visualización y la monitorización del transelevador de un almacén automático.

El transelevador debe permitir el movimiento en los tres ejes de coordenadas. Como no se dispone del almacén físico, se ha optado por emularlo mediante tres motores controlados en posición de forma independiente que emulen el comportamiento de cada uno de los ejes. De este modo, el transelevador presenta un control de posición de sus movimientos, permitiéndose el movimiento de varios ejes a la vez.

El control del almacén se realiza mediante el control del transelevador el cual debe gestionar el almacenaje de los pales.

Para ello, dispone de dos modos de control, automático y manual.

- El modo manual debe controlar manualmente de forma independiente cada uno de los motores. Debe permitir el control de posición de cada uno de los ejes de forma manual así como el acceso al almacén de forma manual.
- El modo automático debe permitir el control del almacén mediante el apilado y desapilado de pales de forma automática. Por programa se indicará la posición donde se debe almacenar o desde la cual se quiere sacar y automáticamente el control realiza la secuencia.

Por último, el sistema SCADA debe permitir la monitorización y visualización continua del sistema así como la gestión del stock en el que se indicara el número de palés almacenados así como el tiempo que lleva cada palé almacenado.

1.2 Antecedentes

Un almacén es un punto clave en cualquier empresa ya que una buena gestión de este reportará unos mayores beneficios y una mejor relación con los clientes.

Los primeros almacenes se gestionaban únicamente con la fuerza del personal, con el paso del tiempo se empezó a pensar en la reducción de costes, sobre los años 50 y debido principalmente a la subida de los salarios se comenzó a buscar la manera de reducir costes, gracias a la ayuda del palé y los primeros sistemas mecánicos.

En nuestros días el almacén es uno de los puntos más importantes de las empresas. Esto se debe al aumento de la competencia, ya que una mejor gestión del producto desemboca en un tiempo de respuesta más reducido y un mejor servicio de cara al cliente.

Actualmente y gracias a la evolución de la electrónica y la informática se han conseguido unas mejoras importantes obteniendo una mayor calidad del servicio y una reducción de costes, asumiendo un mayor coste inicial para desarrollar el sistema autónomo de almacenamiento. Podemos ver como los sistemas de almacenamiento autónomos son utilizados desde en farmacias, para la gestión de pequeños productos, almacenaje automáticos de palés e incluso en almacenaje de vehículos.

Los principales beneficios que se obtienen al contar con un sistema de almacenamiento inteligente son un ahorro de tiempo de proceso, un mayor control de entrada y salida de mercancías, necesidad de menor espacio y un ahorro en el coste de mano de obra.

Los autómatas programables juegan un papel muy impórtate en los almacenes automáticos ya que estos son los encargados de tratar las entradas e indicar a los actuadores cuando y como deben funcionar para realizar la tarea seleccionada. En el caso de tener que modificar el proceso, ya sea por cambio de producto o por mejora de la calidad, solo tendremos que modificar el programa o ampliar o sustituir únicamente los elementos necesarios, de esta manera reducimos los costes.

1.3 Justificación

La realización de este proyecto se puede demostrar desde el marco académico y el funcional.

1.3.1 Académica

Desde el punto de vista académico la realización del presente proyecto tiene como objetivo la obtención de la titulación Grado en Ingeniería Eléctrica. Por otro lado también se ampliarán los conocimientos obtenidos en el laboratorio de accionamientos electromecánicos del departamento de ingeniería eléctrica.

El proyecto ha sido realizado en Escuela Técnica Superior de Ingeniería del Diseño en la Universitat Politècnica de València.

1.3.2 Funcional

Desde el ámbito funcional es muy importante tener unos conocimientos lo más amplios posibles sobre todo lo relacionado con los autómatas programables, sistemas de monitorización y control y redes industriales ya que hoy en día se utilizan de una manera muy amplia, sobre todo en el ámbito industrial, es por ello que se pretenden obtener conocimientos sobre:

- Creación y control de sistemas SCADA (Supervisory Control and Data Acquisition).
- Conocimientos de los diferentes lenguajes de programación (LD, ST, grafcet, etc.).
- Sensores y actuadores utilizados actualmente en la industria.

1.3.3 Personal

Actualmente trabajo en el sector del mantenimiento en una empresa de automoción. He podido observar como día a día este tipo de controles son más utilizados en la industria y he pensado que una forma de conocer estos sistemas y continuar ampliando conocimientos en el ámbito del control de accionamientos y la programación de autómatas será mediante la utilización de motores, variadores, autómatas y otros sensores y actuadores.

1.4 Factores a considerar

El proceso de control debe realizarse de manera que pueda ser utilizado en cualquier aplicación de este tipo. Si se quisiera utilizar para un almacén de diferente tamaño solo se tendrían que realizar pequeñas modificaciones en el autómata.

Si se amplían el número de celdas de carga, únicamente indicando el número de celdas totales y actualizando la cantidad de pulsos necesarios para llegar a dichas celdas, el programa funcionaria de la misma manera. Esto se debe al empleo de matrices y una programación genérica que realiza la misma operación, independientemente de la celda seleccionada, únicamente cambiado los pulsos enviados al servoamplificador según la distancia que deba recorrer cada eje.

1.4.1 Especificaciones del encargo

EL software debe permitir un control total sobre los accionamientos que controlan los motores que permiten el movimiento de los diferentes ejes, X, Y y Z. En nuestro caso se realizara un control de posición. Los controles estarán duplicados ya que se podrán utilizar desde el Scada o desde la cabina embarcada.

A continuación se detallan los modos de control de los que dispondrá el sistema.

1.4.1.1 Modo manual

Desde este modo tendremos un control total de los tres ejes del transelevador. Podremos subir y bajar, eje Y, avanzar y retroceder, eje X, y mover las horquillas en ambas direcciones, eje Z.

1.4.1.2 Modo automático

El modo automático de control permite dos tipos de operaciones: almacenar y vaciar. Para ambos modos de funcionamiento el operario debe seleccionar la celda donde almacenar el nuevo palé o la celda desde la cual retirar el palé a descargar.

- Para llenar almacén: Cuando la carga esté en posición y no habiendo ningún objeto cortando la barrera de seguridad ni ninguna puerta abierta, el operario, seleccionará la celda deseada y pulsará el botón almacenar, al pulsar el botón "marcha" el transelevador recogerá la carga y la depositara en la celda seleccionada. Cuando una nueva carga vuelva a estar en posición se podrá repetir el proceso.
- Para vaciar almacén: Cuando la mesa de carga esté vacía se seleccionada la celda que se desea descargar, después se pulsa el botón "descarga" y

finalmente el botón "marcha", el transelevador recogerá la carga de la celda deseada y la preparará para que el operario puede retirarla.

1.5 Descripción del proceso a controlar

Se va a realizar el control de un almacén automático, esto se hará gracias a un transelevador que se encargará de coger la carga y depositarla en la celda indicada o retirar un pale de la celda cargada con anterioridad y gracias a un autómata programable en el que se realizara el programa y será el encargado de dar instrucciones a los motores de dicho transelevador.

El transelevador almacenara en sus dos estanterías adyacentes, cada estantería tendrá un capacidad de 30 palés, ya que dispondrá de 5 niveles (eje y) y 6 columnas (eje x).

En los siguientes apartados se van a analizar las partes de un sistema de almacenamiento autónomo, el transelevador elegido es el fabricado por la empresa Mecalux, desde la figura 1.1 hasta la 1.7 se han utilizado las imágenes del catálogo del fabricante, www.mecalux.es.

1.5.1 Control del almacén automático

Para el control del almacén se dispondrá de un scada en el que se podrán ejecutar el modo manual y el automático.

1.5.2 Transelevador para palés

En la siguiente imagen se puede ver el transelevador seleccionado para el almacén.

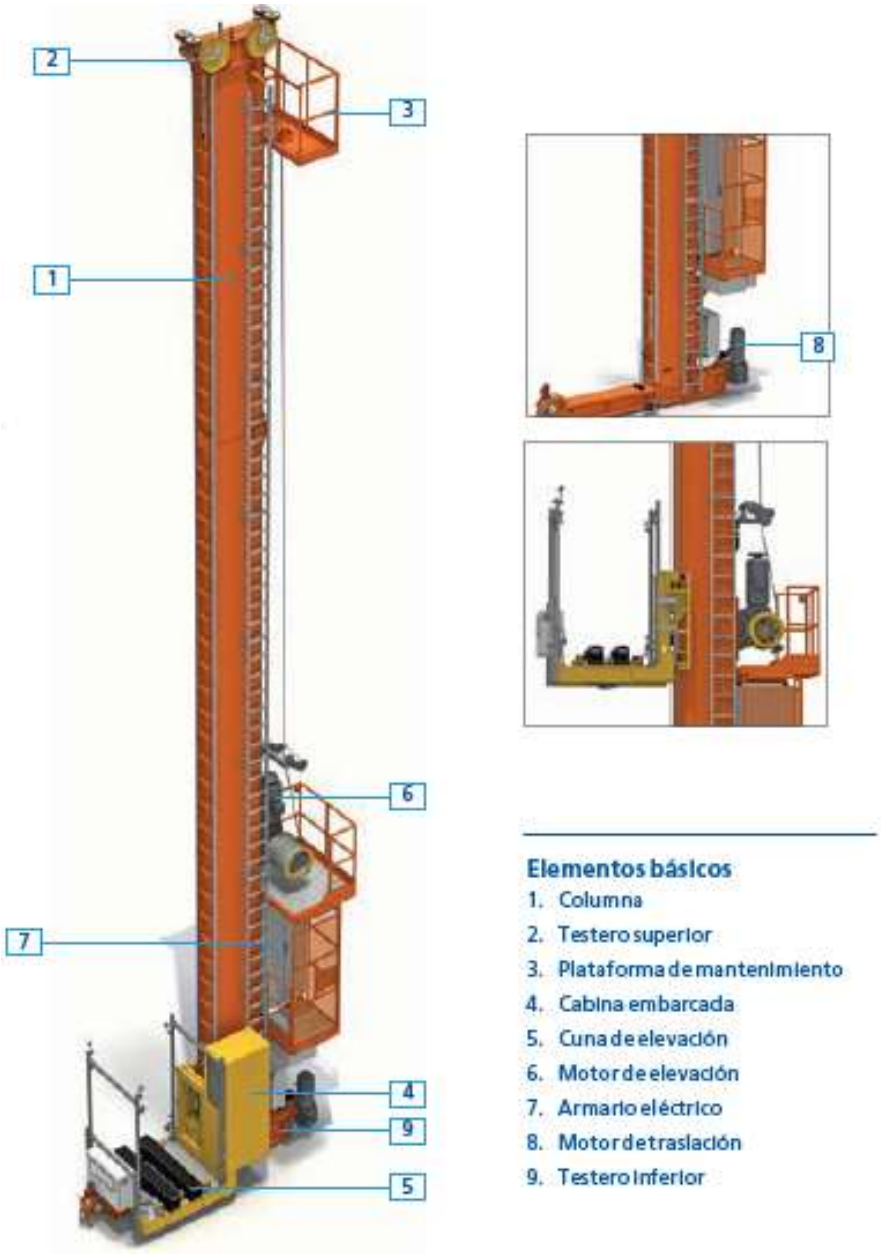


Figura 1. 1: Partes de un transelevador.

1.5.2.1 Columna

La columna está fabricada mediante chapas de acero de alta resistencia soldadas entre si formando un cajón en forma rectangular (viga). En el interior se alojan unos refuerzos dispuestos en celosía, de esta manera se consigue una mayor resistencia a la torsión y a la flexión.

Para realizar el mantenimiento, en la columna se dispone de una escalera de emergencia, dicha escalera cuenta con un cable de seguridad, línea de vida, según indica la normativa de seguridad vigente.

En la parte superior e inferior de la columna se dispone de dos finales de carrera, en el caso de fallo del control de posición, la maniobra se detendrá antes de que se realice el impacto al final de la columna.

1.5.2.2 Testero superior

El testero superior está formado por placas soldadas, en estas se sitúan las ruedas horizontales que guiaran al transelevador por el carril superior. En el bastidor superior también disponemos de las poleas de reenvío del cable de elevación.

Figura 1. 2: Detalle testero superior.

1.5.2.3 Plataforma de mantenimiento

Esta plataforma se sitúa en el superior de la columna, gracias a esta plataforma se podrán realizar los mantenimientos y/o reparaciones en el testero superior de una manera segura.

1.5.2.4 Cabina embarcada

La cabina embarcada se sitúa justo al lado de la cuna de elevación, esta cabina se utiliza para que el operario pueda ir junto con la carga para realizar ajustes, mantenimientos o poder determinar el correcto funcionamiento en la operación.

En la cabina se dispondrá de un panel de control en el que se podrá mover de una forma manual todos los motores y se podrá seleccionar el modo de trabajo. Mediante un led luminoso sabremos en todo momento el modo de trabajo seleccionado. También se dispondrá de una seta de emergencia.

1.5.2.5 Accionamiento de elevación

El objeto del accionamiento de elevación es el de impulsar el bastidor móvil en su movimiento vertical. Se sitúa junto al armario eléctrico y está compuesto por un motor de corriente alterna equipado con un encoder y un electrofreno.

Figura 1. 3: Detalle del accionamiento de elevación.

El motor se encuentra acoplado a un reductor, sobre el eje del reductor se acopla un tambor donde se enrollan los cables de elevación.

1.5.2.6 Testero inferior

Se trata de una estructura con forma de cajón resistente a la torsión y a la flexión gracias a los refuerzos soldados en su interior. En el bastidor se fijan los cabezales de la rueda motriz y la rueda libre. Con el cabezal de la rueda libre se permite aplomar la columna de una manera sencilla.

En la rueda motriz se encuentra acoplado un motor de corriente alterna equipado con un electrofreno y un encoder. Para acoplar dicho motor, en el lado de la rueda motriz, se dispone de un brazo.

Para el movimiento del transelevador en sentido longitudinal se dispone de unas ruedas ubicadas a los lados del rail de rodadura y próximas tanto a la rueda motriz como a la rueda libre.

1.5.3 Cuna de elevación

El bastidor móvil de elevación (cuna) tiene la función de transportar la carga y la cabina y efectuar la carga y la descarga mediante las horquillas extensibles instaladas en el mismo. Entre las horquillas y los marcos del bastidor se dispone de un suelo de chapas, estas están dimensionadas para que soporte el peso de un operario mientras realiza las labores de mantenimiento.

Incorpora unos frenos paracaídas de emergencia que se activan en el caso de que el transelevador supera la velocidad de servicio.

1.5.3.1 Sistema de extracción

El parámetro fundamental a considerar es la capacidad de extensión de la horquilla, en nuestro caso el sistema de extracción será de simple fondo ya que necesitamos depositar únicamente un palé en cada lado del transelevador. De esta manera aunque tengamos menos capacidad de almacenaje obtendremos una gran agilidad en la carga y descarga del almacén.

Figura 1. 4: Horquilla extendida.

Figura 1. 5: Horquilla.

La horquilla telescópica está compuesta por dos brazos unidos entre si y accionados por un motor. Dispondrá de dos finales de carreara, una para en cada lado, para que en caso de fallo la maniobra se detenga.

1.5.4 Equipo de pasillo

1.5.4.1 Carril inferior

Este carril es tipo RN-45 y viene fijado a la losa de hormigón mediante placas antivibración, el número y la distancia entre ellas, depende de la masa total para la correcta distribución de las cargas.

Figura 1. 6: Carril inferior RN-45.

1.5.4.2 Carril guía superior

Formado por perfil HEA120 o tubo estructural. Este está fijado mediante soldadura a los perfiles de la estantería.

Al final de las guías en ambos lados se dispone de un final de carrera para que en caso de fallo antes de impactar la maniobra se detenga.

Figura 1. 7: Carril superior. Perfil HEA120.

1.5.5 Mesa de carga/descarga

En esta mesa será en la que se depositará el palé en caso de almacenaje o de la que se retirara en caso de retirada.

Por seguridad solo estará accesible por un lado, siempre que el transelevador se encuentre en funcionamiento la barrera de seguridad no debe detectar nada, en caso de que detecte algo la maniobra parará.

También dispondrá de un detector capacitivo para saber si la mesa está cargada con algún palé.

1.5.6 Puerta de acceso a pasillo

Para el acceso al pasillo por donde realizará operaciones el transelevador se dispondrá de dos puertas, una puerta de seguridad en cada lado del pasillo, estas dispondrán de un detector inductivo. Si una de las puertas está abierta el transelevador no se moverá en ningún caso.

1.5.7 Estanterías

Serán las encargadas de almacenar los palés. A nivel de software se implementarán unos contadores de tiempo para saber el tiempo que dicha carga está almacenada. Por ultimo también se dispondrá de dos contadores, un contador total de huecos libres y uno de huecos ocupados.

1.5.8 Sensores y actuadores

A continuación se indican todos los detectores y actuadores de los que dispondremos para de esta forma poder realizar la programación y el cableado hasta el sistema de control.

ENTRADAS				
Lugar	Elemento	Descripción	Nombre variable	Dirección
Columna	Final carrera	Límite superior	YLS	%IX1.4
		Límite inferior	YLI	%IX1.5
Cabina embarcada	Pulsador	Manual	MANUAL	%IX2.1
		Subir	AVANZARY	%IX2.2
		Bajar	RETROCEDERY	%IX2.3
		Avanzar	AVANZARX	%IX2.4
		Retroceder	RETROCEDERX	%IX2.5
		Horquilla Izquierda	RETROCEDERZ	%IX2.6
		Horquilla Derecha	AVANZARZ	%IX2.7
		Posición X a cero	RESETX	%IX3.0
		Posición Y a cero	RESETY	%IX3.1
		Posición Z a cero	RESETZ	IX3.2
	Seta	Paro emergencia	PARO	%IX0.2
	Final carrera	Limite derecho	ZLS	%IX1.6
		Limite izquierdo	ZLI	%IX1.7
Guía superior	Final carrera	Limite derecho	XLS	%IX1.2
		Limite izquierdo	XLI	%IX1.3
Mesa carga/descarga	Detector capacitivo	Mesa ocupada/libre	MCARGA	%IX0.6
	Barrera seguridad	Zona trabajo libre	BARRERA	%IX0.7
General instalación	Puerta principal	Puerta principal cerda/abierta	PUERTAP	%IX1.0
	Puerta secundaria	Puerta secundaria cerda/abierta	PUERTAS	%IX1.1
Cuadro de mandos*	Pulsador	Automático	AUTOMATICO	%IX0.0
		Marcha/pausa	MARCHA	%IX0.1
		Reset	RESET	%IX0.3
		Carga	CARGA	%IX0.4
		Descarga	DESCARGA	%IX0.5
		Paro	PARO	%IX2.0

Tabla 1. 1. Entradas al autómata.

*El programa se ha preparado para poder utilizar un cuadro de mandos aunque en este caso se realizaría todo desde el Scada dispuesto junto a la mesa de carga/descarga.

SALIDAS				
Lugar	Elemento	Descripción	Nombre variable	Dirección
Regenerar instalación	pulsador	Activar motores	ACTIVO	miDC541_IO.OUT0
Testero inferior	Motor	Sentido X	sentx	miDC541_IO.OUT6
		Pulsos movimiento x	GFREQ_X	miDC541_IO.OUT7
Accionamiento de elevación	Motor	Sentido Y	senty	miDC541_IO.OUT4
		Pulsos movimiento Y	GFREQ_Y	miDC541_IO.OUT1
Cuna	Motor	Sentido Z	sentz	miDC541_IO.OUT2
		Pulsos movimiento Z	GFREQZ	miDC541_IO.OUT3
Cabina embarcada	LED	Automático	MANUAL	%QX38.0
		Manual	AUTOMATICO	%QX38.1

Tabla 1. 2. Salidas del autómata.

En el anexo 2 se muestra el programa realizado para la gestión de los sensores y actuadores, gracias a este programa se podrán modificar la dirección de las entradas y las salidas sin tener que modificar el programa principal.

1.6 Planteamiento de soluciones alternativas, descripción de los criterios de selección y justificación de la solución adoptada

1.6.1 Elección del PLC

Un PLC o controlador lógico programable es un elemento electrónico programable mediante personal eléctrico o electrónico sin grandes conocimientos informáticos. Un PLC está formado principalmente mediante una CPU, una memoria y unos bloques de entradas y salidas. Este gestiona las entradas, con estas y la programación realizada mediante el programa de control realizara contajes, cálculos, temporizaciones y activará o desactivará actuadores.

Actualmente hay muchos tipos de autómatas programables en el mercado pero en general los podemos diferenciar en compactos y modulares. En el caso de los compactos dispondremos de un número de entradas/salidas determinadas ya que no podremos ampliar, por el contrario, los sistemas modulares nos permiten ir acoplando diferentes modelos según nuestras necesidades, gracias a esto podremos adaptarlo a las nuevas tecnologías simplemente realizando ampliaciones. Siempre tendremos que utilizar un PLC que nos permita tener entradas y salidas libres y una potencia algo superior a la necesaria para que funcione correctamente. También tendremos que tener en cuenta que a más prestaciones y más potencia el PLC se encarecerá.

El autómata es el elemento principal de cualquier proceso de control ya que gestiona y controla tanto sensores como actuadores, por ello, la elección de autómata es muy importante para poder realizar el control correctamente. En el mercado se pueden encontrar una gran variedad de autómatas programables con características muy variadas y de diferentes fabricantes. En nuestro caso se analizarán los autómatas disponibles en el departamento de ingeniería eléctrica y se utilizará el que más se adapte a las necesidades del presente proyecto.

A continuación se detallan los autómatas disponibles en el laboratorio:

- S7-1200, Siemens.

Este PLC de tipo compacto se adecua a numerosas aplicaciones principalmente gracias a su capacidad de ampliación y a su gran flexibilidad.

Figura 1. 8: Autómata Siemens S7-1200.

El s7-1200 1211c incorpora las siguientes características:

- Calculo de 64 bits.
- Interfaz Ethernet/profinet integrada.
- 6 entradas y 4 salidas digitales integradas.
- 2 entradas analógicas integradas.
- Entradas de alta velocidad para contaje y medición.
- Salidas de alta velocidad para regulación de velocidad, posición y punto de operación.
- Bloques de función para control de movimiento.
- Funcionalidad PID para lazos de regulación.

- CJ2M, Omron.

Se trata de un PLC de tipo modular. Dispone de una gran cantidad de módulos de ampliación para la comunicación serie o en caso de que sean necesarias más entradas y salidas. Esta serie dispone de total compatibilidad con las tecnologías de redes abiertas más usuales.

Figura 1. 9: Autómata Omron CJ2M.

El PLC CJ2M de Omron tiene las siguientes características:

- Puerto USB.
- Puerto Ethernet.
- Módulos de E/S de pulsos, con estos módulos se dispondrá de control de frecuencia de pulso, control de ancho de pulso y entradas de interrupción entre otras.

- ACS500 PM571, ABB.

Este es un PLC muy flexible ya que dispone de una gran capacidad de expansión y tiene un alto rendimiento.

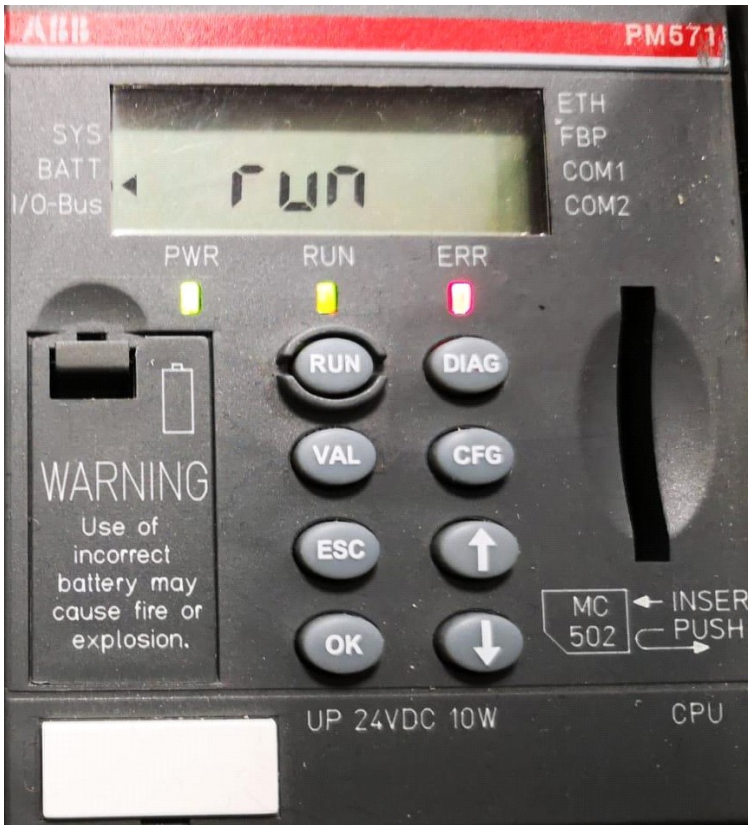


Figura 1. 10. Autómata ABB PM571 con modulo E/S DC541

Las características principales del Acs500 PM571 de ABB son:

- Compatibilidad total con CPU´s AC500 y AC500-eCo.
- Compatibilidad total con módulos E/S S500 y S500-eCo.
- 64 KB de memoria de programa.
- 8 entradas/salidas configurables mediante el módulo DC541.
- Posibilidad de expansión con hasta 10 módulos.

Para la realización del proyecto se ha seleccionado el PLC de la marca ABB, este cumple con las especificaciones necesarias para la realización del proyecto y disponemos de material en el laboratorio para poder conectar los motores y realizar el control de posición.

	S7-1211	cj2m	pm571
1000 instrucciones lógicas (ms)	0,085	0,04	0,05-0,06
100 instrucciones "Word" (ms)	1,7	0,04	0,05- 0,09
100 instrucciones "coma-flotante" (ms)	2,5	0,06	0,05- 0,70
Grado de protección	IP20	IP20	IP20
Entradas integradas	6 digitales y 2 analógicas (0 a 10V)		
Salidas integradas	4 digitales		
Número máximo de entradas analógicas	1 SB y 3 módulos de expansión	Máximo 40 módulos de expansión	160 AI, 160A0. Max 10 módulos de expansión
Número máximo de entradas digitales			320 DI, 240 DO. Max 10 módulos de expansión
Voltaje	24V CC	5V CC	24V CC

Tabla 1. 3. Características autómatas.

1.6.2 Lenguaje de programación

La norma IEC 61131 se creó para estandarizar la programación de los autómatas programables, al principio cada fabricante utilizaba su lenguaje de programación con lo que para programar cada marca había que aprender a usar su lenguaje especifico.

EN el estándar IEC 61131-3 publicado en 1993 se define:

- Tipos de programas, funciones y bloques de función.
- Tipos de datos soportados.
- 5 Lenguajes de programación de autómatas programables: LD, FBD, SFC, IL y ST.

A continuación se detallan los lenguajes de programación.

1.6.2.1 Ladder (LD)

Este lenguaje de programación presenta grandes semejanzas con los diagramas eléctricos utilizados para elaborar cuadros eléctricos. Este lenguaje es muy utilizado para el cambio de un sistema de control por relés al control mediante PLC.

Cuando se utiliza el modo visualización del PLC se puede observar cómo se iluminan los contactos activados, las salidas y las líneas de programa activadas. De esta manera se puede ver de una forma muy rápida e intuitiva los fallos de programa o las averías en las instalaciones.

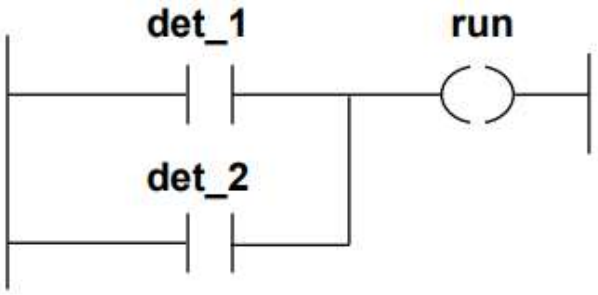


Figura 1. 11: Programación en LD

1.6.2.2 Diagrama de funciones (FBD)

Lenguaje basado en bloques de funciones enlazados unos con otros. Es sencillo pero presenta limitaciones a la hora de crear programas complejos.

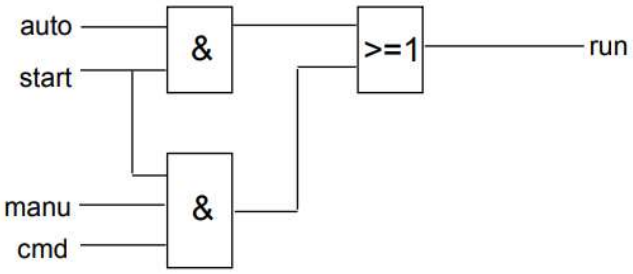


Figura 1. 12: Programación en FBD.

1.6.2.3 Grafcet (SFC)

Este lenguaje está diseñado para la automatización de procesos secuenciales. En este lenguaje a medida que se cumple la etapa baja a la siguiente y así sucesivamente. Entre la transición de etapas se pueden colocar saltos condicionales de etapas, de este modo cuando se cumpla cierta condición se podrán saltar etapas o retroceder.

Muy útil a la hora de conocer el proceso y controlarlo aun sin tener conocimientos de automatismos.

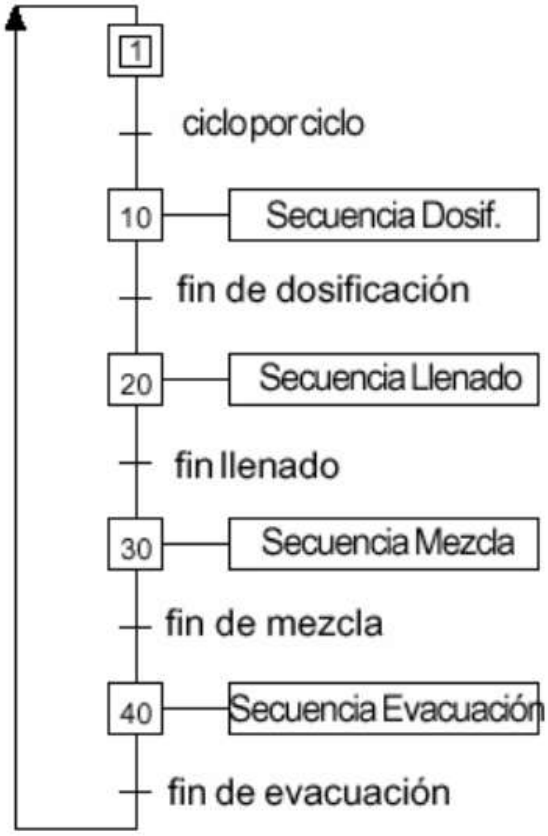


Figura 1. 13: Programación en SFC.

1.6.2.4 Lista de funciones (IL)

En este lenguaje se utilizan una serie de símbolos nemotécnicos, es una transcripción elemental e inmediata de las instrucciones del lenguaje máquina. Se permite una operación por línea.

```
000   LD    %I0.1   Bp. inicio ciclo
      AND   %I0.0   Dp. presencia vehículo
      AND   %M3     Bit autorización reloj calendaric
      AND   %I0.5   Fc. alto rodillo
      AND   %I0.4   Fc. detrás pórtico
005   S     %M0     Memo inicio ciclo
      LD    %M2
      AND   %I0.5
      OR    %I0.2   Bp. parada ciclo
      R     %M0
010   LD    %M0
      ST    %O0.0   Piloto ciclo
```

Figura 1. 14: Programación en IL.

Este lenguaje es muy potente y capaz pero es muy complejo y difícil de utilizar. Adecuado para pequeñas aplicaciones y optimización.

1.6.2.5 Texto estructurado (ST)

Es un lenguaje en el que se utilizan un conjunto de palabras clave o instrucciones. Se realiza una sucesión de enunciados para la asignación de variables y el control de funciones y bloques de funciones. Con este lenguaje se pueden realizar rápidamente sentencias complejas que manejen variables con un amplio rango de diferentes tipos de datos, ya sean digitales o analógicos.

```
21 IF "Entrada3" AND "Entrada4"=true THEN
22   "Salida3":=true;
23 ELSE
24   "Salida3":=FALSE;
25 END_IF;
26
```

Figura 1. 15: Programación en ST.

1.6.2.6 Continuos Function Chart (CFC)

Este es un lenguaje de programación grafico en el que disponemos de unos bloques predefinidos que iremos insertando e interconectando según nuestras necesidades. Es un lenguaje de programación muy intuitivo y no se necesita mucha experiencia para poder programar con este lenguaje. Su principal inconveniente es que a medida que el programa comienza a ser complejo se convierte muy complicado trabajar con él, al igual que en el momento que se necesiten realizar labores de mantenimiento o posteriores ampliaciones.

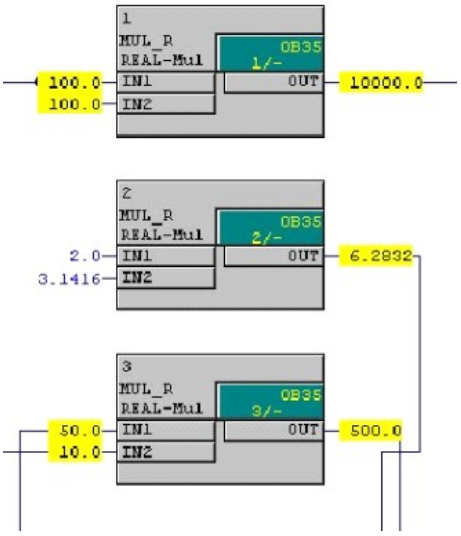


Figura 1. 16 Programación en CFC.

Aunque se podría haber empleado cualquier lenguaje de los indicados anteriormente se ha decido realizar todo el proyecto en ST (texto estructurado) ya que es muy habitual ver este lenguaje en la industria, este permite realizar rápidamente

sentencias complejas con diferentes variables y datos. El único inconveniente es la dificultad para el personal poco habituado a este tipo de programación puesto que no es tan intuitivo como otros lenguajes de programación.

1.6.3 Software de programación

Como se ha indicado en el apartado 1.6.1 el autómata utilizado será el ACS500 PM571 de la marca ABB. Para realizar la programación se utilizará el software CoDeSyS. Dicho software es utilizado para la programación del autómata, también dispone de la posibilidad de la creación del Scada. Cuando se realiza la configuración se puede elegir el lenguaje de programación, podremos elegir entre Ladder (LD), diagrama de funciones (FBD); Grafcet (SFC), lista de funciones (IL), texto estructurado (ST) y Continuos funtion chart (CFC), todos ellos explicados en el apartado anterior. Por otro lado dicho software dispone de un modo simulación, gracias a este modo podremos trabajar y probar sin la necesidad de tener el plc conectado, esto es muy útil si queremos trabajar en varios sitios para no tener que llevar siempre el autómata.

En el anexo 5 se explica con detalle la creación de un nuevo proyecto así como la configuración para la conexión con el autómata.

1.6.4 Scada

Scada (Control Supervisor y Adquisición de datos) es cualquier software que permita el control del proceso usando la comunicación disponible en cada caso y el acceso remoto a datos de un proceso. El sistema scada proporciona información del proceso a operadores, supervisores, mantenimiento, programador, etc. Esta aplicación software se comunica con los dispositivos de campo y controla el proceso de una forma automática pero con la posibilidad de operar directamente desde la pantalla.

Los objetivos del sistema Scada son:

- Economía: Se monitoriza la planta de una forma más eficiente ya que se necesitan menos operarios.
- Accesibilidad: Se puedo controlar el proceso desde la pantalla incluso actuar sobre él.
- Mantenimiento: Se visualizan y registran datos de fallos o alarmas de aviso para la realización de revisiones.
- Gestión: Los datos son recopilados y se muestran en forma de listas, graficas, etc.
- Flexibilidad: Se puede modificar diferentes características del proceso sin gasto económico ya que no modifican físicamente el proceso.

1.6.4.1 WinCC de Siemens

Dentro del software de la marca Siemens, TIA PORTAL, la marca ha incorporado un Scada en el que se puede generar un sistema de monitorización y control de procesos.

- Características y funciones.

Dispone de todo tipo de objetos para poder realizar las configuraciones deseadas, estos se encuentran en sus librerías, también se pueden representar gráficos y tablas para el tratamiento de datos.

- Programación en C.
- Configuración de manera intuitiva mediante Ethernet, profibus, etc.
- Comunicación con autómatas de otras marcas mediante OPC.
- Soporte de tecnologías Active X.

1.6.4.2 CX-Supervisor de Omron

Este es el Scada que ha desarrollado la marca Omrom, nos permite trabajar de un modo flexible con uno o varios autómatas. La programación se desarrolla sobre un entorno de Windows.

- Características y funciones.

- Programación en entorno de Windows mediante Scripts o ventanas.
- Configuración de manera intuitiva mediante Ethernet, profibus, etc.
- Comunicación con autómatas de otras marcas mediante OPC.
- Recetas. Algunos procesos utilizan una configuración de variables que son siempre las mismas. Con las recetas podemos definir las diferentes combinaciones de variables para configurar la producción de la planta de forma mucho más sencilla.

1.6.4.3 CoDeSyS Visualization

El programa CoDeSyS también nos permite la creación del Scada, con este podremos crear objetos para la gestión y el control del proceso.

- Visualización a través de la web mediante TCP/IP.

- Para los controladores con pantalla podremos cargar la aplicación junto a la visualización.
- CoDeSyS HMI, para la visualización mediante PC.
- Posibilidad de insertar varias formas geométricas como, rectángulos, elipses, círculos y polígonos, también tablas, gráficas y botones.

En nuestro caso hemos optado por la utilización de la visualización que nos brinda CoDeSyS ya que cumple con todos nuestros requerimientos y tenemos el extra de la facilidad de conexión ya que todos los elementos utilizados son de la marca ABB.

1.6.5 Tipos de comunicación

A continuación se van a detallar los sistemas de comunicación entre el Scada y el autómata y entre el autómata y el pc.

1.6.5.1 Comunicación con el PLC

- Ethernet.

Este protocolo de comunicación utiliza la interfaz de acceso rj-45, con este obtenemos unas velocidades de comunicación muy elevadas. Podremos conectar los elementos que componen nuestro proceso en línea, en anillo o en estrella.

- Modbus.

Modbus es un protocolo de comunicación basado en la estructura maestro-esclavo. Para iniciar la comunicación el maestro manda un mensaje para escribir o leer a todos los esclavos, el esclavo elegido envía otro mensaje con la información solicitada por el primero o modifica sus estados según le indica el maestro o ambas cosas.

Esto protocolo se suele implementar sobre redes de comunicación RS-485, aunque también sobre redes que usan RS-232.

- Serie.

Este protocolo que utiliza un conector tipo DB-25 (25 pines), o versión de 9 pines, es un sistema de comunicación serie, asíncrono y punto a punto que puede funcionar en Full-dúplex. A continuación se describen sus características:

- Punto a punto: Solamente dos dispositivos pueden estar conectados entre sí.

- Full_duplex: Los dispositivos pueden estar recibiendo y enviando información al mismo tiempo, esto se debe a que se disponen de dos conexiones separadas según el sentido de la información.

- CS31.

Utilizamos la interfaz RS-485, esta difiere de la anterior en su conexión multipunto, más de dos dispositivos, y una protección mayor frente al ruido electromagnético.

- ASCII.

Este protocolo se utiliza para comunicaciones serie y usa los caracteres ASCII para la comunicación.

ASCII es un código de caracteres basado en el alfabeto latino, el protocolo utiliza 7 bits para representar los caracteres con lo que optemos 128 posibles combinaciones, obteniendo así todo el alfabeto, números decimales, símbolos y acciones para controlar diversos dispositivos.

Se ha optado por la comunicación mediante Ethernet, esto se debe a que el ordenador lo tenemos en red y el autómata también. De esta manera se puede realizar la comunicación mediante cualquier ordenador del aula.

1.6.5.2 Comunicación con el Scada

- Comunicación mediante OPC.

OPC es un protocolo estándar para la interconectividad de sistemas que proporciona una interfaz común para las comunicaciones entre diferentes productos de distintos proveedores, este protocolo está basado en las tecnologías estándares de Microsoft como son COM, DCOM, OLE Automation y ActiveX.

Esta comunicación se realiza a través de la arquitectura CLIENTE-SERVIDOR. De esta manera el servidor OPC es la fuente de datos y el Cliente puede acceder al servidor y obtener o modificar los datos del servidor.

- Comunicación mediante DDE.

DDE o Dynamic Data Exchange en una tecnología de comunicación entre aplicaciones bajo el sistema operativo Microsoft Windows y OS/2.

Mediante este protocolo de comunicación se pueden enviar mensajes entre distintas aplicaciones. Para usar DDE se deben conocer los comandos DDE que el servidor soporta, este no ha sido generalmente estandarizado.

- Comunicación entre equipos del mismo fabricante.

Cuando la comunicación se efectúa entre dispositivos de la misma marca no es necesario otro software para realizar la comunicación. Cada marca tiene su propio protocolo de comunicación para sus dispositivos.

En nuestro caso la comunicación se realizará entre el autómata de ABB y el programa CoDeSyS, ya que son del mismo fabricante, no necesitaremos otro software para la comunicación entre estos.

1.6.6 Servodriver

El servodriver es el encargado de recibir las instrucciones del PLC, este gestiona las señales y según la configuración gestiona el movimiento de los motores.

A continuación se analizaran tres servoamplificadores de las marcas Siemens, ABB y Omron.

1.6.6.1 SmartStep 2 de Omron

A continuación se detallan sus características:

- Control de posición mediante entrada de pulsos.
- Dos límites de par.
- Filtros adaptables para eliminar la vibración y la resonancia.
- Tamaño ultra compacto.

1.6.6.2 Sinamics V90 de Siemens

A continuación se detallan las principales características:

- Control de posición con entrada de tren de pulsos.
- Control de par.
- Límite de velocidad y par.

- Supresión de vibraciones a baja frecuencia.
- Resistencia de frenado externa.
- Comunicación Modbus.

1.6.6.3 BSD0200 de ABB

A continuación se detallan las principales características del amplificados BSD0200 de ABB:

- Comando de velocidad analógica +/- 10 voltios.
- Comando de par analógico +/- 10 voltios.
- Comando de tren de pulsos hasta 1 MHz.
- Encoder serie desde 131.072 pulsos / giro (estándar).
- Filtro antivibración.
- Sintonización automática.
- Ajuste automático de compensación.
- Salida de encoder simulada configurable.

Cualquiera de los tres servodrivers explicados anteriormente serian válidos para la realización del presente proyecto. Se ha decidido utilizar el de la marca ABB por disponer de los tres amplificadores en el laboratorio disponibles para su utilización. A continuación se muestra la placa de características:

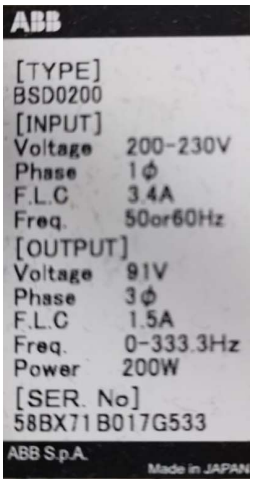


Figura 1. 17. Placa de características del Servodriver.

1.6.7 Servomotor

Al seleccionar el servodriver de la marca ABB se han utilizado tres motores de la misma marca disponibles en el taller para simular los tres controles de posición necesarios para la aplicación del presente proyecto. Los tres motores disponen de encoder para conocer la posición de la aplicación.

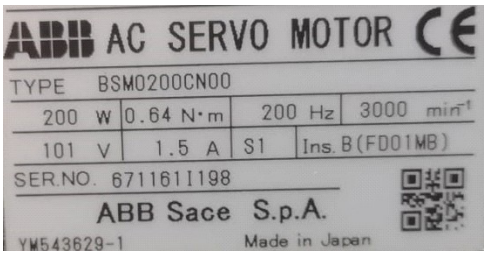


Figura 1. 18. Placa de características del Servomotor.

1.7 Descripción detallada de la solución adoptada

1.7.1 Introducción

Los autómatas programables son, hoy en día, los responsables de la gran mayoría de los controles industriales, esto se debe, principalmente, a sus grandes posibilidades. Mediante redes se pueden comunicar con una gran cantidad de sensores, actuadores, ordenadores, SCADAS, etc. Gracias a esto conseguimos un gran control sobre el proceso, con esto obtenemos un producto de mayor calidad y de una forma más rápida.

Otro de los beneficios de los autómatas programables es su capacidad de poderse adaptar a las condiciones de cada momento, de esta manera se podrán ir mejorando los productos gracias a la monitorización y también se podrán realizar cambios en el proceso utilizando nuevos sensores o actuadores sin que esto suponga una gran inversión.

1.7.2 Estrategia de control utilizada

A continuación se muestra la pirámide de control:

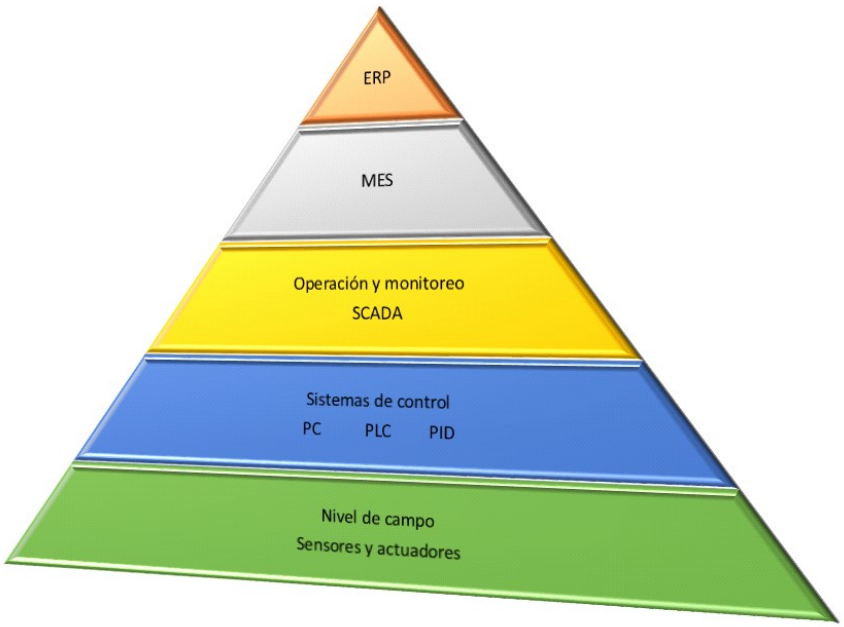


Figura 1. 19 Niveles de control.

Los dos niveles superiores se encargan de la planificación (MES) y de la gestión (ERP) en el apartado actual se va a detallar los tres niveles de control utilizados en el presente proyecto:

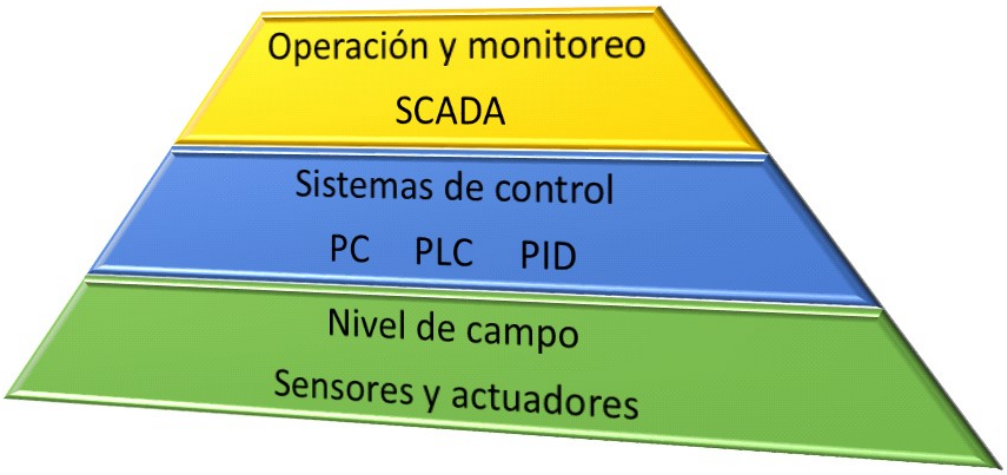


Figura 1. 20: Niveles de control utilizados.

- Nivel de campo o primer nivel (Hardware): En este nivel se engloban todos los elementos que interactúan directamente con el proceso, ya sea para recibir datos, mediante sensores, o para activar o desactivar actuadores. Los sensores pueden ser digitales o analógicos y podremos realizar mediciones eléctricas, mecánicas, térmicas, etc. Dentro de estos también tenemos pulsadores, finales de carrera, etc. Los actuadores también pueden ser digitales o analógicos y podremos encontrar de muchos tipos, pueden ser, neumáticos, hidráulicos, eléctricos, etc.

- Nivel de control o segundo nivel (Proceso): En este nivel encontramos los controladores, en nuestro caso el autómata programable que gestionan los sensores y los actuadores, entradas y salidas.

- Nivel de supervisión o tercer nivel (Proceso): En este nivel encontramos al sistema encargado de acceder a los datos de proceso para poder monitorizarlo y/o controlarlo (SCADA).

1.7.3 Solución adoptada a nivel de campo

1.7.3.1 Comunicación del autómata con tres servoamplificadores

A continuación se detalla la conexión entre el servomotor y el servodriver. Se puede observar la comunicación con el autómata mediante el CN1, por otro lado, se observa la comunicación mediante USB al PC, desde el PC y mediante el software DSB-configurator configuraremos el servodriver, en nuestro caso, para el control de posición.

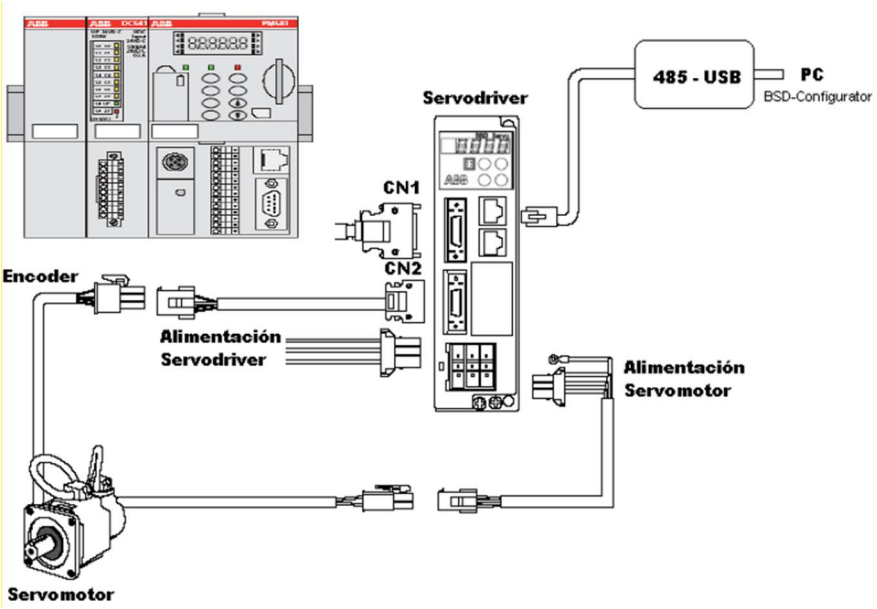


Figura 1. 21: "BSD user manual". Conexión Servomotor, servodriver, pc y autómata.

Para la conexión de los tres servodriver con un solo autómata se ha necesitado un conector especial que se ha tenido que realizar. Se ha cableado una caja de control junto a unas señalizaciones que indican si el motor está operativo y si hay algún final se carrera activado, también se han cableado seis pulsadores que simulan los finales de carrera de los que dispondría el transelevador. En los anexos se detallan las conexiones de dicha caja. Anexo 4.

1.7.3.2 Motores para el control de los ejes

Para el control del transelevador se han empleado tres motores, cada uno de ellos será el encargado del movimiento de cada uno de los ejes X, Y y Z. Las salidas del autómata utilizadas para el control de los motores que se encargarán de mover el transelevador por los diferentes ejes serán las indicadas anteriormente en el apartado 1.5.8 junto al resto de entradas y salidas.

- Motor del eje X.

Este motor se encargará del movimiento del transelevador a lo largo de las diferentes columnas, en nuestro caso serán seis columnas y la séptima será la mesa de carga/descarga.

- Motor del eje Y.

El motor del eje Y será el encargado de que la cabina embarcada se desplace `por los diferentes niveles del almacén automático, en nuestro caso se dispondrá de 5 niveles. Con el movimiento de este motor también cogeremos y dejaremos los diferentes palés.

- Motor del eje Z.

Este motor será el encargado del movimiento de las horquillas. Según el sentido de marcha del motor, definido por el autómata en cada caso, este cargará y/o descargará la carga de la estantería seleccionada.

1.7.4 Solución adoptada a nivel de control

Para la programación se ha utilizado el programa de control de la marca ABB, CoDeSyS, en el anexo 5 se detalla la configuración del programa así como la configuración para la comunicación con el PLC.

A continuación se explicará la programación que se ha implementado en el autómata para la realización de las diferentes maniobras. Estás cumplen con lo indicado en el apartado 1.4.1.

Se ha intentado realizar el proyecto utilizando bloques de funciones para que este resultara más sencillo. Se ha tenido que descartar esta idea puesto que después de la realización de los bloques de funciones el programa no funcionaba correctamente. Esto se debe al tener que repetir la función del generador de pulsos en varios bloques, una vez activaba el generador de pulsos de una determinada salida desde un bloque de funciones este quedaba definido y al ser llamado desde otro bloque de funciones no funcionaba. La solución adoptada ha sido generar todo el programa en la página principal, de esta manera solo se ha tenido que definir un generador de pulsos para cada salida, uno por eje (X, Y y Z).

1.7.4.1 Generador de pulsos

Con esta función indicamos la cantidad de pulsos necesarios para llegar a la posición deseada y la frecuencia que definirá la velocidad del motor. Se han implementado tres generadores de pulsos, uno por eje.

```
(*GENERADOR DE PULSOS*)

GFREQ_Z(SLOT:= 1,  CH:= 3,EN_VISU:= TRUE,EN:= MARCHA,START:= MARCHAZ,  STOP:= PARADA.Q1,FREQ:= vel,PULSE:= PULZ,
GFREQ_Y(SLOT:= 1,  CH:= 1,EN_VISU:= TRUE,EN:= MARCHA,START:= MARCHAY,  STOP:= PARADA.Q1,FREQ:= vel,PULSE:= PULY,
GFREQ_X(SLOT:= 1,  CH:= 7,EN_VISU:= TRUE,EN:= MARCHA,START:= MARCHAX,  STOP:= PARADA.Q1,FREQ:= vel,PULSE:= PULX,
```

Figura 1. 22. Programación generador de pulsos.

Ha continuación se muestran los parámetros de los tres generadores de pulsos.

SLOT	Se define la posición del módulo de salidas DC541
CH	Se selecciona el canal de salida
EN_VISU	Habilitación del control a través de la visualización integrada del bloque visuDC541_FREQ_OUT
EN	Habilitación del bloque
START	Iniciar envió de frecuencia
STOP	Finalizar envió de frecuencia
FREQ	Valor de frecuencia deseado
PULSE	Numero de pulsos (>0 - número de pulsos, =0 - Pulsos infinitos)

Tabla 1. 4 Parámetros de configuración del generador de pulsos

1.7.4.2 Temporizador celdas ocupadas

Su función es la temporización de todas las celdas de almacenamiento. Cuando un palé es almacenado activa una señal luminosa para saber que esa celda está cargada y también activa un contador de tiempo.

El contador de tiempo utilizado es el RTC. Cuando este temporizador es activado empieza a contar desde una fecha definida. Como nosotros necesitábamos un contador desde 0 se ha realizado una resta para que el resultado sea 0 y desde este momento se empiece a contar.

```
IF (p>=1) AND (p<=60) THEN
RTC2[p](EN:=Ocupado[p] , PDT:=DT#1970-01-01-00:00:00, Q=> , CDT=> );
con[p]:=RTC2[p].CDT-RTC2[p].PDT;
p:=p+1;
ELSE
p:=1;
END_IF
```

Figura 1. 23: Programación contador celdas ocupadas.

1.7.4.3 Posición de los ejes

La siguiente función nos permitirá saber en todo momento donde se encuentra la máquina. En el SCADA dispondremos de los contadores de posición de los ejes.

Para esto se ha utilizado R_TRIG. Su función es la mandar un pulso cada vez que recibe un flanco de subida. Cada vez que el generador de pulsos envié un pulso este lo detectará y de esta manera sumará o restará. Para saber si tiene que sumar o restar se tendrá en cuenta la salida que indica el sentido "sentx, senty y/o sentz".

```
(*CONTADORES DE POSICION DE LOS EJES*)

R_TRIGX(CLK:=miDC541_IO.IN7 , Q=>CONX );
R_TRIGY(CLK:=miDC541_IO.IN1 , Q=> CONY);
R_TRIGZ(CLK:=miDC541_IO.IN3 , Q=> CONZ);

IF sentx AND CONX THEN
posX:=posX+1;
END_IF
IF (NOT sentx) AND CONX THEN
posX:=posX-1;
END_IF

IF senty AND CONY THEN
posY:=posY+1;
END_IF
IF (NOT senty) AND CONY THEN
posY:=posY-1;
END_IF

IF sentz AND CONZ THEN
posZ:=posZ+1;
END_IF
IF (NOT sentz) AND CONZ THEN
posZ:=posZ-1;
END_IF
```

Figura 1. 24: Programación posición de los ejes.

1.7.4.4 Modo manual

- Flujograma

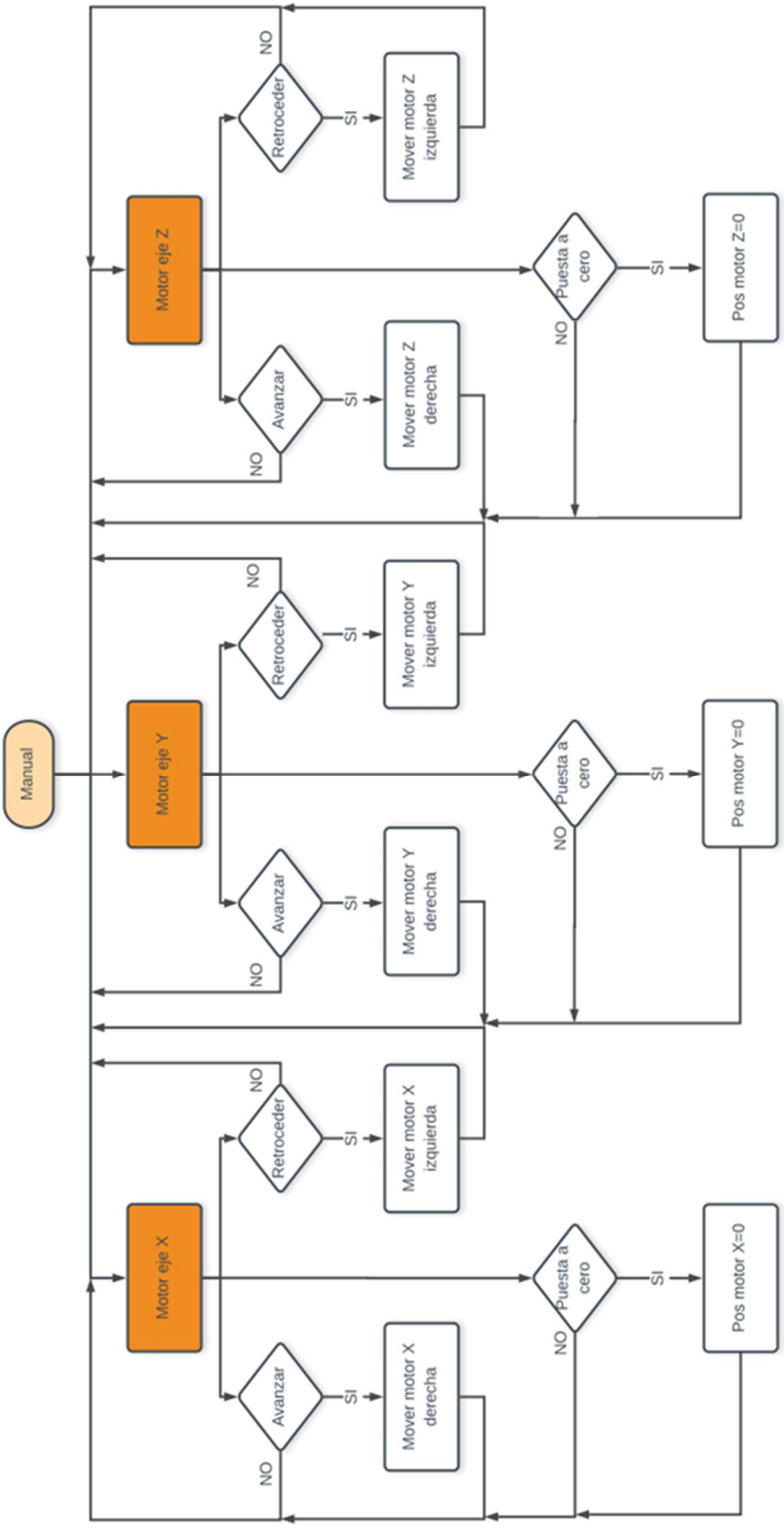


Figura 1. 25: Flujograma modo manual.

Tendremos la posibilidad del control manual de la máquina. Se podrán mover los ejes individualmente y también se podrá reiniciar la posición para que en el caso de tener que sustituir un motor o algún elemento mecánico que pueda desajustar la maquina podamos llevarla al punto cero y reiniciar a posición 0.

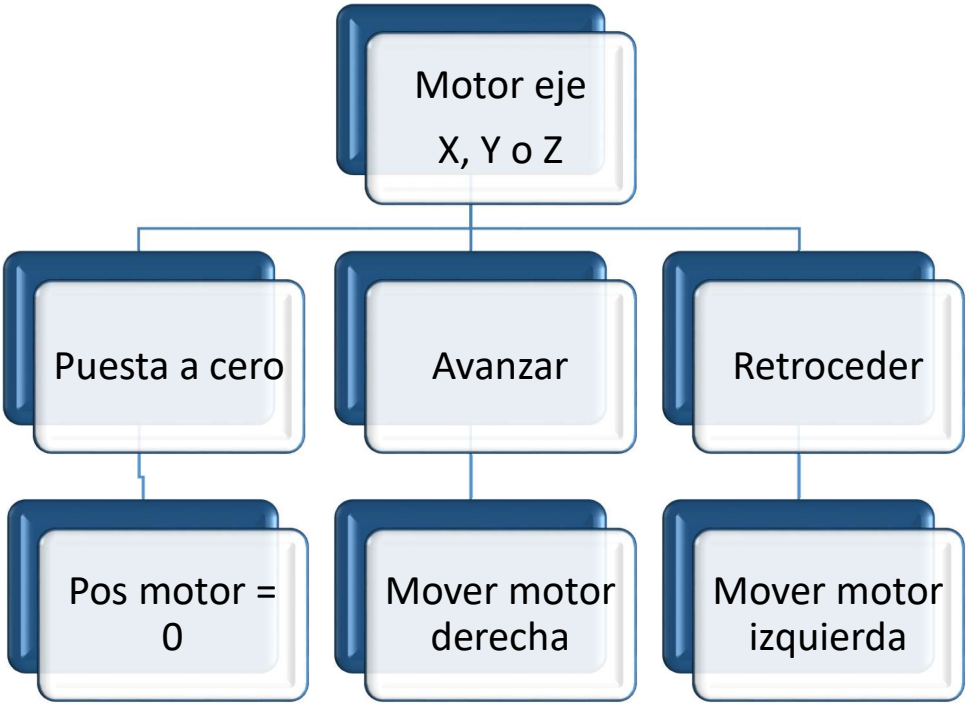


Figura 1. 26. Secuencia modo manual

Al activar el modo manual activaremos SRM. Este es el "set_reset" que define el modo de funcionamiento, en este caso manual.

```
SRM(SET1:=MANUAL , RESET:=AUTOMATICO OR (NOT MANUAL), Q1=> );
```

Figura 1. 27: Programación "set_reset" selección manual.

- Mover ejes

Para el movimiento de los ejes dispondremos de dos botones por eje, uno para el avance y otro para el retroceso. Se ha definido que el movimiento será el de 2 pulsos ya que la opción de movimiento manual ha sido creada para el movimiento en caso de fallos y/o colisiones. En el caso que superemos algún final de carrera la maquina se parará por seguridad.

```
(*MODO MANUAL*)

(*Movimiento X*)

IF SRM.Q1 AND AVANZARX AND (NOT RETROCEDERX) THEN
vel:=10;
MARCHA:=TRUE;
MARCHAX:=TRUE;
PULX:=2;
sentx:=TRUE;
END_IF

IF SRM.Q1 AND RETROCEDERX AND (NOT AVANZARX) THEN
MARCHA:=TRUE;
MARCHAX:=TRUE;
PULX:=2;
sentx:=FALSE;
END_IF

IF SRM.Q1 AND GFREQ_X.RDY AND (NOT AVANZARX)  AND (NOT RETROCEDERX) THEN
MARCHAX:=FALSE;
RX:=TRUE;
RY:=TRUE;
RZ:=TRUE;
END_IF
```

Figura 1. 28: Programación modo manual. Movimiento eje X.

```
(*Movimiento Y*)

IF SRM.Q1 AND AVANZARY AND (NOT RETROCEDERY) THEN
vel:=10;
MARCHA:=TRUE;
MARCHAY:=TRUE;
PULY:=2;
senty:=TRUE;
END_IF

IF SRM.Q1 AND RETROCEDERY AND (NOT AVANZARY) THEN
MARCHA:=TRUE;
MARCHAY:=TRUE;
PULY:=2;
senty:=FALSE;
END_IF

IF SRM.Q1 AND GFREQ_Y.RDY AND (NOT AVANZARY)  AND(NOT RETROCEDERY) THEN
MARCHAY:=FALSE;
RX:=TRUE;
RY:=TRUE;
RZ:=TRUE;
END_IF
```

Figura 1. 29: Programación modo manual. Movimiento eje Y.

```
(*Movimiento Z*)

IF SRM.Q1 AND AVANZARZ AND (NOT RETROCEDERZ) THEN
vel:=10;
MARCHA:=TRUE;
MARCHAZ:=TRUE;
PULZ:=2;
sentz:=TRUE;
END_IF

IF SRM.Q1 AND RETROCEDERZ AND (NOT AVANZARZ) THEN
MARCHA:=TRUE;
MARCHAZ:=TRUE;
PULZ:=2;
sentz:=FALSE;
END_IF

IF SRM.Q1 AND GFREQ_Z.RDY AND (NOT AVANZARZ) AND (NOT RETROCEDERZ)  THEN
MARCHAZ:=FALSE;
RX:=TRUE;
RY:=TRUE;
RZ:=TRUE;
END_IF
```

Figura 1. 30: Programación modo manual. Movimiento eje Z.

- Reiniciar posición

Con este pulsador reiniciaremos el contador de posición y lo pondremos a cero.

```
(*Reiniciar contadores de posicion*)

IF SRM.Q1 AND RESETX THEN
posX:=0;
END_IF

IF SRM.Q1 AND RESETY THEN
posY:=0;
END_IF

IF SRM.Q1 AND RESETZ THEN
posZ:=0;
END_IF
```

Figura 1. 31: Programación reiniciar contadores de posición.

1.7.4.5 Modo automático

- Flujograma

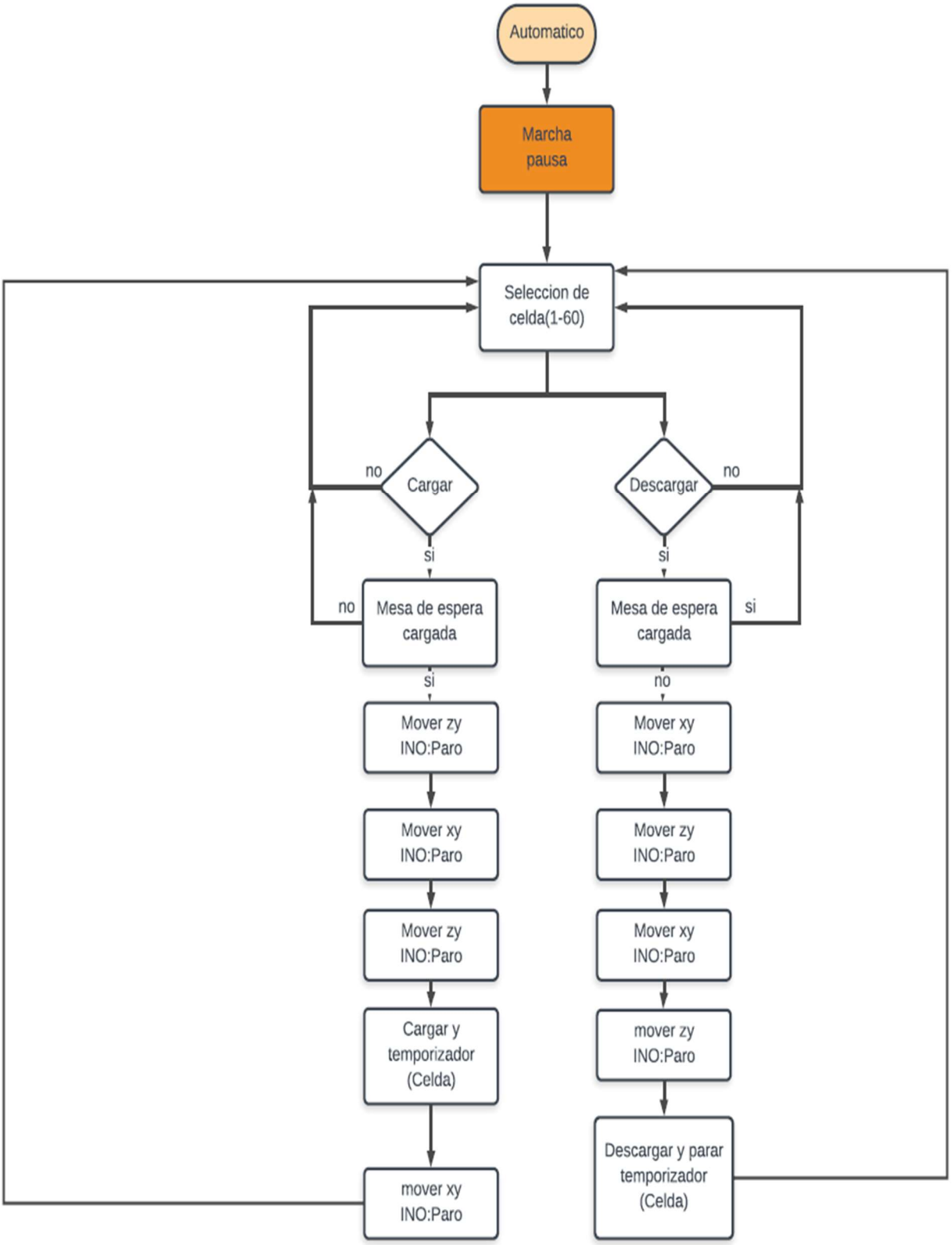


Figura 1. 32. Flujograma modo automático.

Al activar el modo automático activaremos SRA. Este es el "set_reset" que define el modo de funcionamiento, en este caso automático.

```
SRA(SET1:=AUTOMATICO , RESET:=MANUAL OR (NOT AUTOMATICO), Q1=> );
```

Figura 1. 33. Programación "set_reset" selección automático.

El modo automático realizará la acción de carga o descarga de una manera completamente autónoma. A continuación se detalla la programación para la carga y descarga y para determinar si la celda se encuentra a la derecha o a la izquierda según el número de celda elegida.

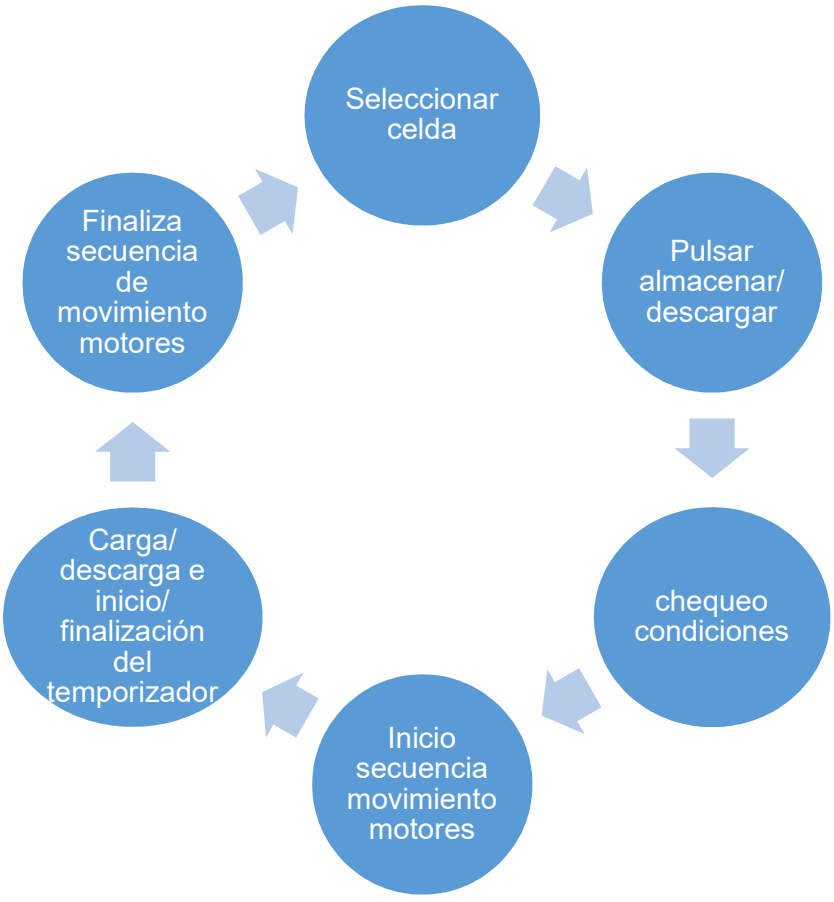


Figura 1. 34: Secuencia modo automático.

- Sentido según celda

Esta función únicamente será utilizada por el eje Z, ya que este tendrá que girar a levógiro o dextrógiro según donde se encuentre la celda, de la celda 1 a la 30 a la izquierda y de la celda 31 a la 60 a la derecha.

```
(*SENTIDO DE GIRO SEGUN CELDA SELECCIONADA*)

IF  scelda<31 THEN
sentzc:=0;
END_IF
IF scelda>30 THEN
sentzc:=1;
END_IF
```

Figura 1. 35: Programación sentido de giro según celda seleccionada.

- Almacenar

Esta función realizara el almacenaje de un palé. Primero seleccionaremos la celda en la queremos almacenar el palé, hay que tener en cuenta que en caso que la celda se encuentre ocupada la maquina no empezará la maniobra. A continuación se detallan las fases de este proceso.

```
(*ALMACENAR*)
TX(CLK:=GFREQ_X.RDY , Q=> );
RTY(CLK:=GFREQ_Y.RDY , Q=> );
RTZ(CLK:=GFREQ_Z.RDY , Q=> );
SRY(SET1:=RTY.Q , RESET:=RY, Q1=> );
SRZ(SET1:=RTZ.Q , RESET:=RZ, Q1=> );
SX(SET1:=TX.Q , RESET:=RX, Q1=> );

IF SRA.Q1 AND CARGA AND MCARGA  AND NOT CARGAR  AND NOT DESCARGAR AND NOT Ocupado[scelda] THEN
CARGAR:=TRUE;
RX:=FALSE;
RY:=FALSE;
RZ:=FALSE;
R1:=TRUE;
END_IF
```

Figura 1. 36: Programación modo automático. Almacenar.

- Mover Z e Y

Para ello primero moverá el eje Z, siempre hacia el lado derecho ya que ahí se encuentra la mesa de carga/descarga, después moverá el Y, siempre en sentido derecha ya que está cogiendo el palé y después el Z en el sentido contrario. De esta forma tendremos el palé cargado en el bastidor móvil y listo para ir a la posición deseada.

```
(*Mover ZY*)

IF CARGAR AND R1 THEN
vel:=5;
sentz:=TRUE;
PULZ:=8;
MARCHAZ:=TRUE;
R2:=TRUE;
R1:=FALSE;
END_IF

IF R2 AND SRZ.Q1 THEN
PULY:=5;
senty:= TRUE;
sentz:=FALSE;
MARCHAY:=TRUE;
MARCHAZ:=FALSE;
R3:=TRUE;
RZ:=TRUE;
R2:=FALSE;
END_IF


IF R3 AND SRY.Q1 THEN
MARCHAZ:= TRUE;
MARCHAY:= FALSE;
R4:=TRUE;
RY:= TRUE;
RZ:= FALSE;
R3:= FALSE;
END_IF

IF R4 AND SRZ.Q1 THEN
MARCHAZ:= FALSE;
RY:= FALSE;
R4:= FALSE;
R5:=TRUE;
END_IF
```

Figura 1. 37: Programación modo automático. Almacenar. Paso 1.

- Mover X e Y

Ahora moverán los ejes X e Y, moverán ambos a dextrógiro. Estos ejes moverán al mismo tiempo yo no hay ninguna restricción entre ellos. Estos llegaran a la celda seleccionada anteriormente.

```
(*Mover XY*)

IF R5 THEN
RZ:=TRUE;
sentz:=TRUE;
sentx:=TRUE;
PULX:=vecposx[scelda];
PULY:=vecposy[scelda];
    IF vecposy[scelda]=0 THEN
    MARCHAY:=FALSE;
    END_IF
    IF vecposy[scelda]>0 THEN
    MARCHAY:=TRUE;
    END_IF
MARCHAX:=TRUE;
R6:=TRUE;
R5:=FALSE;
END_IF

IF R6 AND ((SRY.Q1 OR vecposy[scelda]=0) AND SX.Q1) THEN
MARCHAX:=FALSE;
MARCHAY:=FALSE;
RZ:=FALSE;
R7:=TRUE;
R6:=FALSE;
END_IF
```

Figura 1. 38: Programación modo automático. Almacenar. Paso 2.

- Mover Z e Y

En primer lugar moverá el eje Z, para determinar hacia qué lado gira el motor utilizará la función anteriormente explicada, cuando finalice el movimiento moverá el eje Y, en sentido izquierdo, para después volver a mover el eje Z en el sentido contrario que lo hizo anteriormente.

```
(*Mover ZY*)

IF R7 THEN
sentz:=sentzc;
MARCHAZ:=TRUE;
RX:=TRUE;
RY:=TRUE;
R8:=TRUE;
R7:=FALSE;
END_IF

IF R8 AND SRZ.Q1 THEN
PULY:=5;
senty:= FALSE;
sentz:=NOT sentzc;
MARCHAY:=TRUE;
MARCHAZ:=FALSE;
R9:=TRUE;
RZ:=TRUE;
RX:=FALSE;
RY:=FALSE;
R8:=FALSE;
END_IF



IF R9 AND SRY.Q1 THEN
OCUPADO[scelda]:=TRUE;
MARCHAZ:= TRUE;
MARCHAY:= FALSE;
R10:=TRUE;
RY:= TRUE;
RZ:= FALSE;
R9:= FALSE;
END_IF

IF R10 AND SRZ.Q1 THEN
MARCHAZ:= FALSE;
RY:= FALSE;
RZ:= TRUE;
R10:= FALSE;
R11:=TRUE;
END_IF
```

Figura 1. 39: Programación modo automático. Almacenar. Paso 3.

- Mover X e Y

Finalmente moverán los ejes X e Y en sentido izquierdo hasta llegar a la posición de reposo.

```
(*Mover XY*)

IF R11 THEN
sentz:=FALSE;
sentx:=FALSE;
PULX:=vecposx[scelda];
PULY:=vecposy[scelda];
    IF vecposy[scelda]=0 THEN
    MARCHAY:=FALSE;
    END_IF
    IF vecposy[scelda]>0 THEN
    MARCHAY:=TRUE;
    END_IF
MARCHAX:=TRUE;
R12:=TRUE;
R11:=FALSE;
END_IF

IF R12 AND ((SRY.Q1 OR vecposy[scelda]=0) AND SX.Q1) THEN
MARCHAX:=FALSE;
MARCHAY:=FALSE;
RZ:=FALSE;
R12:=FALSE;
RY:=TRUE;
RX:=TRUE;
CARGAR:=FALSE;
END_IF
```

Figura 1. 40: Programación modo automático. Almacenar. Paso 4.

- Descargar

Este función es la opuesta a la de almacenar. Para poder iniciar esta maniobra se deberá tener la mesa de carga/descarga libre y la celda seleccionada para la descarga tendrá que estar ocupada. Los "set_reset" utilizados para la función de descargar son los mismos que se han utilizado en la función "almacenar", ver figura 1.40.

```
(*DESCARGAR*)

IF SRA.Q1 AND DESCARGA AND (NOT MCARGA)  AND NOT CARGAR  AND NOT DESCARGAR AND Ocupado[scelda] THEN
DESCARGAR:=TRUE;
RX:=FALSE;
RY:=FALSE;
RZ:=FALSE;
D1:=TRUE;
vel:=5;
END_IF
```

Figura 1. 41. Programación modo automático. Descargar.

- Mover X e Y

Primero moverán los ejes X e Y, moverán ambos a dextrógiro. Estos ejes moverán al mismo tiempo yo no hay ninguna restricción entre ellos. Estos llegaran a la celda seleccionada anteriormente.

```
(*Mover XY*)

IF D1 THEN
senty:=TRUE;
sentx:=TRUE;
PULX:=vecposx[scelda];
PULY:=vecposy[scelda];
    IF vecposy[scelda]=0 THEN
    MARCHAY:=FALSE;
    END_IF
    IF vecposy[scelda]>0 THEN
    MARCHAY:=TRUE;
    END_IF
MARCHAX:=TRUE;
D2:=TRUE;
D1:=FALSE;
END_IF

IF D2 AND ((SRY.Q1 OR vecposy[scelda]=0) AND SX.Q1) THEN
MARCHAX:=FALSE;
MARCHAY:=FALSE;
D3:=TRUE;
D2:=FALSE;
END_IF
```

Figura 1. 42. Programación modo automático. Descargar. Paso 1.

- Mover Z e Y

En primer lugar moverá el eje Z, para determinar hacia qué lado gira el motor, utilizara la función anteriormente explicada, cuando finalice el movimiento moverá el eje Y, en sentido derecho para levantar el palé, para después volver a mover el eje Z en el sentido contrario que lo hizo anteriormente.

```
(*Mover ZY*)

IF D3 THEN
sentz:=sentzc;
MARCHAZ:=TRUE;
RX:=TRUE;
RY:=TRUE;
D4:=TRUE;
D3:=FALSE;
END_IF

IF D4 AND SRZ.Q1 THEN
PULY:=5;
senty:= TRUE;
sentz:=NOT sentzc;
MARCHAY:=TRUE;
MARCHAZ:=FALSE;
D5:=TRUE;
RZ:=TRUE;
RX:=FALSE;
RY:=FALSE;
D4:=FALSE;
END_IF

IF D5 AND SRY.Q1 THEN
OCUPADO[sceIda]:=FALSE;
MARCHAZ:= TRUE;
MARCHAY:= FALSE;
D6:=TRUE;
RY:= TRUE;
RZ:= FALSE;
D5:= FALSE;
END_IF

IF D6 AND SRZ.Q1 THEN
MARCHAZ:= FALSE;
RY:= FALSE;
RZ:= TRUE;
D6:= FALSE;
D7:=TRUE;
END_IF
```

Figura 1. 43Programación modo automático. Descargar. Paso 2.

- Mover X e Y

Ahora moverán los ejes X e Y en sentido izquierdo los mismos pulsos que movieron al inicio de la descarga.

```
(*Mover XY*)

IF D7 THEN
senty:=FALSE;
sentx:=FALSE;
PULX:=vecposx[scelda];
PULY:=vecposy[scelda];
    IF vecposy[scelda]=0 THEN
    MARCHAY:=FALSE;
    END_IF
    IF vecposy[scelda]>0 THEN
    MARCHAY:=TRUE;
    END_IF
MARCHAX:=TRUE;
D8:=TRUE;
D7:=FALSE;
END_IF

IF D8 AND ((SRY.Q1 OR vecposy[scelda]=0) AND SX.Q1) THEN
MARCHAX:=FALSE;
MARCHAY:=FALSE;
RZ:=FALSE;
D8:=FALSE;
RY:=TRUE;
RX:=TRUE;
D9:=TRUE;
END_IF
```

Figura 1. 44. Programación modo automático. Descargar. Paso 3.

- Mover Z e Y

A continuación moverá el eje Z, siempre hacia el lado derecho ya que ahí se encuentra la mesa de carga/descarga, después moverá el Y, siempre en sentido izquierdo para dejar el palé y después el Z en el sentido contrario. De esta forma tendremos el palé descargado y el bastidor móvil listo para iniciar otra maniobra.

```
(*Mover ZY*)

IF D9 THEN
sentz:=TRUE;
MARCHAZ:=TRUE;
RX:=TRUE;
RY:=TRUE;
D10:=TRUE;
D9:=FALSE;
END_IF

IF D10 AND SRZ.Q1 THEN
PULY:=5;
senty:= FALSE;
sentz:=FALSE;
MARCHAY:=TRUE;
MARCHAZ:=FALSE;
D11:=TRUE;
RZ:=TRUE;
RX:=FALSE;
RY:=FALSE;
D10:=FALSE;
END_IF

IF D11 AND SRY.Q1 THEN
OCUPADO[scelda]:=FALSE;
MARCHAZ:= TRUE;
MARCHAY:= FALSE;
D12:=TRUE;
RY:= TRUE;
RZ:= FALSE;
D11:= FALSE;
END_IF

IF D12 AND SRZ.Q1 THEN
MARCHAZ:= FALSE;
RY:= FALSE;
RZ:= TRUE;
D12:= FALSE;
DESCARGAR:=FALSE;
END_IF
```

Figura 1. 45. Programación modo automático. Descargar. Paso 4.

1.7.5 Solución adoptada a nivel de supervisión: Scada

A continuación se van a mostrar las dos pantallas del scada. Tendremos una pantalla principal en la que tendremos el control de Stock y los controles automáticos. En la otra tendremos los controles manuales.

1.7.5.1 Modo manual

En la parte central, derecha en la imagen siguiente, podemos observar la posición de la cuna de elevación. Se puede ver la representación de las celdas y de la mesa de carga. En primer lugar vemos una vista de planta en los que podremos ver donde se encuentran los ejes X e Y. Junto a esto podremos ver los cinco niveles de los que se dispone para ver el movimiento del eje X. En la mesa de carga vemos una señalización luminosa, verde cuando no está cargada y roja cuando lo está. En verde/rojo se han representado las seguridades, así podremos saber si están en condiciones de operación o no respectivamente. En la parte superior vemos la selección de modo manual. A la izquierda tenemos quince botones y tres contadores, un contador y cinco botones para cada eje. Cada contador nos indicará la posición de cada motor. Al pulsar avanzar/retroceder se moverá el motor correspondiente dos pulsos. En el momento que se pulse el botón de reinicio de posición el contador se pondrá a cero y veremos con en la visualización de la posición que la cuna de elevación se ha ido a su posición cero. También se dispone de dos pulsadores que simularán los finales de carrera superior e inferior, si pulsamos el final de carrera superior no nos dejara subir pero si bajar, si pulsamos el inferior ocurrirá al contrario. A la parte derecha podemos ver el estado de los finales de carrera, en verde si no está activado y rojo si se encuentra activo, en las siguiente imágenes podemos ver la pantalla del Scada "modo manual". Dicha pantalla se ha dividido en dos para poder observarla correctamente. Al pulsar el límite superior y el inferior de los ejes X y Z respectivamente Abajo a la izquierda tendremos el botón de masa cargada y el de pasar a la visualización del modo automático.

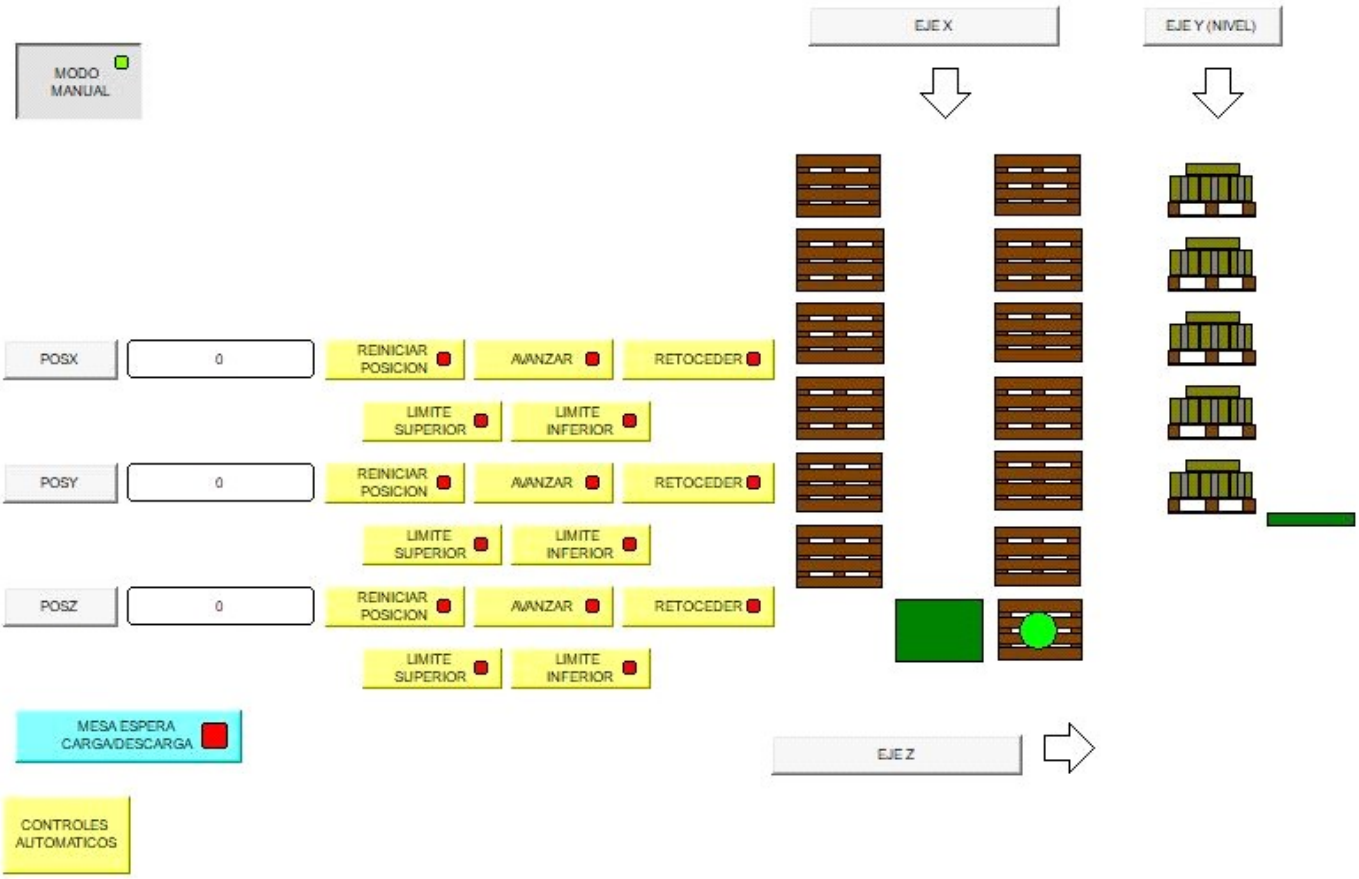


Figura 1. 46. Scada, controles manuales.

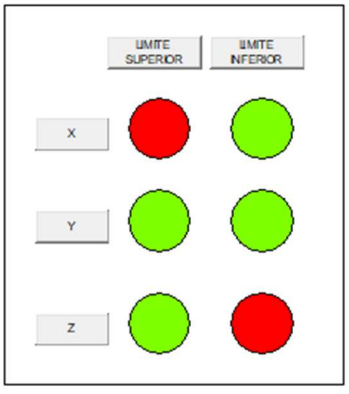


Figura 1. 47. Scada, controles manuales. Detalle estado finales de carrera.

1.7.5.2 Modo automático

Esta pantalla la tenemos diferenciada en dos zonas, en la zona derecha podemos ver las celdas de almacenamiento, están disponen de una señalización luminosa, roja cuando está ocupada y verde cuando está disponible, también se dispone de un contando de tiempo por celda que indica el tiempo total de almacenamiento.

Figura 1. 48. Scada, control de stock y tiempos.

En la parte superior del lado izquierdo dispondremos de los controles automáticos, selección de modo automático, "marcha/Pausa" "paro" y "reset", justo debajo de la selección de modo encontramos el número de celda seleccionado para realizar la carga/descarga, si pulsamos encima del número nos aparecerá un teclado numérico para seleccionar la celda, este está limitado a 60. Debajo de los controles podemos observar la posición de la cuna de elevación de la misma manera que se ha explicado anteriormente. A la izquierda tenemos 5 contadores, tres de ellos nos indican la posición de los motores y los otros dos las celdas ocupadas y las celdas vacías. Bajo estos están las seguridades y el pulsador de mesa cargada, ya que estos los simularemos desde el Scada. Abajo del todo está el pulsador para cambiar a la pantalla del modo manual.

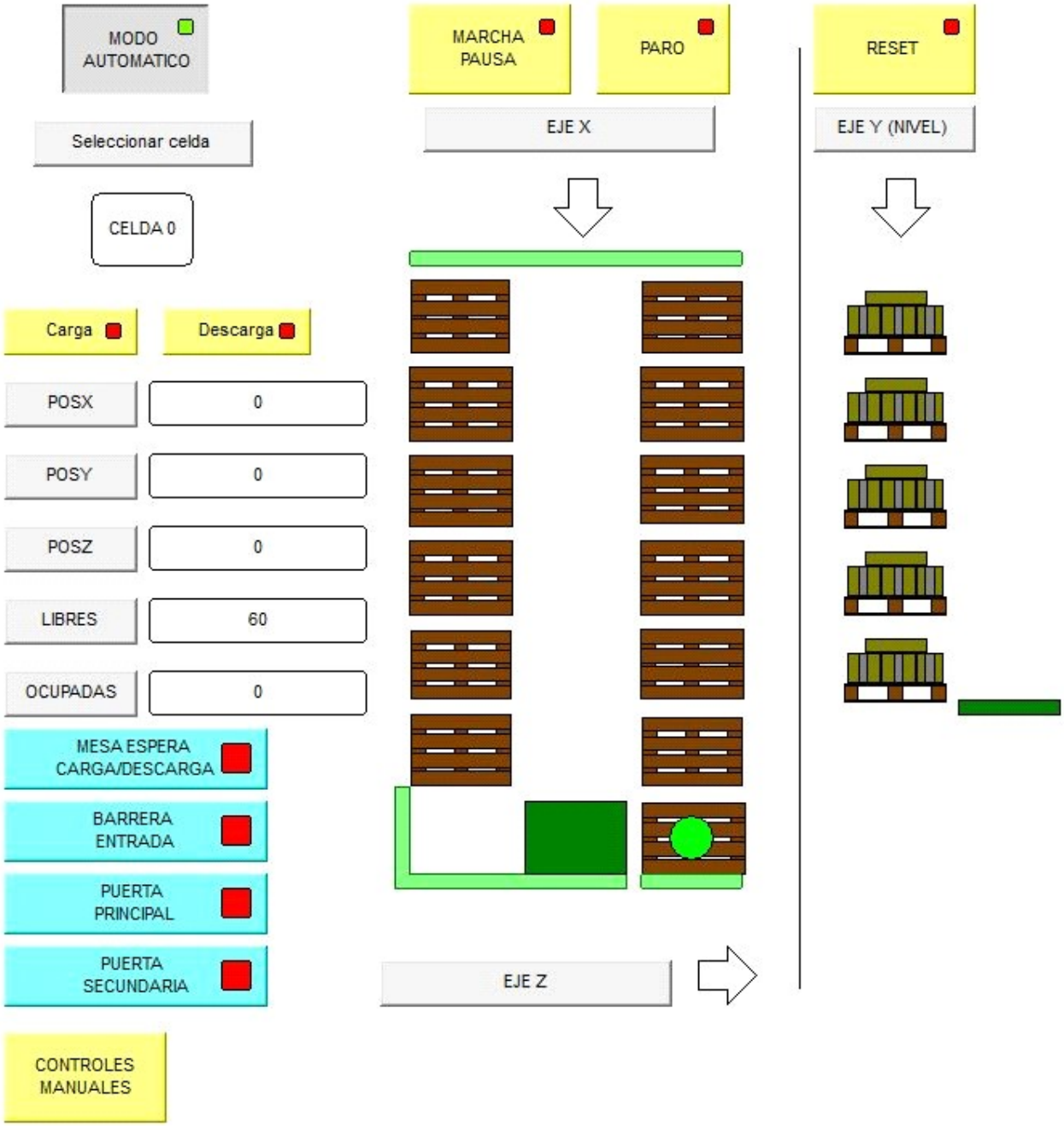


Figura 1. 49. Scada, controles automáticos.

1.8 Verificación y funcionamiento

Una vez realiza la programación y las conexiones necesarias se han comprobado en ausencia de tensión, el cableado y las conexiones efectuadas entre autómata, servoamplificador, motores y pc. Para protección contra contactos directos e indirectos se han utilizado las protecciones disponibles en el laboratorio de accionamientos electromecánicos.

Se ha comprobado el funcionamiento de todos los elementos instalados realizando desde el Scada las acciones de carga y descarga, movimientos en manual y activación de las diferentes protecciones programadas en el autómata.

Después de realizar las pruebas de verificación y funcionamiento se ha podido observar el funcionamiento esperado de todos los elementos y de la programación realizada.

1.9 Conclusiones

Gracias a la realización del almacén automático se obtienen unas mejoras frente a un almacén convencional tanto en seguridad, ya que tenemos un proceso muy controlado, espacio, ya que no necesitamos espacio para maniobrar con maquinaria y rapidez al poder tener la maquina siempre funcionando. Por otro lado también se necesita menos mano de obra y obtenemos un mayor control del stock.

La programación del transelevador se ha llevado a cabo con el objetivo de obtener un control automático sobre los tres ejes de dicho transelevador así como una monitorización y un control de Stock del almacén.

La programación se ha realizado de manera que en caso de querer utilizarla en un almacén de diferentes características los cambios sean mínimos y de esa manera poder rentabilizar el trabajo y no tener que realizar el proyecto desde cero en el caso de querer utilizarla en más almacenes.

El lenguaje seleccionado para llevar a cabo la programación ha sido el texto estructurado (ST). Se ha seleccionado este lenguaje ya que es muy habitual ver este lenguaje en la industria y con él se pueden realizar acciones complejas con diferentes variables y datos de una manera más rápida. También cabe destacar que era un lenguaje que no había usado con anterioridad y me parecía muy interesante poder conocer un lenguaje tan usual como este.

Una de las mejoras que no se ha podido llevar a cabo sería la de conectar los finales de carrera con el autómata para que en el caso que se active el final de carrera pare el motor y la visualización y no solo el motor como ocurre ahora, ya que la señal del final de carrera actúa directamente sobre el servoamplificador. En el presente proyecto no se ha podido llevar a cabo por la falta de entradas en el autómata. Se han diseñado unos botones en el Scada para simular dicha acción.

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

UNIVERSITAT POLITECNICA DE VALENCIA
ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA DEL DISEÑO

CONTROL Y VISUALIZACIÓN DE UN ALMACÉN AUTOMÁTICO

2. PLIEGO DE CONDICIONES

ÍNDICE DEL PLIEGO DE CONDICIONES

2.1 Introducción

El presente pliego de condiciones se ha diseñado únicamente del prototipo realizado en el laboratorio así como de los elementos utilizados para la realización del mismo. En el caso que el presente proyecto se quisiera utilizar para el control de un transelevador se debería realizar el pliego de condiciones teniendo en cuenta todos los elementos de la maquinaria necesaria.

2.2 Normativa vigente

El proyecto ha sido realizado respetando y acatando la normativa vigente.

En lo que respecta a la parte eléctrica y electrónica se ha utilizado la normativa expuesta en el Reglamento Electrotécnico para Baja Tensión, aprobado por el Real Decreto 842/2002, de 2 de agosto y publicado en el BOE nº224, de 18 de septiembre de 2002. Y de sus Instrucciones Técnicas Complementarias basadas en las normas UNE.

Cabe destacar las siguientes instrucciones:

ITC-BT-01: Terminología.

ITC-BT-19: Instalaciones interiores o receptoras. Prescripciones generales.

ITC-BT-20: Instalaciones interiores o receptoras. Sistemas de instalación.

ITC-BT-22: Instalaciones interiores o receptoras. Protección contra sobreintensidades.

ITC-BT-23: Instalaciones interiores o receptoras. Protección contra sobretensiones.

ITC-BT-24: Instalaciones interiores o receptoras. Protección contra contactos directos e indirectos.

ITC-BT-43: Instalación de receptores. Prescripciones generales.

ITC-BT-47: Instalación de receptores. Motores. También se ha tenido en cuenta la normativa sobre Seguridad Industrial expuesta por el Ministerio de Industria, Turismo y Comercio.

Así como la Directiva 2006/42/CE del parlamento europeo y del consejo sobre máquinas y la normativa para la seguridad de las maquinas UNE-EN ISO 13850 del 2016.

Debido al carácter educacional de este proyecto no se tendrán en cuenta las medidas de seguridad que se especifican en la norma de operación remota sobre máquinas.

2.3 Condiciones a satisfacer por los componentes

2.3.1 Cableado eléctrico

Para proteger la instalación y evitar riesgos eléctricos para las máquinas y las personas todos los conductores utilizados dispondrán de aislamiento de PVC y se tendrán en cuenta las secciones necesarias en cada caso.

Los conductores utilizados para la para la alimentación del autómata así como para la señales al servoamplificador serán de 0,5mm^2 y las mangueras utilizadas para la potencia de los motores serán de 1,5mm^2 según nos indica el fabricante. Por otro lado para la alimentación del servoamplificador como para la de la fuente de alimentación se utilizara una manguera de 2,5mm^2.

Todos los elementos utilizados que se indican el presente apartado den cumplir con el RD 560/2010. Por ello todos los conductores serán no propagadores de llama y libres de alógenos.

2.3.2 Resistencias

Entre la salidas del autómata y la entrada del servoamplificador que se conectan a las entradas de pulsos, se han colocado unas resistencias de 1200 Ohmios para mantener los valores de tensión necesarios en el servodriver y evitar errores y posibles averías. Las resistencias cumplirán la normativa sobre resistencias UNE 20545.

2.3.3 Interruptores y pulsadores

Todos los interruptores utilizados en el presente proyecto cumplirán con las normas UNE 61058 y UNE 60669.

2.3.4 Ordenador personal

El ordenar necesario contará con una CPU, un monitor, un ratón y un teclado. Hay que tener en cuenta las condiciones del entorno en el que va a ser utilizado el

ordenador ya que si se usa en un entorno industrial podría estar trabajando en ambientes con suciedad, humedad, temperatura u otros.

A continuación se detallan las características técnicas:

- -Microprocesador Intel core 2 duo a 1.8GHZ.
- -Memoria RAM de 4096 Mb.
- -Espacio libre en el disco duro de 2Gb.
- -Sistema operativo Windows 7.

A nivel de software necesitaremos el programa CoDeSyS para la realización de la programación del autómata y el Scada, y el programa BSD para la realización de la programación del servoamplificador.

2.3.5 Fuente de alimentación

Para la alimentación del autómata programable se ha utilizado una fuente de alimentación que nos proporciona 24V de corriente continua, está se conecta a 230v de corriente alterna a la red.

2.3.6 Autómata programable

El autómata programable será alimentado por una fuente de alimentación de 24V tal y como se indica en el apartado anterior. El autómata programable junto con el módulo de entradas/salidas y la fuente de alimentación irán colocados en el carril DIN.

2.4 Pruebas de funcionamiento

Para asegurar la correcta colocación y conexión de los elementos utilizados se llevarán a cabo unas pruebas de funcionamiento y verificación. Estas pruebas se realizarán teniendo en cuenta lo que indica la normativa del REBT en su instrucción técnica complementaria número 5, ITC-BT-05.

2.4.1 Revisión visual y de continuidad

Se revisará y verificara la correcta sujeción de los elementos utilizados y la correcta conexión del cableado. También se revisará que las partes móviles no entren en contacto con otros elementos.

2.4.2 Pruebas en tensión

Se conectarán los elementos y se comprobará que los elementos de seguridad funcionan de una manera adecuada, una vez se verifique el correcto funcionamiento se procederá a probar el resto de elementos.

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

UNIVERSITAT POLITECNICA DE VALENCIA
ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA DEL DISEÑO

CONTROL Y VISUALIZACIÓN DE UN ALMACÉN AUTOMÁTICO

3. PRESUPUESTO

ÍNDICE DEL PRESUPUESTO

3.1 Introducción

En el presente documento se van a detallar los gastos necesarios para la obtención de los materiales, software y de la mano de obra para poder realizar el montaje en el laboratorio. No se contemplan los gastos de instalación y maquinaria en el caso que este proyecto se quisiera llevar a cabo para una aplicación real de un transelevador.

3.2 Coste de los materiales

En la siguiente tabla se muestran los elementos utilizados, el cableado y pequeño material engloba el cableado para la conexión de los dispositivos así como el tablero, el carril DIN, los pulsadores, etc.

MATERIALES			
Concepto	cantidad(ud.)	precio unitario(€)	total(€)
Autómata ABB ACS500 PM571	1	774,00 €	774,00 €
Modulo de E/S DC541	1	241,00 €	241,00 €
Servodriver ABB DSB0200	3	489,90 €	1.469,70 €
Servomotor ABB BSM0200CN00	3	272,58 €	817,74 €
Fuente de alimentación CP-C 24/5.0	1	126,00 €	126,00 €
Cable señal encoder	3	52,49 €	157,47 €
Cable potencia servomotor	3	32,59 €	97,77 €
Otro cableado y pequeño material	1	100,00 €	100,00 €
TOTAL MATERIALES			3.528,44 €

Tabla 3. 1. Materiales.

3.3 Coste Software

Las licencias son anuales, con lo que se dividirá el precio anual por las horas de trabajo en el presente proyecto para luego según las horas de uso de cada programa poder aplicar un precio justo de coste por licencia. Contando que se trabaja de lunes a viernes y descontando 30 días de vacaciones obtenemos 225 días de trabajo al año, de esta manera, trabajando 8 horas diarias, obtenemos 1800 horas de trabajo.

A continuación se detalla el precio del software necesario para la programación del autómata, creación del Scada y configuración de servodriver.

SOFTWARE					
Concepto	precio unitario(€)	Horas trabajo/año	Precio hora	Horas utilización	total(€)
CoDeSyS	1.200,00 €	1800	0,67 €	200	133,33 €
BSD	400,00 €		0,22 €	13	2,89 €
Office	126,00 €		0,07 €	90	6,30 €
TOTAL SOFTWARE					142,52 €

Tabla 3. 2. Software.

3.4 Coste mano de obra

En la siguiente tabla se detallan los costes en montaje, programación, cableado y otras labores que debe realizar el operario. Se ha estimado un precio de 20€/hora

MANO DE OBRA			
Concepto	cantidad(h)	precio unitario(€)	total(€)
Montaje	15	20,00 €	300,00 €
Programación del autómata	135		2.700,00 €
Programación del SCADA	55		1.100,00 €
Programación del servodriver	3		60,00 €
Pruebas y verificaciones	20		400,00 €
Redacción del proyecto	90		1.800,00 €
TOTAL MANO DE OBRA			4.560,00 €

Tabla 3. 3. Mano de obra.

3.5 Gastos generales

Los gastos generales son los gastos del negocio no relacionados directamente con el producto, luz, agua, personal administrativo, servicios, etc. Estos gastos siempre son necesarios independientemente del volumen de producción de la empresa. En este caso se aplicará el 20% en concepto de gastos generales.

Para poder calcular los gastos generales debemos obtener en primer lugar el coste de software, materiales y mano de obra para después poder calcular el porcentaje de gastos generales:

GASTOS GENERALES		
Concepto	ud.	precio
Software		142,52 €
Materiales		3.528,44 €
Mano de obra		4.560,00 €
Total fabricación		8.230,96 €
Gastos generales	20%	1.646,19 €

Tabla 3. 4. Gastos generales.

3.6 Beneficio industrial

A continuación se aplicará el beneficio que obtendrá la empresa por realizar el presente proyecto. Se ha estimado un 25% del coste del proyecto en concepto de beneficio industrial.

BENEFICIO INDUSTRIAL		
Concepto	ud.	precio
Software		142,52 €
Materiales		3.528,44 €
Mano de obra		4.560,00 €
Total fabricación		8.230,96 €
Beneficio	25%	2.057,74 €

Tabla 3. 5. Beneficio industrial.

3.7 Coste total

Para obtener el coste total sumaremos el total fabricación, los gastos generales y el beneficio industrial y aplicaremos el 21% en concepto de IVA y de esta manera conseguiremos el coste total del proyecto.

COSTE TOTAL		
Concepto	ud.	precio
Total fabricación		8.230,96 €
Gastos generales		1.646,19 €
Beneficio		2.057,74 €
Total sin IVA		11.934,90 €
IVA	21%	2.506,33 €
COSTE TOTAL		14.441,22 €

Tabla 3. 6. Coste total.

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

UNIVERSITAT POLITECNICA DE VALENCIA
ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA DEL DISEÑO

CONTROL Y VISUALIZACIÓN DE UN ALMACÉN AUTOMÁTICO

4. ANEXOS

ÍNDICE DE LOS ANEXOS

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

UNIVERSITAT POLITECNICA DE VALENCIA
ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA DEL DISEÑO

CONTROL Y VISUALIZACIÓN DE UN ALMACÉN AUTOMÁTICO

4.1 ANEXO 1: Listado de programa

ÍNDICE DEL ANEXO 1

4.1.1 Declaración de variables.

```
PROGRAM PLC_PRG
VAR
    miDC541_IO: DC541_IO;

    MARCHA, PARO, MARCHAX, MARCHAY, MARCHAZ: BOOL;
    vel: LREAL;
    PULX, PULY, PULZ: DWORD;
    sentz, sentx, senty, sentzc: BOOL;
    MCARGA, CARGA, CARGAR, DESCARGA, DESCARGAR: BOOL;
    PUERTAP, PUERTAS, BARRERA: BOOL;
    XLS, XLI, ZLS, ZLI, YLS, YLI: BOOL;

    R1,R2,R3,R4,R5,R6,R7,R8,R9,R10,R11,R12: BOOL;
    D1,D2,D3,D4,D5,D6,D7,D8,D9,D10,D11,D12: BOOL;

    vecposx: ARRAY [1..60] OF INT:=
    10,20,30,40,50,60,10,20,30,40,50,60,10,20,30,40,50,60,10,20,30,40,50,60,
    10,20,30,40,50,60,10,20,30,40,50,60,10,20,30,40,50,60,10,20,30,40,50,60,10,20,30,40,50,60;
    vecposy: ARRAY [1..60] OF INT :=6(0),6(30),6(60),6(90),6(120),6(0),6(30),6(60),6(90),6(120);
    scelda: INT;
    Ocupado: ARRAY [1..60] OF BOOL;
    con: ARRAY [1..60] OF TIME;
    RTC2:ARRAY [1..60] OF RTC;

    R_TRIGSUMA, R_TRIGRESTA, R_TRIGX,R_TRIGY,R_TRIGZ, RTZ, RTY, TX, TY: R_TRIG;
    PARADA,SRM,SRA,SRY,SRZ,SY,SX: SR;
    CONX,CONY,CONZ:BOOL;
    posX,posY,posZ: INT;

    AVANZARX, AVANZARY, AVANZARZ, RETROCEDERX, RETROCEDERY, RETROCEDERZ:BOOL;
    RX, RY, RZ:BOOL;

    GFREQ_Z: DC541_FREQ_OUT;
    GFREQ_Y: DC541_FREQ_OUT;
    GFREQ_X: DC541_FREQ_OUT;

    RESETY, RESETZ, RESETX: BOOL;
    MANUAL, AUTOMATICO: BOOL;
    RESET: BOOL;
    p:INT;
    ACTIVO: BOOL;
    Stock: INT;
    RESTA, SUMA: BOOL;
END_VAR
VAR_INPUT
END_VAR
```

4.1.2 Programa de control.

```
miDC541_IO(EN:=TRUE,SLOT:=1);
miDC541_IO.OUT6:=sentx;
miDC541_IO.OUT4:=senty;
miDC541_IO.OUT2:=sentz;
miDC541_IO.OUT0:=ACTIVO;
SRM(SET1:=MANUAL , RESET:=AUTOMATICO OR (NOT MANUAL), Q1=> );
SRA(SET1:=AUTOMATICO , RESET:=MANUAL OR (NOT AUTOMATICO), Q1=> );
PARADA(SET1:=PARO OR PUERTAP OR PUERTAS OR BARRERA , RESET:=RESET, Q1=>);
IF SRM.Q1 OR SRA.Q1 THEN
ACTIVO:=TRUE;
ELSE
ACTIVO:=FALSE;
END_IF

(*PARADAS DE EMERGENCIA*)

IF PARADA.Q1 THEN
MARCHA:=TRUE;
END_IF

IF XLS AND sentx THEN
MARCHAX:=FALSE;
END_IF
IF XLS AND (NOT sentx) THEN
MARCHAX:=FALSE;
END_IF

IF YLS AND senty THEN
MARCHAY:=FALSE;
END_IF
IF YLS AND (NOT senty) THEN
MARCHAY:=FALSE;
END_IF

IF ZLS AND sentz THEN
MARCHAZ:=FALSE;
END_IF
IF ZLS AND (NOT sentz) THEN
MARCHAZ:=FALSE;
END_IF

(*SENTIDO DE GIRO DEL EJE Z SEGUN LA CELDA ELEGIDA PARA LA CARGA O DESCARGA*)

IF  scelda<31 THEN
sentzc:=0;
END_IF
IF scelda>30 THEN
sentzc:=1;
END_IF

(*ALMACENAR*)
TX(CLK:=GFREQ_X.RDY , Q=> );
RTY(CLK:=GFREQ_Y.RDY , Q=> );
RTZ(CLK:=GFREQ_Z.RDY , Q=> );
SRY(SET1:=RTY.Q , RESET:=RY, Q1=> );
SRZ(SET1:=RTZ.Q , RESET:=RZ, Q1=> );
SX(SET1:=TX.Q , RESET:=RX, Q1=> );

IF SRA.Q1 AND CARGA AND MCARGA  AND NOT CARGAR  AND NOT DESCARGAR AND NOT
Ocupado[scelda] THEN
CARGAR:=TRUE;
RX:=FALSE;
RY:=FALSE;
RZ:=FALSE;
```

```
R1:=TRUE;
END_IF

(*mover zy*)

IF CARGAR AND R1 THEN
vel:=5;
sentz:=TRUE;
PULZ:=8;
MARCHAZ:=TRUE;
R2:=TRUE;
R1:=FALSE;
END_IF

IF R2 AND SRZ.Q1 THEN
PULY:=5;
senty:= TRUE;
sentz:=FALSE;
MARCHAY:=TRUE;
MARCHAZ:=FALSE;
R3:=TRUE;
RZ:=TRUE;
R2:=FALSE;
END_IF

IF R3 AND SRY.Q1 THEN
MARCHAZ:= TRUE;
MARCHAY:= FALSE;
R4:=TRUE;
RY:= TRUE;
RZ:= FALSE;
R3:= FALSE;
END_IF

IF R4 AND SRZ.Q1 THEN
MARCHAZ:= FALSE;
RY:= FALSE;
R4:= FALSE;
R5:=TRUE;
END_IF

(*mover xy*)

IF R5 THEN
RZ:=TRUE;
sentz:=TRUE;
sentx:=TRUE;
PULX:=vecposx[scelda];
PULY:=vecposy[scelda];
    IF vecposy[scelda]=0 THEN
    MARCHAY:=FALSE;
    END_IF
    IF vecposy[scelda]>0 THEN
    MARCHAY:=TRUE;
    END_IF
MARCHAX:=TRUE;
R6:=TRUE;
R5:=FALSE;
END_IF

IF R6 AND ((SRY.Q1 OR vecposy[scelda]=0) AND SX.Q1) THEN
MARCHAX:=FALSE;
MARCHAY:=FALSE;
RZ:=FALSE;
R7:=TRUE;
R6:=FALSE;
```

```
END_IF

(*mover zy*)

IF R7 THEN
sentz:=sentzc;
MARCHAZ:=TRUE;
RX:=TRUE;
RY:=TRUE;
R8:=TRUE;
R7:=FALSE;
END_IF

IF R8 AND SRZ.Q1 THEN
PULY:=5;
senty:= FALSE;
sentz:=NOT sentzc;
MARCHAY:=TRUE;
MARCHAZ:=FALSE;
R9:=TRUE;
RZ:=TRUE;
RX:=FALSE;
RY:=FALSE;
R8:=FALSE;
END_IF

IF R9 AND SRY.Q1 THEN
OCUPADO[scelda]:=TRUE;
MARCHAZ:= TRUE;
MARCHAY:= FALSE;
R10:=TRUE;
RY:= TRUE;
RZ:= FALSE;
R9:= FALSE;
END_IF

IF R10 AND SRZ.Q1 THEN
MARCHAZ:= FALSE;
RY:= FALSE;
RZ:= TRUE;
R10:= FALSE;
R11:=TRUE;
END_IF

(*mover xy*)

IF R11 THEN
sentz:=FALSE;
sentx:=FALSE;
PULX:=vecposx[scelda];
PULY:=vecposy[scelda];
    IF vecposy[scelda]=0 THEN
    MARCHAY:=FALSE;
    END_IF
    IF vecposy[scelda]>0 THEN
    MARCHAY:=TRUE;
    END_IF
MARCHAX:=TRUE;
R12:=TRUE;
R11:=FALSE;
END_IF

IF R12 AND ((SRY.Q1 OR vecposy[scelda]=0) AND SX.Q1) THEN
MARCHAX:=FALSE;
MARCHAY:=FALSE;
RZ:=FALSE;
```

```
R12:=FALSE;
RY:=TRUE;
RX:=TRUE;
CARGAR:=FALSE;
END_IF

(*DESCARGAR*)

IF SRA.Q1 AND DESCARGA AND (NOT MCARGA)  AND NOT CARGAR  AND NOT DESCARGAR AND
Ocupado[scelda] THEN
DESCARGAR:=TRUE;
RX:=FALSE;
RY:=FALSE;
RZ:=FALSE;
D1:=TRUE;
vel:=5;
END_IF

(*mover xy*)

IF D1 THEN
senty:=TRUE;
sentx:=TRUE;
PULX:=vecposx[scelda];
PULY:=vecposy[scelda];
    IF vecposy[scelda]=0 THEN
    MARCHAY:=FALSE;
    END_IF
    IF vecposy[scelda]>0 THEN
    MARCHAY:=TRUE;
    END_IF
MARCHAX:=TRUE;
D2:=TRUE;
D1:=FALSE;
END_IF

IF D2 AND ((SRY.Q1 OR vecposy[scelda]=0) AND SX.Q1) THEN
MARCHAX:=FALSE;
MARCHAY:=FALSE;
D3:=TRUE;
D2:=FALSE;
END_IF

(*mover zy*)

IF D3 THEN
sentz:=sentzc;
MARCHAZ:=TRUE;
RX:=TRUE;
RY:=TRUE;
D4:=TRUE;
D3:=FALSE;
END_IF

IF D4 AND SRZ.Q1 THEN
PULY:=5;
senty:= TRUE;
sentz:=NOT sentzc;
MARCHAY:=TRUE;
MARCHAZ:=FALSE;
D5:=TRUE;
RZ:=TRUE;
RX:=FALSE;
RY:=FALSE;
D4:=FALSE;
END_IF
```

```
IF D5 AND SRY.Q1 THEN
OCUPADO[scelda]:=FALSE;
MARCHAZ:= TRUE;
MARCHAY:= FALSE;
D6:=TRUE;
RY:= TRUE;
RZ:= FALSE;
D5:= FALSE;
END_IF

IF D6 AND SRZ.Q1 THEN
MARCHAZ:= FALSE;
RY:= FALSE;
RZ:= TRUE;
D6:= FALSE;
D7:=TRUE;
END_IF

(*mover xy*)

IF D7 THEN
senty:=FALSE;
sentx:=FALSE;
PULX:=vecposx[scelda];
PULY:=vecposy[scelda];
    IF vecposy[scelda]=0 THEN
    MARCHAY:=FALSE;
    END_IF
    IF vecposy[scelda]>0 THEN
    MARCHAY:=TRUE;
    END_IF
MARCHAX:=TRUE;
D8:=TRUE;
D7:=FALSE;
END_IF

IF D8 AND ((SRY.Q1 OR vecposy[scelda]=0) AND SX.Q1) THEN
MARCHAX:=FALSE;
MARCHAY:=FALSE;
RZ:=FALSE;
D8:=FALSE;
RY:=TRUE;
RX:=TRUE;
D9:=TRUE;
END_IF

(*mover zy*)

IF D9 THEN
sentz:=TRUE;
MARCHAZ:=TRUE;
RX:=TRUE;
RY:=TRUE;
D10:=TRUE;
D9:=FALSE;
END_IF

IF D10 AND SRZ.Q1 THEN
PULY:=5;
senty:= FALSE;
sentz:=FALSE;
MARCHAY:=TRUE;
MARCHAZ:=FALSE;
D11:=TRUE;
RZ:=TRUE;
```

```
RX:=FALSE;
RY:=FALSE;
D10:=FALSE;
END_IF

IF D11 AND SRY.Q1 THEN
OCUPADO[scelda]:=FALSE;
MARCHAZ:= TRUE;
MARCHAY:= FALSE;
D12:=TRUE;
RY:= TRUE;
RZ:= FALSE;
D11:= FALSE;
END_IF

IF D12 AND SRZ.Q1 THEN
MARCHAZ:= FALSE;
RY:= FALSE;
RZ:= TRUE;
D12:= FALSE;
DESCARGAR:=FALSE;
END_IF


(*CONTADORES DE POSICION DE LOS EJES*)

R_TRIGX(CLK:=miDC541_IO.IN7 , Q=>CONX );
R_TRIGY(CLK:=miDC541_IO.IN1 , Q=> CONY);
R_TRIGZ(CLK:=miDC541_IO.IN3 , Q=> CONZ);

IF sentx AND CONX THEN
posX:=posX+1;
END_IF
IF (NOT sentx) AND CONX THEN
posX:=posX-1;
END_IF

IF senty AND CONY THEN
posY:=posY+1;
END_IF
IF (NOT senty) AND CONY THEN
posY:=posY-1;
END_IF

IF sentz AND CONZ THEN
posZ:=posZ+1;
END_IF
IF (NOT sentz) AND CONZ THEN
posZ:=posZ-1;
END_IF

(*MODO MANUAL*)

(*Movimiento X*)

IF SRM.Q1 AND AVANZARX AND (NOT RETROCEDERX) AND (NOT XLS) THEN
vel:=10;
MARCHA:=TRUE;
MARCHAX:=TRUE;
PULX:=2;
sentx:=TRUE;
END_IF

IF SRM.Q1 AND RETROCEDERX AND (NOT AVANZARX) AND (NOT XLI) THEN
MARCHA:=TRUE;
```

```
MARCHAX:=TRUE;
PULX:=2;
sentx:=FALSE;
END_IF

IF SRM.Q1 AND GFREQ_X.RDY AND (NOT AVANZARX)  AND (NOT RETROCEDERX) THEN
MARCHAX:=FALSE;
RX:=TRUE;
RY:=TRUE;
RZ:=TRUE;
END_IF

(*Movimiento Y*)

IF SRM.Q1 AND AVANZARY AND (NOT RETROCEDERY) THEN
vel:=10;
MARCHA:=TRUE;
MARCHAY:=TRUE;
PULY:=2;
senty:=TRUE;
END_IF

IF SRM.Q1 AND RETROCEDERY AND (NOT AVANZARY) THEN
MARCHA:=TRUE;
MARCHAY:=TRUE;
PULY:=2;
senty:=FALSE;
END_IF

IF SRM.Q1 AND GFREQ_Y.RDY AND (NOT AVANZARY)  AND(NOT RETROCEDERY) THEN
MARCHAY:=FALSE;
RX:=TRUE;
RY:=TRUE;
RZ:=TRUE;
END_IF

(*Movimiento Z*)

IF SRM.Q1 AND AVANZARZ AND (NOT RETROCEDERZ) THEN
vel:=10;
MARCHA:=TRUE;
MARCHAZ:=TRUE;
PULZ:=2;
sentz:=TRUE;
END_IF

IF SRM.Q1 AND RETROCEDERZ AND (NOT AVANZARZ) THEN
MARCHA:=TRUE;
MARCHAZ:=TRUE;
PULZ:=2;
sentz:=FALSE;
END_IF

IF SRM.Q1 AND GFREQ_Z.RDY AND (NOT AVANZARZ) AND (NOT RETROCEDERZ)  THEN
MARCHAZ:=FALSE;
RX:=TRUE;
RY:=TRUE;
RZ:=TRUE;
END_IF

(*Reiniciar contadores de posición*)

IF SRM.Q1 AND RESETX THEN
posX:=0;
END_IF
```

```
IF SRM.Q1 AND RESETY THEN
posY:=0;
END_IF

IF SRM.Q1 AND RESETZ THEN
posZ:=0;
END_IF

(*CONTADORES Y CELDAS OCUPADAS*)

IF (p>=1) AND (p<=60) THEN
RTC2[p](EN:=Ocupado[p] , PDT:=DT#1970-01-01-00:00:00, Q=> , CDT=> );
con[p]:=RTC2[p].CDT-RTC2[p].PDT;
p:=p+1;
ELSE
p:=1;
END_IF

(*CONTROL STOCK*)
R_TRIGSUMA(CLK:=R9 , Q=>SUMA );
R_TRIGRESTA(CLK:=D11 , Q=> RESTA);

IF SUMA THEN
Stock:=Stock +1;
END_IF

IF RESTA THEN
Stock:= Stock -1;
END_IF

(*GENERADOR DE PULSOS*)

GFREQ_Z(SLOT:= 1,          CH:= 3,EN_VISU:= TRUE,EN:= MARCHA,START:= MARCHAZ,       STOP:=
PARADA.Q1,FREQ:= vel,PULSE:= PULZ);

GFREQ_Y(SLOT:= 1,          CH:= 1,EN_VISU:= TRUE,EN:= MARCHA,START:= MARCHAY,       STOP:=
PARADA.Q1,FREQ:= vel,PULSE:= PULY);

GFREQ_X(SLOT:= 1,          CH:= 7,EN_VISU:= TRUE,EN:= MARCHA,START:= MARCHAX,       STOP:=
PARADA.Q1,FREQ:= vel,PULSE:= PULX);
```

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

UNIVERSITAT POLITECNICA DE VALENCIA
ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA DEL DISEÑO

CONTROL Y VISUALIZACIÓN DE UN ALMACÉN AUTOMÁTICO

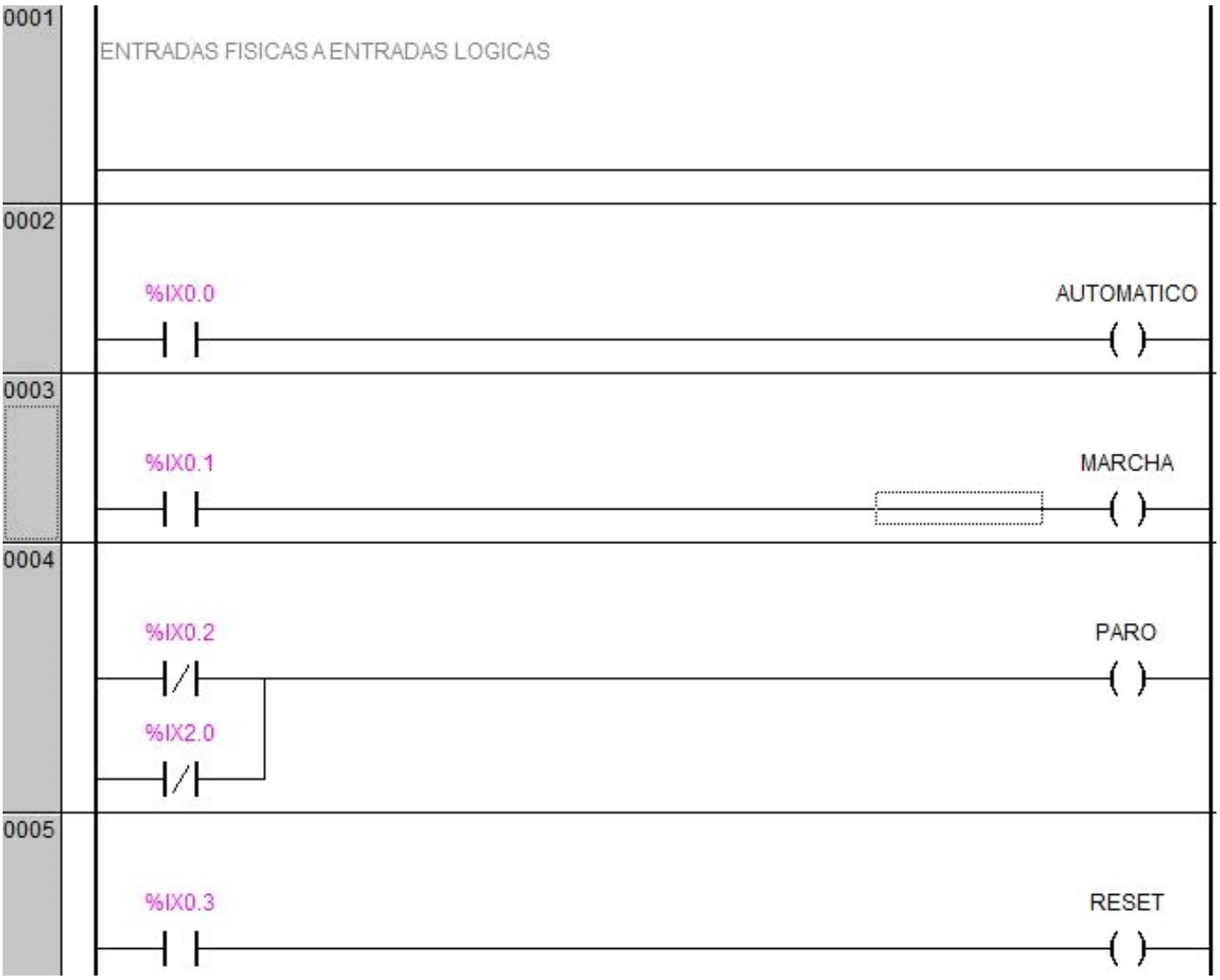
4.2 ANEXO 2: Programa para la gestión de E/S

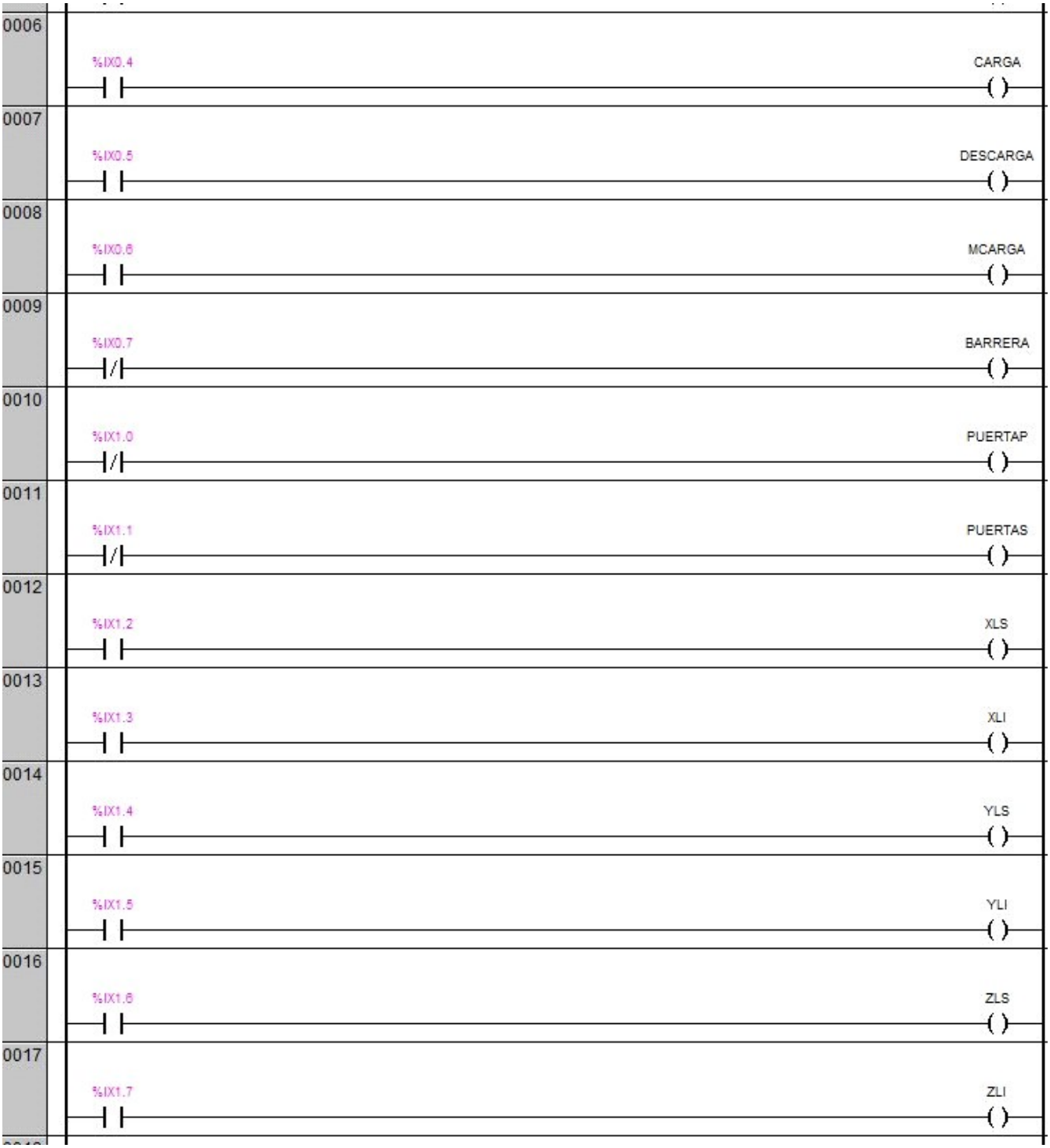
ÍNDICE DEL ANEXO 2

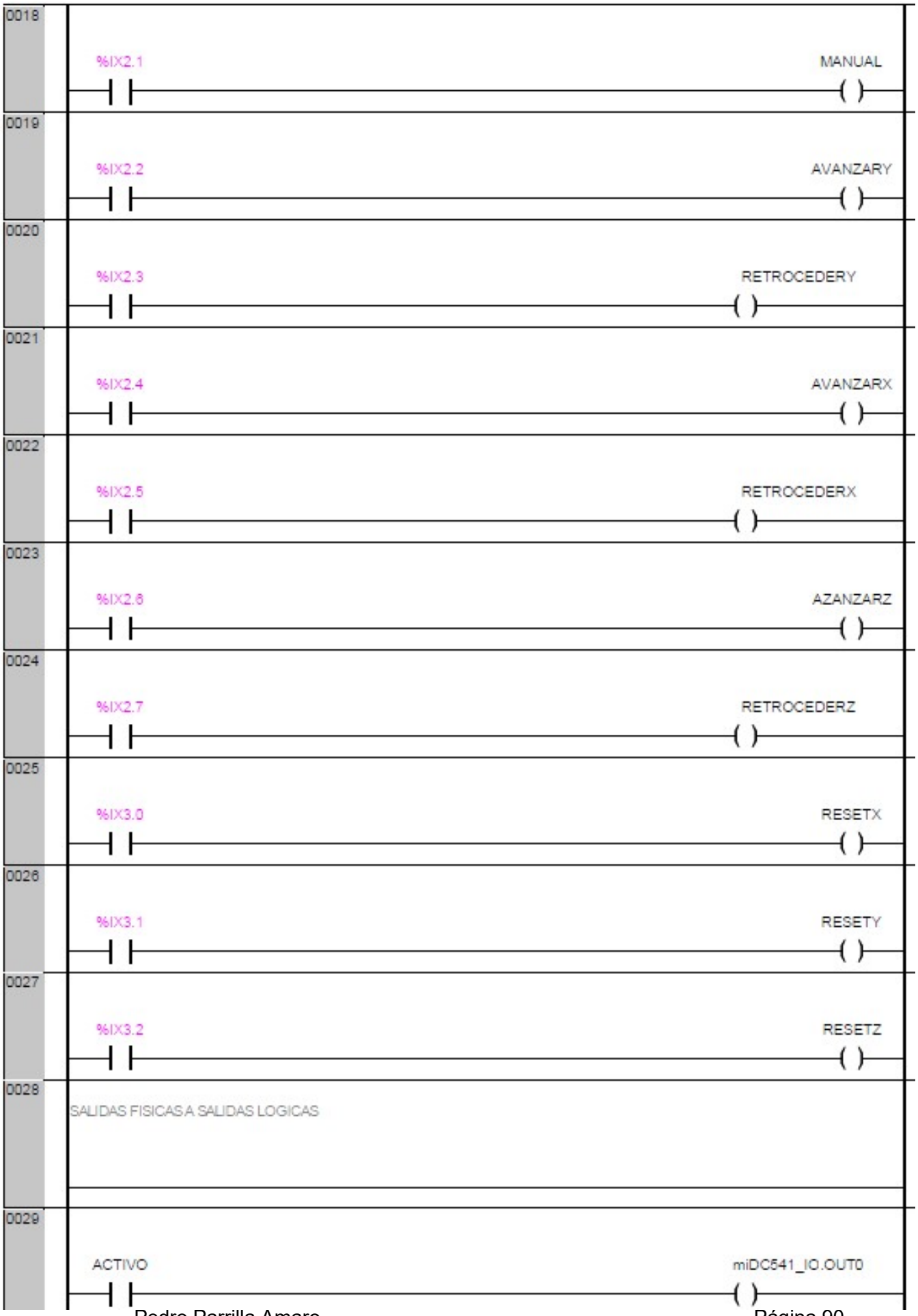
4.2.1 Introducción

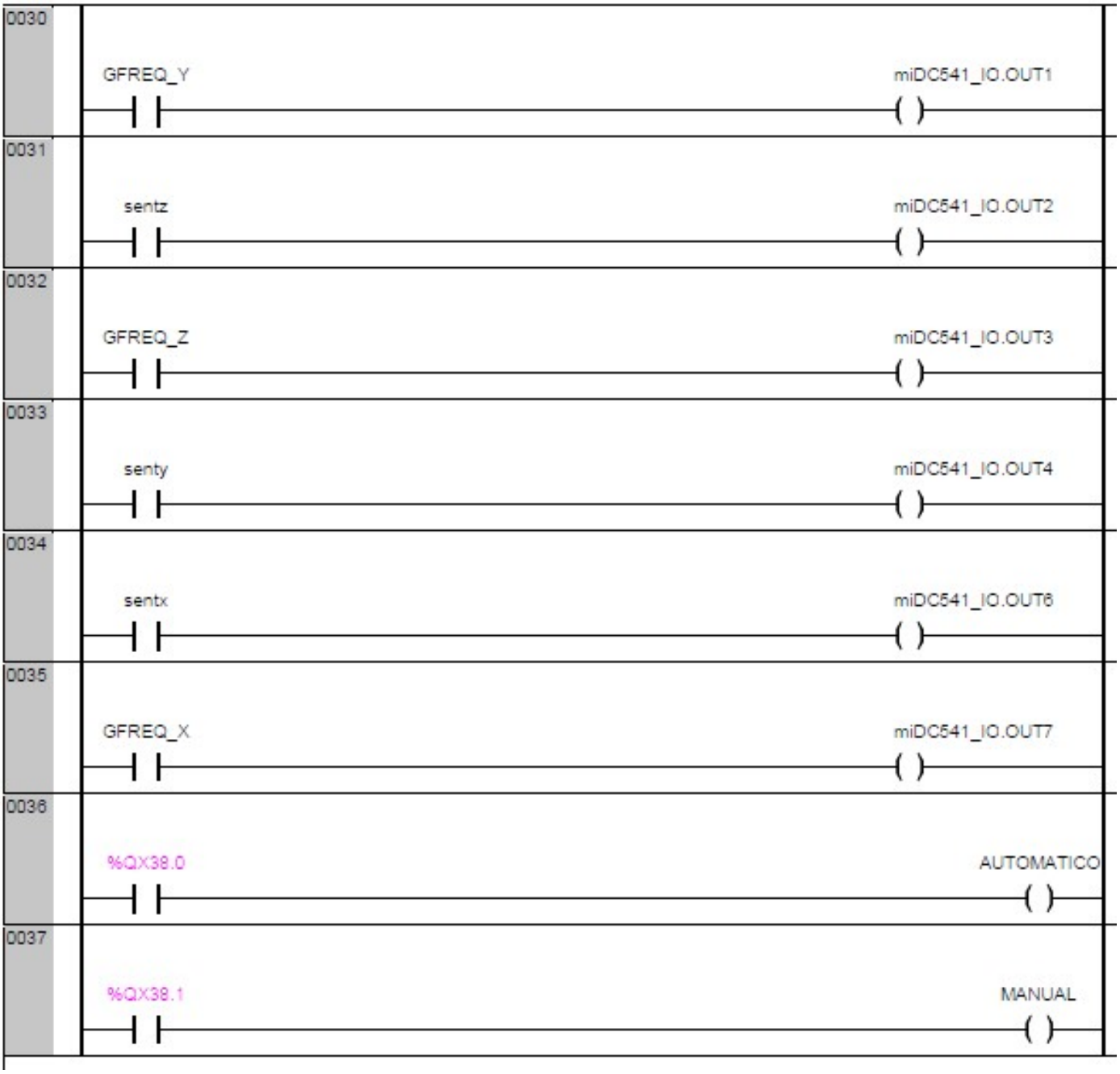
En el siguiente anexo se muestra la programación realizada para la gestión de entradas y salidas. Gracias a la programación mostrada a continuación podremos cambiar las direcciones utilizadas para los sensores y actuadores. De esta manera se consigue que la utilización del programa para diferentes sistemas sea mucho más fácil e intuitiva.

4.2.2 Listado de programa




0006
%IX0.4
CARGA
0007
%IX0.5
DESCARGA
0008
%IX0.6
MCARGA
0009
%IX0.7
BARRERA
0010
%IX1.0
PUERTAP
0011
%IX1.1
PUERTAS
0012
%IX1.2
XLS
0013
%IX1.3
XLI
0014
%IX1.4
YLS
0015
%IX1.5
YLI
0016
%IX1.6
ZLS
0017
%IX1.7
ZLI


0018
%IX2.1
MANUAL
0019
%IX2.2
AVANZARY
0020
%IX2.3
RETROCEDERY
0021
%IX2.4
AVANZARX
0022
%IX2.5
RETROCEDERX
0023
%IX2.6
AZANZARZ
0024
%IX2.7
RETROCEDERZ
0025
%IX3.0
RESETX
0026
%IX3.1
RESETY
0027
%IX3.2
RESETZ
0028
SALIDAS FISICAS A SALIDAS LOGICAS
0029
ACTIVO
miDC541_IO.OUT0


0030
GFREQ_Y
miDC541_IO.OUT1
0031
sentz
miDC541_IO.OUT2
0032
GFREQ_Z
miDC541_IO.OUT3
0033
senty
miDC541_IO.OUT4
0034
sentx
miDC541_IO.OUT6
0035
GFREQ_X
miDC541_IO.OUT7
0036
%QX38.0
AUTOMATICO
0037
%QX38.1
MANUAL

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

UNIVERSITAT POLITECNICA DE VALENCIA
ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA DEL DISEÑO

CONTROL Y VISUALIZACIÓN DE UN ALMACÉN AUTOMÁTICO

4.3 ANEXO 3: Configuración del servodriver

ÍNDICE DEL ANEXO 3

4.3.1 Configuración servodriver, BSD

No	Set item	Set value	Set range	Initial Value
1	Command pulse correction Alpha	7282	1 to 32767	16
2	Command pulse correction Beta	2	1 to 32767	1
3	*Pulse train input form	0	0:Command pulse/Sign 1:Forward/Reverse 2:Tv	1
4	*Rotate Direction/Out pulse pha	0	0:CCW/B-phase advance 1:CW/B-phase advan	0
5	Tuning mode	0	0:Auto tuning 1:Semi auto tuning 2:Manual tunin	0
6	Load inertia ratio	5.0	0.0 to 100.0	5.0
7	Auto tuning gain	10	1 to 20	10
8	Auto forward gain	5	1 to 20	5
9	*Control mode change	0	0:Position 1:Speed 2:Torque 3:Pos./Speed 4:Po:	0
10	*CONT1 signal assignment	1	0:Not assigned 1:RUN 2:RST 3:+OT 4:-OT 5:E	1
11	*CONT2 signal assignment	0	7:Deviation clear 8:External fault input 9:Anti-res	2
12	*CONT3 signal assignment	3	10:Anti-resonant freq. 1 11:INH 12:Alpha select I	0
13	*CONT4 signal assignment	4	14:CSEL 15:FWD 16:REV 17:X1 18:X2 19:AC	0
14	*CONT5 signal assignment	0	22:Reserved for maker (Please refer to parameter	0
15	*OUT1 signal assignment	6	0:Not assigned 1:RDY 2:PSET 3:ALM(contact-/	1
16	*OUT2 signal assignment	1	5:Dynamic braking 6:OT detection 7:Forced stop	2
17	*OUT3 signal assignment	0	9:NZERO 10:Torque limit detection 11:Brake tim	4
18	*OUT4 signal assignment	0	13:Reserved for maker (Please refer to parameter	0
19	*Output pulse count	2048	16 to 32768[pulse]	2048
20	*Z-phase offset	0	0 to 65535[*2pulse]	0
21	Deviation zero width	400	1 to 2000[pulse]	400
22	Deviation over width	20000	10 to 65535[*100pulse]	20000
23	Speed zero width	50	10 to 5000[r/min]	50
24	Judgment time of positioning enc	0.000	0.000 to 1.000[sec]	0.000
25	Maximum current	300	0 to 300[%]	300
26	*Alarm detection at under voltag	1	0:No detection 1:Detect	1
27	*Operation at under voltage	0	0:Rapidly decelerates to stop 1:Free-run	0
28	*Electronic thermal of braking res	0	0:Invalid 1:Valid(for thin form resistor(WSR-***-TD	0
29	Parameter rewriting inhibit	0	0:Rewriting allowed 1:Rewriting inhibit	0
30	*Initial display of keypad panel	0	0 to 21 (Please refer to parameter information for c	0
31	Manual feed speed 1(test runnin	100.0	0.1 to 5000.0[r/min]	100.0
32	Manual feed speed 2	500.0	0.1 to 5000.0[r/min]	500.0
33	Manual feed speed 3	1000.0	0.1 to 5000.0[r/min]	1000.0
34	Maximum speed	5000.0	0.1 to 5000.0[r/min]	5000.0

Tabla 4. 1. Configuración servodriver.

A continuación se detallan los valores modificados para el correcto funcionamiento de los servomotores.

- Parámetros 1 (α) y 2(β).

Estos parámetros son los encargados de determinar la distancia que moverá el motor por cada pulso recibido desde el autómata programable, esto se determina con la siguiente formula:

$$\frac{\propto}{\beta} = \frac{131072}{Distancia} \cdot Paso$$

$$\frac{\propto \cdot 360}{\beta \cdot 2^{17}} = Paso$$

Hemos decidido que cada pulso del autómata se traduzca en un movimiento de 10º en servomotor con lo que:

$$\frac{\propto}{\beta} = \frac{10 \cdot 2^{17}}{360} \rightarrow \alpha = 3641 \cdot \beta$$

Para cumplir la igualdad anterior se han tomado los valores de 7282 y 2 para α y β respectivamente.

- Parámetros del 10 al 14 (cont).

Aquí podemos configurar las entradas al servoamplificador, en nuestro caso hemos configurada las entradas número 10, 12 y 13, en las entradas 11 y 14 hemos colocado un 0 ya que nos utilizamos.

En el parámetro 10 hemos colocado un 1 puesto que por esta entrada vamos a darle la señal de inicio al servodriver.

Los parámetros 12 y 13 los hemos utilizados para los finales de carrera que se han colocado en la botonera exterior, OT+ y OT-.

- Parámetros del 15 al 18 (OUT)

Aquí configuraremos las salidas del servodriver, en los parámetros número 17 y 18 hemos colocado un 0 ya que estas salidas no las utilizaremos.

Los parámetros número 15 y 16 serán los encargados de encender los led de señalización de la caja exterior, el primero se encenderá cuando un final de carrera esté

activado indicando que el motor no se moverá, el 16 se activara cuando el motor esté preparado para iniciar el movimiento.

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

UNIVERSITAT POLITECNICA DE VALENCIA
ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA DEL DISEÑO

CONTROL Y VISUALIZACIÓN DE UN ALMACÉN AUTOMÁTICO

4.4 ANEXO 4: Conector PLC a tres servomotores

ÍNDICE DEL ANEXO 4

4.4.1 Introducción.

Para la conexión desde el autómata programable a los tres servomotores, para poder controlar los tres ejes desde un autómata, se ha cableado una caja de control en la que se han conectado también 2 señalizaciones por motor y 2 interruptores. La señalización nos indica si el motor está preparado para trabajar o si está en fallo, los interruptores serán la simulación de los finales de carrera que dispone el transelevador.

4.4.2 Entradas Servodriver.

A continuación se detallan las entradas del servodriver, estas están disponibles en el manual del servoamplificador "User´s guide".

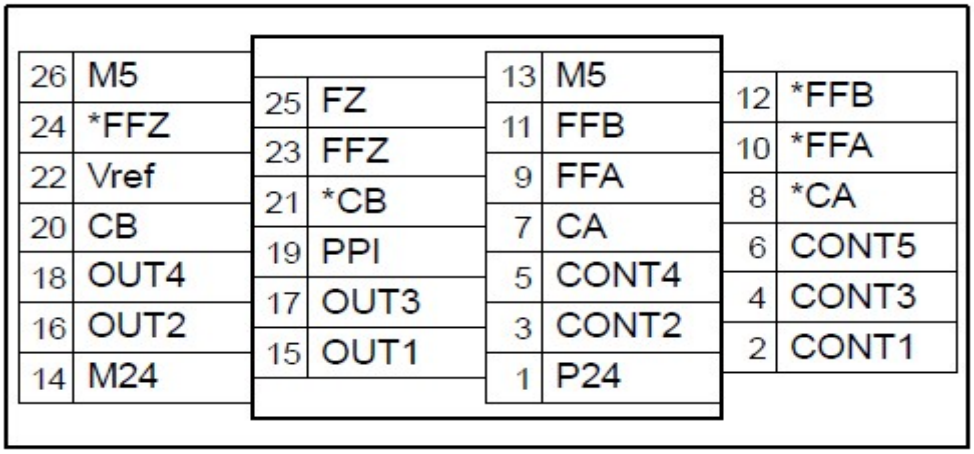


Figura 4. 1.Entradas servoamplificador desde conector CN1.

4.4.3 Conexiones realizadas.

En la siguiente tabla se indican las conexiones realizadas entre el autómata programable y los tres conectores CN1.

SERVOAMPLIFICADOR		CN1		
PIN	VARIABLE	X	Y	Z
1	0V	1.9		
2	MARCHA	1.0		
4	FINAL DE CARRERA +	INT1	INT3	INT5
5	FINAL DE CARRERA -	INT2	INT4	INT6
7	FRECUENCIA	1.7	1.1	1.3
8	0V	1.9		
13				
14	24V PLC	24V		
15	MOTOR LISTO	LED1	LED3	LED5
16	FINAL DE CARRERA ACTIVO	LED2	LED4	LED6
20	SENTIDO	1.6	1.4	1.2
21	0V	1.9		

Tabla 4. 2. Conexiones autómata con servodriver x-y-z.

Se han colocado unas resistencias para limitar la tensión que envía el plc al servodriver, el autómata ya disponía de estas en las salidas 1.6 y 1.7 que se habían colocado anteriormente, estas resistencias las hemos colocado en las salidas desde la 1.1 a la 1.4, de esta manera tenemos cubiertas las tres señales de sentido y las tres señales del tren de pulsos.

Por otro lado los 0v voltios que necesita el servodriver y los que se necesitan para las lámparas se han cogido de la salida 1.9, 0V, del autómata. Los 24V se han extraído del autómata. En el plano numero uno podemos observar la conexión de los servoamplificadores con los motores, en el plano número dos el detalle de las conexiones realizadas entre el autómata programable y los tres servodrivers así como los interruptores y señalizaciones instaladas en la caja de control.

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

UNIVERSITAT POLITECNICA DE VALENCIA
ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA DEL DISEÑO

CONTROL Y VISUALIZACIÓN DE UN ALMACÉN AUTOMÁTICO

4.5 ANEXO 5: Creación de programa y comunicación para CoDeSyS

ÍNDICE DEL ANEXO 5

4.5.1 ANEXO 5: Introducción.

Para la programación el autómata PM571 de ABB utilizaremos el programa CoDeSys, este programa nos da la opción de programar mediante cualquier lenguaje de programación explicado en el apartado 1.6.2, lista de funciones, Ladder, diagrama de funciones, Grafcet, Continuos Function Chart y texto estructurado. Por otro lado nos permite la creación del Scada para la visualización y el control del proceso y nos proporciona las herramientas necesarias para probar y depurar cualquier sistema de automatización.

A continuación se va a explicar cómo crear un programa con el software de programación CoDeSys.

4.5.2 Creación y configuración de un nuevo proyecto.

Una vez abierto el software nos dirigimos a la derecha de la barra superior, seleccionamos "File" y en el desplegable elegimos "New", nos aparecerá el siguiente cuadro.

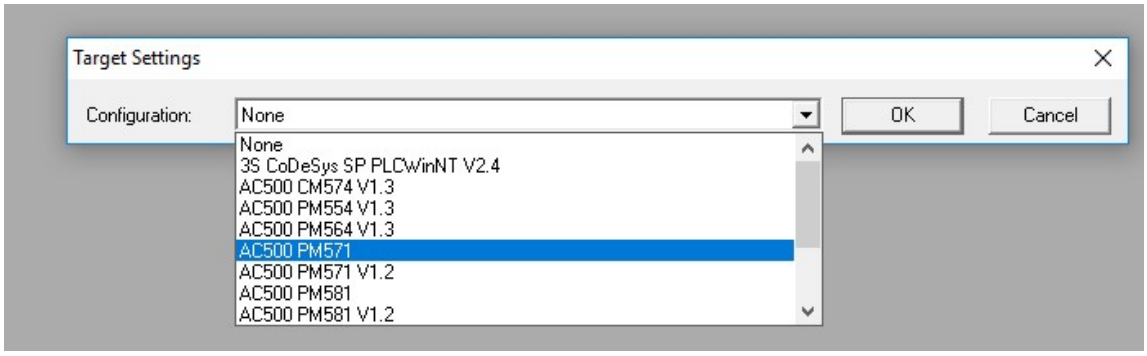


Figura 4. 2: Selección del autómata utilizado.

En el desplegable podemos ver todas las CPU's permitidas por CoDeSys, en nuestro caso seleccionamos "ACS500 PM571" y pulsamos "OK".

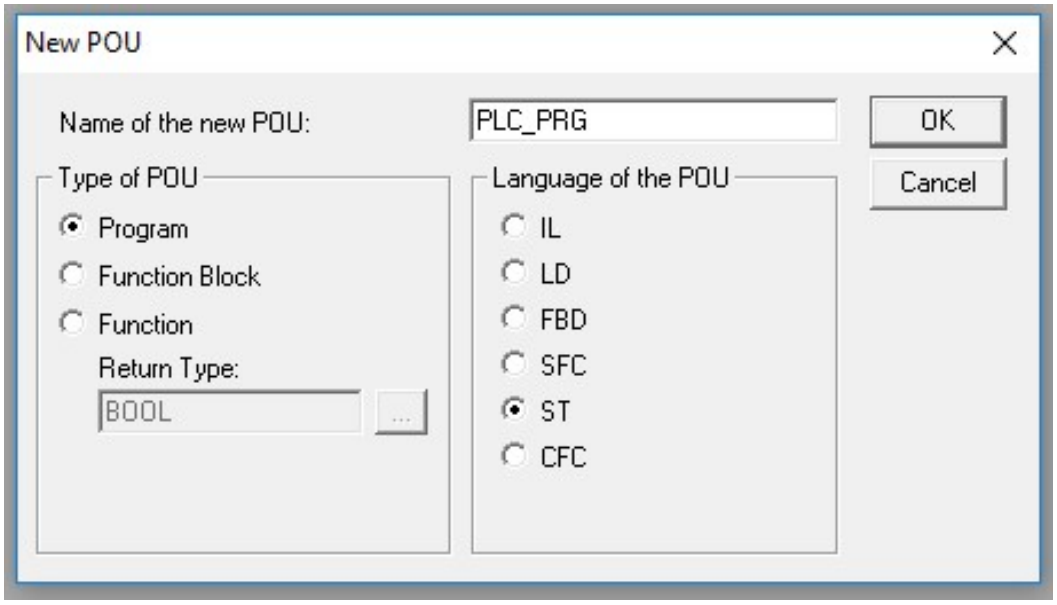


Figura 4. 3: Selección del lenguaje de programación.

Ahora tenemos que seleccionar el lenguaje de programación que vamos a usar, se ha decidido realizar la programación en texto estructurado con lo que se selecciona "ST" y pulsamos "OK".

A continuación definiremos el sistema de comunicación. En la parte inferior izquierda seleccionamos la pestaña "Resource". Seleccionamos "PLC Configuration" y elegimos "Ethernet" como se puede ver en la siguiente imagen.

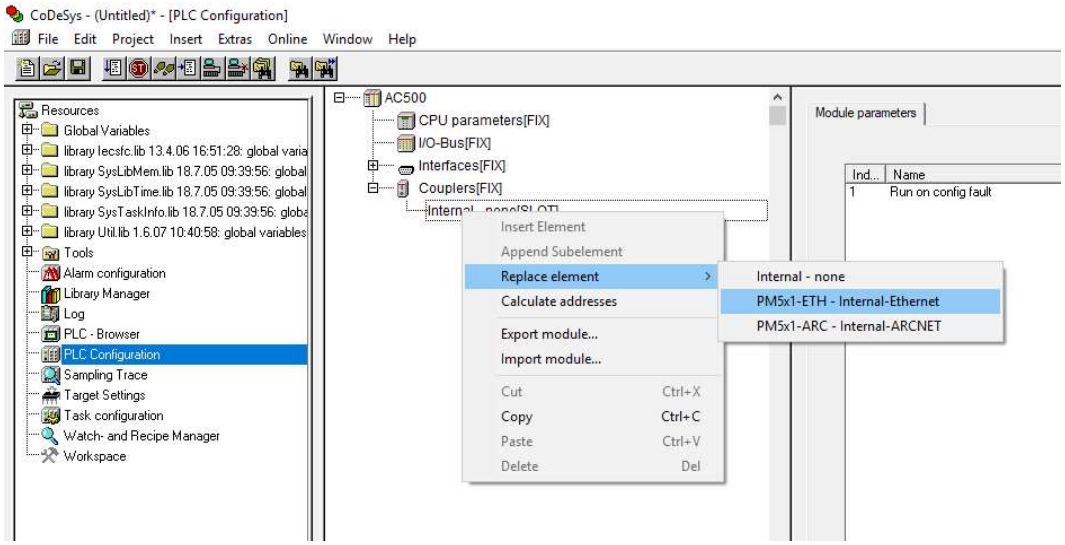


Figura 4. 4: Definición de las comunicaciones del autómata.

4.5.3 Definición del módulo de entradas/salidas DC541

Por ultimo vamos a definir el módulo de entradas/salidas DC541. Dentro del apartado "PLC configuration" pulsamos con el botón derecho encima de "couplers" seleccionamos "Append Subelement" y elegimos el módulo "DC541" como se puede ver en la siguiente imagen.

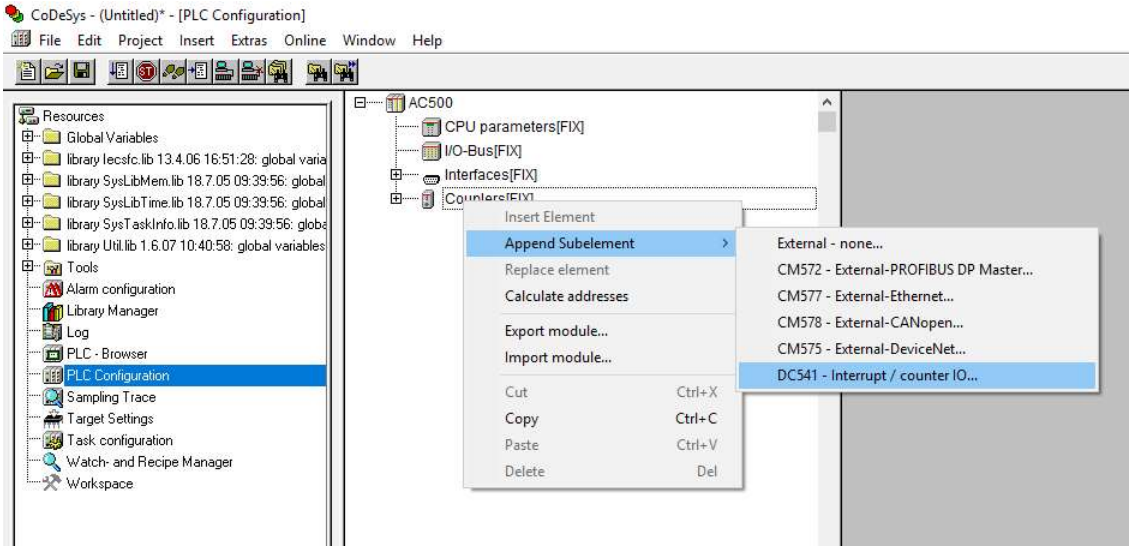


Figura 4. 5: Definición del módulo DC541.

Una vez se ha definido el módulo de entradas/salidas debemos definir la función de cada entra/salida. Por defecto aparecen todos en modo entrada "Input", en nuestro caso configuraremos 5 canales en modo salida "Output" y 3 los reservamos para poder realizar el control de posición de los motores mediante el generador de pulsos "Frequency output".

Index	Name	Value	Default	Min.	Max.
2	Channel 0	Output	Input		
3	Channel 1	Frequency output	Input		
4	Channel 2	Output	Input		
5	Channel 3	Frequency output	Input		
6	Channel 4	Output	Input		
7	Channel 5	Output	Input		
8	Channel 6	Output	Input		
9	Channel 7	Frequency output	Input		

Figura 4. 6: Definición de entradas y salidas del módulo DC541.

4.5.4 Asignación de IP

Finalmente vamos a seleccionar la IP del autómata y definirla en el programa para poder llevar a cabo la comunicación entre el PLC y CoDeSyS.

En primer lugar se define la IP en el autómata. Hay que tener en cuenta que dentro de la misma red no se pueden configurar 2 autómatas con la misma IP ya que esto nos generaría errores. En la parte inferior izquierda seleccionamos la pestaña "Resource". Seleccionamos "Tools" y a continuación "IP config". Nos aparecerá una pantalla como la que vemos a continuación

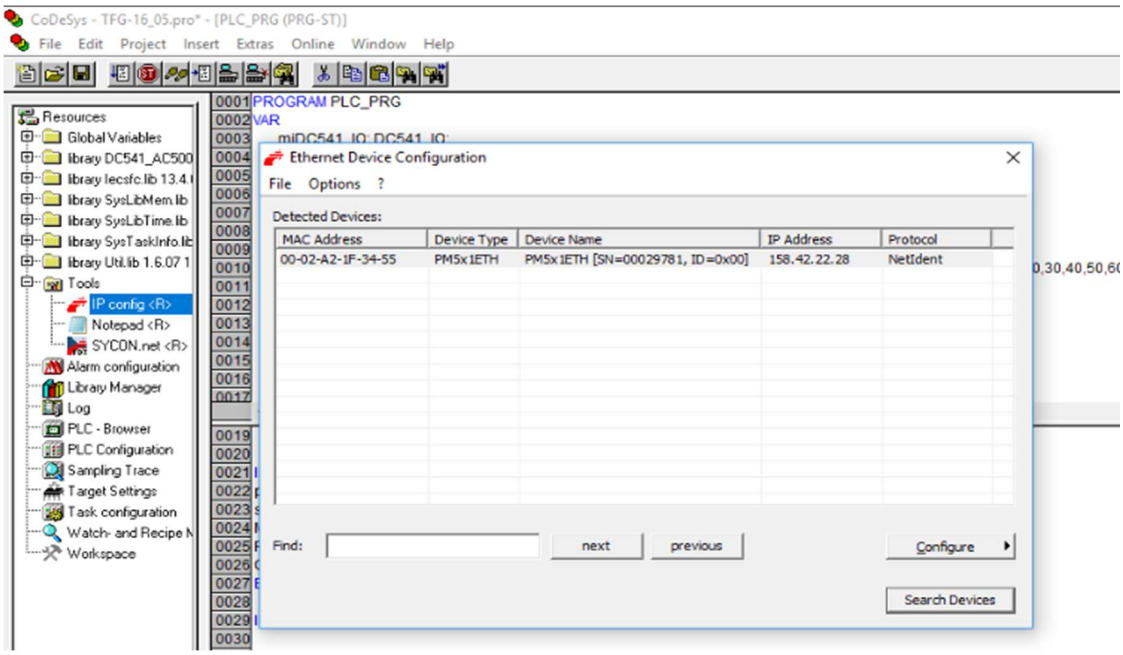


Figura 4. 7: Definición de la IP del autómata.

Pulsamos "Search devices" ahora aparecerán todos los autómatas conectados a nuestra red, en el caso de que aparezcan varios autómatas tendremos que determinar cuál es el nuestro mediante la dirección MAC. Seleccionamos nuestro autómata pulsamos "configure" y después "Set ip adress" y definimos la dirección IP.

Para acabar con la configuración nos dirigimos a la barra superior y pulsamos "online" y después "Communication parameters", nos aparecerá la siguiente ventana:

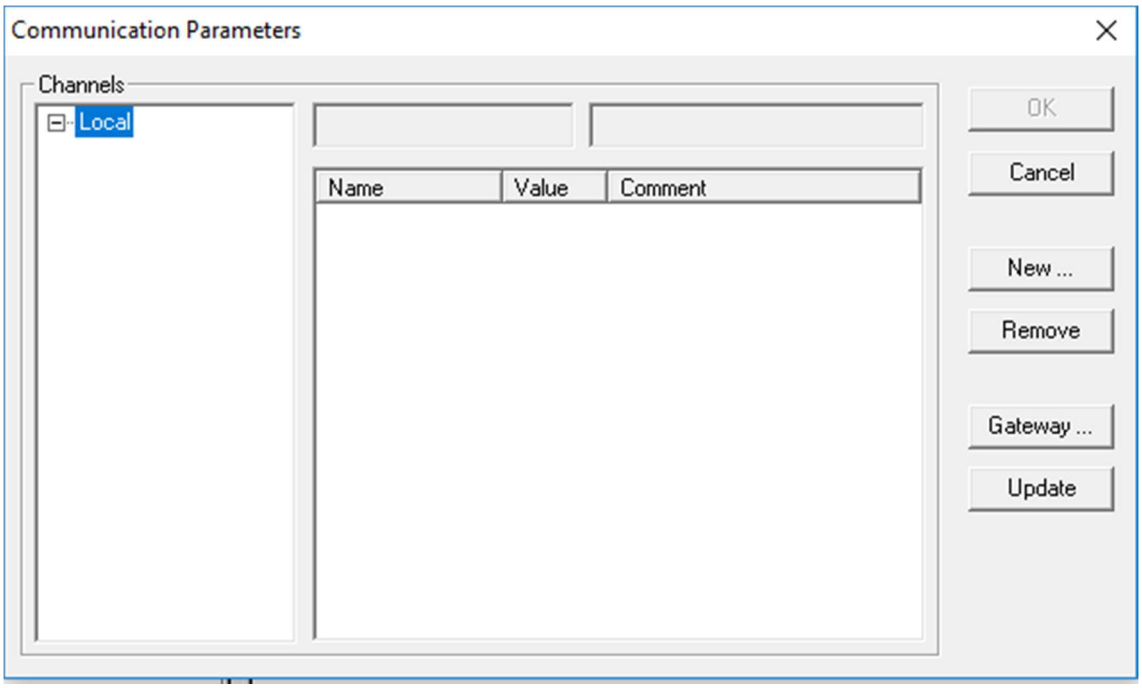


Figura 4. 8: Parámetros de comunicación del autómata.

A continuación pulsamos "New" y en la ventana que nos aparece seleccionamos "Tcp/Ip" le damos nombre a la conexión y pulsamos "Ok"

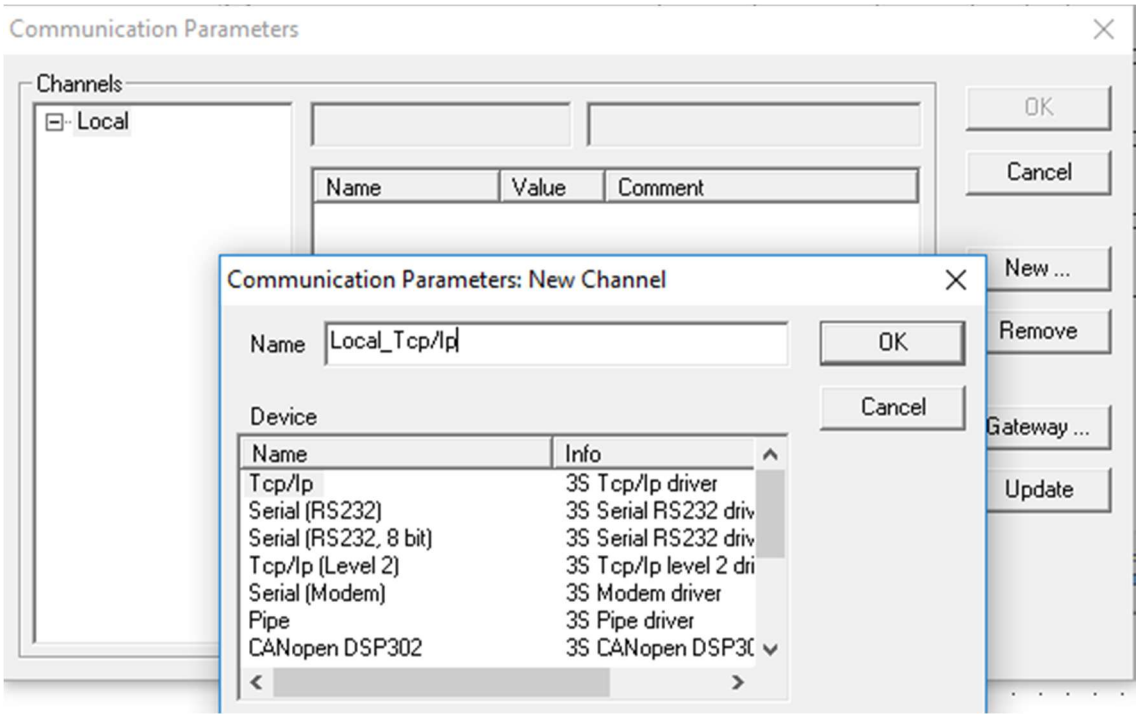


Figura 4. 9: Parámetros de comunicación del autómata Crear nuevo canal.

Finalmente en "Adress" ponemos la misma Ip que hemos puesto en el PLC, en "Port" ponemos el 1201 y en "Motorola byteorder" pulsamos dos veces para poner "YES". Quedando de la siguiente forma.

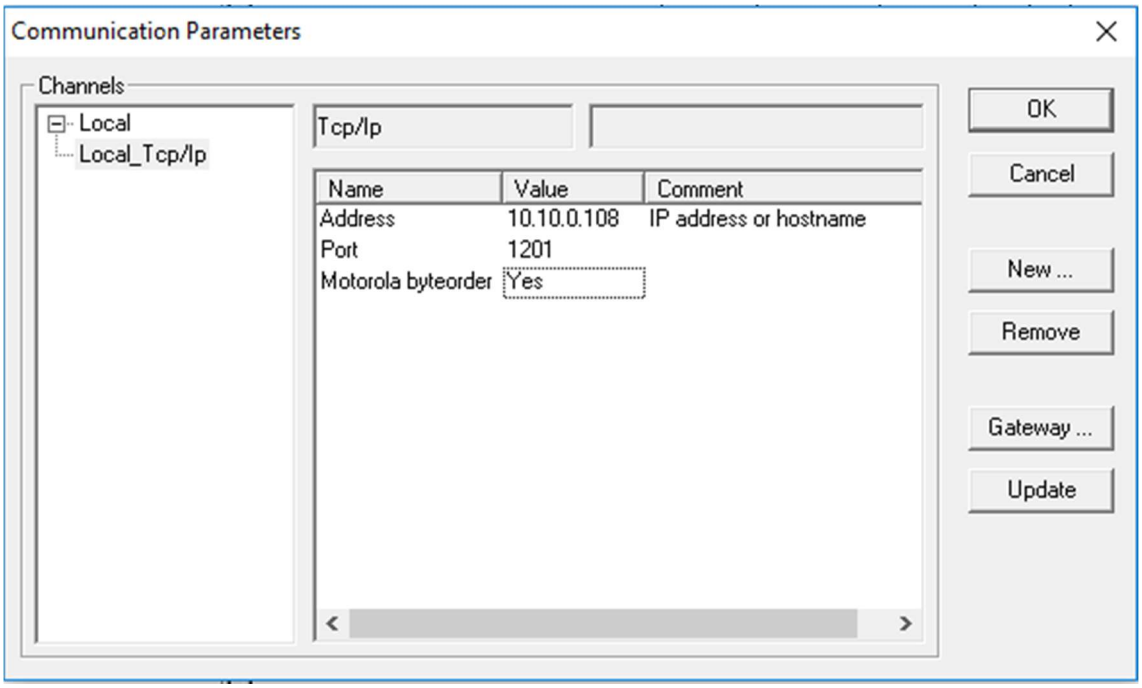


Figura 4. 10: Parámetros de comunicación del autómata. Definir IP.

Ya solo queda pulsar "Update". De esta manera el programa está listo para comunicar con el plc.

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

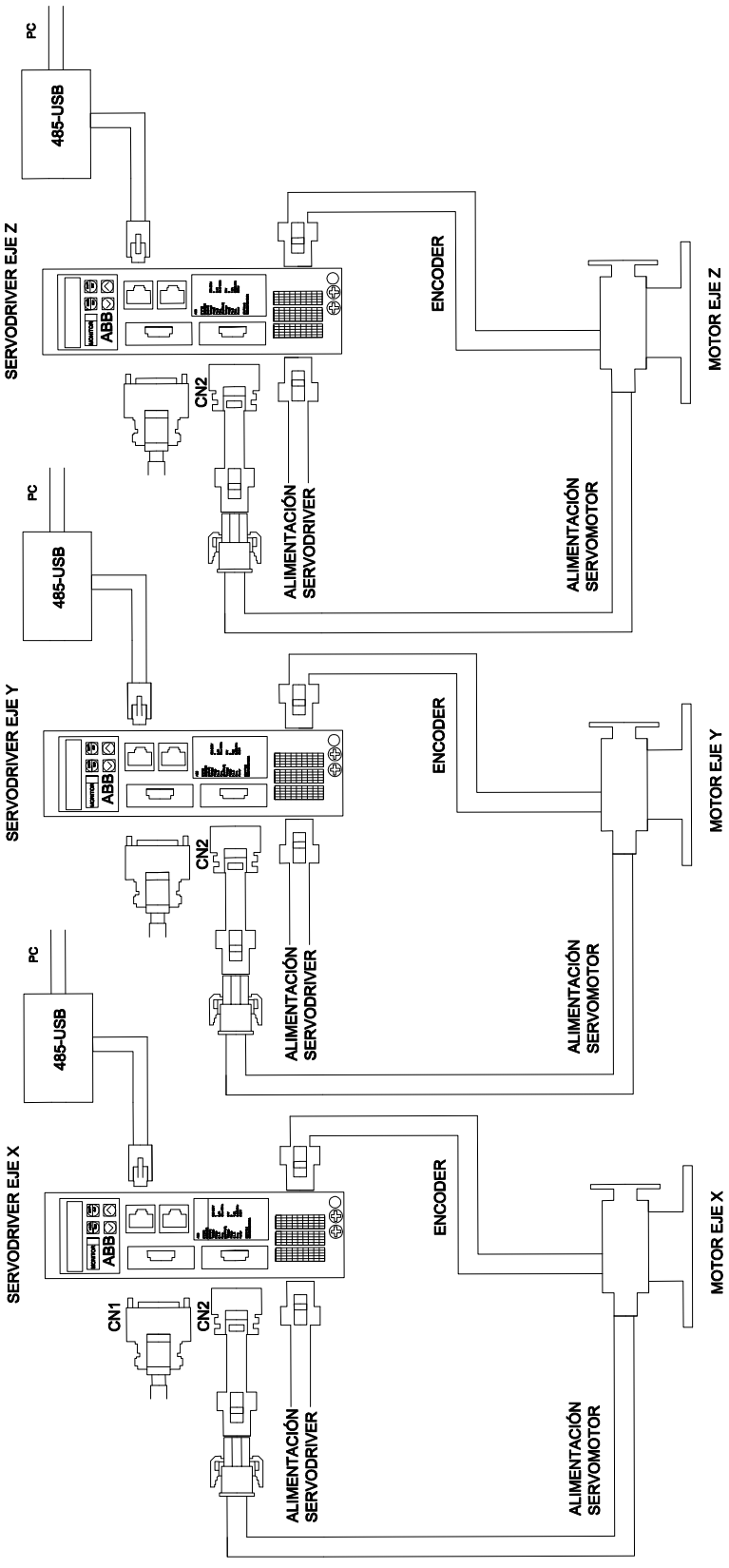
Escuela Técnica Superior de Ingeniería del Diseño

UNIVERSITAT POLITECNICA DE VALENCIA
ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA DEL DISEÑO

CONTROL Y VISUALIZACIÓN DE UN ALMACÉN AUTOMÁTICO

4.6 ANEXO 6: Esquemas eléctricos.

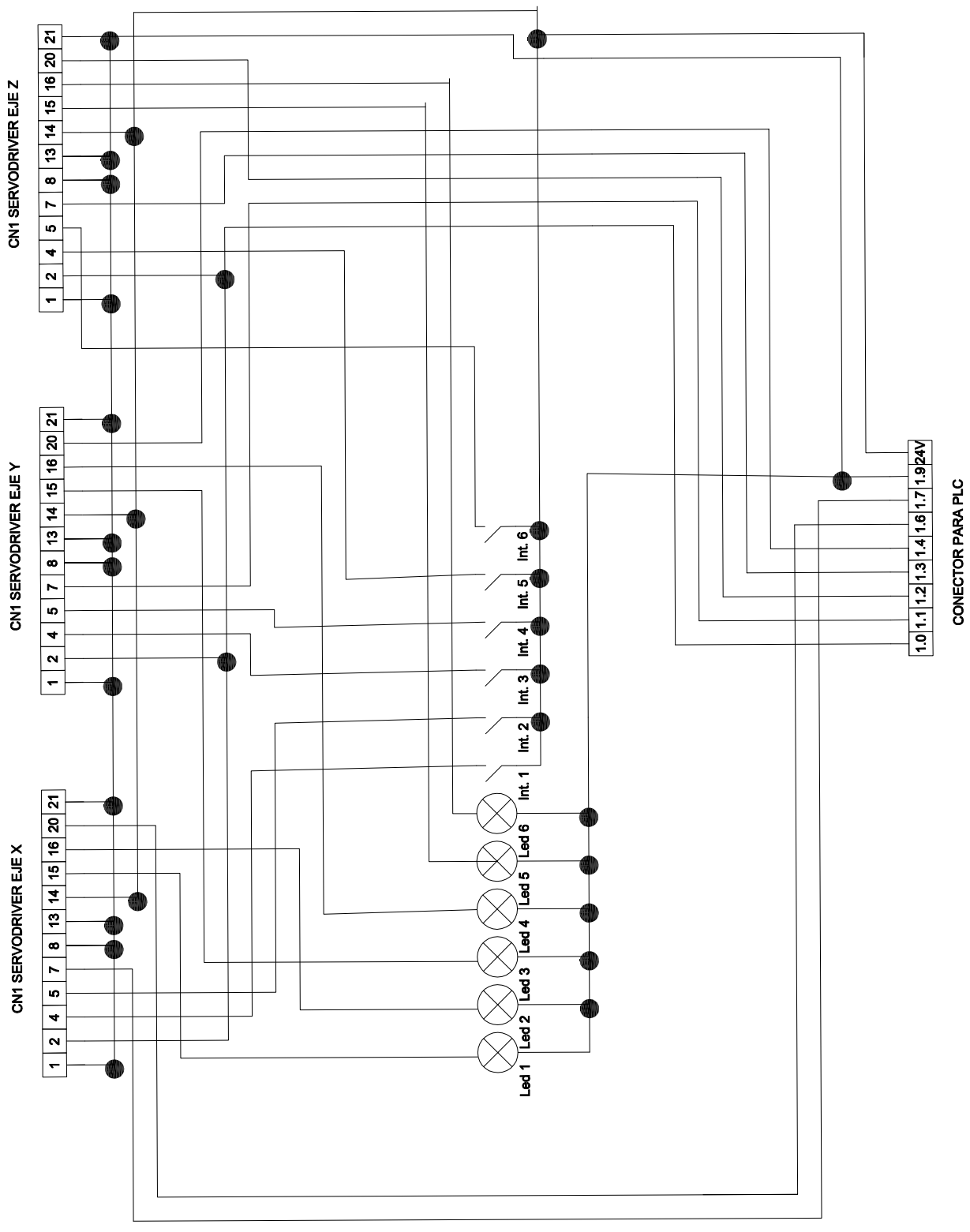
ÍNDICE DEL ANEXO 6



Universitat Politècnica de València

Escuela técnica superior de ingeniería del diseño

Proyecto: **Control y visualización de un almacén automático**		Escala: **-/-**
Nombre: **Pedro Parrilla Amaro**		
Fecha: Marzo 2019	Titulo de plano: **CONEXIÓN SERVOAMPLIFICADORES**	Nº Plano: **1**



Universitat Politècnica de València

Escuela técnica superior de ingeniería del diseño

Proyecto: **Control y visualización de un almacén automático**		Escala: **-/-**
Nombre: **Pedro Parrilla Amaro**		
Fecha: Marzo 2019	Titulo de plano: **DETALLE CONEXIONES CONECTORES CN1 A AUTOMATA PROGRAMABLE**	Nº Plano: **2**