UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

CONTROL Y VISUALIZACIÓN DE UN ALMACÉN ELEVADOR AUTOMATIZADO

Grado en Ingeniería Eléctrica.

Curso 19/20.

Director
Rubén Puche Panadero

Codirector
Ángel Sapena Bano

Adrián Aparici Micó
adapmi@etsid.upv.es

Índice

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

CONTROL Y VISUALIZACIÓN DE UN ALMACÉN ELEVADOR AUTOMATIZADO

Memoria

Director
Rubén Puche Panadero

Codirector
Ángel Sapena Bano

Adrián Aparici Micó
adapmi@etsid.upv.es

Índice

1.Introducción

Desde hace años se ha estado aplicando la tecnología de los PLC´s en el ámbito de la ingeniería industrial. Esto debido a su robustez, versatilidad y fácil adaptabilidad e implementación.

La automatización de procesos mediante autómatas programables ha hecho que estos puedan realizar de manera automática y eficiente trabajos muy monótonos, repetitivos aburridos y/o peligrosos, aumentando en gran medida la eficiencia sobre todo en el ámbito industrial. Esto ha llevado a que el mercado de los PLC´s sea un mercado en alza ya que cada vez son más las industrias que eligen automatizar sus procesos con el objetivo de aumentar su eficiencia y competitividad.

La automatización de procesos no ha sido ajena al avance tecnológico. Los autómatas programables han ido evolucionando, ofreciendo mayores posibilidades de control. De esta manera mientras que la industria ha ido evolucionando y requiriendo controles más precisos y complejos, los autómatas programables lo han hecho con ella llevando a cabo dichas demandas (temperatura, posición, PID´s…). Esto ha hecho que sea indispensable hoy en día la presencia de PLC´s en una industria si esta quiere ser competitiva y adaptarse a los nuevos tiempos.

No sólo se emplean autómatas para el ámbito industrial, sino que son ampliamente utilizados en todo tipo de campos (medicina, transporte, domótica, etc.) Por ello su estudio es tan importante en el ámbito de la ingeniería.

2. Objetivo

El objetivo del trabajo de fin de grado es realizar un programa de control mediante un PLC de una maqueta de un almacén vertical inteligente disponible en el control de máquinas eléctricas del Departamento de Ingeniería Eléctrica.

Dicho control es visualizado y monitorizado desde un sistema SCADA comunicado desde un ordenador vía cable USB.

El programa realizado no debe centrarse únicamente para el control de la maqueta, sino que ha de permitir el uso del mismo programa para otros controles de tres ejes, garantizando así una mayor versatilidad de usos de este.

Debido a la existencia de un modo general, el SCADA ha de permitir una configuración de los motores (desde su velocidad nominal hasta la existencia de los finales de carrera) y la existencia de un modo manual.

3. Justificación

Se puede justificar la realización de este proyecto desde dos puntos de vista distintos. Estos pueden ser:

3.1. Académico

El desarrollo de este trabajo de fin de grado está directamente relacionado con la obtención del título de Graduado en Ingeniería Eléctrica en base a la normativa vigente impuesta por el ministerio.

3.2. Funcional

Debido a lo comentado en la introducción, los autómatas programables están muy presentes en la mayoría de los ámbitos, por ello, es muy importante para un graduado aumentar los conocimientos y profundizar en dicho campo.

Mediante el total control de un modelo no simulado se profundiza también en conocimientos sobre lenguajes de programación, comunicación de autómatas y detección y solución de problemas propios de la ingeniería (conexiones, fallo de los encoders, limitaciones de harware, etc.).

Por último, existe el motivo de aprender a como planificar un proyecto desde el desarrollo de la idea hasta la aplicación de mejoras y resolución de errores. Todo ello a partir de un proceso lo más ordenado posible haciendo que tanto el SCADA como la programación sean lo más completos y sencillos posible.

4. Necesidades del proyecto

Como ya se ha destacado en anteriores apartados, el programa no solo ha de poder controlar un sistema de almacenaje, sino que también ha de servir para otros procesos como motor de 3 ejes. Todo esto acorde a la normativa vigente.

El software ha de controlarse enteramente desde un sistema SCADA donde se tienen que haber 2 pantallas claramente diferenciadas. Estas son las del modo general, y específico. Desde este SCADA, aparte de monitorizar los procesos, se ha de conocer la posición de los ejes en la medida que los elementos de posicionamiento que tengan los motores lo permitan. Como último, el software ha de incluir una opción funcional de simulación para ambos modos

4.1. Control específico

Es el control para el que ha sido diseñado en un principio el programa. Ya que el proyecto ha sido hecho en base al control de la maqueta de almacenamiento vertical inteligente. Este modo tiene que constar de 4 funciones. La función “Depositar” permite colocar un objeto en el estante deseado, la función “Recoger” retira un el objeto seleccionado de su estante, la función “Mover” se dedica a trasladar un objeto de un estante a otro que esté vacío y la función “Origen” mueve la bandeja controlada mediante los 3 ejes al punto de origen desde donde se depositan los objetos y donde los ubica el modo “Recoger”.

Debido a que es un sistema de almacenamiento se ha de conocer en todo momento el estado de cada estante (ocupado o libre) para saber qué objeto mover y donde poder dejarlo. Estos estados de los estantes también se han de poder alterar de forma manual sin la necesidad de pasar por ningún proceso. Esto se debe a que en un almacén se puede dar el caso de que un objeto se ha retirado o colocado de manera manual ajeno al control automático por muchos factores (corte en el suministro eléctrico, avería, mantenimiento, etc.).

4.2. Control general

El control general representa el esqueleto desde el cual el específico añade sus configuraciones y procesos. Este control general tiene que constar de dos partes claramente diferenciadas.

La primera y más básica es el control manual. Este consta de una botonera donde se han de poder activar los tres ejes del motor hacia la dirección que se quiera.

La segunda parte del control general es el modo automático. En este modo se ha de abrir una tabla donde se introducen las posiciones de los tres ejes eligiendo también el tamaño de dicha tabla. Tras esto se le da a la macha y el sistema funciona igual que el modo específico, pero realizando un proceso diferente. Este sistema ha de permitir guardar y cargar hasta tres procesos definidos previamente por el usuario por si se requiere.

Otra función muy importante del modo general ha de ser la configuración de los tres ejes. Si esta configuración no garantiza un control apropiado el proceso no se puede poner en marcha. En esta configuración se ha de introducir la velocidad nominal y el modo de control de posición elegido junto a los datos que complementan a este. Dichos modos son por encoders, por tiempo o por finales de carrera finales ya que los iniciales han de estar siempre para garantizar la posición 0,0,0. Los finales de carrera finales son compatibles con los encoders y el control por tiempo, pero el control por tiempo y los encoders no lo son entre ellos.

5. Descripción del proceso a controlar

La maqueta se basa en una máquina cartesiana, es decir actúa en los tres ejes cartesianos (X, Y, Z). Está formada por tes elementos móviles que se desplazan cada uno en un eje mediante unos motores y unos tornillos sin fin. En el control específico los ejes X e Y trabajan mediante un sistema de coordenadas y un control de posición mientras que el eje Z trabaja mediante finales de carrera. Como ya se ha dicho, en el control general todo esto es configurable y no tiene por qué obedecer a la descripción anterior.

La maqueta, como ya se ha dicho es una maqueta de un almacén elevado automatizado de 24V de la empresa Fischer Technik, una empresa alemana dedicada a la fabricación de maquetas y elementos de robótica con fines educativos.

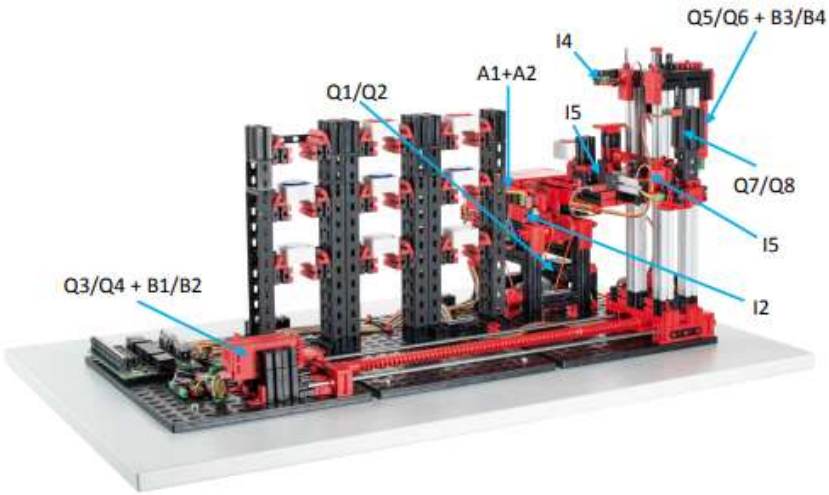


Figura 1. Almacén elevado automatizado vista general

Este almacén posee la siguiente asignación de los bornes de alimentación, entrada y salida son los siguientes:

N.º de borne	Función	Entrada/salida
1	Alimentación de corriente (+) actuadores	24 V CC
2	Alimentación de corriente (+) sensores	24 V CC
3	Alimentación de corriente (-)	0 V
4	Alimentación de corriente (-)	0 V
5	Pulsador de referencia horizontal	I1
6	Barrera de luz interior	I2
7	Barrera de luz exterior	I3
8	Pulsador de referencia vertical	I4
9	Sensor de pistas (señal 1, abajo)	A1
10	Sensor de pistas (señal 2, arriba)	A2
11	Codificador horizontal impulso 1	B1
12	Codificador horizontal impulso 2	B2
13	Codificador vertical impulso 1	B3
14	Codificador vertical impulso 2	B4
15	Pulsador de referencia brazo giratorio delante	I5
16	Pulsador de referencia brazo giratorio atrás	I6
17	Motor cinta transportadora hacia delante	Q1 (M1)
18	Motor cinta transportadora hacia atrás	Q2 (M1)
19	Motor horizontal hacia el estante	Q3 (M2)
20	Motor horizontal hacia la cinta transportadora	Q4 (M2)
21	Motor vertical hacia abajo	Q5 (M3)
22	Motor vertical hacia arriba	Q6 (M3)
23	Motor brazo giratorio hacia delante	Q7 (M4)
24	Motor brazo giratorio hacia atrás	Q8 (M4)

Figura 2. Bornes

Estos bornes se envían y reciben la información de los actuadores y sensores al autómata y viceversa mediante cables conectados a unos relés.

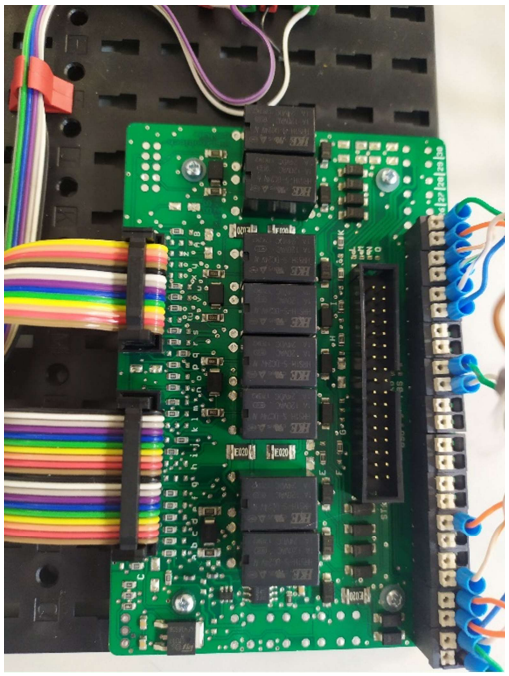


Figura 3. Relés y conexión de los bornes

A la derecha la última figura se pueden apreciar los bornes ya conectados al autómata. Junto a estos se puede ver una ranura donde realizar una conexión. Esta conexión es en esencia la misma que la ya realizada con los bornes. Existe para ofrecer la posibilidad de conectar todos los bornes mediante una sola conexión. Esta conexión responde al siguiente esquema:

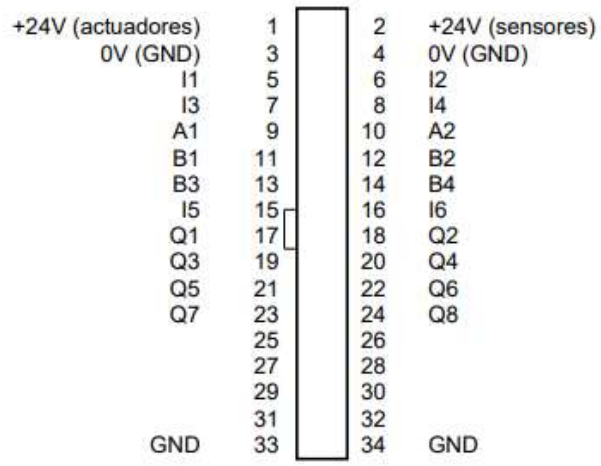


Figura 4. Esquema de bornes

5.1. Sensores y Actuadores

En esta maqueta podemos encontrar los siguientes sensores y actuadores:

5.1.1. Motores

Estos constituyen los únicos actuadores mediante los cuales se mueve todo el sistema. Existen dos tipos de motores presentes en el sistema.

El primero de ellos es el motor codificador. Dicho se encuentra tanto en el eje X como en el Y. Se trata de máquinas de corriente continua e imanes permanentes que, con ayuda de sensores Hall, posibilitan una medición angular incremental. Los motores de codificador presentan una tensión nominal de 24 V y una potencia máxima de 2,03 W con una velocidad de 214 r.p.m. El consumo de corriente a la máxima potencia es de 320 mA. El engranaje integrado tiene una transmisión de 25:1, es decir que el codificador genera tres impulsos por

revolución del árbol motor o 75 impulsos por revolución del árbol de salida del engranaje. Como se registran dos impulsos desfasados, el codificador utilizado puede distinguir en qué sentido gira el motor.

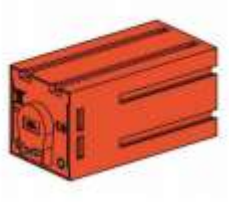


Figura 5. Motor codificador

La conexión se realiza mediante un cable de cuatro conductores: el conductor rojo debe conectarse a una salida de 24 V y el conductor verde, a masa. Los cables negro y amarillo transmiten los impulsos (salida pushpull, máx. 1 kHz, máx. 10 mA).

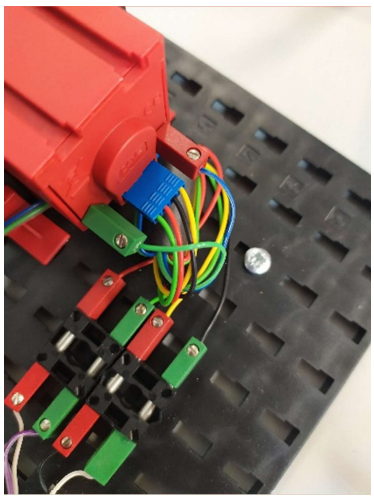


Figura 6. Conexión del motor codificador

El otro tipo de motor que se encuentra en el proceso es el motor S de 24V, el cual se emplea para el avance y retroceso del eje Z y la cinta transportadora de la bandeja de entrada. El brazo giratorio del dispositivo de control del almacén elevado se acciona con un motor S. Este motor compacto es una máquina de corriente continua e imanes permanentes, que se puede utilizar junto con un engranaje reductor insertable. El motor funciona con una tensión nominal de 24

V CC, y el consumo de corriente es de 300 mA como máximo. De ello resultan un par de giro máximo de 5 mNm y una velocidad en vacío de 10700 r.p.m. El engranaje reductor dispone de una transmisión de 64,8:1 y una salida lateral.

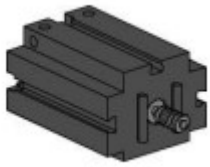


Figura 7. Motor S de 24V

Su conexión consta únicamente de un conductor rojo conectado a la salida de 24V y un conector verde a masa.

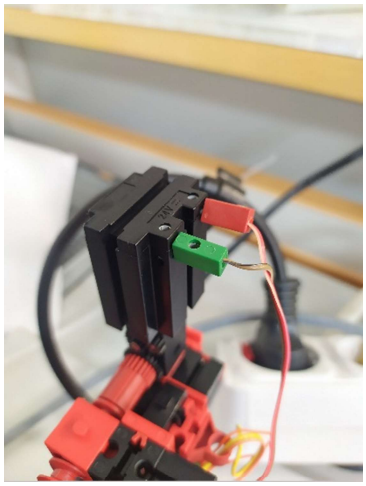


Figura 8. Conexión motor compacto

5.1.2. Encoder

Los encoders son aparatos empleados para medición de posición y velocidad de giro de un motor. Como ya se ha mencionado en el apartado de los motores, los encoders se encuentran en los motores codificadores en forma de sensores Hall. Estos sensores son de tipo incremental posibilitando el conocimiento tanto de velocidad como de posición enviando 75 pulsos por cada vuelta del motor antes de la relación de transmisión.

A diferencia del encoder absoluto, un encoder incremental envía dos pulsos diferentes mediante los cuales se puede tomar una referencia mediante la

conocer la posición además de la velocidad. Para resumir, conociendo el número de pulsos por vuelta y que, si envía el pulso A y luego el B gira hacia un lado y que, si hace el B y luego el A gira hacia el otro, se puede conocer tanto la velocidad como la posición.

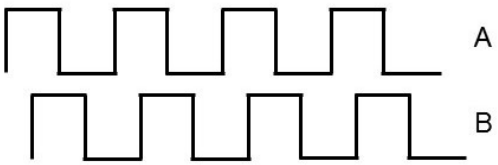


Figura 9. Pulsos encoder incremental

5.1.3. Fototransistor

Los fototransistores se utilizan en el almacén elevado automatizado como barreras de luz en la bandeja. A la vez se aprovecha que, a partir de una cierta luminosidad, el fototransistor conduce corriente. No obstante, si este umbral de luminosidad no se alcanza, el fototransistor pierde su conductividad. Junto con una lámpara de lente, que se contrapone al fototransistor, este último conduce normalmente corriente y, con ello, se puede utilizar como barrera de luz. Para reducir la influencia de la luz ambiente, se puede usar una cubierta contra luz parásita.

Figura 10. Fototransistor

A la hora de conectar el fototransistor a la alimentación de corriente, se debe observar la polaridad correcta. El polo positivo debe conectarse a la marca roja en el fototransistor.

5.1.4. Minipulsador

En el manipulador de aspiración al vacío se emplean minipulsadores como interruptores de referencia (finales de carrera). Al aplicar métodos de medición incremental, un interruptor de referencia sirve para determinar la posición absoluta o el ángulo absoluto. Se pueden encontrar dos minipulsadores como final de carrera inicial y final en el eje Z, uno en el eje X y otro en el Y. Estos dos últimos se emplean como sendos finales de

carrera iniciales. El minipulsador allí utilizado se puede usar tanto como contacto normalmente cerrado como normalmente abierto.

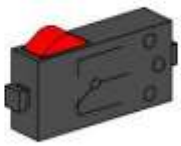


Figura 11. Minipulsador

Su conexionado se realiza conectando el conductor rojo a la fuente de alimentación de 24V y el verde al retorno. Cuando se acciona el pulsador, existe una conexión conductora entre el contacto 1 y el contacto 3, mientras que la conexión entre el contacto 1 y el contacto 2 se interrumpe. En la figura 12 se muestra el esquema de conexiones del minipulsador.

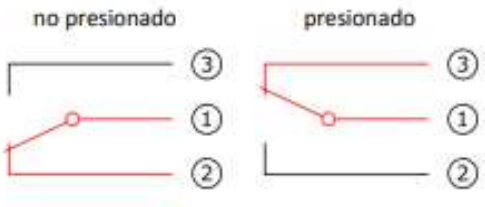


Figura 12. Esquema de conexiones del minipulsador

5.1.5. Sensor de pistas IR

El sensor de pistas IR es un sensor digital infrarrojo para detectar una pista negra sobre un fondo blanco a una distancia de 5-30 mm ubicado en la bandeja de entrada. Está constituido de dos elementos de transmisión y dos de recepción. Las señales se efectúan como salidas push-pull. La conexión se realiza con cuatro cables.

Figura 12. Sensor de pistas IR

El cable rojo debe conectarse a la salida 9 V CC y el cable verde, a masa. Los cables negro y amarillo transmiten las señales. La placa de conexión asume la transformación de la tensión y la adaptación de nivel de 24 V CC a 9 V CC.

5.2. Secciones del almacén

Como ya se ha estado dejando caer, es visiblemente divisible en 3 secciones claramente diferenciadas.

La primera es la bandeja de entrada. Esta consta de un fototransistor que detecta si existe un objeto en la bandeja, un motor que hace mover una cinta transportadora y un sensor de pistas que detecta si existe algún objeto ubicado al final de la cinta. Esta parte constituye la entrada y salida de objetos (en este caso cajas) al almacén.

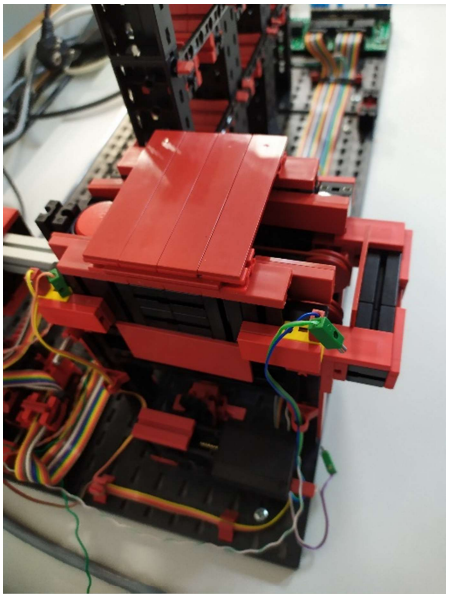


Figura 13. Bandeja de entrada

La segunda sección y más importante (donde se centran principalmente tanto procesos como programación) son los ejes. Como ya se ha mencionado estos son los ejes X, Y, Z.

Se le llama eje X al eje horizontal del sistema. Consta de un motor codificador que hace mover un objeto a lo largo de 292mm en el eje horizontal mediante un tornillo sin fin. También posee un final de carrera para detectar la posición 0 del eje.

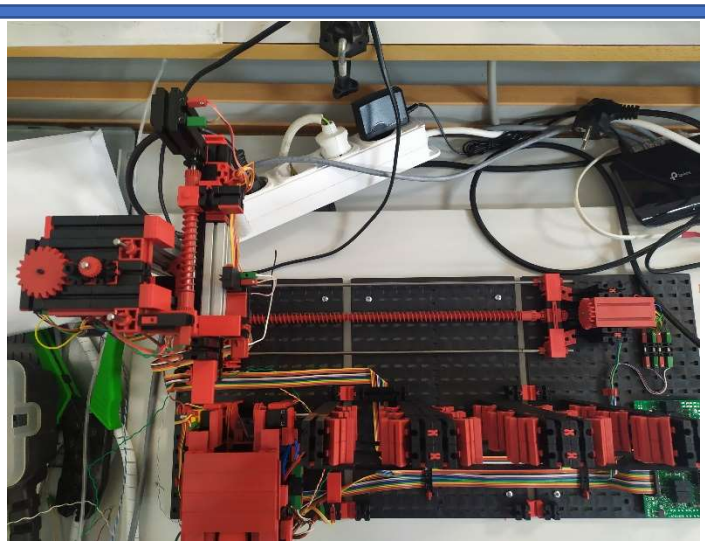


Figura 14. Eje X en la posición 0

Se le atribuye el nombre de eje Y al eje que regula la posición vertical del objeto. Este, al igual que el X, posee un motor que desplaza el objeto en el eje Y mediante un tornillo sin fin y un final de carrera para detectar la posición 0. Este eje hace desplazar al objeto a lo largo de 135mm.

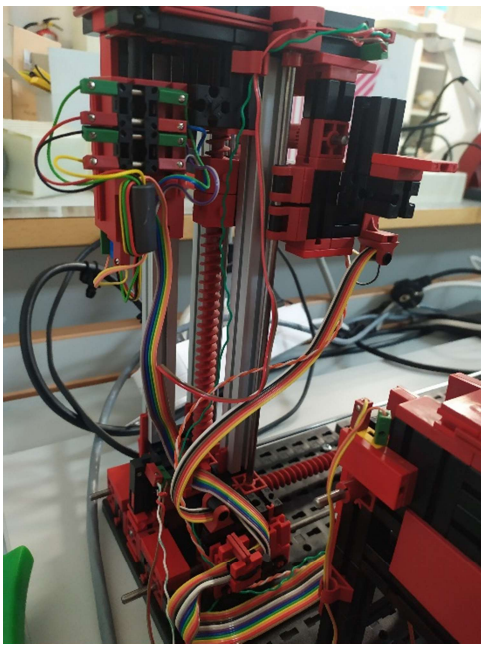


Figura 15. Eje Y en la posición 0

Por último, el eje Z es el que desplaza el brazo donde se ubica el objeto hacia adelante y hacia atrás. Este consta de un motor compacto y dos finales de

carrera que determinan las posiciones absolutas de hacia adelante (1) y hacia atrás (0).

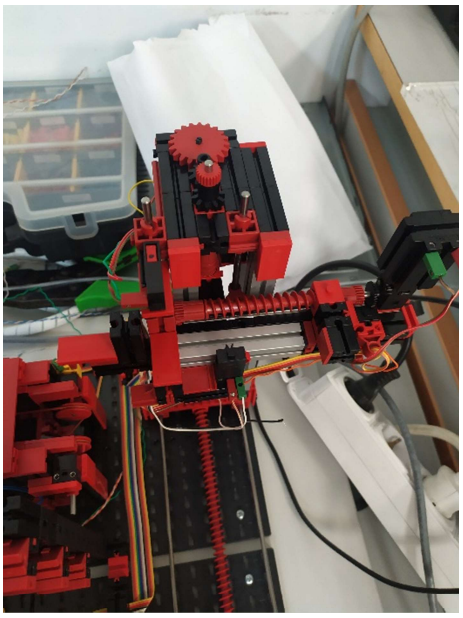


Figura 16. Eje Z en la posición 0

La última sección del almacén es la de los estantes. Esta sección consta de nueve ranuras dispuestas en un cuadrado de 3x3 donde poder depositar y recoger las cajas mediante el sistema de tres ejes. Los estantes no cuentan con ningún tipo de actuador ni sensor.

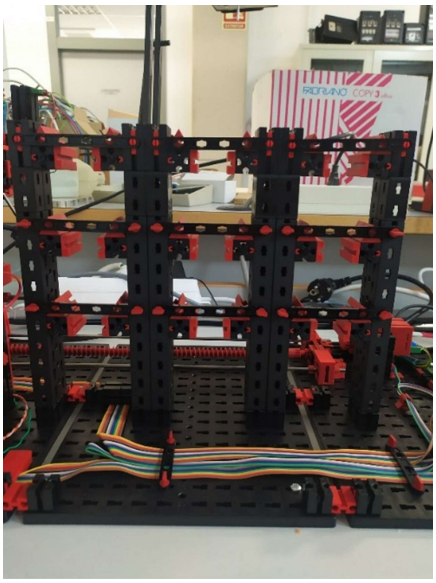


Figura 17. Estantes

	Columna 1	Columna 2	Columna 3
Fila 1	115, 3	200, 3	292, 3
Fila 2	115, 50	200, 50	292, 50
Fila 3	115, 124	200, 124	292, 124
Bandeja de entrada	0, 92		

Figura 18. Ubicación de cada estante con respecto al 0,0 en mm

6. Selección del equipo

La selección del equipo necesario para que el proyecto pueda desarrollarse tiene que cumplir las necesidades de este expresadas anteriormente. Además, tiene que satisfacer unas necesidades propias del contexto en el que se está realizando el trabajo. Las primeras necesidades propias para cumplir son la accesibilidad y la familiarización. El equipo ha de estar accesible para su uso, instalación y/o programación, además el alumno debe estar familiarizado con este para poder realizar así las labores requeridas.

6.3. Selección de PLC

El autómata programable es el elemento principal de todo el sistema de control. Es el encargado de recibir las señales y procesarlas para mostrar la información requerida e influir en los actuadores de manera correspondiente. Por tanto, es donde se realiza toda la programación requerida para el control del proceso.

Para la selección del autómata programable se ha tomado en cuenta, la accesibilidad del software como la disponibilidad a él en el laboratorio y las limitaciones de este para llevar a cabo en mayor o menor medida el proceso.

Debido a toda la demanda exigente tanto dentro como fuera de un contexto industrial existen multitud de autómatas programables de diversos fabricantes. Cada uno consta de características propias que van desde las entradas y salidas que poseen hasta el software que emplea. El estudio de selección se ha realizado a partir de los autómatas existentes disponibles en el laboratorio:

-SISMATIC S7-200 de Siemens

Se trata de un PLC modular por tanto permite adaptar el tipo y numero de entradas y salidas en base a las necesidades del proyecto. Por tanto, es capaz de poseer hasta 16 entradas y 16 salidas digitales pudiendo así satisfacer con creces las entradas y salidas que demanda el modelo a controlar. Como características principales se pueden encontrar:

·Puerto de comunicación RS4485.

·Comunicación PROFIBUS, AS-Interface y vía internet.

·Adaptación para comunicación con servidor OPC.

·Software STEP 7-Micro/WIN con librería add-on Micro/WIN.

·Módulo de posicionamiento para control de motores y servomotores.

Figura 19. SIMATIC S7-200

-PM550 gamma AC500 de ABB

Se trata de un PLC pequeño y compacto que posee 8 entradas digitales y 6 salidas también digitales. Al ser compacto no se puede ampliar con más módulos, pero posee suficientes entradas y salidas para poder realizar el proceso (explicado más adelante). Sus características son las siguientes:

·Puerto para cable programable USB

·Programación en FB y ST

·Librerías de FB

Este es el autómata que se ha elegido para realizar el control pese a sus limitaciones y el parecer una opción menos acertada. La principal razón para elegirlo es el software que emplea ya que este es el Codesys, un programa con el cual el alumno está familiarizado (empleado en la asignatura PROMAQ) y que se puede tener en el ordenador. Ambos programas requieren de una licencia para su uso pero la diferencia es que la del Codesys se le puede proporcionar al alumno mientras que la del TIA-Portal (software que emplea Siemens) no. Esto último es muy importante a la hora de trabajar en el proyecto desde casa y más en el contexto actual de pandemia. En el departamento existen ordenadores con el software ya instalado y listo para usar, pero en un contexto de pandemia es inviable depender del acceso al laboratorio para la realización de la totalidad del proyecto.

Este autómata se alimenta a partir de una fuente de continua de 24V.

Figura 19. PM554

El PLC se comunica con el ordenador donde se encuentra el programa mediante un cable programable TK503 también de ABB.

Figura 20. Cable TK503

Existen dos mangueras de cables que van desde los bornes al autómata. La primera corresponde a las entradas y la segunda a las salidas e independientemente existen dos cables que van del positivo al negativo de la alimentación. Las conexiones de entradas, salidas y alimentación del autómata se realizan de la siguiente manera:

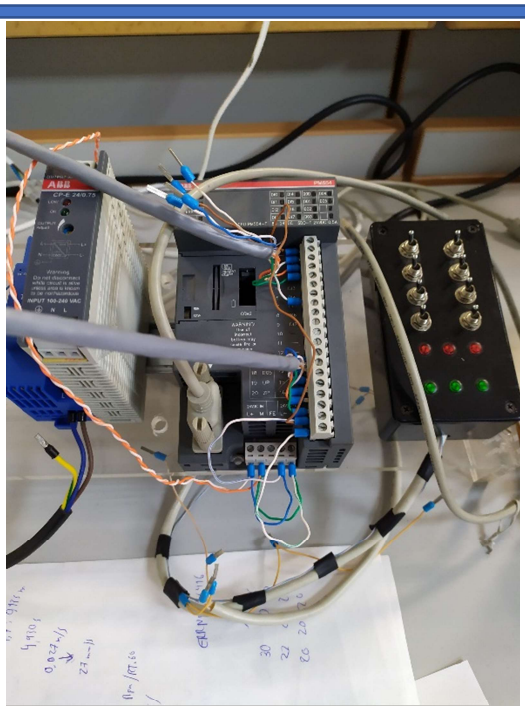


Figura 21. Conexión del autómata

6.2. Fuente de alimentación

Tanto el autómata como el almacén necesitan ser alimentados por una fuente de 24V en corriente continua. Esta fuente ha de proveer de suficiente corriente para alimentar al autómata y a todos los sensores y actuadores de la maqueta. Para ello ha empleado una fuente rectificadora conectada a la red que transforma los 230V de corriente alterna de esta a los 24V de corriente continua que se necesitan. La fuente elegida es la fuente DRL30-24-1 del fabricante TDK-Lambda.

Dicha fuente es regulable entre 24V y 1.25A y 28V y 1.07A en corriente continua a partir de una conexión de entre 100 y 240V en corriente alterna. Esto se regula mediante una ranura de destornillador presente en la parte superior izquierda. La conexión se efectúa enviando un cable positivo y otro negativo (blanco/azul y azul respectivamente) a la maqueta y otro positivo y negativo (blanco/naranja y naranja respectivamente) al autómata.

Figura 22. Fuente de alimentación y conexiones

7. Limitaciones, problemas y soluciones propuestas

A lo largo de la realización del trabajo se han encontrado diferentes condiciones y problemas con los que hay que lidiar para realizar el trabajo con el objetivo de cumplir todas las necesidades expuestas anteriormente.

7.1. SARS-CoV-2

El primer problema y más evidente es el contexto de pandemia actual. Este 2020 debido a la pandemia mundial de SARS-CoV-2 que se ha vivido no ha permitido realizar el trabajo con normalidad. Esto se debe a que la universidad ha tenido que cerrar sus puertas en varias ocasiones durante semanas o incluso meses y no se han podido efectuar las pruebas necesarias en un plazo normal de TFG. Lo cual ha conllevado que la presentación haya tenido que ser aplazada con el fin de realizar dichas pruebas. Otro problema es a la hora de presentar ya que, a fecha de redactar esto, la defensa ha de ser por videollamada, limitando así la exposición del trabajo en gran medida. Por todo ello gran parte del trabajo ha estado centrado en la simulación más que en las pruebas de campo.

7.2. Maqueta

Otro problema encontrado ha sido que el montaje de la maqueta recibido no constaba de ningún tipo de alimentación de los pulsadores, de manera que se ha tenido que cablear (independientemente al diseño del fabricante) una alimentación a cada pulsador desde un cable ya existente que proveyese a todos los minipulsadores de los 24V de continua necesarios.

Con respecto a la maqueta se aprecia un problema añadido al trabajo y es los fallos en la disposición mecánica de los elementos. Estos se acumulan principalmente en la zona de los tornillos sin fin ya que estos están se componen por piezas independientes que unidas en una varilla forman los tornillos sin fin. Estas varillas no disponen de la fijación necesaria y provocan un juego en ellas que desembocan desde un desplazamiento del engranaje que impide que el

motor lo haga girar hasta una separación entre las piezas del tornillo sin fin que provoca que solo gire una parte o atascos. Esto se ha intentado solucionar o impedir ajustando lo mejor posible todas las partes mecánicas con el fin de que no se den estos errores.

7.3. Señales booleanas

En cuanto a las limitaciones, la primera y más evidente es que para efectuar el control solo se disponen de señales booleanas con lo cual no se puede enviar ningún tipo de dato de carácter analógico (medidas, velocidades, etc.). De todos los datos de carácter analógico que no se pueden tener directamente el más relevante para la realización de este proceso de control es la velocidad.

Para obtener los datos de velocidad se requiere obtener los pulsos del encoder y ya en el programa traducirlos. Esto nos lleva a otro condicionante y es que un contador al uso no puede recibir pulsos a tanta velocidad (75 por vuelta a 215rpm) lo que hace que se necesite un contador rápido para ello que pueda detectar todos estos pulsos.

Por otra parte, a la hora de enviar la velocidad que requiere el sistema únicamente se dispone de salidas booleanas así que se ha de recurrir a un control por ancho de pulso (PWM). Este control se basa en abrir y cerrar rápidamente la señal al motor haciendo que este esté arrancando y frenando constantemente de manera controlada. El abrir y cerrar controlada y constantemente la señal se traduce a que al final el motor acaba girando la velocidad deseada inferior a la nominal. Dicho PWM tampoco se puede realizar de la manera ordinaria en el programa así que se han de emplear funciones y elementos concretos que tiene el programa para ello.

Volviendo al contador rápido, también es necesario a la hora de conocer la posición mediante los encoders.

7.4. Entradas y salidas limitadas

Otra limitación evidente es la clara falta de entradas y salidas con respecto a los sensores y actuadores que tiene a disposición el almacén (12 entradas y 8

salidas por las 8 entradas y 6 salidas que tiene el autómata). De manera que la primera decisión tomada ha sido eliminar de la programación la parte menos importante del sistema, es decir, la bandeja de entrada. De manera que ahora las cajas son colocadas y recogidas directamente del brazo giratorio cuando este llegue a la bandeja de entrada. Tras esto se elimina el número exacto de entradas y salidas que se requieren para el control quedando así 8 entradas y 6 salidas.

El objetivo del trabajo es que el programa sea lo más adaptable posible a otros procesos, de esta manera cabe la posibilidad que a la hora de establecer tipos de control en la configuración de los motores en su modo general falten entradas para satisfacer todas las necesidades. Para abordar este escenario se ha de limitar la configuración se dispone de 5 entradas disponibles configurables entre 3 para finales de carrera finales y 2 para contadores rápidos y 3 entradas fijas para los finales de carrera iniciales (indispensable conocer la posición 0 de cada eje). De esta manera se pretende limitar la configuración para que pueda llevarse a cabo el proceso.

Como ya se ha podido observar los encoders emiten dos señales que han de ser recibidas por contadores rápidos. Para economizar el proceso y no requerir de tantos contadores se opta por eliminar una de estas dos señales. Ya que si se está enviando un “forward” al sistema y el motor gira se deduce fácilmente que el sistema gira hacia adelante, de manera que las dos señales del encoder pasan a ser una redundancia. La eliminación de la segunda señal de cada encoder provoca que se tengan dos entradas libres en la prueba del laboratorio, quedando así el sistema en 6 entradas y 6 salidas.

Tras haber conocido todas las limitaciones de entradas y salidas y proceder a configurarlas se encuentra otra limitación. La limitación encontrada es tal vez la más determinante a la hora de realizar el control y es la falta de entradas configurables como contadores rápidos y salidas como PWM (solo hay una para contador rápido y ninguna para PWM). Al tratarse de un autómata compacto no se le pueden añadir módulos de manera que esta limitación es del todo determinante. Para llevar a cabo el sistema con este gran lastre se opta por

eliminar el control de velocidad centrando todo el control en la posición e introduciendo el control por tiempo en el modo específico.

El control por tiempo es un control mucho menos preciso a la práctica ya que no es del todo exacto y no se puede recibir ningún tipo de señal que confirme la posición, es decir, si a mitad de un proceso hay un error que impide que el motor avance esto lo desconocerá el control y no tomará medidas en base a ello. Por ello se ha dejado el control de posición por pulsos disponible en la configuración para que si en algún momento el programa puede ser exportado a otro autómata con más posibilidades de entradas, salidas y contadores rápidos se pueda establecer un control más preciso que el empleado en el PM-554.

7.5. Estantes

Como ya se ha mencionado, los estantes no poseen ningún tipo de sensor que detecte si están ocupados o no, de manera que se ha de programar que al acabar según qué proceso el estante o los estantes afectados cambien de estado (de ocupado a libre y viceversa). También existe la posibilidad de que alguno de los estantes vea su estado alterado por medios ajenos al proceso de control (manualmente, accidentes, mantenimiento…) así que en la programación también ha de existir una forma para alterar el estado de un estante de manera manual y ajena a los procesos.

8. Programación del PLC

La programación del PLC es la base del trabajo y donde han ido a parar la mayor parte de los esfuerzos. El programa empleado es el Codesys V1.3. Esta es la principal razón por la que se ha decidido avanzar con el autómata PM-554 pese a todas las limitaciones encontradas, ya que el PM-554 es el PLC disponible en el laboratorio y con más entradas y salidas disponibles. El resto de los autómatas presentes en el laboratorio con más entradas y salidas o directamente modulables no son compatibles con la versión de Codesys que se ha proporcionado al alumno. Los otros programas compatibles con dichos PLC´s no ha sido posible obtener ya sea por no tener clave para obtenerlo de la manera habitual o el desconocimiento para obtenerlo por otras vías. Otra razón importante para elegir Codesys V1.3 es la necesidad de poder programar desde casa debido a la, ya mencionada anteriormente, situación de pandemia.

8.1. Configuración del programa y conexión

Para configurar el programa lo primero es decidir en qué lenguaje se va a programar. Los lenguajes de programación de autómatas disponibles son los siguientes:

-Diagrama de contactos (LD)

-Lenguaje por lista de instrucciones (IL)

-Lenguaje Grafcet (SFC)

-Texto estructurado (ST)

-Bloques de función (FBD)

-Lenguaje de diagrama de función continua (CFC)

De todos los lenguajes mentados, el estudiante está familiarizado con el diagrama de contactos y con el texto estructurado. De entre estos dos se ha elegido el texto estructurado ya que existe una mayor familiarización y además provee de una programación más profunda y compacta que el diagrama de contactos pese a que este, visualmente hablando, es más accesible.

El lenguaje en texto estructurado adapta la programación del lenguaje Pascal. Dicho lenguaje se caracteriza por su estructura fuertemente tipificada, con lo cual el código se divide en porciones de fácil lectura llamadas funciones y todas las variables han de ser declaradas para su uso. Es un lenguaje muy simple y flexible que permite la programación en diversos estilos.

Con todo sabido, lo primero es crear un documento con la configuración AC500 PM554 V1.3 en texto estructurado como se ve en las siguientes imágenes:

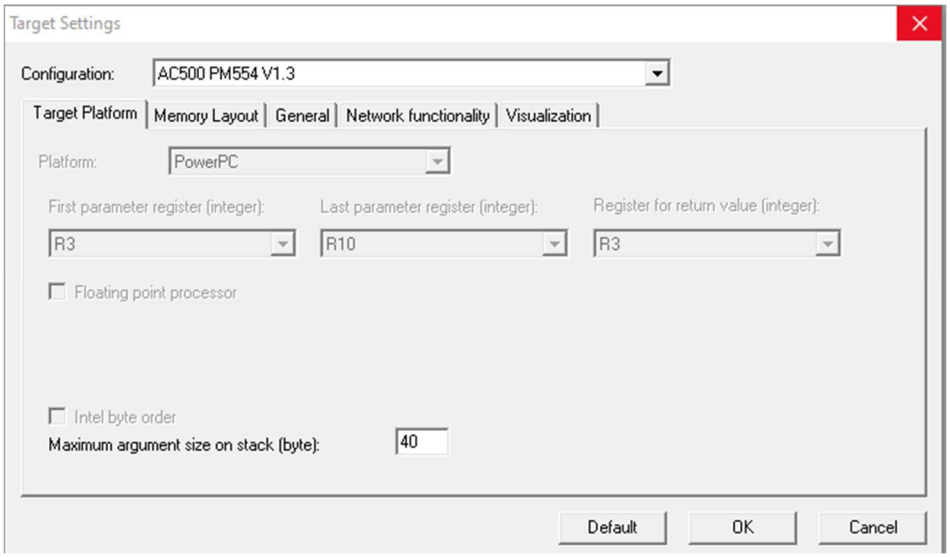


Figura 23. Target Settings

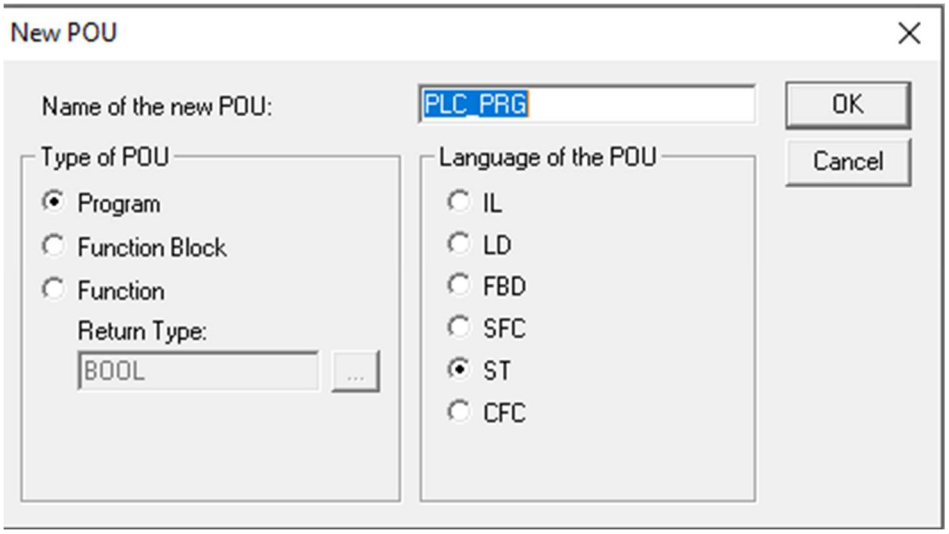


Figura 24. Lenguaje de programación

Tras abrir el programa con la configuración elegida se ha de configurar el cable que conecta el PC con el autómata. Como ya se ha dicho el cable se trata de un TK503 y para que sea reconocido por el PC y poder configurarlo hay que instalar los divers del cable. Al tener el cable reconocido por el ordenador se ha de ir al “Administrador de Dispositivos” ubicado en el “Panel de control” y buscar el cable en “Puertos (COM y LPT)”.

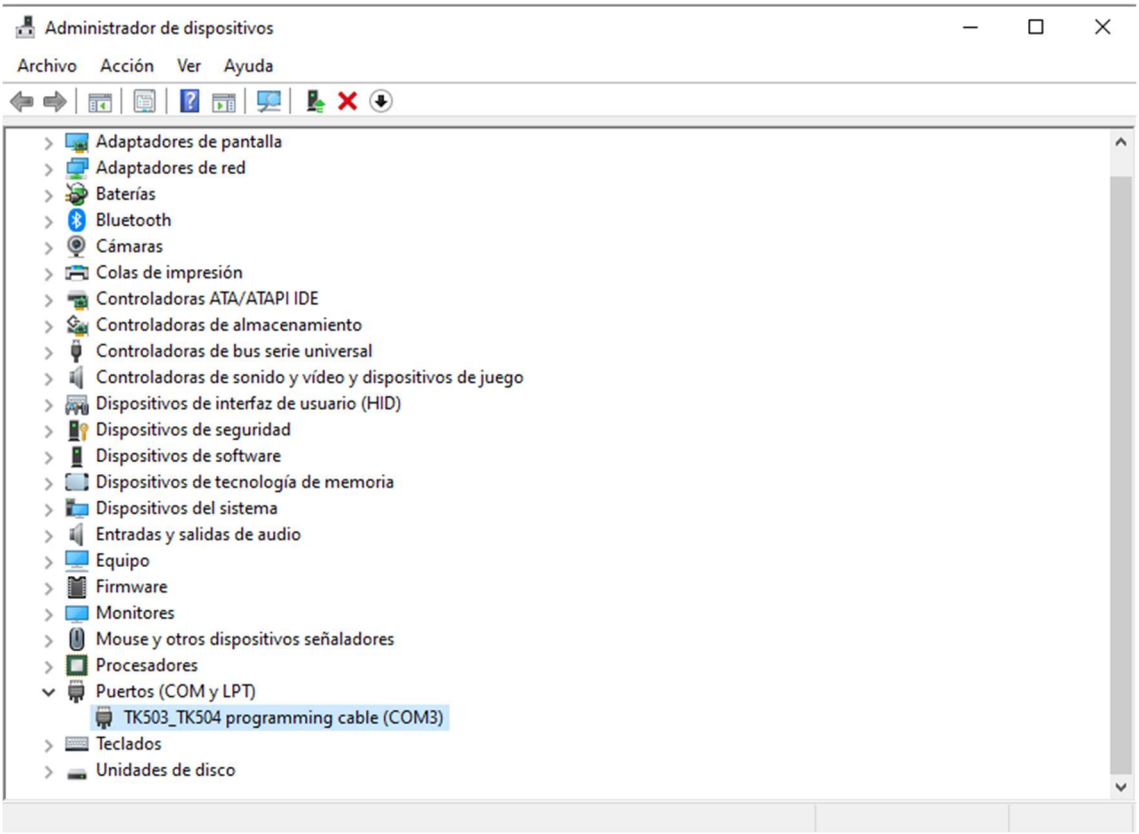


Figura 25. Administrador de dispositivos

Tras encontrar el dispositivo hay que fijarse en el COM que posee, cambiarlo en el caso que se desee o sea necesario e introducir los siguientes datos al entrar en el dispositivo dentro de “Propiedades”-> “Configuración de puerto”:

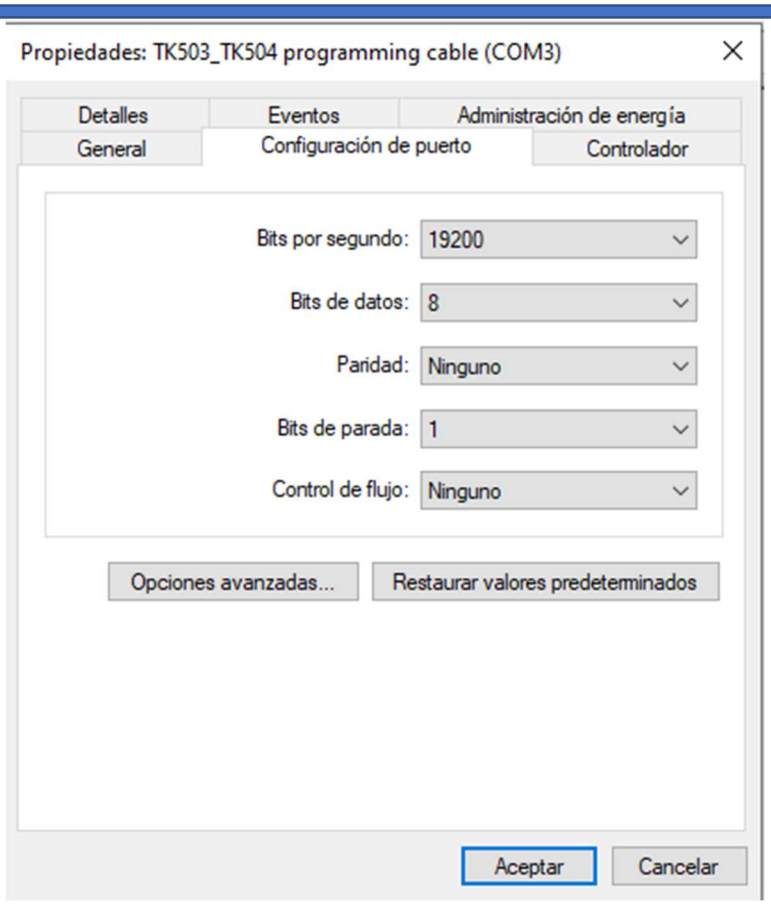


Figura 26. Propiedades del dispositivo

Al entrar en el apartado de “configuración de puerto” se introducen los datos vistos en la figura 25. Tras tener el dispositivo ya configurado es momento de configurar el programa para que reconozca el dispositivo. Para ello, lo primero es ir a la barra superior del programa y entrar en “Online”->”Communication Parameters…”. En la ventana emergente que aparece se ha de crear un nuevo canal “Serial (RS232)”. Al crearlo se introducen los datos conforme a la configuración del cable en el panel de control quedando de esta manera:

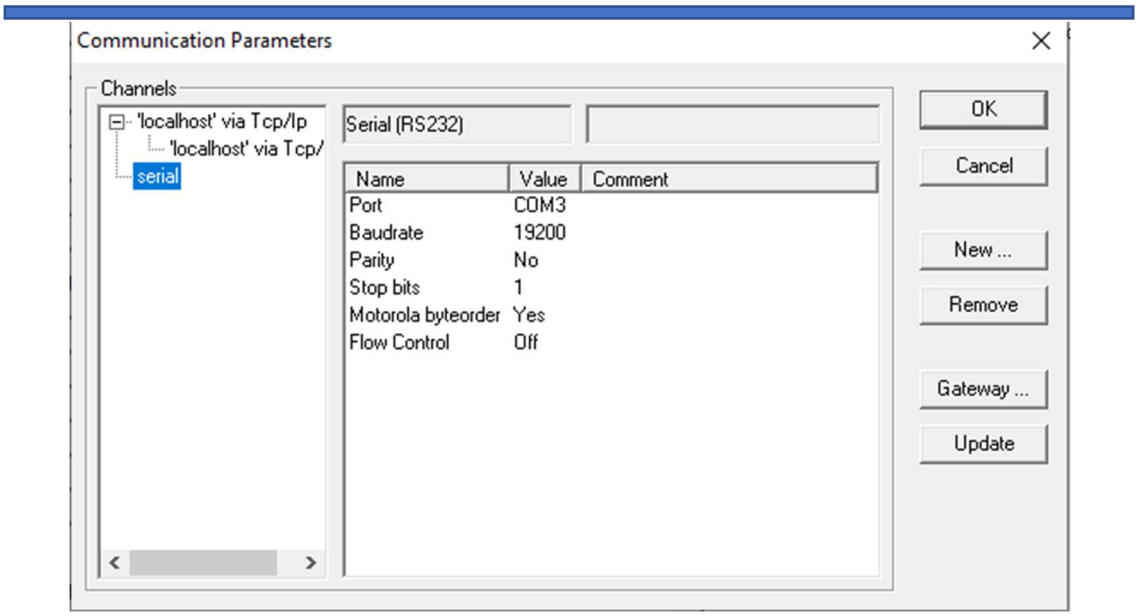


Figura 27. Communication parameters

Ahora solo queda configurar el puerto del autómata. Para ello se ha de acceder a “Resources”, “PLC Configuration”, “Interfaces”, “COM1”. El COM hace referencia al puerto del autómata donde de conecta el cable, en este caso el número 1. En dicho puerto se ha de introducir la misma configuración puesta en el panel de control y en “Communication Parameters…”.

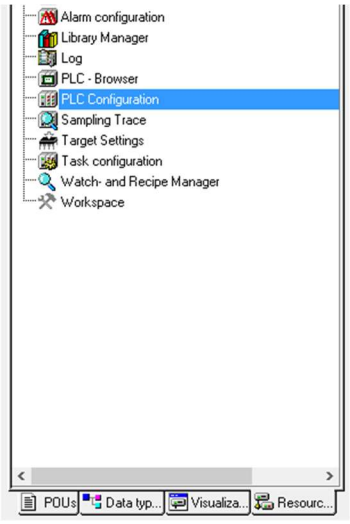


Figura 28. PLC Configuration

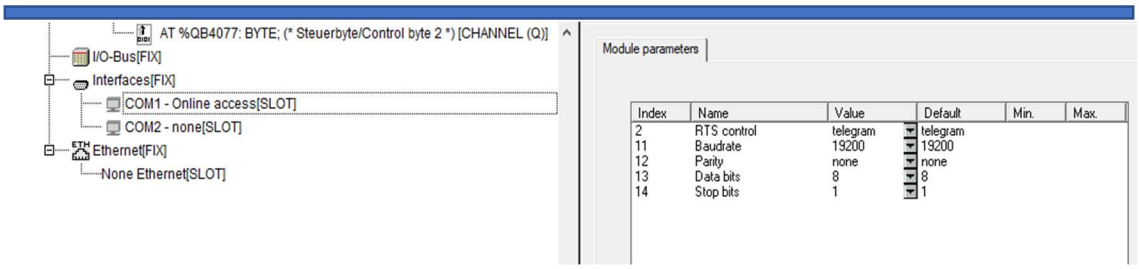


Figura 29. Interfaces-COM1

8.2. Programa

A la parte principal de la programación se le ha llamado “PLC_PRG” dentro del Codesys. Es donde se integran todas las entradas, las funciones y los bloques de función. También programa principal es donde se ha programado el avance del proceso entre sus coordenadas con una diferencia notable entre el modo general y específico. Esta diferencia es que, en el modo específico, el sistema no realiza movimientos diagonales (se mueve en una sola dimensión a la vez) mientras que en el modo general sí. Es debido a que para preservar la estabilidad de las cajas en el brazo giratorio del almacén y que dicha caja no caiga al suelo. Esto no es necesario en el modo general ya que únicamente limitaría movimientos en este modo.

La programación principal obedece al modelo del siguiente flujograma:

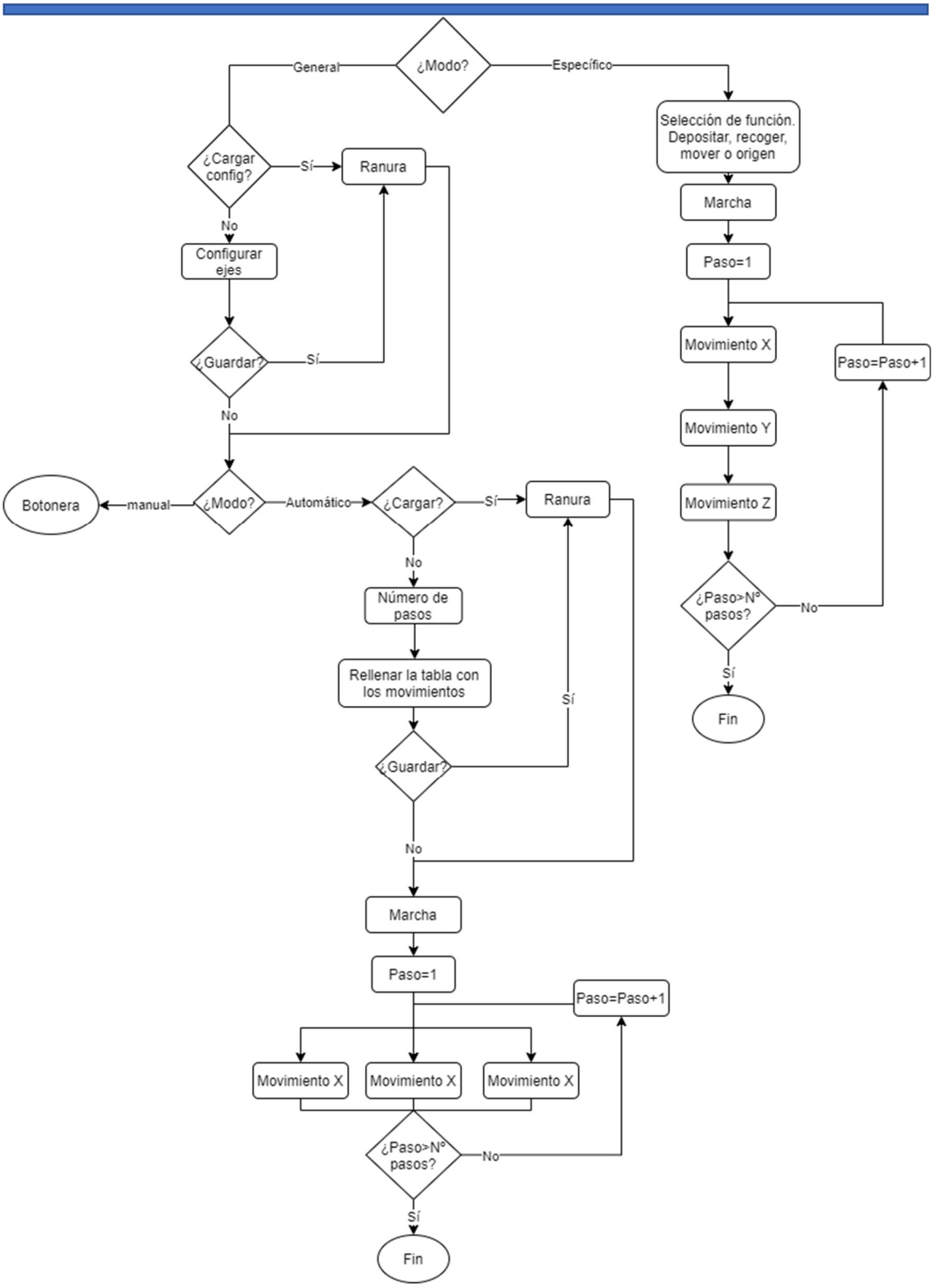


Figura 30. Flujograma principal

En el flujograma también se aprecian las opciones de guardado en el modo general tanto de configuración como de movimientos.

8.2.1 Variables

Las primeras variables creadas y empleadas para el programa son las entradas y salidas directas del autómata. También se han creado otras posibles entradas y salidas por si se emplean métodos de control distintos a los empleados en el método específico. Estas son:

-MX_FWD, MY_FWD Y MZ_FWD: Variables booleanas que controlan la señal a los motores hacia adelante (forward).

-MX_REV, MY_REV Y MZ_REV: Variables booleanas que controlan la señal a los motores hacia atrás (reverse).

-MX_FCI, MY_FCI y MZ_FCI: Variables booleanas que reciben la señal enviada desde los finales de carrera iniciales de los motores.

-MX_FCF, MY_FCF y MZ_FCF: Variables booleanas que reciben la señal enviada desde los finales de carrera (si los hay) finales de los motores.

-MX_P, MY_P y MZ_P: Variables booleanas que reciben los pulsos de los encoders (si los hay) de los motores.

Para el control de los motores se han diseñado dos tipos de bloques de función (se desarrollan más adelante).

-MOTOR_X, MOTOR_Y y MOTOR_Z: Son variables tipo MOTOR que se encargan de controlar los movimientos del motor en base a su tipo de control, límites y posición.

-M_CONFIG_X, M_CONFIG_Y, M_CONFIG_Z: Se trata de variables tipo M_CONFIG encargadas de recoger los datos de configuración de cada motor y traducirlos a valores interpretables por los bloques de función tipo MOTOR. También se ocupan de las acciones de guardar y cargar configuraciones.

Para comunicar las distintas posibles entradas del motor que pueden configurarse se han creado las siguientes variables para limitar las configuraciones y asociarlas a las entradas correspondientes:

N_ENCODER: Variable tipo INT que se encarga de contar cuantos encoders están en uso.

COUNTER: Vector que va del 1 al 2 de variables tipo BOOL. Son las variables por las que entran directamente la señal de los posibles encoders.

AUX_I: Vector que va del 1 al 3 de variables tipo BOOL. Por estas variables entran directamente las posibles señales de los finales de carrera finales.

El programa (excepto en el modo manual) emplea vectores con coordenadas preestablecidas para realizar los movimientos necesarios de un proceso en concreto. De manera que cuando cumple las coordenadas de la primera parte del vector pasa a la segunda y así hasta que acaba el vector. Estas son las variables relacionadas con dicho proceso:

-PASO: Variable tipo INT con un valor inicial de 1 encargada de indicar la parte del vector en la que se encuentra el proceso.

-N_PASO: Variable tipo INT que configura el tamaño del vector.

-CAMINO: Vector tipo Step que va desde el 1 al 150. Es donde se introducen las coordenadas en número de pulsos al sistema.

-CAMINO_MM: Vector tipo Step que va desde el 1 al 150. Es donde se introducen las coordenadas en milímetros al sistema.

-VACIO: Constituye un vector igual que el CAMINO o el CAMINO_MM, pero este siempre permanece en 0.

-Step: STRUCT creada en la cual se introducen coordenadas X, Y, Z en forma de valores tipo INT.

-CONFIG: Vector que va desde el número 1 al 3 (el 1 pertenece al motor X; el 2, al Y y el 3, al Z) de tipo CONFIG encargado de recoger las configuraciones de

los motores X, Y, Z. Las variables tipo CONFIG son STRUCTs creadas que albergan los siguientes datos de configuración de un motor:

a) ENCODER: Variable booleana que informa de la existencia de un encoder.

b) VN: Variable tipo REAL donde se determina la velocidad nominal del motor en rpm.

c) NPV: Variable entera que configura el número de pulsos por vuelta que envía el encoder.

d) FCF: Variable booleana que informa de la existencia de un final de carrera final.

e) D_MAX_MM: Variable tipo REAL que determina máxima distancia que alcanza el motor en milímetros.

f) D_MAX: Variable tipo INT que determina la máxima distancia alcanzable por el motor en número de pulsos.

g) ANVANCE: Variable entera que configura los milímetros que avanza el motor por vuelta.

h) TIEMPO: Variable booleana que informa de la existencia de un control por temporizadores.

i) V: Variable tipo REAL relacionada de manera directa con la velocidad de avance del motor en milímetros por segundo.

-X, Y, Z: Variables tipo INT que informan de la posición en la que se encuentran los motores en número de pulsos.

-X_MM, Y_MM, Z_MM: Variables tipo REAL que informan de la posición en la que se encuentran los motores en milímetros.

-Run: Variable booleana encargada de emitir la señal para dar la marcha a todo el proceso.

-FIN: Variable tipo flanco positivo (R_TRIG) encargada de informar sobre la finalización de todo el proceso.

-RESET: Variable booleana encargada de reiniciar todo el proceso antes de que este llegue a su fin.

-CLEAN: Variable tipo BOOL que envía una señal que limpia el vector enviando todos sus valores a 0.

Como ya se ha dicho en reiteradas ocasiones el programa cuenta con un modo general y un modo específico y en cada modo existen varias funciones claramente diferenciadas. Estas son las variables que inciden en las funciones del modo específico:

-MOV, DROP, LOOT y ORIGEN: Variables booleanas que indican el modo que se ha elegido (mover, dejar, recoger u origen).

-CASILLAS: Vector que va del 0 al 10 de tipo “Slot” donde se indican las coordenadas y los estados de cada estante del almacén (del 1 al 9) y la ubicación de la bandeja de entrada.

-Slot: Tipo de variable STRUCT creada que indica la ubicación, mediante las variables X, Y de tipo INT, y el estado de cada estante mediante la variable booleana “Ocupado”.

-N: Variable entera encargada de elegir el estante donde realizar la función seleccionada. Su valor inicial es 10 ya que este señala la bandeja de entrada.

-N2: Variable entera encargada de elegir la segunda ubicación cuando se selecciona el modo “Mover”. Al igual que la variable N y por las mismas razones su valor inicial es 10.

-BOLCK: Variable booleana que impide iniciar una marcha si no se ha seleccionado ningún modo o este se ha seleccionado de manera errónea.

Por otra parte, existen variables exclusivas del modo general y sus funciones. Su mayor diferencia radica en las funciones de guardado existentes tanto en los procesos como en las configuraciones. Las variables del modo general son las siguientes:

-MANUAL, AUTO: Variables booleanas mediante las que se selecciona el modo requerido.

-SAVE: Variable vectorial tipo STEP_SAVE encargada de almacenar y cargar procesos en 3 espacios de guardado (del 1 al 3).

-STEP_SAVE: Tipo de variable creada STRUCT encargada de almacenar un proceso completo. Se compone de un vector STEP (visto anteriormente) donde se informa de las coordenadas y una variable N_STEP (vista anteriormente) que determina el tamaño del vector.

-CONFIG_SAVE: Variable vectorial tipo CONFIG encargada de almacenar y cargar configuraciones en 5 espacios de guardado (del 1 al 5).

-GUARDAR: Señal booleana mediante la que se informa al programa de guardar el proceso creado en la ranura seleccionada.

-GUARDAR: Señal booleana mediante la que se informa al programa de cargar el vector de la ranura seleccionada.

-N_SAVE: Variable entera que determina el número de ranura de guardado de proceso seleccionada.

-PILOTO: Variable booleana que indica si la ranura de guardado seleccionada se encuentra vacía o no.

-GUARDAR_X, GURADAR_Y y GURADAR_Z: Variables booleanas encargadas de guardar la configuración creada de los motores X, Y o Z en la ranura de guardado seleccionada.

-CARGAR_X, CARGAR_Y y CARGAR_Z: Variables tipo BOOL encargadas de cargar en la configuración de los motores X, Y o Z una configuración guardada en la ranura indicada.

-NCX, NCY y NCZ: Variables enteras mediante las que se selecciona la ranura donde cargar o guardar una configuración para los motores X, Y o Z.

También se ha de tener en cuenta que a la hora de cambiar de modo no se queden funciones activas residuales que puedan causar problemas en el funcionamiento del programa. Para ello se ha creado una variable booleana

llamada AUX_PANT encargada de limpiar estas funciones y evitar errores en el cambio de modo.

8.2.2. Código

La primera parte por la que empieza el código es por los bloques de función tipo MOTOR que controlan los movimientos de cada motor. Se ha optado por emplear bloques de función ya que los tres motores funcionan de la misma manera al poderse configurar tanto las características de estos como los métodos de control que se emplean. Más adelante se profundiza en este tipo de FB mostrando los procesos que ocurren en su interior. Las FBs mostradas a continuación se referencian en el flujograma visto previamente con los bloques "movimiento X", "movimiento Y" y "movimiento Z". En las siguientes imágenes se pueden apreciar las entradas y salidas de los FBs:

```
MOTOR_X(
    N:=CAMINO[PASO].X ,
    N_ACTUAL:= X,
    RUN:=Run,
    FWD:= MX_FWD,
    REV:=MX_REV,
    MANUAL:=MANUAL,
    P:=MX_P ,
    FCI:= MX_FCI,
    FCF:=MX_FCF,
    FCF_CONFIG:=CONFIG[1].FCF,
    ENCODER:=CONFIG[1].ENCODER,
    D_MAX:= CONFIG[1].D_MAX,
    NPV:=CONFIG[1].NPV,
    MM:=CAMINO_MM[PASO].X,
    D_MAX_MM:=CONFIG[1].D_MAX_MM,
    MM_ACTUAL:=X_MM,
    AVANCE:=CONFIG[1].AVANCE,
    V:=CONFIG[1].V,
    TIEMPO:=CONFIG[1].TIEMPO
    );
```

Figura 31. FB motor X

```
MOTOR_Y(
    N:=CAMINO[PASO].Y ,
    N_ACTUAL:=Y ,
    RUN:=(MOTOR_X.FIN AND (MOV OR DROP OR LOOT OR ORIGEN)) OR (RUN AND NOT (MOV OR DROP OR LOOT OR ORIGEN)),
    FWD:= MY_FWD,
    REV:=MY_REV,
    MANUAL:=MANUAL,
    P:=MY_P,
    FCI:= MY_FCI,
    FCF:=MY_FCF,
    FCF_CONFIG:=CONFIG[2].FCF,
    ENCODER:=CONFIG[2].ENCODER,
    D_MAX:= CONFIG[2].D_MAX,
    NPV:=CONFIG[2].NPV,
    MM:=CAMINO_MM[PASO].Y,
    D_MAX_MM:=CONFIG[2].D_MAX_MM,
    MM_ACTUAL:=Y_MM,
    AVANCE:=CONFIG[2].AVANCE,
    V:=CONFIG[2].V,
    TIEMPO:=CONFIG[2].TIEMPO
    );
```

Figura 32. FB motor Y

```
MOTOR_Z(
    N:=CAMINO[PASO].Z ,
    N_ACTUAL:=Z ,
    RUN:=(MOTOR_Y.FIN AND (MOV OR DROP OR LOOT OR ORIGEN)) OR (RUN AND NOT (MOV OR DROP OR LOOT OR ORIGEN)),
    FWD:= MZ_FWD,
    REV:=MZ_REV,
    MANUAL:=MANUAL,
    P:=MZ_P,
    FCI:= MZ_FCI,
    FCF:=MZ_FCF,
    FCF_CONFIG:=CONFIG[3].FCF,
    ENCODER:=CONFIG[3].ENCODER,
    D_MAX:= CONFIG[3].D_MAX,
    NPV:=CONFIG[3].NPV,
    MM:=CAMINO_MM[PASO].Z,
    D_MAX_MM:=CONFIG[3].D_MAX_MM,
    MM_ACTUAL:=Z_MM,
    AVANCE:=CONFIG[3].AVANCE,
    V:=CONFIG[3].V,
    TIEMPO:=CONFIG[3].TIEMPO
    );
```

Figura 33. FB motor Z

Como se puede apreciar el RUN de los motores Y y Z cambian en base a si está alguna de las funciones del modo específico o no haciendo que dependan del FIN del motor anterior o directamente del RUN respectivamente. Esto se traduce a que en el modo específico el sistema no traza diagonales (no activa más de un motor a la vez) mientras que en el general sí.

Seguido al control de los motores se encuentran los FBs que constituyen la configuración de los tres motores y sus opciones de guardar y cargar. Los bloques de función son referenciados en el flujograma en el bloque de “Configurar ejes” y en el proceso de guardar y cargar la configuración del modo

general. En el específico no se referencia ya que la configuración viene predefinida. La razón de emplear FBs obedece a las mismas razones que los FBs previamente vistos y también se desarrolla este FB más adelante. En las siguientes figuras se pueden apreciar las entradas y salidas de los bloques de función:

```
    M_CONFIG_X(
GUARDAR:=GUARDAR_X ,
CARGAR:=CARGAR_X ,
ENCODER:=CONFIG[1].ENCODER ,
NPV:=CONFIG[1].NPV ,
AVANCE:=CONFIG[1].AVANCE ,
D_MAX_MM:=CONFIG[1].D_MAX_MM ,
MM:=CAMINO_MM[PASO].X ,
VN:=CONFIG[1].VN ,
FCF:=CONFIG[1].FCF ,
TIEMPO:=CONFIG[1].TIEMPO,
ENCODER_S:= CONFIG_SAVE[NCX].ENCODER,
NPV_S:=CONFIG_SAVE[NCX].NPV ,
AVANCE_S:=CONFIG_SAVE[NCX].AVANCE ,
D_MAX_MM_S:=CONFIG_SAVE[NCX].D_MAX_MM ,
VN_S:=CONFIG_SAVE[NCX].VN ,
FCF_S:=CONFIG_SAVE[NCX].FCF ,
TIEMPO_S:=CONFIG_SAVE[NCX].TIEMPO,
PULSOS=>CAMINO[PASO].X ,
D_MAX=>CONFIG[1].D_MAX ,
V=>CONFIG[1].V
    );
```

Figura 34. FB configuración motor X

```
    M_CONFIG_Y(
GUARDAR:=GUARDAR_Y ,
CARGAR:=CARGAR_Y ,
ENCODER:=CONFIG[2].ENCODER ,
NPV:=CONFIG[2].NPV ,
D_MAX_MM:=CONFIG[2].D_MAX_MM ,
MM:=CAMINO_MM[PASO].Y ,
VN:=CONFIG[2].VN ,
FCF:=CONFIG[2].FCF ,
AVANCE:=CONFIG[2].AVANCE,
TIEMPO:=CONFIG[2].TIEMPO,
ENCODER_S:= CONFIG_SAVE[NCY].ENCODER,
NPV_S:=CONFIG_SAVE[NCY].NPV ,
D_MAX_MM_S:=CONFIG_SAVE[NCY].D_MAX_MM ,
VN_S:=CONFIG_SAVE[NCY].VN ,
FCF_S:=CONFIG_SAVE[NCY].FCF ,
AVANCE_S:=CONFIG_SAVE[NCY].AVANCE,
TIEMPO_S:=CONFIG_SAVE[NCY].TIEMPO,
PULSOS=>CAMINO[PASO].Y ,
D_MAX=>CONFIG[2].D_MAX ,
V=>CONFIG[2].V
    );
```

Figura 35. FB configuración motor Y

```
    M_CONFIG_Z(
GUARDAR:=GUARDAR_Z ,
CARGAR:=CARGAR_Z ,
ENCODER:=CONFIG[3].ENCODER ,
NPV:=CONFIG[3].NPV ,
D_MAX_MM:=CONFIG[3].D_MAX_MM ,
MM:=CAMINO_MM[PASO].Z ,
VN:=CONFIG[3].VN ,
FCF:=CONFIG[3].FCF ,
AVANCE:=CONFIG[3].AVANCE,
TIEMPO:=CONFIG[3].TIEMPO,
ENCODER_S:= CONFIG_SAVE[NCZ].ENCODER,
NPV_S:=CONFIG_SAVE[NCZ].NPV ,
D_MAX_MM_S:=CONFIG_SAVE[NCZ].D_MAX_MM ,
VN_S:=CONFIG_SAVE[NCZ].VN ,
FCF_S:=CONFIG_SAVE[NCZ].FCF ,
AVANCE_S:=CONFIG_SAVE[NCZ].AVANCE,
TIEMPO_S:=CONFIG_SAVE[NCZ].TIEMPO,
PULSOS=>CAMINO[PASO].Z ,
D_MAX=>CONFIG[3].D_MAX ,
V=>CONFIG[3].V
    );
```

Figura 36. FB configuración motor Z

En estos bloques se aprecia más claramente como mediante el vector CONFIG se recogen los datos de configuración de los motores X, Y, Z clasificándolos en el vector del 1 al 3.

Tras el control y la configuración se encuentran las configuraciones preestablecidas para el modo específico:

```
IF MOV OR DROP OR LOOT OR ORIGEN THEN
    CONFIG[1].ENCODER:=FALSE;
    CONFIG[1].NPV:=3;
    CONFIG[1].VN:=324;
    CONFIG[1].FCF:=FALSE;
    CONFIG[1].D_MAX_MM:=292;
    CONFIG[1].TIEMPO:=TRUE;
    CONFIG[1].AVANCE:=5;

    CONFIG[2].ENCODER:=FALSE;
    CONFIG[2].NPV:=3;
    CONFIG[2].VN:=324;
    CONFIG[2].FCF:=FALSE;
    CONFIG[2].D_MAX_MM:=135;
    CONFIG[2].TIEMPO:=TRUE;
    CONFIG[2].AVANCE:=5;

    CONFIG[3].ENCODER:=FALSE;
    CONFIG[3].NPV:=3;
    CONFIG[3].VN:=324;
    CONFIG[3].FCF:=TRUE;
    CONFIG[3].D_MAX_MM:=0;
    CONFIG[3].TIEMPO:=FALSE;
END_IF
```

Figura 37. Configuraciones modo específico

Como se puede apreciar los motores X e Y se controlan mediante control por tiempo mientras que el motor Z se controla únicamente con finales de carrera. También se puede apreciar en las configuraciones de los motores X e Y sendas medidas del recorrido en milímetros.

Tras esto se encuentra la llamada al bloqueo para cuando se pretende iniciar la marcha en el modo específico sin haber elegido modo correctamente:

```
BLOCK:= NOT ORIGEN AND NOT MOV AND NOT DROP AND NOT LOOT;
BLOCK:= ((N=10 OR N=0) AND NOT ORIGEN) OR ((N2=10 OR N2=0 OR N2=10) AND MOV);
```

Figura 38. Bloqueos modo específico

Al acabar la configuración del modo específico se introducen los distintos caminos y número de pasos en función del modo elegido. En todos los modos

menos en el ORIGEN (el brazo va a la bandeja de entrada) el brazo va a un punto establecido, sube ligeramente para coger la caja, se dirige a donde la tenga que depositar y baja ligeramente para depositar el objeto en el sitio elegido. Todas estas funciones son referenciadas en el flujograma en el bloque de "Selección de función" presente en el modo específico.

```
IF DROP THEN
    CAMINO_MM[3].X:=CASILLAS[N].X;
    CAMINO_MM[3].Y:=CASILLAS[N].Y;
    CAMINO_MM[2].X:=CAMINO_MM[1].X;
    CAMINO_MM[4].X:=CAMINO_MM[3].X;
    CAMINO_MM[2].Y:=CAMINO_MM[1].Y-15;
    CAMINO_MM[4].Y:=CAMINO_MM[3].Y+5;
    CAMINO_MM[1].Z:=1;
    CAMINO_MM[2].Z:=0;
    CAMINO_MM[3].Z:=1;
    CAMINO_MM[4].Z:=0;
    CAMINO_MM[1].X:=CASILLAS[10].X;
    CAMINO_MM[1].Y:=CASILLAS[10].Y+5;

    N_PASO:=4;
END_IF
```

Figura 39. Caminos modo depositar

```
IF LOOT THEN
    CAMINO_MM[1].X:=CASILLAS[N].X;
    CAMINO_MM[1].Y:=CASILLAS[N].Y+5;
    CAMINO_MM[2].X:=CAMINO_MM[1].X;
    CAMINO_MM[4].X:=CAMINO_MM[3].X;
    CAMINO_MM[2].Y:=CAMINO_MM[1].Y-15;
    CAMINO_MM[4].Y:=CAMINO_MM[3].Y+10;
    CAMINO_MM[1].Z:=1;
    CAMINO_MM[2].Z:=0;
    CAMINO_MM[3].Z:=1;
    CAMINO_MM[4].Z:=0;
    CAMINO_MM[3].X:=CASILLAS[10].X;
    CAMINO_MM[3].Y:=CASILLAS[10].Y-5;

    N_PASO:=4;
END_IF
```

Figura 40. Camino modo recoger

```
IF MOV THEN
    CAMINO_MM[1].X:=CASILLAS[N].X;
    CAMINO_MM[1].Y:=CASILLAS[N].Y+5;
    CAMINO_MM[3].X:=CASILLAS[N2].X;
    CAMINO_MM[3].Y:=CASILLAS[N2].Y-5;
    CAMINO_MM[2].X:=CAMINO_MM[1].X;
    CAMINO_MM[4].X:=CAMINO_MM[3].X;
    CAMINO_MM[2].Y:=CAMINO_MM[1].Y-15;
    CAMINO_MM[4].Y:=CAMINO_MM[3].Y+10;
    CAMINO_MM[1].Z:=1;
    CAMINO_MM[2].Z:=0;
    CAMINO_MM[3].Z:=1;
    CAMINO_MM[4].Z:=0;

    N_PASO:=4;
END_IF
```

Figura 41. Camino modo mover

Como se observa la función depositar envía la caga al sitio elegido desde la bandeja, la función recoger envía a la bandeja la caja del estante seleccionado y la función mover mueve una caja de un estante seleccionado a otro también seleccionado.

```
IF ORIGEN THEN
    CAMINO_MM[1].X:=CASILLAS[10].X;
    CAMINO_MM[1].Y:=CASILLAS[10].Y;
    CAMINO_MM[1].Z:=1;

    N_PASO:=1;
END_IF
```

Figura 42. Camino modo origen

Como se ve la función origen es una función simple donde únicamente se mueve el brazo a la bandeja de entrada.

Tras finalizar los caminos y configuraciones se crea el siguiente modo de avance de los pasos:

```
IF ((MOTOR_Z.FIN AND (DROP OR LOOT OR MOV OR ORIGEN)) OR (MOTOR_X.FIN AND MOTOR_Y.FIN  AND MOTOR_Z.FIN
AND NOT (LOOT OR DROP OR MOV OR ORIGEN))) AND RUN THEN
    PASO:= PASO +1;
END_IF
```

Figura 43. Avance de los pasos

Aquí también se observa la misma diferencia entre el modo general y el específico, donde el modo general traza diagonales y avanza al finalizar el movimiento de los tres motores mientras que el específico solo necesita que finalice el del motor Z ya que los anteriores ya lo habrán hecho. Por ello, en el flujograma, tras pulsar “marcha” el sistema de ordenar a los tres movimientos en serie o en paralelo dependiendo si se encuentra el programa en el modo específico o general.

Tras esto se escribe el código que controla la opción de guardar y cargar los vectores que trazan el camino del proceso del motor junto al número de pasos que necesita visto previamente al bloque de “Marcha” en el flujograma

```
IF GUARDAR THEN
    SAVE[N_SAVE].STEP:=CAMINO_MM;
    SAVE[N_SAVE].N_STEP:=N_PASO;
END_IF

IF CARGAR THEN
    CAMINO_MM:=SAVE[N_SAVE].STEP;
    N_PASO:=SAVE[N_SAVE].N_STEP;
END_IF

IF SAVE[N_SAVE].N_STEP>=1 THEN
    PILOTO:=TRUE;
END_IF
```

Figura 44. Guardar y Cargar

El piloto se activa cuando se ha guardado un número de pasos ya que esto es indispensable para guardar el resto del vector y, por consiguiente, si se ha guardado un número de pasos también se habrán guardado unos caminos.

El AUX_PANT se encarga de limpiar al cambiar de modo funciones y configuraciones residuales para que estas no generen problemas de la siguiente forma:

```
IF AUX_PANT THEN
    CAMINO_MM:=VACIO;
    N_PASO:=0;
    MANUAL:=FALSE;
    AUTO:=FALSE;
END_IF
```

Figura 45. AUX_PANT

También, si se desea limpiar el vector existe la variable CLEAN (previamente mencionada) que se encarga de esta función de la siguiente manera:

```
IF CLEAN THEN
    CAMINO_MM:=VACIO;
END_IF
```

Figura 46. CLEAN

Para finalizar los procesos se hade configurar la manera que se han de reiniciar o acabar y esto ocurre cuando se le da al botón de “Reset”, cuando se cambia de modo abruptamente o cuando finaliza el último paso de un proceso. En el modo específico, solamente este último escenario desemboca en un cambio de estado en los estantes (de libre a ocupado y viceversa) como se puede apreciar:

```
FIN(CLK:= PASO>N_PASO OR RESET OR AUX_PANT);

IF FIN.Q AND DROP AND NOT (RESET OR AUX_PANT) THEN
    CASILLAS[N].Ocupado:= TRUE;
END_IF

IF FIN.Q AND LOOT AND NOT (RESET  OR AUX_PANT) THEN
    CASILLAS[N].Ocupado:= FALSE;
END_IF

IF FIN.Q AND MOV AND NOT (RESET OR AUX_PANT) THEN
    CASILLAS[N].Ocupado:= FALSE;
    CASILLAS[N2].Ocupado:= TRUE;
END_IF

IF FIN.Q THEN
    N:=0;
    N2:=0;
    Run:= FALSE;
    PASO:=1;
    DROP:=FALSE;
    LOOT:=FALSE;
    MOV:=FALSE;
    ORIGEN:=FALSE;
END_IF
```

Figura 47. Finalización de los procesos

Tras esta finalización el valor de la variable PASO vuelve a su valor inicial.

```
IF PASO>N_PASO THEN
    PASO:= 1;
END_IF
```

Figura 48. PASO a valor inicial

Para acabar solo queda concluir la programación con las direcciones de las entradas y salidas del autómata. Para ello, primero los finales de carrera finales y los encoders existentes se tienen que repartir las entradas correspondientes. Esto sucede al enviar el cada bloque de función de configuración unos valores enteros que indican el número de entradas que necesitan y tras ser indicados se reparten a patir de una prioridad descendente desde X hasta Z. Además, calcula el número de encoders configurados para impedir que existan más de dos.

```
IF NOT SIM THEN
    IF CONFIG[1].ENCODER THEN
        MX_P:=COUNTER[1];
    END_IF
        IF CONFIG[1].FCF THEN
        MX_FCF:=AUX_I[1];
    END_IF

    IF CONFIG[2].ENCODER THEN
        MY_P:=COUNTER[M_CONFIG_X.N_ENCODER+1];
    END_IF
        IF CONFIG[2].FCF THEN
        MY_FCF:=AUX_I[M_CONFIG_X.AUX_USO+1];
    END_IF

    IF  CONFIG[3].ENCODER THEN
        MZ_P:=COUNTER[M_CONFIG_Y.N_ENCODER+1];
    END_IF
        IF CONFIG[3].FCF THEN
        MZ_FCF:=AUX_I[M_CONFIG_X.AUX_USO+M_CONFIG_Y.AUX_USO+1];
    END_IF
END_IF

N_ENCODER:=M_CONFIG_X.N_ENCODER+M_CONFIG_Y.N_ENCODER+M_CONFIG_Z.N_ENCODER;
```

Figura 49. Reparto de entradas de FCFs y esncoders

Finalmente se vinculan las entradas y salidas en base a las que da el autómata. Las salidas se condicionan principalmente a que no esté activado el fin de cada motor.

```
IF NOT SIM THEN
    MX_FCI:=%IX4000.0;
    MY_FCI:=%IX4000.1;
    MZ_FCI:=%IX4000.2;
    COUNTER[1]:=%IX4000.3;
    COUNTER[2]:=%IX4000.4;
    AUX_I[1]:=%IX4000.5;
    AUX_I[2]:=%IX4000.6;
    AUX_I[3]:=%IX4000.7;
END_IF

%QX4000.0:=MX_FWD AND NOT MOTOR_X.FIN;
%QX4000.1:=MX_REV AND NOT MOTOR_X.FIN;
%QX4000.2:=MY_FWD AND NOT MOTOR_Y.FIN;
%QX4000.3:=MY_REV AND NOT MOTOR_Y.FIN;
%QX4000.4:=MZ_FWD AND NOT MOTOR_Z.FIN;
%QX4000.5:=MZ_REV AND NOT MOTOR_Z.FIN;
```

Figura 50. Vinculación a entradas y saldas del autómata

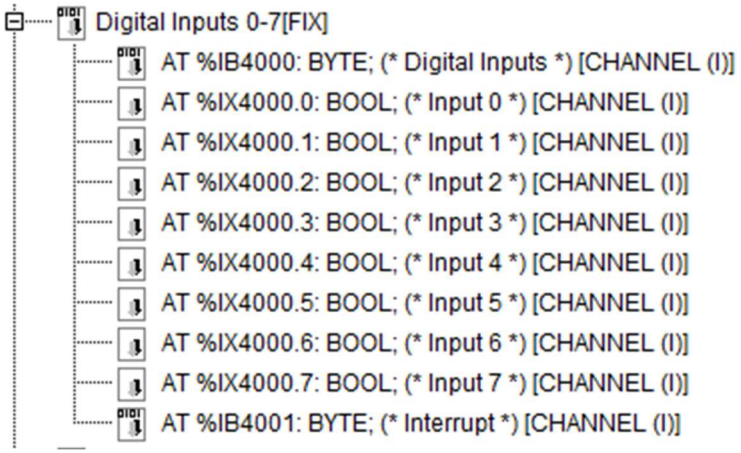


Figura 51. Entradas del autómata

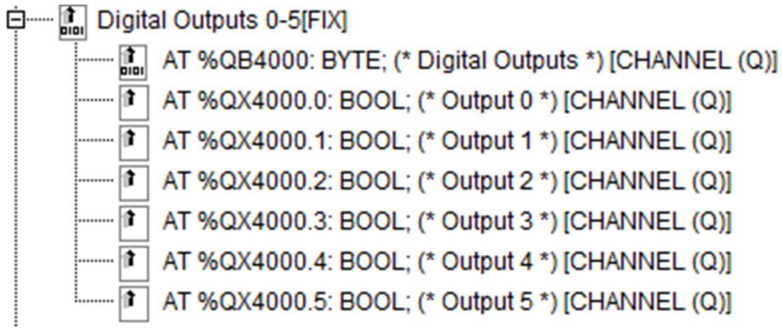


Figura 52. Salidas del autómata

8.2.3. Bloque de función MOTOR

El bloque de función (FB) tipo MOTOR se encarga de controlar el movimiento del motor en base a las instrucciones enviadas y posiciona al motor a raíz de este movimiento. La razón de crear el bloque defunción ha sido por la eficiencia de evitar repetir el mismo código tres veces ya que los tres motores se han de poder controlar y configurar de la misma manera. El bloque de función ha de seguir, a grandes rasgos, el siguiente flujograma:

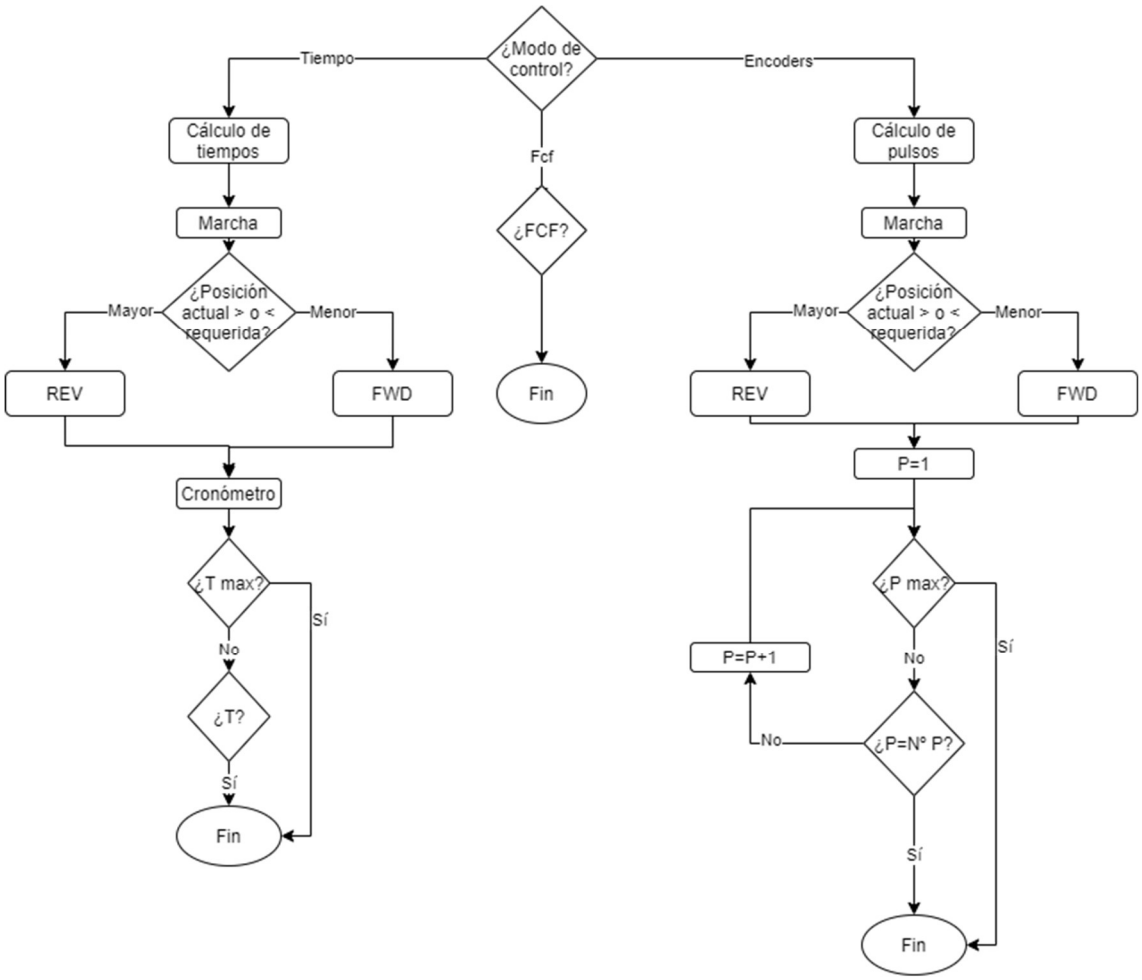


Figura 53. Flujograma funciones del motor

8.2.3.1. Variables

En el bloque de función se ha de declarar desde el principio como actúa una variable. Pueden ser entradas, entradas/salidas, salidas o variables internas. La

existencia de estas últimas es lo que diferencia a un bloque de función (FB) de una función (FC).

Para empezar, se han de definir las variables de entrada que son las siguientes:

-N: Variable entera que introduce la posición la laque ha de ir el motor en número de pulsos.

-P: Variable booleana encargada de recibir los pulsos del encoder.

-RUN: Variable booleana que autoriza la marcha.

-ENCODER: Variable booleana que indica la existencia de un control por encoder.

-TIEMPO: Variable booleana que indica un control por tiempo.

-FCI: Variable tipo BOOL que recibe la señal del final de carrera inicial.

-FCF: Variable tipo BOOL que recibe la señal del final de carrera final.

-MANUAL: Variable booleana que recibe la orden de pasar al modo de control manual.

-FCF_CONFIG: Variable booleana que indica la existencia de un final de carrera final.

-D_MAX: Variable entera (INT) que acota la distancia máxima a la que puede llegar el motor en número de pulsos.

-AVANCE: Variable entera que indica la distancia lineal que avanza el motor en una vuelta en milímetros

-NPV: Variable tipo int que informa del número de pulsos que son recibidos por el encoder en una misma revolución.

-V: Variable real que indica la velocidad lineal a la que se desplaza el motor en rpm.

-D_MAX_MM: Variable real que acota la distancia máxima a la que puede llegar el motor en miímetros.

-MM: Variable entera que indica la distancia a la que tiene que llegar el motor con respecto del 0 en milímetros.

Tras tener las entradas definidas sean de definir las entradas/salidas. Estas son entradas que también pueden ser modificadas por la FB y enviarlas como salida con su nuevo valor. Las entradas/salidas presentes en el bloque de función son las siguientes:

-N_ACTUAL: Variable entera que indica la posición en la que se encuentra el motor respecto de la posición 0 en número de pulsos.

-FWD: Variable booleana que controla la orden del motor de ir hacia adelante (forward).

-REV: Variable booleana que controla la orden del motor de ir hacia atrás (reverse).

-N_ACTUAL_MM: Variable real que indica la posición en la que se encuentra el motor respecto de la posición 0 en milímetros.

En cuanto a las salidas del bloque de función solo existe una que es:

-FIN: Variable booleana que indica el fin del proceso realizado por el motor.

Para finalizar las variables tan solo queda definir las variables internas del bloque defunción. Dichas variables, como ya se ha dicho es lo que hace único al bloque de función y son las siguientes:

-TEMP_MAX y TEMP: Variables TON de los temporizadores encargados de cronometrar el tiempo hasta la distancia máxima y la distancia que sea de alcanzar respectivamente.

-PULSO: Flanco positivo (R_TRIG) encargado de traducir los pulsos recibidos del encoder a un solo flanco con el fin de realizar la cuenta más exacta posible de pulsos.

-FIN_MANUAL: Flanco negativo encargado de decretar la finalización del modo manual.

-FIN_RUN: Flanco negativo encargado de decretar la finalización de la marcha.

-T, D_MAX_T: Variables tipo TIME encargadas de acotar el tiempo necesario para llegar desde la posición actual hasta la posición requerida o la distancia máxima.

-TIMER: Variable tipo TON encargada de actualizar la posición del motor en su modo manual.

-T_FIN y T_FIN_MAX: Variables booleanas encargadas de señalizar el fin del TEMP y TEMP_MAX respectivamente.

-R_TEMP y R_TEMP_MAX: Flancos positivos (R_TRIG) encargados de traducir T_FIN y T_FIN_MAX a un único pulso.

-R_AUX: Flanco positivo encargado de emitir un solo pulso auxiliar cuando se altera la orden de posición y ya no es la misma que la posición actual.

-R_RUN: Flanco positivo encargado de reiniciar los temporizadores TEMP y TEMP_MAX cuando se inicia de nuevo el sistema.

-PASO: Variable real cuya función es saber la distancia lineal que avanza el motor con un único pulso.

8.2.3.2. Código

La primera parte en programarse ha sido la parte del control por pulsos (representado en la parte derecha del flujograma). En ella a partir de los pulsos procedentes del flanco positivo PULSOS se suman o se restan pulsos a la variable N_ACTUAL en base a si el motor va hacia adelante o hacia atrás. El forward y el reverse se activan (en el caso de no estar en manual) si la posición requerida está a una distancia mayor o menor respectivamente respecto del 0. Cuando se llega la posición requerida, está el reverse activado y llega al final de carrera inicial o está el forward y se llega al final de carrera final el bloque de función envía el fin.

Mientras todo el proceso sucede, la posición actual se traduce a milímetros multiplicando N_ACTUAL por la variable PASO obteniendo la variable N_ACTUAL_MM. La variable PASO se obtiene al dividir AVANCE entre NPV.

```
IF ENCODER THEN
    PULSO(CLK:=P );

    PASO:=INT_TO_REAL(AVANCE/NPV);


    IF PULSO.Q AND FWD AND NOT FCF THEN
        N_ACTUAL:= N_ACTUAL+1;
    END_IF

    IF PULSO.Q AND REV AND NOT FCI THEN
        N_ACTUAL:= N_ACTUAL-1;
    END_IF

    MM_ACTUAL:=N_ACTUAL*PASO;

    IF NOT MANUAL THEN
        IF D_MAX>0 THEN
            FIN:= (RUN AND N=N_ACTUAL) OR (RUN AND N=D_MAX) OR (FCI AND  REV);
        END_IF

        FWD:= RUN AND N>N_ACTUAL;
        REV:= RUN AND N<N_ACTUAL;
    END_IF
END_IF
```

Figura 54. Modo encoders

La segunda parte programada es la de los finales de carrera finales (representado en la parte central del flujograma). El control por finales de carrera finales es el único control compatible con el resto de los modos por ello, el control del forward y del reverse está condicionado a que no exista otro control. En el caso de no haberlos, el forward se activa cuando se requiere una posición superior a 0 hasta tocar el final de carrera final y el reverse cuando se requiere la posición 0 hasta llegar a esta y pulsar el final de carrera inicial. Si el control por finales de carrera finales no se combina con ningún otro control no se puede conocer la posición del motor.

En cualquier caso, cuando se pulsa el final de carrera final, al igual que el inicial, activa la señal de FIN.

```
IF FCF_CONFIG THEN

    IF NOT MANUAL THEN
        IF NOT ENCODER AND NOT TIEMPO THEN
            IF MM>0 THEN
                FWD:=RUN AND NOT FCF;
                FIN:= (FCI AND (MM=0)) OR FCF;
            END_IF

            IF MM=0 THEN
                REV:=RUN AND NOT FCI;
                FIN:= FCI OR (FCF AND (MM>0));
            END_IF
        END_IF
    END_IF
    IF MANUAL THEN
        FIN:=FALSE;

        IF FCF THEN
            FWD:=FALSE;
        END_IF
    END_IF
END_IF
```

Figura 55. Control por FCF

El último de los controles es el control de posición por tiempos (representado en el sector izquierdo del flujograma). Este control es mucho más impreciso que el control por pulsos, pero sale al paso cuando se requiere de control de posición, pero no se tienen medios para realizar uno por pulsos (como es el caso). Este control se basa en calcular el tiempo que se tarda en llegar a la posición requerida a partir de la distancia a esta y la velocidad lineal del motor. Tras el cálculo y darle a la marcha se inician los TON TEMP y TEMP_D_MAX y realizan la acción de forward o reverse (dependiendo de la posición que hay que alcanzar). Cuando uno de los dos finaliza el bloque de función emite un FIN. Tras actualizarse la distancia que hay que recorrer al avanzar el paso de la secuencia de movimientos los TON se reinician para poder volver a realizar el proceso.

En cuanto a la posición del motor, cuando no se encuentra en modo manual el sistema, se actualiza cuando se detiene a partir de sumarle o restarle (dependiendo del tipo de movimiento) la distancia recorrida. Esta distancia se calcula a partir de la velocidad y el tiempo transcurrido de la marcha. Mientras

que, cuando se encuentra en el modo manual, la posición se actualiza cada 200ms a partir de la velocidad lineal también. Esta actualización de posición no es compatible fuera de un modo manual ya que entra en conflicto con los temporizadores adulterando el proceso.

```
IF TIEMPO THEN
    IF NOT MANUAL THEN

        R_AUX(CLK:=MM_ACTUAL<MM*0.999 OR MM_ACTUAL>MM*1.001);
        FIN_RUN(CLK:=RUN);

        T:=REAL_TO_TIME(1000*ABS(MM-MM_ACTUAL)/V);
        D_MAX_T:=REAL_TO_TIME(1000*ABS(D_MAX_MM-MM_ACTUAL)/V);

        IF FIN_RUN.Q AND NOT FIN AND FWD THEN
            MM_ACTUAL:=MM_ACTUAL+V*TIME_TO_REAL(TEMP.ET)/1000;
        END_IF
        IF FIN_RUN.Q AND NOT FIN AND REV THEN
            MM_ACTUAL:=MM_ACTUAL-V*TIME_TO_REAL(TEMP.ET)/1000;
        END_IF

        TEMP(IN:=RUN AND NOT T_FIN_MAX AND NOT R_AUX.Q AND NOT R_RUN.Q , PT:= T, Q=>T_FIN);
        TEMP_MAX(IN:=RUN AND NOT T_FIN AND MM>=D_MAX_MM AND NOT R_RUN.Q, PT:=D_MAX_T, Q=>T_FIN_MAX);

        FWD:= RUN AND NOT FIN AND (MM>MM_ACTUAL);
        REV:= RUN AND NOT FIN AND (MM<MM_ACTUAL) AND NOT FCI;

        R_TEMP(CLK:=T_FIN);
        R_RUN(CLK:=RUN);
        R_TEMP_MAX(CLK:=T_FIN_MAX);
```

Figura 56. Control por tiempo parte 1

```
        IF R_TEMP_MAX.Q  AND FWD THEN
            MM_ACTUAL:=MM_ACTUAL+V*TIME_TO_REAL(TEMP_MAX.PT)/1000;
        END_IF
        IF R_TEMP_MAX.Q  AND REV  THEN
            MM_ACTUAL:=MM_ACTUAL-V*TIME_TO_REAL(TEMP_MAX.PT)/1000;
        END_IF

        IF R_TEMP.Q AND FWD THEN
            MM_ACTUAL:=MM_ACTUAL+V*TIME_TO_REAL(TEMP.PT)/1000;
        END_IF
        IF R_TEMP.Q AND REV THEN
            MM_ACTUAL:=MM_ACTUAL-V*TIME_TO_REAL(TEMP.PT)/1000;
        END_IF

        FIN:=T_FIN OR T_FIN_MAX AND MM_ACTUAL>MM*0.999 AND MM_ACTUAL>MM*1.001;

    END_IF
    IF MANUAL THEN
        TIMER(IN:=NOT TIMER.Q, PT:=T#200MS);

        IF FWD AND TIMER.Q AND MM_ACTUAL<D_MAX_MM THEN
            MM_ACTUAL:=MM_ACTUAL+0.2*V;
        END_IF

        IF REV AND TIMER.Q THEN
            MM_ACTUAL:=MM_ACTUAL-0.2*V;
        END_IF
    END_IF


END_IF
```

Figura 57. Control por tiempo parte 2

Por último, existen partes del control del motor que comparten los tres tipos de control. La primera es el parar cualquier tipo de señal enviada a la salida hacia motores al salir del modo manual por razones de seguridad. La segunda consiste en cortar la salida reverse y llevar la posición del motor a 0 cuando se pulsa el final de carrera inicial. Otra parte del control consiste en que cuando hay una distancia máxima y se alcanza se ha de cortar la señal de forward. Por último y para ser precisos en el posicionamiento en el caso de errores se sabe que la posición no puede ser menor que 0.

```
FIN_MANUAL(CLK:=MANUAL);

IF FIN_MANUAL.Q THEN
    FWD:=FALSE;
    REV:=FALSE;
END_IF

IF FCI THEN
    MM_ACTUAL:=0;
    REV:=FALSE;
END_IF

IF D_MAX_MM>0 AND MM>=D_MAX_MM THEN
    FWD:=FALSE;
END_IF

IF MM_ACTUAL<0 THEN
    MM_ACTUAL:=0;
END_IF
```

Figura 58. Características generales de la FB MOTOR

Como se ha podido apreciar el control de los motores elegido está diseñado para que se pueda pausar, reanudar y reiniciar en cualquier momento tanto el mismo motor como los elementos que dependan de la activación de la salida FIN.

8.2.4. Bloque de función M_CONFIG

El bloque de función M_CONFIG es el encargado de traducir ciertas partes de la configuración de los motores para ser interpretada por el bloque de función tipo MOTOR. Además, se encarga de las acciones de guardar y cargar configuraciones y de enviar información sobre la configuración que se puede emplear o de si la empleada es correcta o no. Este bloque de función, al igual que el tipo MOTOR, es común para los tres motores al poder configurarse de la misma manera.

8.2.4.1. Variables

Al igual que el bloque de función MOTOR, el bloque M_CONFIG también posee entradas, salidas, entradas/salidas y variables internas. Todas estas guardan relación con el recibimiento y el envío de información sobre la configuración del motor.

Lo primero es definir las variables de entradas al bloque de función que son las siguientes:

-GUARDAR: Señal booleana de entrada que da la orden de guardar la configuración realizada en la ranura seleccionada.

-CARGAR: Señal booleana de entrada que da la orden de cargar la configuración de la ranura seleccionada.

-MM: Entrada de tipo entero que introduce la posición que ha de alcanzar el motor en milímetros.

Tras haber definido las variables solo de entrada se definen las variables de entrada que pueden ser modificables, es decir, las de entrada/salida:

-NPV y NPV_S: Valores de tipo entero que definen el número de pulsos por vuelta que recibe el motor de la configuración realizada y la ranura de guardado seleccionada respectivamente.

-VN y VN_S: Valores de tipo real que definen la velocidad nominal en rpm que posee el motor de la configuración realizada y la ranura de guardado seleccionada respectivamente.

-FCF y FCF_S: Valores de tipo booleano que informan de la existencia de un final de carrera final en el control del motor de la configuración realizada y en el de la ranura de guardado seleccionada respectivamente.

-ENCODER y ENCODER_S: Valores de tipo booleano que informan de la existencia de un control por número de pulsos en el control del motor de la configuración realizada y en el de la ranura de guardado seleccionada respectivamente.

-TIEMPO y TIEMPO_S: Valores de tipo booleano que informan de la existencia de un control por tiempo en el control del motor de la configuración realizada y en el de la ranura de guardado seleccionada respectivamente.

-AVANCE y AVANCE_S: Valores de tipo entero que definen el avance lineal en milímetros tras una revolución del motor de la configuración realizada y de la ranura de guardado seleccionada respectivamente.

-D_MAX_MM y D_MAX_MM_S: Valores de tipo real encargados de definir la distancia máxima que puede alcanzar el motor de la configuración realizada y el de la ranura de guardado respectivamente.

Al tener todas las variables que entran al bloque de función de una forma u otra toca definir las variables que únicamente salen del sistema, es decir, las salidas.

-PUSLOS: Variable de tipo entero que define la distancia a la que ha de dirigirse el motor en número de pulsos.

-D_MAX: Variable de tipo entero que define la distancia máxima a la que puede llegar el motor en número de pulsos.

-AUX_USO: Variable entera auxiliar empleada para definir la entrada que se ha de emplear para el posible final de carrera final.

-ALARMA: Señal booleana que alerta de una errónea configuración del motor.

-PILOTO: Señal booleana que informa de la existencia de una configuración guardada en la ranura seleccionada.

-N_ENCODER: Valor entero que informa del número de encoders que se están empleando en esta configuración (1 o 0)

-V: Valor real que envía la velocidad lineal del motor en mm/s.

Para finalizar la declaración de variables tan solo queda definir la única variable interna que emplea el bloque de función:

-PASO: Valor de tipo real que define el avance lineal por pulso de un motor en milímetros.

8.2.4.2. Código

La primera parte por la que empieza el código del bloque de función M_CONFIG, al igual que el MOTOR, es la del control por número de pulsos. En dicha configuración se traducen las coordenadas recibidas y la distancia máxima alcanzable de milímetros a número de pulsos. También se encarga de limpiar los datos que solo se empleen en la configuración de este control para evitar

problemas en el caso de que no se utilice una configuración por número de pulsos.

```
IF ENCODER THEN
    PASO:=AVANCE/NPV;
    PULSOS:=REAL_TO_INT(MM/PASO);

    IF D_MAX_MM>0 THEN
        D_MAX:=REAL_TO_INT(D_MAX_MM/PASO);
        IF D_MAX*PASO>D_MAX_MM THEN
            D_MAX:=D_MAX-1;
        END_IF
    END_IF
ELSE
    PULSOS:=0;
    PASO:=0;
    NPV:=0;
END  IF
```

Figura 59. M_CONFIG control por número de pulsos

En el método de control por tiempo sólo se necesita obtener la velocidad lineal a partid el avance y la velocidad nominal. También se limpia este valor obtenido cuando no se emplea el control al igual que en el control por número de pulsos.

```
IF TIEMPO THEN
    V:= AVANCE*VN/60;
ELSE
    V:=0;
END_IF
```

Figura 60. M_CONFIG control por tiempo

El método de guardar y cargar es prácticamente igual que el de guardar y cargar procesos ya visto en el código del programa principal.

```
IF GUARDAR THEN
    ENCODER_S:=ENCODER;
    NPV_S:=NPV;
    VN_S:=VN;
    FCF_S:=FCF;
    D_MAX_MM_S:=D_MAX_MM;
    AVANCE_S:=AVANCE;
    TIEMPO_S:=TIEMPO;
END_IF

IF CARGAR THEN
    ENCODER:=ENCODER_S;
    NPV:=NPV_S;
    VN:=VN_S;
    FCF:=FCF_S;
    D_MAX_MM:=D_MAX_MM_S;
    AVANCE:=AVANCE_S;
    TIEMPO:=TIEMPO_S;

END_IF
```

Figura 61. Guardar y cargar configuraciones

Para finalizar solo queda la parte donde se envían distintas informaciones. La primera trata sobre la existencia o no de una configuración guardada en la ranura seleccionada. Esto se conoce ya que si en dicha ranura el valor de alguna característica no es 0 se activa el piloto que lo reporta. Otra información que envía la FB trata sobre la configuración errónea. Una configuración errónea es una donde faltan datos por introducir o se usan a la vez métodos de control incompatibles. Para finalizar envía en forma de entero el número de encoders o finales de carrera finales que es están empleando (0 o 1) para la enlazar estos con la entrada al autómata correspondiente.

```
PILOTO:= FCF_S OR  ENCODER_S OR TIEMPO_S OR AVANCE_S>0 OR NPV_S>0 OR  D_MAX_MM_S>0  OR VN_S>0;

ALARMA:=VN=0 OR (NOT ENCODER AND NOT FCF AND NOT TIEMPO) OR (ENCODER AND TIEMPO) AND
 ((ENCODER AND (D_MAX=0 OR NPV=0 OR AVANCE=0 OR VN=0)) OR (TIEMPO AND (AVANCE=0 OR VN=0 OR D_MAX=0)));

IF FCF THEN
    AUX_USO:=1;
ELSE
    AUX_USO:=0;
END_IF

IF NOT ENCODER THEN
    N_ENCODER:=0;
ELSE
    N_ENCODER:=1;
END_IF

IF NOT TIEMPO AND NOT ENCODER THEN
    D_MAX_MM:=0;
    AVANCE:=0;
END_IF
```

Figura 62. Información del estado de la configuración

8.3. SCADA

SCADA, acrónimo de Supervisory Control And Data Acquisition, es un término empleado para referirse a un software para ordenadores que permite controlar y supervisar procesos industriales de manera remota. Para este proyecto se ha requerido crear un SCADA mediante el cual monitorizar todos los procesos que realice el autómata programable a través de un ordenador.

El SCADA se ha creado mediante el mismo programa que la programación (Cdesys) y se divide en tres ventanas bien diferenciadas. La primera ventana y mediante la cual ha de iniciar es la que atañe al modo general. Esto se debe a que el programa ha sido concebido pensando principalmente en el modo específico y su función de almacén inteligente automatizado.

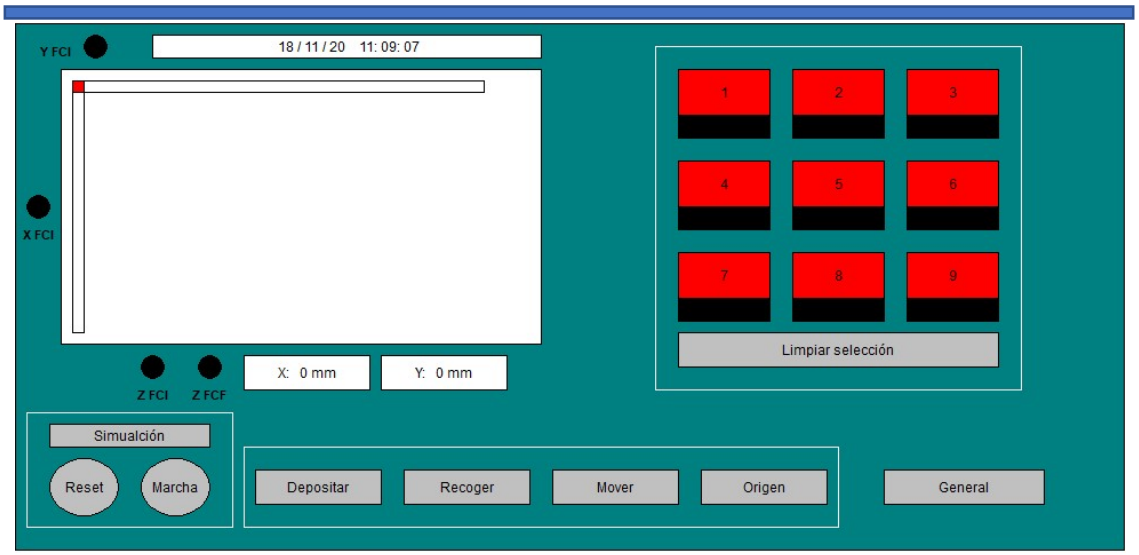


Figura 63. Modo Específico

Como se puede apreciar el modo específico cuenta con las 4 funciones previamente mencionadas anteriormente en la parte inferior. El programa no se puede poner en marcha en el modo específico si no se ha elegido previamente la función que se pretende realizar.

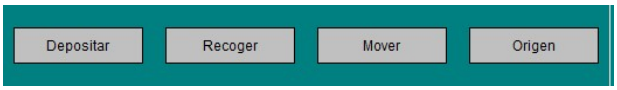


Figura 64. Modo Específico funciones

En la parte inferior izquierda también se puede apreciar los botones de “Marcha” y “Reset” mediante los que se pone en marcha y se pausa el programa con el botón de “Marcha” y se reinicia con el botón de “Reset”. Otra cosa que se puede encontrar en esta parte de la pantalla es el botón “Simulación” que permite realizar simulaciones desde el ordenador sin necesidad de tener presente el PLC

Figura 65. Modo Específico Marcha y Reset y simulación

El último elemento presente en la parte inferior izquierda se encuentra el botón encargado de realizar el cambio de pantalla a la encargada del modo general.

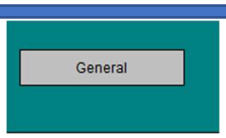


Figura 66. Modo Específico Botón cambio de pantalla modo general

En la parte superior izquierda se encuentra la pantalla donde se muestran los datos principales del programa (posición, finales de carrera, fecha y hora). Los datos de posición se traducen a su vez en un esquema visual donde el eje X es el eje horizontal, el Y es el eje vertical y el cuadrado rojo representa la posición actual del brazo. En cuanto a los pilotos de los finales de carrera estos pueden actuar como botones siempre y cuando se encuentre el programa en modo simulación.

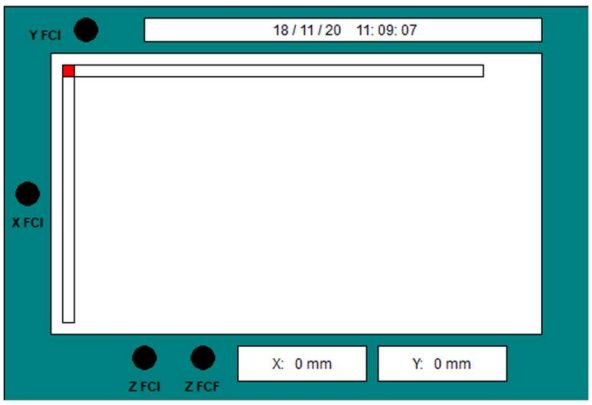


Figura 67. Modo Específico Pantalla de datos

Por último, se encuentra la parte más importante y característica de la pantalla del modo específico, la sección de control de los estantes. En esta sección se pueden apreciar los nueve estantes en forma de botones.

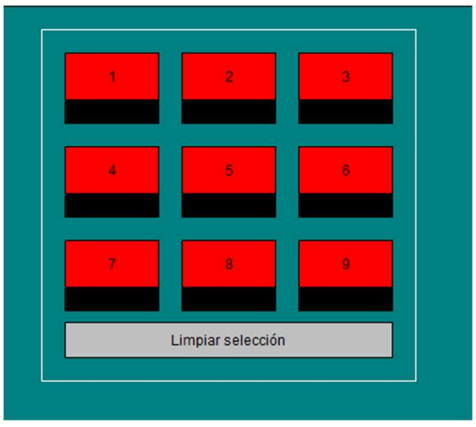


Figura 68. Modo Específico Estantes

Los botones obedecen a un sistema de colores bien diferenciados. El primer color que adopta el botón es el color rojo. Cuando se encuentra en color rojo es cuando no se puede seleccionar dicho estante o bien por el estado del estante en base al modo elegido o por no haber elegido directamente ningún modo.

Figura 69. Botón estante rojo

El siguiente estado presente es el estado de “disponible” representado por el color verde. Cuando se encuentra el botón en color verde significa que dicho estante se encuentra disponible para su selección.

Figura 70. Botón estante verde

Cuando un estante se encuentra seleccionado su botón adopta el color azul.

Figura 71. Botón estante azul

Por último, cuando el botón no se encuentra disponible para seleccionarse no por las causas mencionadas en el rojo sino porque se está efectuando un proceso y ya no se puede alterar, el botón adopta el color naranja.

Figura 72. Botón estante naranja

En el caso de querer limpiar la selección de estantes realizada existe un botón debajo de los mencionados estantes que realiza dicha labor.

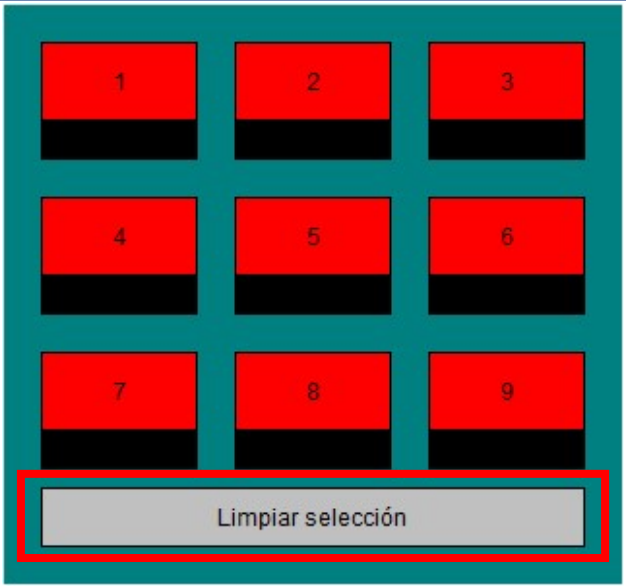


Figura 73. Botón limpiar selección

Para finalizar, debajo de cada botón de los estantes se encuentra un botón con piloto que refleja el estado en el que se encuentra el estante (libre u ocupado). En el caso de estar ocupado el botón lo informa de la siguiente manera:

Figura 74. Estante ocupado

Dichos pilotos cuentan con botón para que puedan ser alterados manualmente sin necesidad de ningún proceso previo si así se requiere.

La segunda pantalla con la que cuenta el SCADA es la que monitoriza el modo general. Esta pantalla es parecida a la del modo específico, pero con notables diferencias.

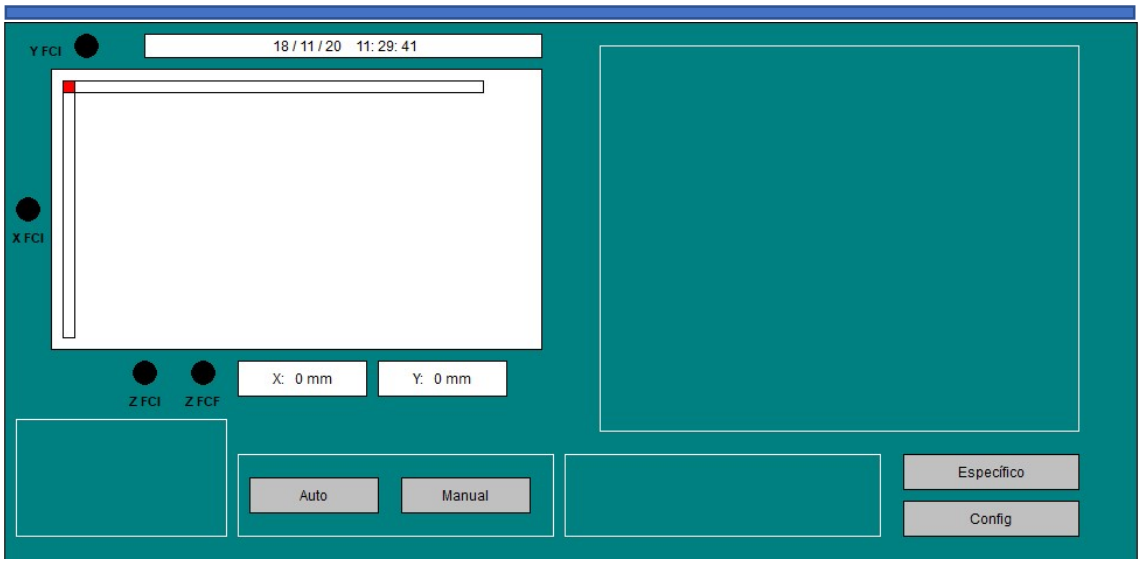


Figura 75. Pantalla modo general

En la pantalla del modo general se vuelve a encontrar en la parte superior izquierda con los mismos elementos que visualizan la información antes mencionada. La diferencia en este caso es que existe una configuración de motores variable así que los datos que se reciben varían en base al tipo de control. Otra consecuencia de las configuraciones variables es la siguiente advertencia cuando estas no son correctas:

Figura 76. Aviso configuración incorrecta

En la parte inferior solo se muestran visibles dos secciones claramente diferenciadas. La primera es donde se selecciona el submodo requerido (manual o automático). Más adelante se incide en estos dos modos.

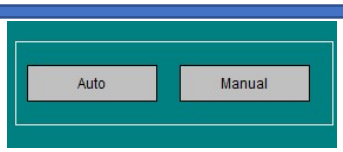


Figura 77. Pantalla modo general modos auto y manual

La segunda cosa visible en un primer momento en la parte inferior son los botones de cambio de pantalla. Su función es cambiar de pantalla a la del modo específico o a la de configuración según el botón pulsado.

Figura 78. Pantalla modo botones cambio de pantalla

Volviendo a los submodos, al seleccionar el botón de auto se habilita el submodo automático dejando visible los siguientes elementos en la pantalla:

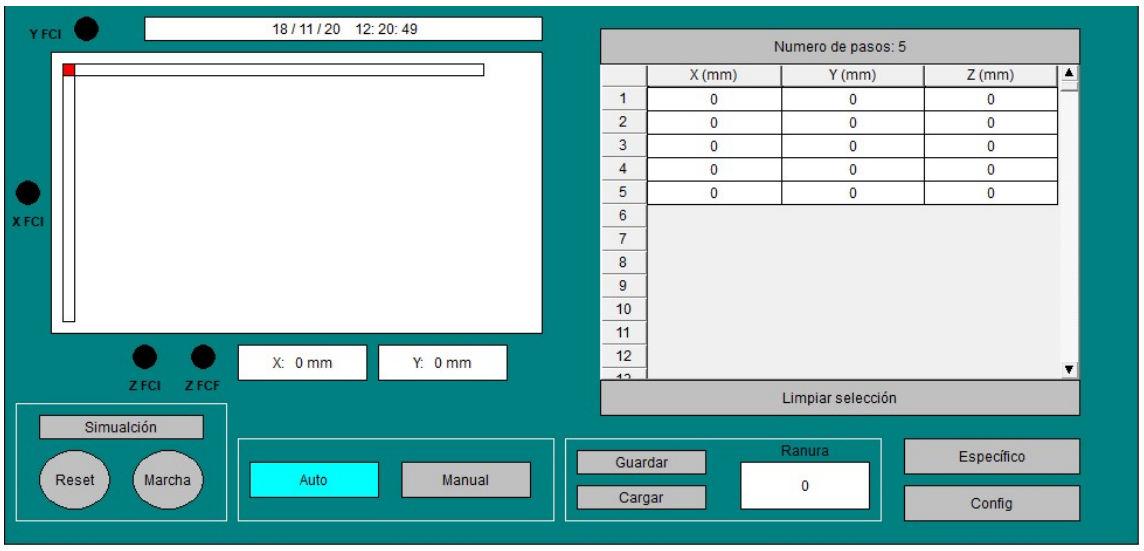


Figura 79. Pantalla modo general submodo atuo

En la parte inferior izquierda se encuentran los mismos botones de “Reset”, “Marcha” y “Simulación” que en el modo específico. Otro elemento aparecido en la barra inferior izquierda es la sección de guardar y cargar los procesos junto al número de ranura donde hacerlo.

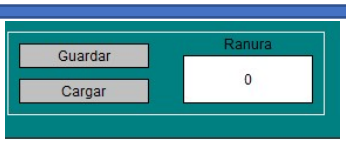


Figura 80. Submodo auto guardar y cargar

La otra parte visible es la referida a la tabla de procesos. En ella se ponen las coordenadas de cada eje requeridas en cada paso. Junto a dicha tabla se encuentra en la parte superior la sección donde introducir el número de pasos (de 1 a 150). La tabla se adapta al número de pasos seleccionado. Justo debajo de la tabla se encuentra un botón mediante el cual es posible hacer una limpieza de la tabla llevando todos los valores introducidos a 0.

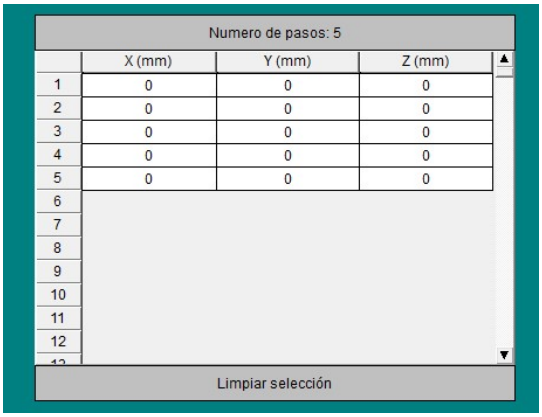


Figura 81. Submodo auto tabla

Por otra parte, al seleccionar el botón correspondiente al submodo manual aparece en la parte izquierda de la pantalla la siguiente botonera mediante la cual es posible controlar manualmente los tres ejes del sistema:

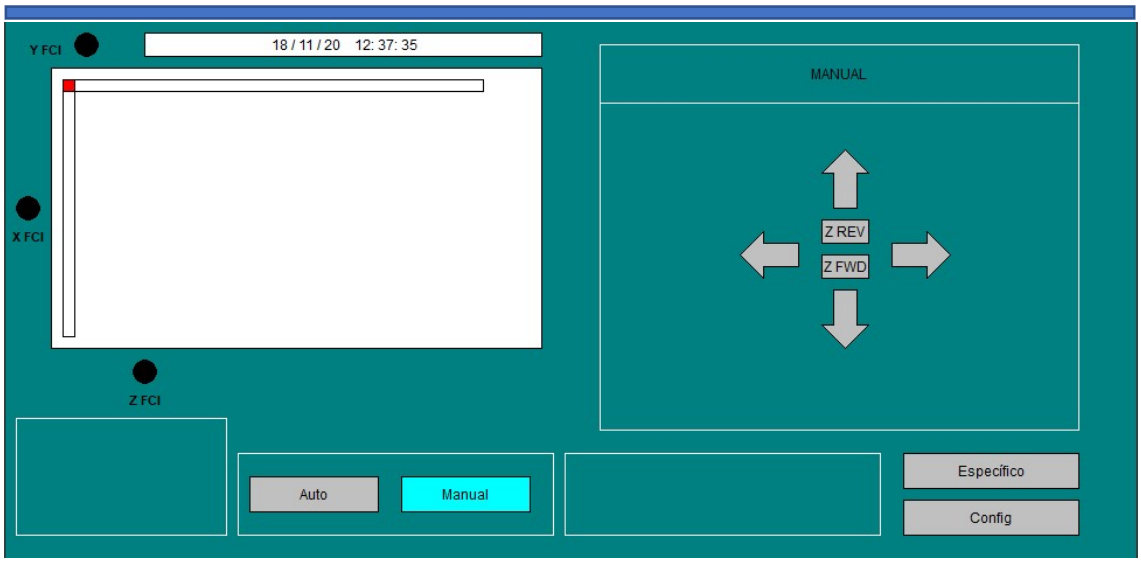


Figura 82. Submodo manual

La última pantalla presente en el SCADA es la correspondiente a la configuración de los motores. Solamente se puede acceder a ella desde el modo general ya que en el modo específico la configuración de los motores viene ya preestablecida. En la pantalla de configuración están en primera instancia los motores con los tres posibles métodos de control.

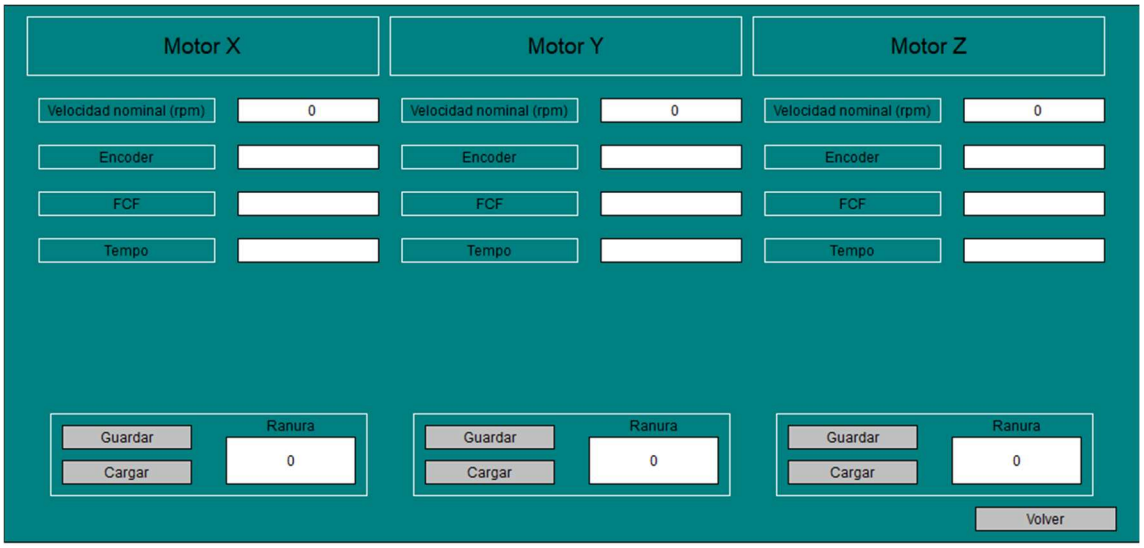


Figura 83. Pantalla de configuración de los motores

En base al método de control seleccionado se visualizan los espacios donde introducir los datos necesarios para cada control.

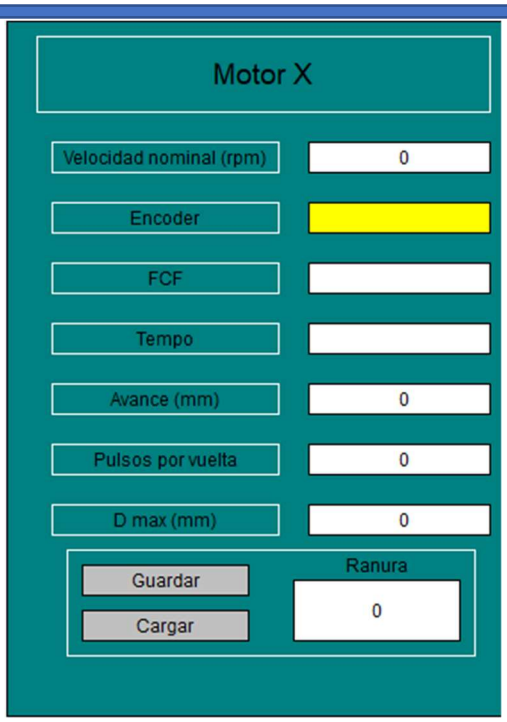


Figura 84. Palla de configuración control por pulsos

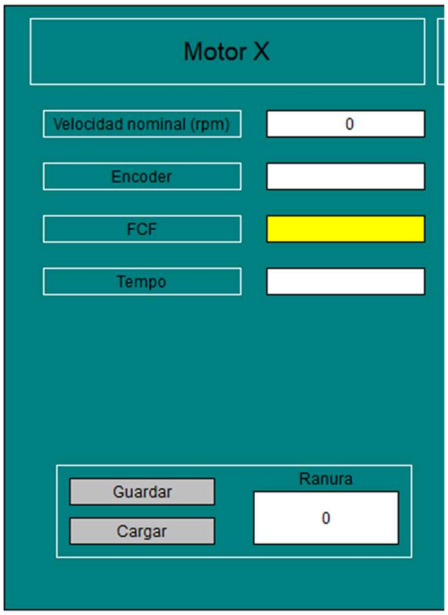


Figura 85. Palla de configuración control por finales de carrera

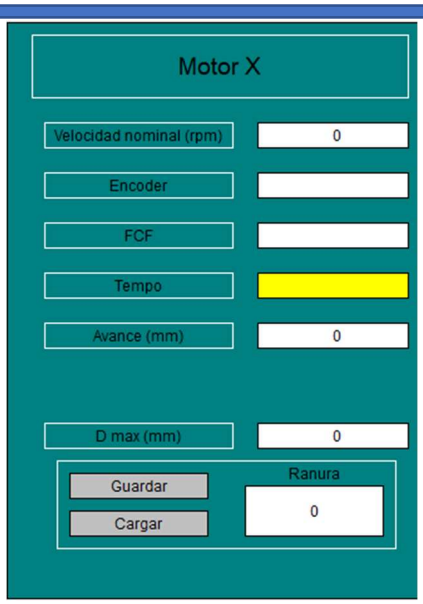


Figura 86. Palla de configuración control por tiempo

Por último, en la parte inferior se encuentran los botones de guardar y cargar configuración para cada motor junto a sus correspondientes ranuras y un botón mediante el cual volver a la pantalla del modo general.

Figura 87. Pantalla de configuración guardar, cargar y volver

9. Conclusión

Tras la realización del trabajo de fin de grado y en conjunto a todo lo aprendido sobre automatización durante el transcurso del grado se concluye que es necesaria una constante formación en el ámbito de los automatismos. Dicha formación ha de estar estrechamente relacionada con práctica para realizar así controles más precisos, completos y eficientes.

Por otra parte, es imprescindible adaptar los PLCs a las nuevas tecnologías con el fin de introducirlos en el mayor número de ámbitos posibles. Para que esta adaptación sea más amplia el software necesario ha de ser más libre con el fin de facilitar el acceso a la gran mayoría. Hoy en día la gran mayoría de softwares no son de libre acceso y como mucho ofrecen una versión capada con la que es casi imposible realizar una programación seria con ella. El software al final no sirve si no existe un autómata que controlar, de manera que los programas sean de acceso libre solo facilita la familiarización con los autómatas funcionando a modo de publicidad para ellos. Por otra parte, que el acceso al software sea tan complicado solo provoca una mayor dificultad a la hora de familiarizarse con los procesos de automatización.

Todo lo mentado desemboca en que en el TFG realizado no se haya tenido acceso por razones de software a un autómata más adecuado para el uso. La falta de contadores rápidos y PWMs necesarios para la realización de un control por número de pulsos eficiente y preciso ha provocado el uso de un control por temporizadores. Este control a la práctica posee muchas imprecisiones al no recibir ningún tipo de retorno por parte del sistema. En el caso de haber tenido acceso al software necesario se hubiera podido emplear un autómata modular con el que satisfacer la necesidad de entradas y de un control preciso.

En cuanto a la maqueta del almacén, es irresponsable recomendarla sin antes recalcar los numerosos fallos mecánicos y de cableado que posee. Esto supone una gran dificultad a la hora de realizar pruebas y de visualizar un control, de manera que al final no acaba de resultar rentable ya que todas las dificultades que genera entorpecen la realización del proyecto. La parte principal de un

proyecto con fines didácticos sobre la automatización de un proceso ha de ser la automatización de este. Es comprensible que existan fallos y dificultades en el sistema a controlar, pero cuando resolver fallos tan grandes (la mayoría no deberían ni estar presentes en una maqueta comprada a una empresa) se vuelve prácticamente la parte principal del proyecto los fallos dejan de ser justificables.

A pesar de todo lo mentado y pese a la pandemia provocada por el COVID-19 se ha realizado el proyecto de la mejor manera que se ha podido con los medios disponibles. El programa es completamente funcional, adaptable y posee funciones extras y mejoras que justifican la realización y entrega de este trabajo de fin de grado.

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

CONTROL Y VISUALIZACIÓN DE UN ALMACÉN ELEVADOR AUTOMATIZADO

Pliego de condiciones

Director
Rubén Puche Panadero

Codirector
Ángel Sapena Bano

Adrián Aparici Micó
adapmi@etsid.upv.es

Índice

1. Condiciones Generales

1.1. Objetivo del Pliego

El objetivo del pliego de condiciones es constituir una extensión del contrato entre propiedad y contratista para la ejecución de un proyecto. Este documento abarca cuatro tipos básicos de condiciones:

- Condiciones técnicas: hacen referencia a los trabajos que hay que realizar, las características y calidad de los materiales, cuidados especiales y detalles concretos a tener presente durante la ejecución, y a los controles y ensayos de calidad preceptivos.

- Condiciones facultativas: hacen referencia a los derechos y obligaciones de las partes y sus representantes en el momento de ejecutar el proyecto.

-Condiciones económicas: hacen referencia a las garantías, la formación de precios, las formas de abono y las indemnizaciones por incumplimiento.

- Condiciones legales: hacen referencia al perfil de contratista, la forma de adjudicación, el tipo de contrato, la obligatoriedad de suscripción de seguros de responsabilidad civil y otros asuntos relacionados.

Con todas estas condiciones dicho documento realiza una descripción detallada de la normativa legal a la que está sujeta el proyecto y la seguridad y calidad tanto del proceso de montaje como de la ejecución de este.

En el caso de que, durante la instalación, montaje, puesta a punto o utilización del sistema apareciera algún contratiempo que no esté reflejado en el documento, es imprescindible que se consulte con el proyectista para la solución de este problema. Debido a que este es un proyecto educacional con el objetivo de la obtención del título de ingeniero técnico por parte del proyectista muchos de los siguientes apartados no tienen sentido ya que se ha realizado el proyecto en base a una maqueta y la fabricación real de la misma no se va a llevar a cabo. Por ello en la redacción del documento se explican las normas que se deberían cumplir en caso de que se llevara a cabo.

1.2. Normativa Vigente

El sistema de control del presente proyecto está directamente conectado a la red de corriente alterna (220v, 50Hz), por ese motivo todos sus elementos, componentes y aparatos se rigen según las normas del Reglamento Electrotécnico de Baja Tensión (RBT) y sus Instrucciones Complementarias. Además, se deben tener en cuenta las normas UNE y DIN. Estas normas son las siguientes:

- ITC-BT-01: Terminología

- ITC-BT-18: instalaciones de puesta a tierra.

- 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.

- ITC-BT-51: instalaciones de sistemas de automatización, gestión técnica de la energía y seguridad.

- UNE-EN IEC 61131-1: información general.

- UNE-EN IEC 61131-2: requisitos y ensayos de los equipos.

-UNE-EN IEC 61131-3: lenguajes de programación.

-UNE-EN IEC 61131-5: comunicaciones.

- UNE-EN IEC 61131-6: seguridad funcional.

-UNE-EN IEC 61131-7: programación en lógica borrosa.

- UNE-EN IEC 61131-9: interfaz digital de comunicación punto a punto para sensores y accionadores pequeños.

UNE-EN IEC 61131-10: formatos de intercambio XML abierto para PLC.

2. Condiciones a satisfacer por los componentes

2.1. Componentes pasivos

2.1.1. Conductores

Todos los conductores utilizados en el sistema de control deben estar aislados mediante PVC para evitar cortocircuitos, derivaciones a tierra y proteger al usuario.

Estos conductores pueden ser de cobre o aluminio en función de la disponibilidad, la sección de cada uno de ellos debe adaptarse a la potencia que circule por el mismo conductor.

Todos los cables provenientes de la fuente de alimentación, así como de los encargados de transmitir las señales han de poder soportar los 24V de alimentación. Todas estas conexiones han de ser mediante cables de 0.25 mm^2. Los cables pueden ser independientes o formar parte de las mangueras

La alimentación, al estar conectada directamente a la red, ha de contar con una manguera de dos conductores de 1.5 mm^2 cada uno.

Además, todos los elementos utilizados y a los que se hace referencia en este apartado deben cumplir el RD 2267/2004: Reglamento de seguridad contra incendios en los establecimientos industriales, por el que los cables deberán ser no propagadores de incendio y con baja emisión de humo y opacidad reducida.

2.3. Elementos electrónicos adicionales

Todos los interruptores y pulsadores que se utilicen han de cumplir con la normativa previamente expuesta.

2.4. Ordenador Personal

El ordenador utilizado debe contar con una CPU, un monitor, un ratón y un teclado. Debido a su utilización en un laboratorio se ha de tomar en cuenta las condiciones en las que este ha de trabajar, principalmente la falta de espacio de

trabajo debido al tamaño de la maqueta. Por ello, se recomienda que el ordenador sea portátil.

El PC utilizado debe cumplir los siguientes requisitos mínimos:

-Sistema operativo Windows 2000 o superior.

-200Mb libres en el disco duro.

-512MB memoria RAM

- Procesador Pentium V, Centrino > 1,8 GHz, Pentium M > 1,0GHz

2.5. El Autómata Programable

El autómata programable esta alimentado con 24V de corriente continua procedentes de una fuente de alimentación conectada a la red eléctrica de 230V en alterna.

3. Condiciones Constructivas

El esquema eléctrico correspondiente a la maqueta ya viene montado de fábrica. Tan solo queda alimentar los finales de carrera mediante los cables ya mencionados previamente. Estos cables recorren los mismos caminos que los ya venidos de fábrica montados.

La longitud de los conductores depende de la distancia entre los elementos conectados. En cuanto a los elementos fijos, la distancia ha de ser justa para que el cable no cuelgue evitando enganches y enredos, mientras que en los elementos móviles se ha de disponer de cierta holgura para posibilitar su movimiento.

4. Pruebas de Funcionamiento

Antes de conectar a la red el proceso se realizan unos ensayos para verificar que todo está colocado y conectado correctamente. A continuación, se van a explicar estos ensayos.

Todas estas pruebas se deben hacer teniendo en cuenta la normativa expuesta en el RBT en la ITC-BT-05: verificaciones e inspecciones.

4.1. Revisión visual y de continuidad

Esta primera revisión consiste en comprobar antes de ninguna conexión que todos los elementos están bien conectados comprobando si las conexiones están bien sujetas y en su sitio correspondiente.

4.2. Pruebas en tensión

Estas pruebas consisten en comprobar que los dispositivos de seguridad funcionan correctamente conectando el sistema a la red de alimentación. También, mediante un sencillo programa de entradas y salidas, se comprueba se estas se conectan y funcionan de manera adecuada.

4.3. Prueba final

En la última prueba se pone en funcionamiento el sistema mediante el control programado. En ella se comprueba primero el modo manual y luego, mediante el modo específico, se realizan las acciones de depositar, mover y recoger. De esta manera se comprueba de manera simulada y luego física que el sistema realiza todos los procesos de manera correcta.

5. Control de calidad

Debido a que todos los elementos empleados son elementos comerciales la tarea de realizar un control de calidad recae en el fabricante. Por ello esta tarea no es competencia ni del contratista ni del proyectista. De todas maneras, se han de comprobar todos los elementos verificando que realizan sus funciones correctamente con el fin de evitar un mal funcionamiento.

Una vez montado todo el sistema, se ha de comprobar que no existen ni cortocircuitos ni derivaciones a tierra. Tras ello se procede a verificar que a la salida de la fuente de alimentación el valor de la tensión es adecuado, en este caso son 24V.

6. Condiciones de ejecución

El montaje de la maqueta consta en asegurarse de que se dispone de todos los elementos, herramientas y materiales necesarios y que siempre se cumplan los ensayos y verificaciones previas vistas en los apartados anteriores.

En el caso de realizar el montaje de la maqueta descrito a lo largo del proyecto se ha de revisar el proceso de ejecución y montaje para mejorar su calidad y productividad.

7. Compra del material

Para comprar los materiales se deben analizar todos los proveedores de los que se disponga, analizando y comparando sus precios y tiempos de espera.

En el caso del montaje realizado en el proyecto se ha intentado gastar lo mínimo disponible en el laboratorio del departamento de ingeniería eléctrica. Por tanto, no se ha realizado ninguna compra directa por parte del alumno ya que todos los aparatos, materiales y herramientas están disponibles en el laboratorio

El mercado y su comercialización en él no está contemplado en un primer momento. En caso de contemplar una posible comercialización se debe estudiar a fondo el mercado. Al proceder las piezas de distintos proveedores hay que prestar mucha atención a los tiempos de espera ya que un posible retraso paraliza todo el proceso de montaje. Con el objetivo de apaliar este problema se debe de contar con un stock permanente que pueda amortiguar los posibles retrasos. El stock se debe someter a un intenso estudio ya que el poseer un stock demasiado grande puede resultar contraproducente y afectar en gran medida al proyecto.

8. Condiciones económicas

8.1. Mejoras del proyecto inicial

En el caso de su fabricación para la comercialización, cualquier intento de mejora o perfeccionamiento realizado por el fabricante debe ser consultado con el proyectista antes de ser realizada. En el caso de que el proyectista estudie la mejora, considere que es técnica y legalmente válida y resulte oportuno introducirla en el sistema se han de renegociar las condiciones del contrato.

8.2. Pagos de los trabajos

En este apartado del pliego de condiciones se establecen las condiciones a seguir por el contratista, el proyectista y el contratante.

- Si el pago se produce entre los 7 días naturales posteriores a la recepción definitiva del prototipo, inclusive con antelación a los mismos, el importe no sufrirá recargo alguno.

- Cuando el pago se efectúa entre 8 y 30 días naturales después de la recepción del producto definitivo por parte del contratante, el producto sufre un recargo del 4% sobre el precio inicial previsto.

- Si se efectúa el pago entre los 31 y 60 días posteriores a la recepción definitiva, este sufre un recargo del 5% sobre el importe establecido.

- Para un aplazamiento del pago entre los 61 y 91 días tras la recepción definitiva, el producto sufre un recargo del 7% sobre el precio final.

- Si se retrasa el pago entre los 91 y 300 días naturales seguidos después de la entrega del producto, el precio de este sufre un recargo del 25% sobre el presupuesto inicial.

- En caso de que el contratante opte por pagar a plazos, debe comunicarlo con anterioridad y establecerse a priori el número de plazos y el intervalo de tiempo

entre cada pago fraccionado, sufriendo cada plazo un recargo en función del tiempo transcurrido y de acuerdo con los puntos anteriores.

- En caso del incumplimiento del pago en el término de tiempo escogido por el cliente, el fabricante tiene derecho a demandarle ante los tribunales.

9. Condiciones legales

9.1. Contrato

El contrato debe realizarse por escrito, cumpliendo todos los requisitos legales y estando firmado por todas las partes implicadas. En el contrato se debe especificar el precio inicial de fabricación del producto y su precio unitario. A este precio se le añaden, en el caso de ser necesario, las tarifas vistas en el anterior apartado. Además, han de aparecer especificadas por escrito todas aquellas cláusulas que deban cumplir alguna de las partes firmantes.

9.2. Adjudicación de la contrata

Los trabajos de fabricación, montaje, instalación, programación y comprobación de los dispositivos que componen el sistema se adjudican siguiendo los criterios que considere oportunos la parte contratante.

9.3. Arbitraje y jurisdicción

Si en el caso de producirse algún desentendimiento entre las dos partes y estas no se pongan por ellas mismo en acuerdo, se nombra a un técnico por cada una de las partes para que estas lleguen a un acuerdo entre ellas. Si este acuerdo no se logra de este modo se ha de recurrir a los tribunales de justicia para que aplicaran la jurisdicción.

9.4. Impuestos

Se exige a la contrata que este al corriente de los pagos de impuestos, tasas y contribuciones necesarios para el desarrollo de sus actividades empresariales. Para la comercialización del producto se ha de añadir un impuesto sobre el precio de este correspondiente al impuesto sobre el valor añadido (I.V.A.), estipulado actualmente en un 21%. En el caso de que este impuesto fuera modificado en un futuro por la administración del estado, se deberá adaptar este impuesto para cumplir con esa modificación.

9.5. Rescisión del contrato

La única causa de rescisión del contrato ha de ser producida por el retraso sin previo aviso de la entrega del producto del contratista al contratante.

10. Condiciones Facultativas

El objetivo de este proyecto es controlar el funcionamiento de una maqueta de almacenamiento inteligente vertical mediante el control de posicionamiento de un brazo giratorio por medio de tres ejes. Dicho control se efectúa desde un PC conectado directamente al autómata programable vía USB.

Esta maqueta ya venía montada y cableada de fábrica, de manera que el trabajo ha consistido en la corrección de los fallos de montaje, el cableado con el autómata, la alimentación y la programación y conexión del método de control.

En caso de que el programa se implante en un sistema real se deben recalcular coordenadas y adaptar las estructuras tanto físicas como de programación al nuevo volumen de proceso.

11. Derechos y deberes del contratista

Debe conocer y cumplir todas las leyes referentes a su actividad empresarial. Los permisos obligatorios para la realización de este proyecto se deben obtener por parte del contratante y no del contratista.

Debe conocer las especificaciones técnicas y normas de seguridad que deben de cumplir todos los elementos del proyecto. Para ello he de cumplir con la normativa establecida en el Reglamento Electrotécnico de Baja Tensión.

Debe de realizar las pruebas necesarias para asegurarse de todos los elementos y el sistema en general funcione correctamente ofreciendo una buena calidad del proceso finalizado.

El contratante no puede reclamar por retrasos producidos en el proceso de fabricación por causas justificadas ajenas al contratista. En el caso de que estos retrasos no se justifiquen el contratante debe abonar un importe del 10% del importe de fabricación.

Los elementos fabricados han de cumplir con los requisitos especificados en el proyecto y solo se pueden ser modificados con la aprobación del proyectista.

El contratante debe facilitar al contratista todos los elementos e información necesaria para una realización eficiente y rápida del proceso de fabricación. Además, debe entregarle por escrito las especificaciones del proyecto y los planos de los elementos y esquemas que componen dicho proyecto.

El trabajo del contratista finaliza cuando después de comprobar que el elemento fabricado funciona correctamente y es puesto en marcha.

No es competencia del proyectista comprobar el cumplimiento de las comprobaciones especificadas por parte de la empresa instaladora.

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

CONTROL Y VISUALIZACIÓN DE UN ALMACÉN ELEVADOR AUTOMATIZADO

Presupuesto

Director
Rubén Puche Panadero

Codirector
Ángel Sapena Bano

Adrián Aparici Micó
adapmi@etsid.upv.es

Índice

1. Introducción

El objetivo de esta parte del proyecto consiste en valorar los costes necesarios para la realización del trabajo de la manera más aproximada y concreta posible. El presupuesto abarca desde los costes materiales hasta los costes humanos necesarios.

El coste de las herramientas necesarias se omite al ser herramientas sencillas que cualquier persona puede tener al alcance y más alguien con las capacidades necesarias para realizar este proyecto de final de carrera (alicates, pelacables, etc.).

2. Presupuesto de ejecución material

El presupuesto de ejecución material o PEM abarca todos los gastos cuantificables no humanos necesarios para llevar a cabo el proceso satisfactoriamente.

2.1. Elementos de control

Este apartado abarca todos los elementos empleados para la realización de la automatización del proceso.

Elemento	Cantidad	Precio ud.	Importe
Almacén elevado automatizado	1	566.10€/ud	566.10€
Autómata PM554	1	289€/ud	289€
Cable programable	1	28.27€/ud	28.27€
Fuente TDK-Lambda	1	26.68€/ud	26.68€
Ordenador empleado (incluye periféricos)	1	709€/ud	709€
		Total	1619.05€

Figura 88. Presupuesto elementos de control

2.2. Software

El software de programación empleado es el Codesys perteneciente a la empresa ABB. Su licencia de uso según su página web oficial se sitúa en los 504€.

2.3. Componentes

En la siguiente tabla se muestran el coste del resto de componentes empleados para el montaje del sistema:

Elemento	Cantidad	Precio Ud.	Importe
Manguera 8x0.25mm	2m	0.95€/m	1.90€
Puntas huecas 0.75mm pack 100	1	2.50€/100 ud.	2.50€
		Total	4.40€

Figura 89. Coste de los componentes

Por consiguiente, el PEM total alcanza la cifra de **2127.45€.**

3. Mano de obra

Para finalizar se hace un cálculo de costes de mano de obra en base a las horas que han sido necesarias para la realización del proyecto. Se estima un coste aproximado total de 11€ la hora (incluyendo seguridad social) obteniendo de esta manera la siguiente tabla:

Trabajo	Horas	Importe
Estudio del sistema	10	110€
Puesta a punto de los dispositivos	5	55€
Programación del PLC	100	1100€
Programación del SCADA	80	880€
Ensayos y pruebas de funcionamiento	50	550€
Resolución de errores	50	550€
Redacción del proyecto	30	330€
Total	325	3575€

Figura 90. Coste mano de obra

4. Gastos generales

Se trata del conjunto de gastos de la empresa durante la realización del proyecto. A diferencia de los gastos previamente vistos estos gastos no se pueden cuantificar con exactitud así que se realiza una estimación aproximada. Estos

gastos abarcan desde desplazamientos hasta consumo energético pasando por el uso de material de oficina. Están estrechamente relacionados con el PEM de manera que se estima en este caso que constituyen un 13% sobre el total del presupuesto de ejecución material. Por ende, los gastos generales alcanzan la cifra de 2127.45*0.13 = 276.57€.

5. Margen de beneficio

Toda empresa tiene el objetivo de obtener un margen de beneficio por los proyectos que le son solicitados. El margen de beneficio depende de diversos factores como pueden ser la oferta, la demanda o el valor objetivo del trabajo realizado durante el proyecto. Pero al final este margen de beneficio no deja de ser un valor subjetivo en gran medida.

En este caso se ha tomado un margen de beneficio del 25% sobre el PEM obteniendo así el siguiente valor:

21217.45*0.25 = 531.86€

6. Total

En este último apartado se recogen en una misma tabla presupuestal todos los gastos mencionados en los apartados anteriores para obtener su suma total y aplicarle los impuestos correspondientes a esta.

A parte, la tabla presupuestal sirve de resumen de los apartados detallando los gastos mediante nombres, modelos y unidades con el fin de facilitar la consulta.

DESCRIPICÓN DEL PROYECTO

Proyecto de final de carrera basado en el control posicional de un modelo de almacén vertical mediante un autómata programable con fines académicos y educativos.

DESCRIPCIÓN	UNIDADES	PRECIO UNITARIO	IMPOERTE
Almacén elevado automatizado 24V Fischer Technik	1	566.10€/ud	566.10€
Cable programable TK503	1	28.27€/ud	28.27€
Autómata programable ABB gama AC500 modelo PM554-TP-ETH	1	289.00€/ud	289.00€
Fuente de alimentación TDK-Lambda DRL 10-100 Series	1	26.68€/ud	26.68€
Manguera de 8 hilos conductores de 0.25mm	2m	0.95€/m	1.90€
Puntas huecas para cables de 0.75mm azules	100	2.50€/100uds	2.50€
Ordenador portátil Dynabook Satellite Pro C50-E-102	1	709.00€/ud	709.00€
Licencia Codesys	1	504.00€/ud	504.00€
Mano de obra	325h	11.00€/h	3575€
Gastos generales	-	-	276.57€
Margen de beneficio	-	-	531.86€
		SUBTOTAL	6510.88€
		IMPUESTOS (IVA 21%)	1367.29€
		TOTAL	7878.17€

Este presupuesto recoge el precio de todo el material empleado para la realización del proyecto sin tener en cuenta las herramientas empleadas en el montaje de todo el sistema.

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

CONTROL Y VISUALIZACIÓN DE UN ALMACÉN ELEVADOR AUTOMATIZADO

Anexo I: Manual de usuario

Director
Rubén Puche Panadero

Codirector
Ángel Sapena Bano

Adrián Aparici Micó
adapmi@etsid.upv.es

Índice

1. Introducción

Un almacén elevado automatizado permite un control del almacenamiento más rápido, preciso y eficiente que uno convencional. Este programa sirve para satisfacer esa necesidad y poder controlar un modelo de almacén mediante el uno de un sistema de tres ejes. El dispositivo de control es perfectamente capaz de realizar acciones de recogida, deposición y movimiento de manera efectiva y rápida.

Al tratarse de un sistema de tres ejes, este se puede adaptar a otros procesos distintos al almacenamiento ofreciendo una configuración de los parámetros de los motores, así como una selección del método de control elegido para cada uno.

Con todo ello se puede establecer un control de posicionamiento mediante tres ejes de manera automática como manual. Todo esto es visualizado y monitorizado desde un sistema SCADA presente en un PC y conectado vía USB al autómata programable donde reside el programa.

2. Modo específico

El modo específico es el relacionado directamente con el control del modelo del almacén vertical inteligente. Este cuenta con una configuración de los motores y su control predeterminada que no se puede alterar. La configuración definida otorga a los ejes X e Y un control por temporizadores mientras que el eje Z consta de un control basado en finales de carrera iniciales y finales. Desde el sistema SCADA se visualiza en la pantalla del ordenador lo siguiente:

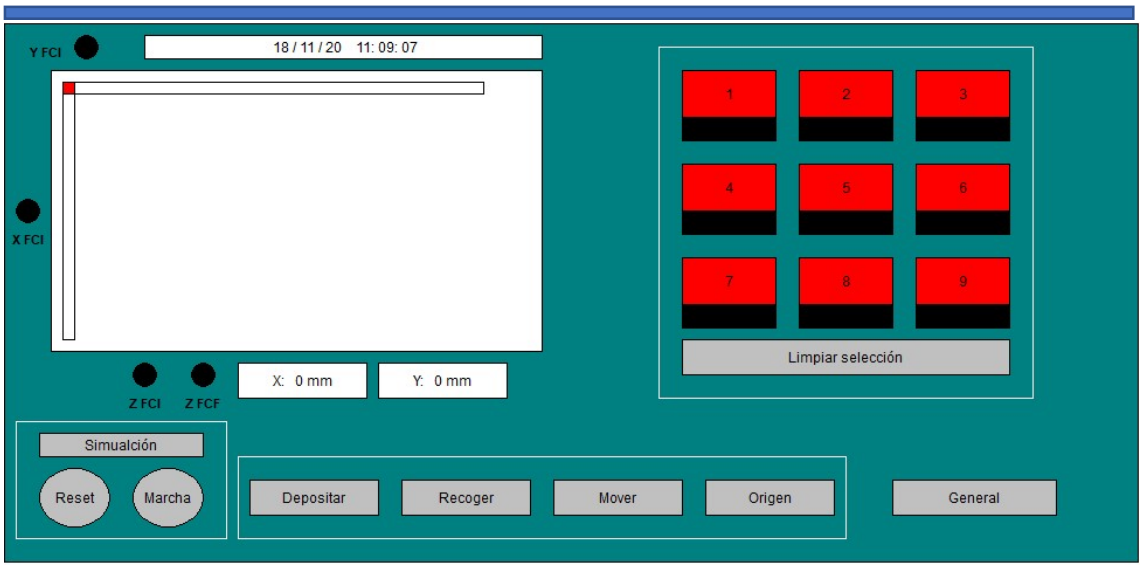


Figura 1.. Modo Específico

Como se puede apreciar el modo específico cuenta con las 4 funciones. La primera función es “depositar”, como bien dice el nombre, deposita un objeto en el estante seleccionado desde la bandeja de entrada. La segunda función es “recoger” que realiza la función opuesta al modo “depositar”. La tercera es el modo “mover” que desplaza un objeto de un estante seleccionado en primer lugar a otro que esté vacío elegido tras seleccionar el estante de origren. Por último, se encuentra la función “origen” que coloca el brazo giratorio en la bandeja de entrada. El programa no se puede poner en marcha en el modo específico si no se ha elegido previamente la función que se pretende realizar.

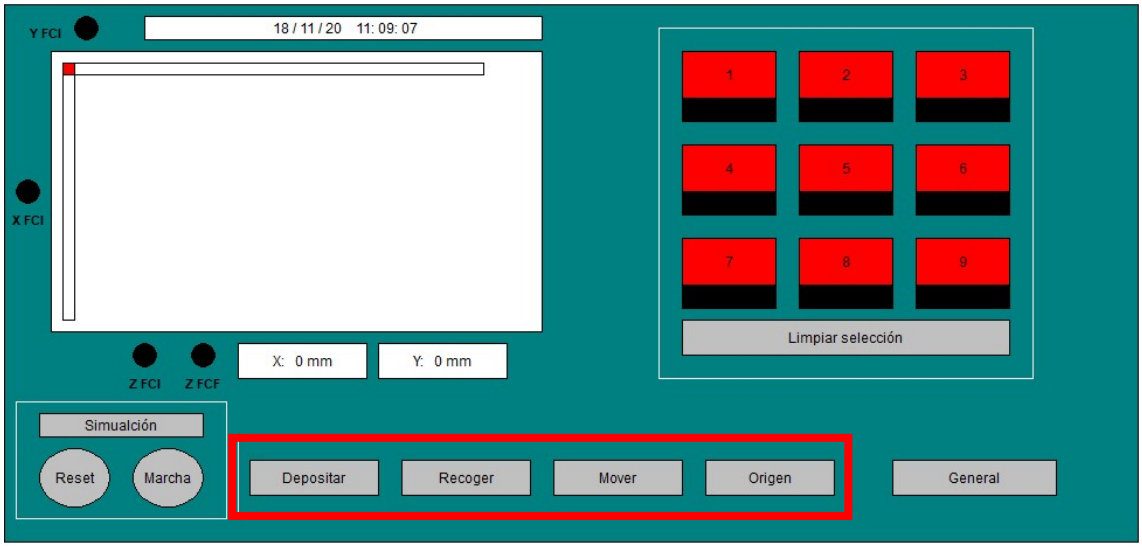


Figura 2. Modo Específico funciones

En la parte inferior izquierda también se puede apreciar los botones de “Marcha” y “Reset” mediante los que se pone en marcha y se pausa el programa con el botón de “Marcha” y se reinicia con el botón de “Reset”. Otra cosa que se puede encontrar en esta parte de la pantalla es el botón “Simulación” que permite realizar simulaciones desde el ordenador sin necesidad de tener presente el PLC

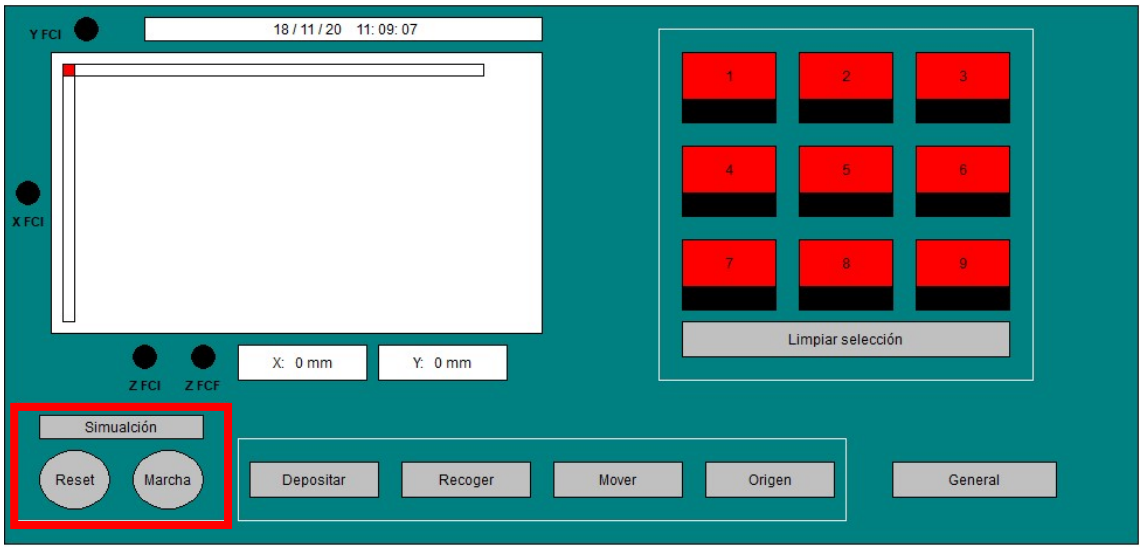


Figura 3. Modo Específico Marcha y Reset y simulación

El último elemento presente en la parte inferior izquierda se encuentra el botón encargado de realizar el cambio de pantalla a la encargada del modo general.

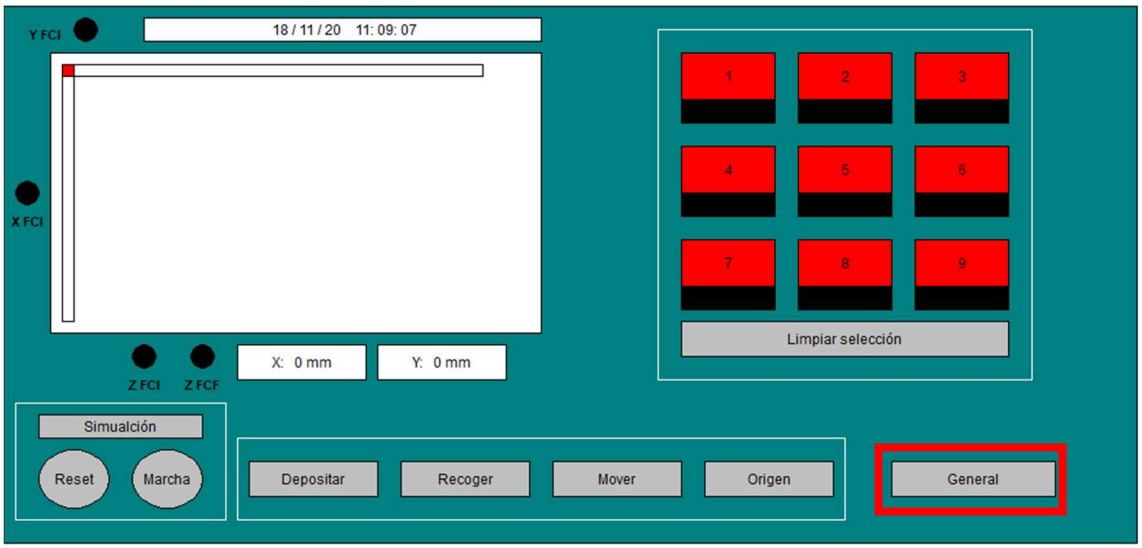


Figura 4. Modo Específico Botón cambio de pantalla modo general

En la parte superior izquierda se encuentra la pantalla donde se muestran los datos principales del programa (posición, finales de carrera, fecha y hora). Los datos de posición se traducen a su vez en un esquema visual donde el eje X es el eje horizontal, el Y es el eje vertical y el cuadrado rojo representa la posición actual del brazo. En cuanto a los pilotos de los finales de carrera (informan de la posición inicial y/o final del eje) estos pueden actuar como botones siempre y cuando se encuentre el programa en modo simulación.

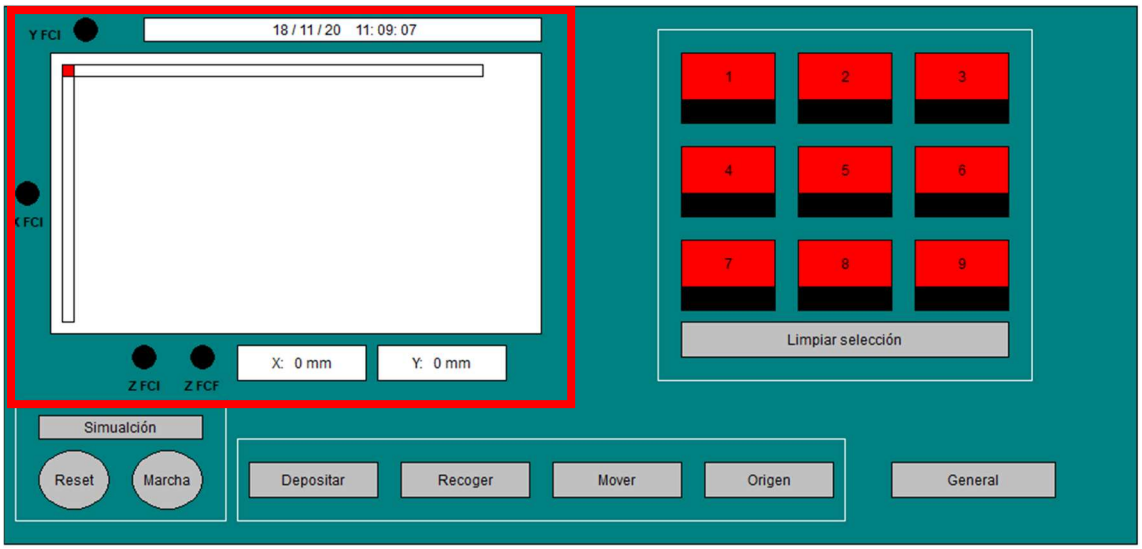


Figura 5. Modo Específico Pantalla de datos

Por último, se encuentra la parte más importante y característica de la pantalla del modo específico, la sección de control de los estantes. En esta sección se pueden apreciar los nueve estantes en forma de botones.

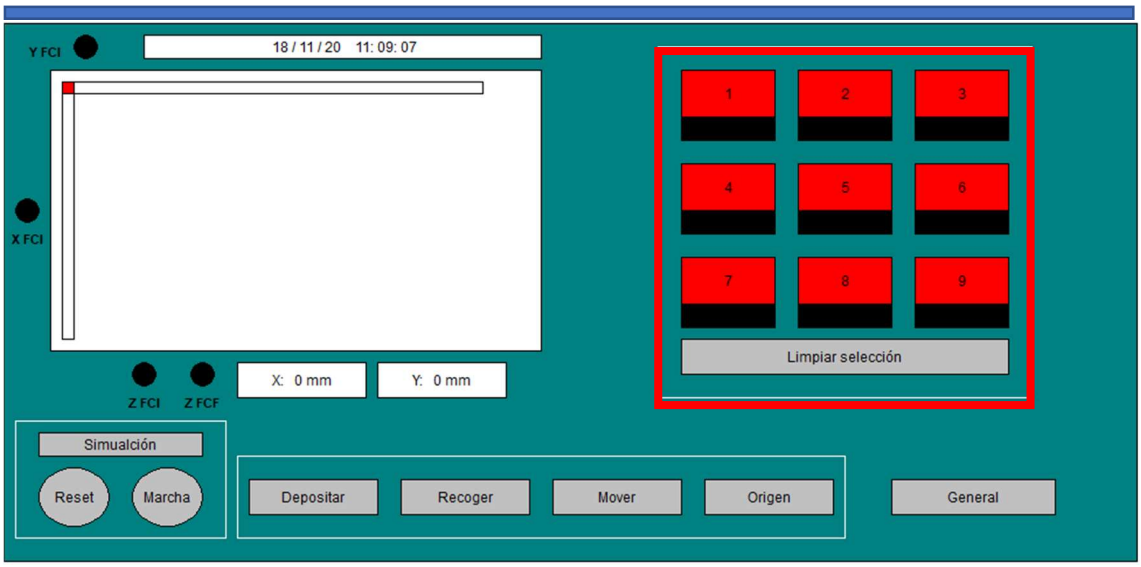


Figura 6. Modo Específico Estantes

Los botones obedecen a un sistema de colores bien diferenciados. El primer color que adopta el botón es el color rojo. Cuando se encuentra en color rojo es cuando no se puede seleccionar dicho estante o bien por el estado del estante en base al modo elegido o por no haber elegido directamente ningún modo.

Figura 7. Botón estante rojo

El siguiente estado presente es el estado de “disponible” representado por el color verde. Cuando se encuentra el botón en color verde significa que dicho estante se encuentra disponible para su selección.

Figura 8. Botón estante verde

Cuando un estante se encuentra seleccionado su botón adopta el color azul.

Figura 9. Botón estante azul

Por último, cuando el botón no se encuentra disponible para seleccionarse no por las causas mencionadas en el rojo sino porque se está efectuando un proceso y ya no se puede alterar, el botón adopta el color naranja.

Figura 10. Botón estante naranja

En el caso de querer limpiar la selección de estantes realizada existe un botón debajo de los mencionados estantes que realiza dicha labor.

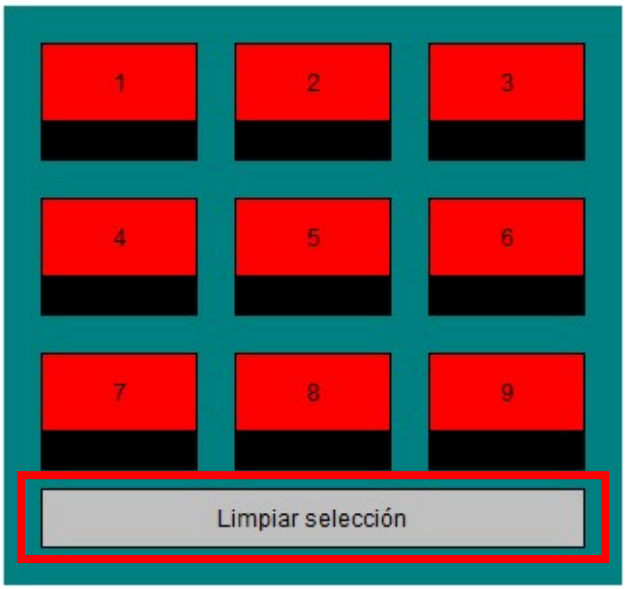


Figura 11. Botón limpiar selección

Para finalizar, debajo de cada botón de los estantes se encuentra un botón con piloto que refleja el estado en el que se encuentra el estante (libre u ocupado). En el caso de estar ocupado el botón lo informa de la siguiente manera:

Figura 12. Estante ocupado

Dichos pilotos cuentan con botón para que puedan ser alterados manualmente sin necesidad de ningún proceso previo si así se requiere.

3. Modo general

La segunda pantalla con la que cuenta el SCADA es la que monitoriza el modo general. El modo general sirve para poder monitorizar mediante el programa procesos ajenos al modelo de almacenamiento del modo específico pero que siguen empleando un sistema de control de tres ejes. Esta pantalla es parecida a la del modo específico, pero con notables diferencias.

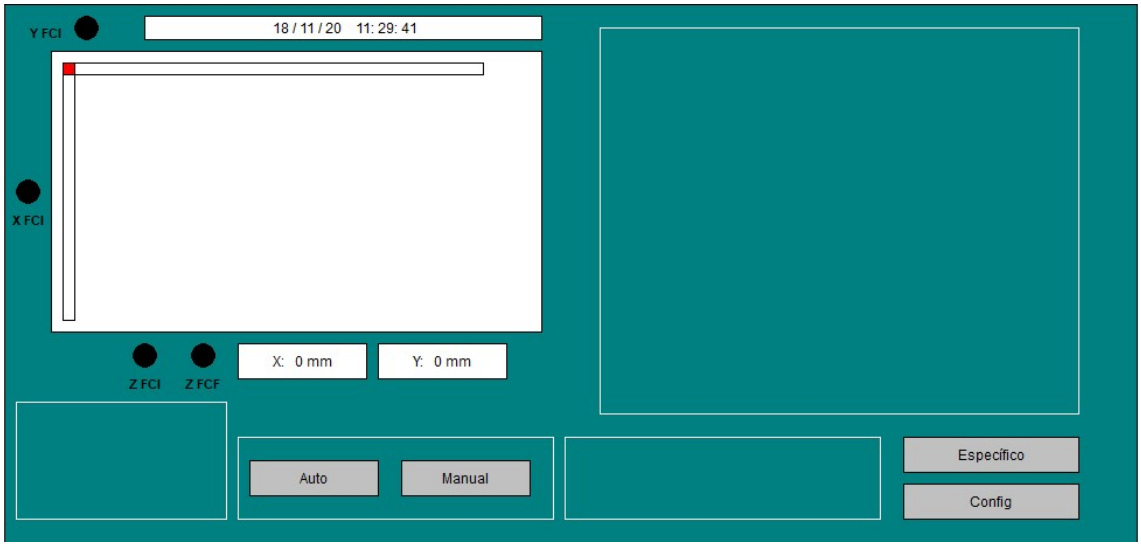


Figura 13. Pantalla modo general

En la pantalla del modo general se vuelve a encontrar en la parte superior izquierda con los mismos elementos que visualizan la información antes mencionada. La diferencia en este caso es que existe una configuración de motores variable así que los datos que se reciben varían en base al tipo de control. Otra consecuencia de las configuraciones variables es la siguiente advertencia cuando estas no son correctas:

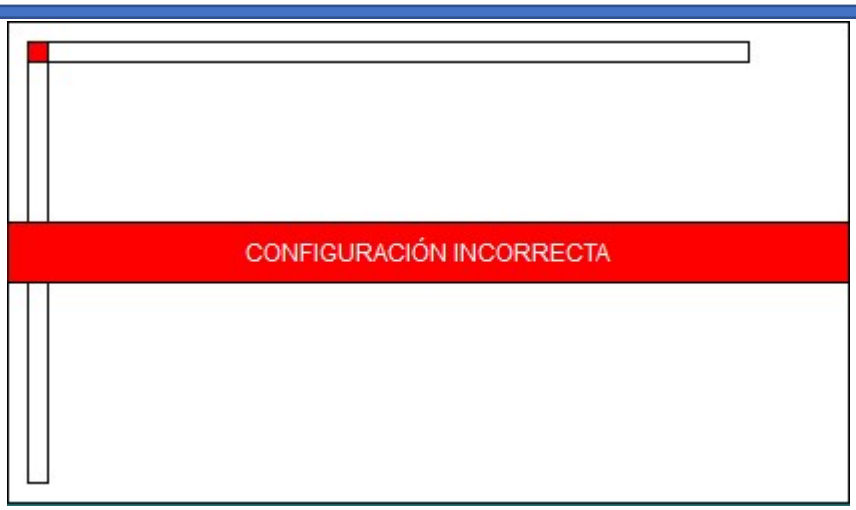


Figura 14. Aviso configuración incorrecta

Una configuración incorrecta es una donde se emplean más controles por pulsos de los del máximo (dos), falta por introducir un método de control, se introducen dos métodos de control incompatibles en el mismo eje o al introducir uno faltan parámetros por configurar.

En la parte inferior solo se muestran visibles dos secciones claramente diferenciadas. La primera es donde se selecciona el submodo requerido (manual o automático). Más adelante se incide en estos dos modos.

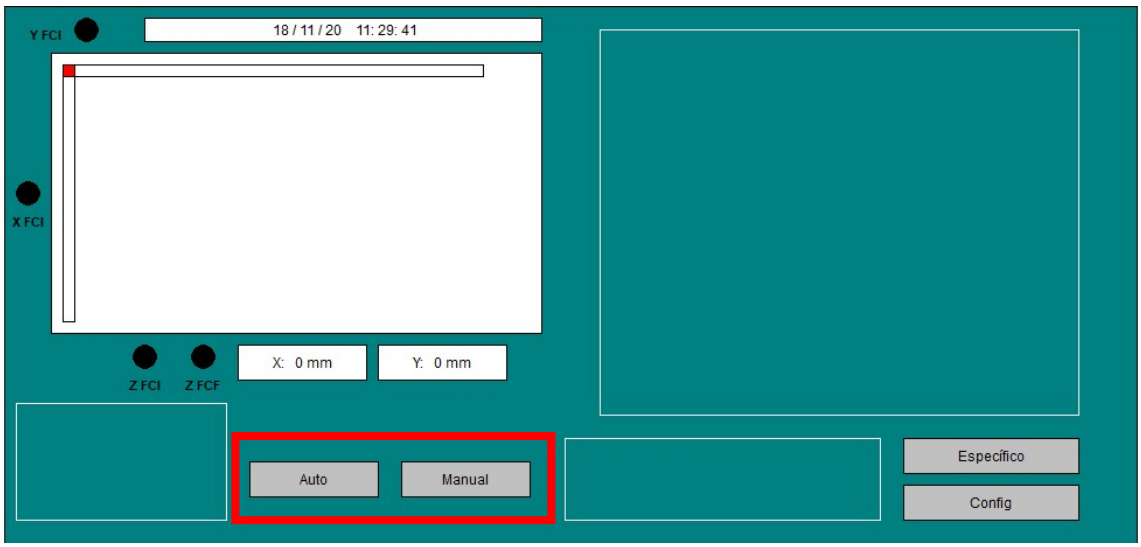


Figura 15. Pantalla modo general modos auto y manual

La segunda cosa visible en un primer momento en la parte inferior son los botones de cambio de pantalla. Su función es cambiar de pantalla a la del modo específico o a la de configuración según el botón pulsado.

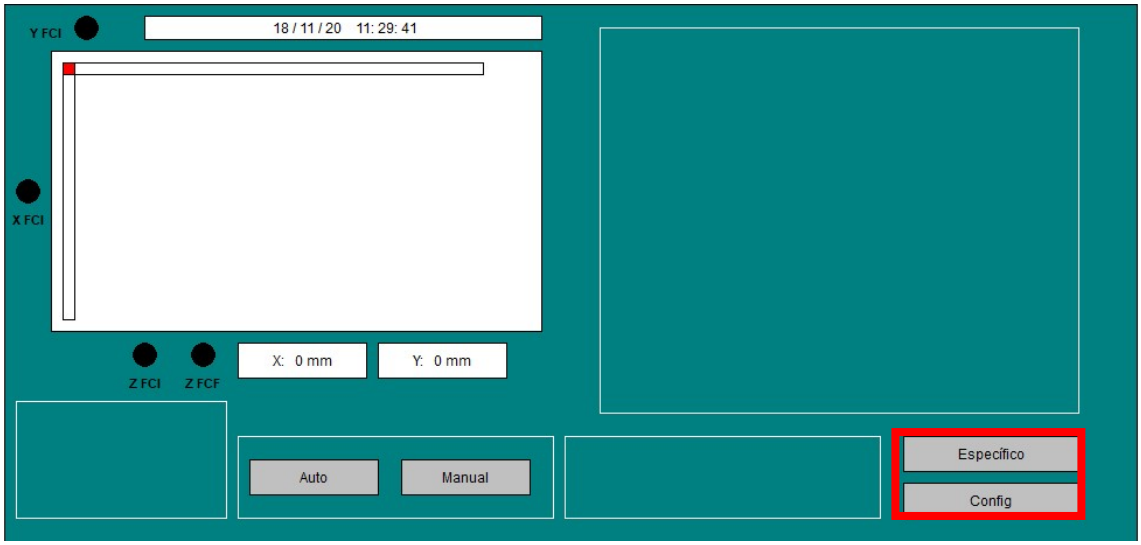


Figura 16. Pantalla modo botones cambio de pantalla

Volviendo a los submodos, al seleccionar el botón de auto se habilita el submodo automático dejando visible los siguientes elementos en la pantalla:

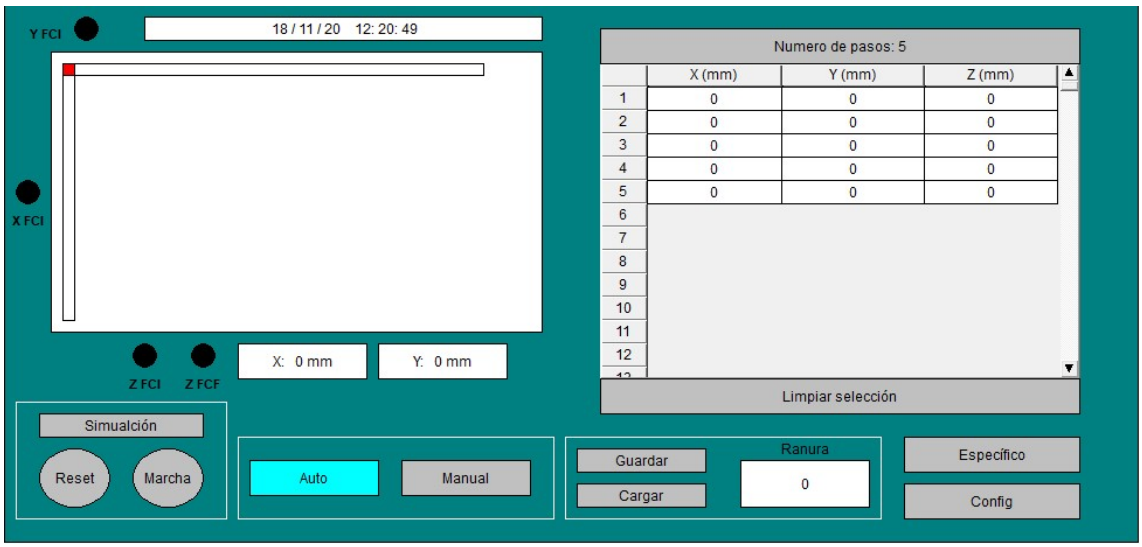


Figura 17. Pantalla modo general submodo atuo

En la parte inferior izquierda se encuentran los mismos botones de “Reset”, “Marcha” y “Simulación” que en el modo específico. Otro elemento aparecido en

la barra inferior izquierda es la sección de guardar y cargar los procesos junto al número de ranura donde hacerlo.

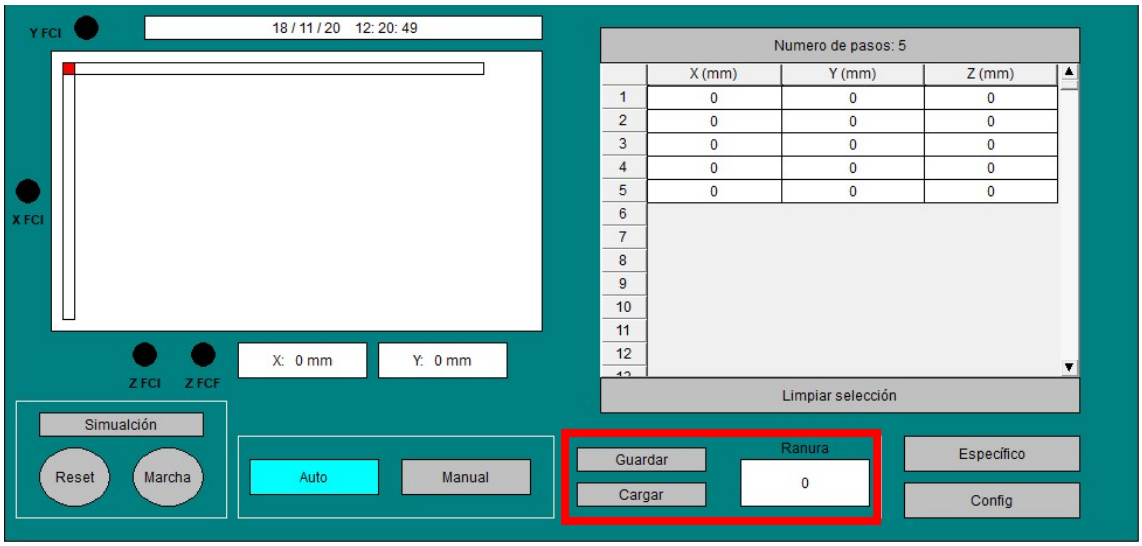


Figura 18. Submodo auto guardar y cargar

La otra parte visible es la referida a la tabla de procesos. En ella se ponen las coordenadas de cada eje requeridas en cada paso. Junto a dicha tabla se encuentra en la parte superior la sección donde introducir el número de pasos (de 1 a 150). En el caso de que exista algún eje sin control de posicionamiento (control por finales de carrera finales), al poner un número superior a 0 se ha de dirigir hasta su posición final y al poner 0 se dirige la posición inicial.

La tabla se adapta al número de pasos seleccionado. Justo debajo de la tabla se encuentra un botón mediante el cual es posible hacer una limpieza de la tabla llevando todos los valores introducidos a 0.

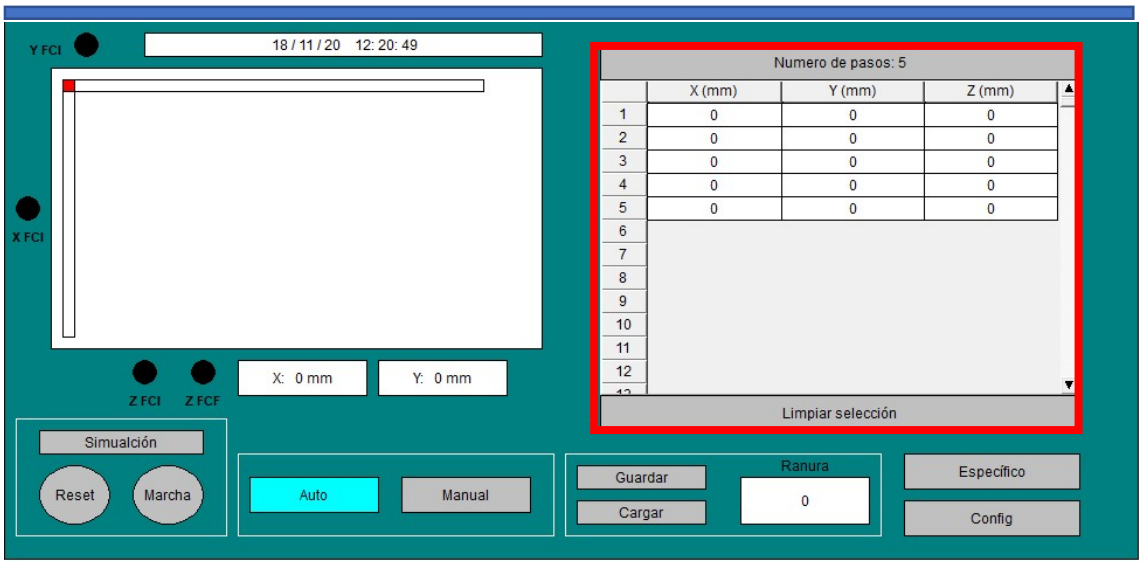


Figura 19. Submodo auto tabla

Por otra parte, al seleccionar el botón correspondiente al submodo manual aparece en la parte izquierda de la pantalla la siguiente botonera mediante la cual es posible controlar manualmente los tres ejes del sistema:

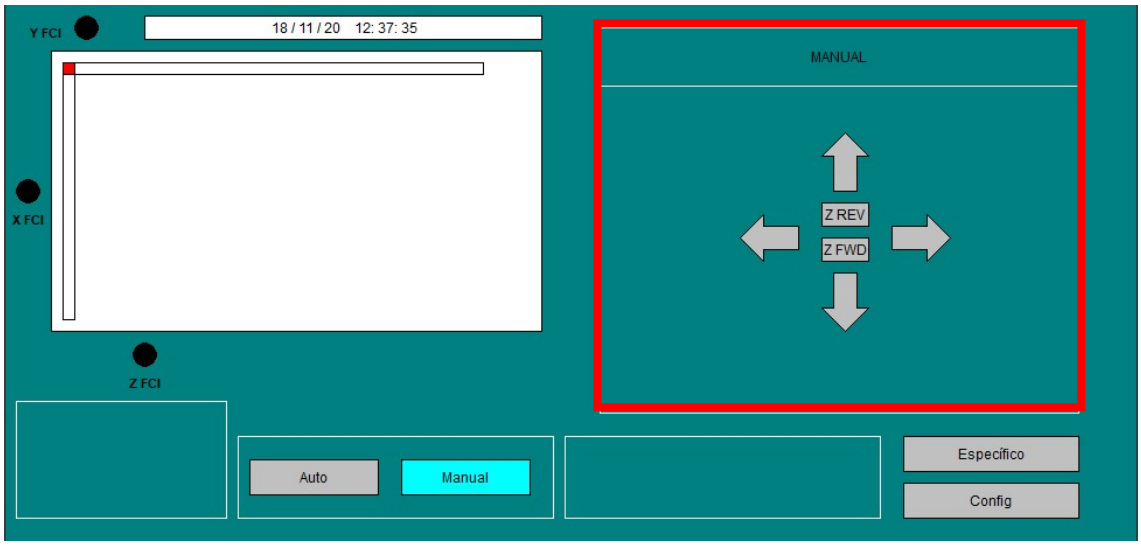


Figura 20. Submodo manual

La última pantalla presente en el SCADA es la correspondiente a la configuración de los motores. Solamente se puede acceder a ella desde el modo general ya que en el modo específico la configuración de los motores viene ya preestablecida. En la pantalla de configuración están en primera instancia los motores con los tres posibles métodos de control.

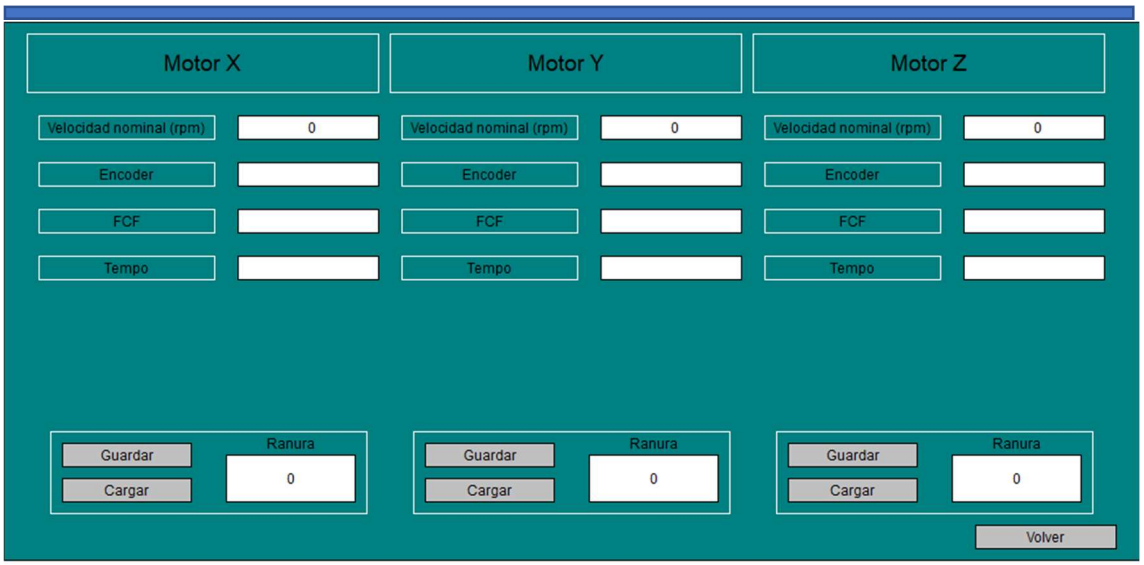


Figura 21. Pantalla de configuración de los motores

En base al método de control seleccionado se visualizan los espacios donde introducir los datos necesarios para cada control.

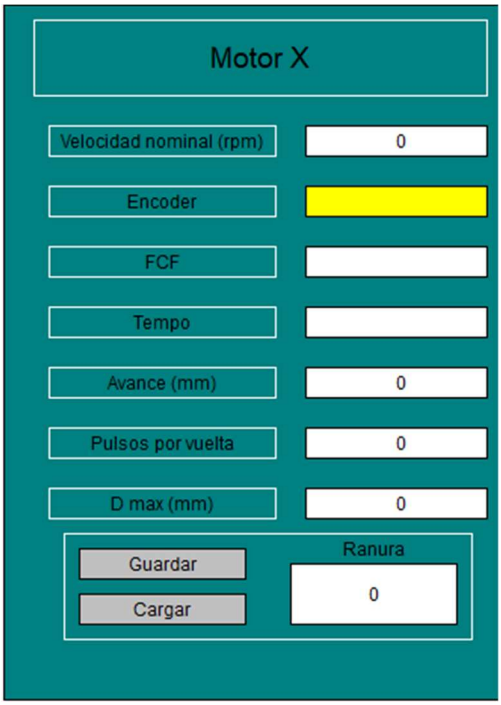


Figura 22. Palla de configuración control por pulsos

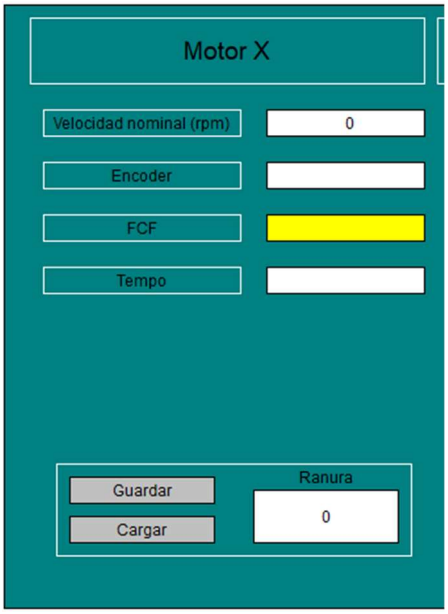


Figura 23. Palla de configuración control por finales de carrera

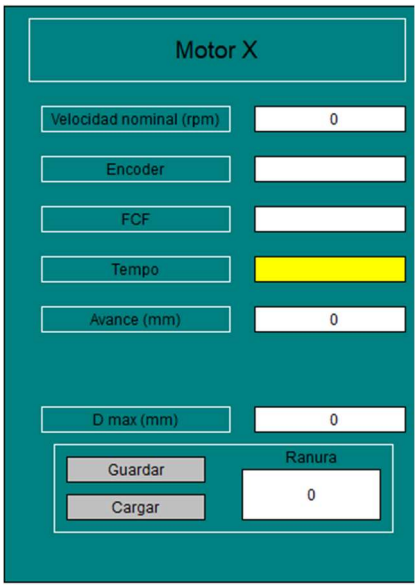


Figura 24. Palla de configuración control por tiempo

Por último, en la parte inferior se encuentran los botones de guardar y cargar configuración para cada motor junto a sus correspondientes ranuras y un botón mediante el cual volver a la pantalla del modo general.

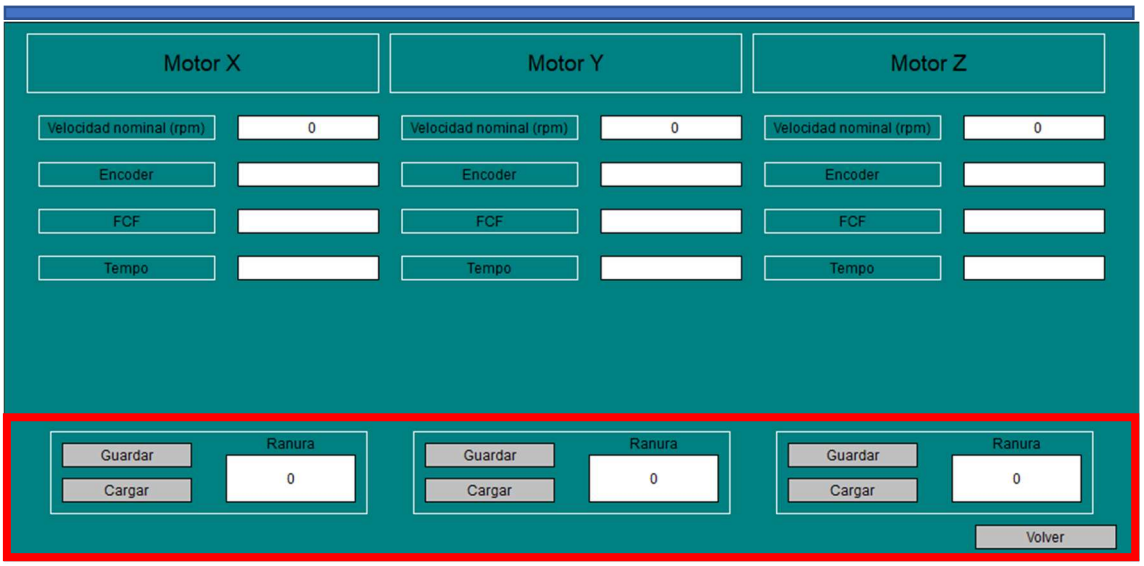


Figura 25. Pantalla de configuración guardar, cargar y volver

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

CONTROL Y VISUALIZACIÓN DE UN ALMACÉN ELEVADOR AUTOMATIZADO

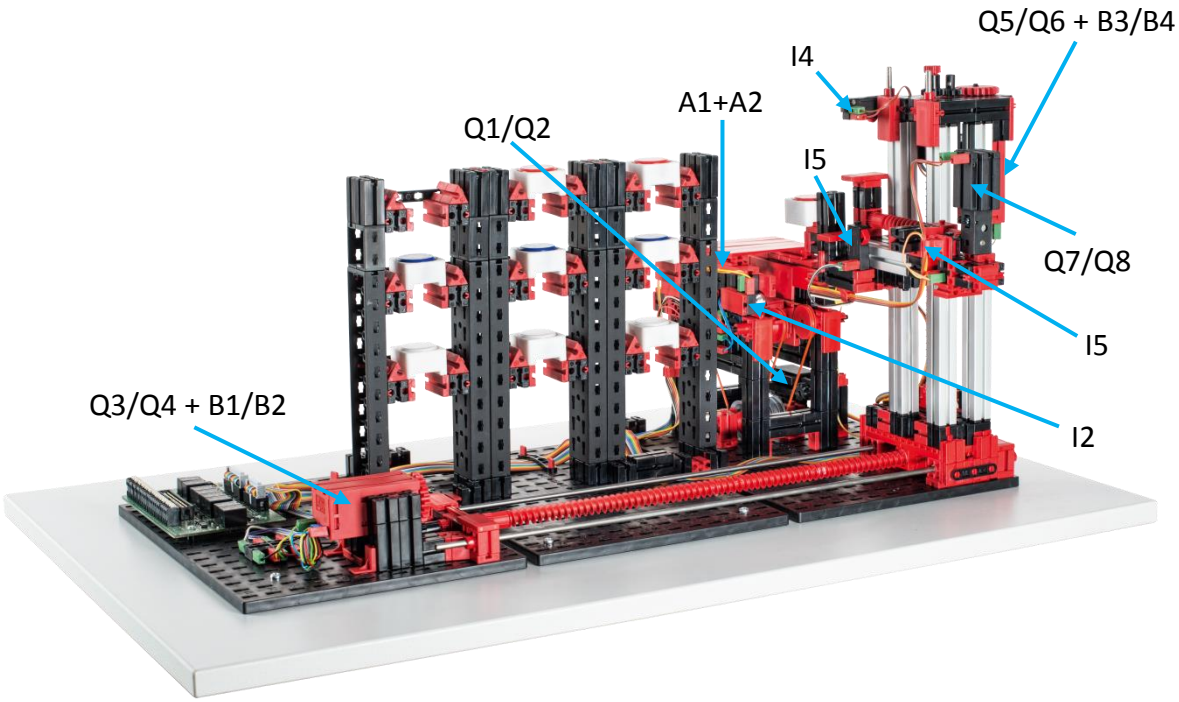
Anexo II: Ficha técnica del almacén elevado automatizado de 24V

Director
Rubén Puche Panadero

Codirector
Ángel Sapena Bano

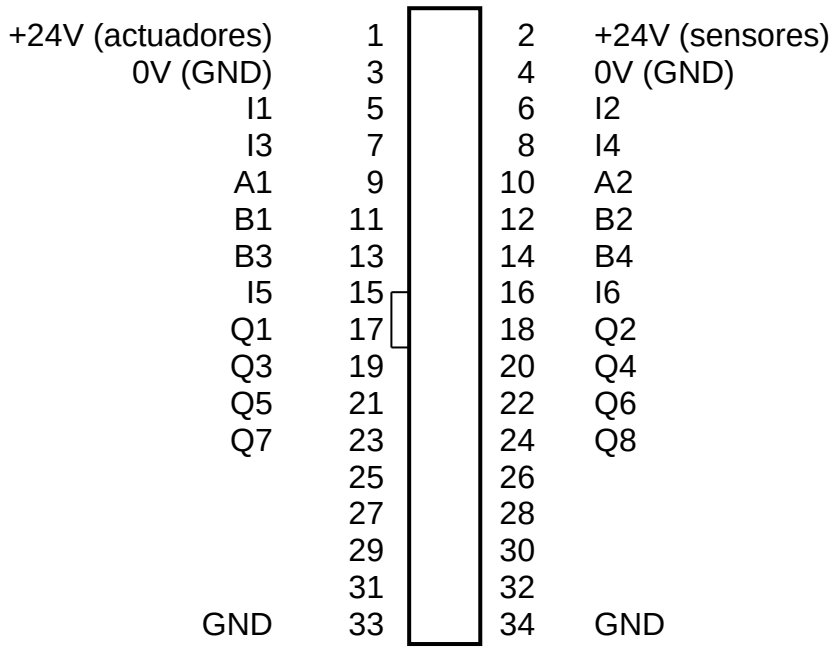
Adrián Aparici Micó
adapmi@etsid.upv.es

536631
Almacén elevado automatizado 24V

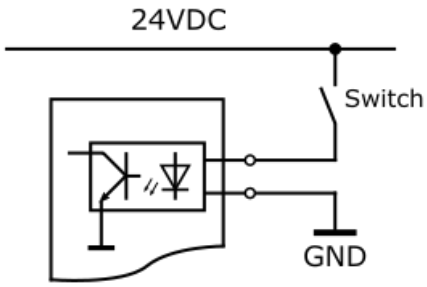
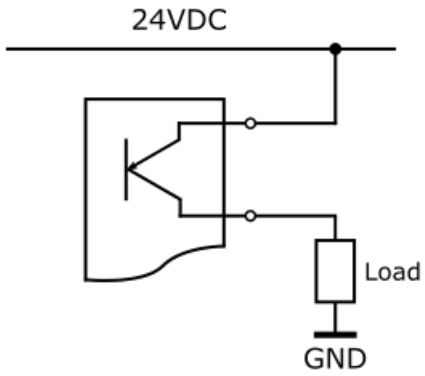


Esquema de asignación del almacén elevado automatizado

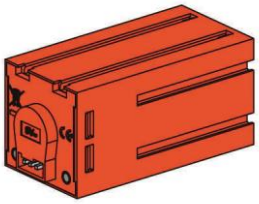
N.º de borne	Función	Entrada/salida
1	Alimentación de corriente (+) actuadores	24 V CC
2	Alimentación de corriente (+) sensores	24 V CC
3	Alimentación de corriente (-)	0 V
4	Alimentación de corriente (-)	0 V
5	Pulsador de referencia horizontal	I1
6	Barrera de luz interior	I2
7	Barrera de luz exterior	I3
8	Pulsador de referencia vertical	I4
9	Sensor de pistas (señal 1, abajo)	A1
10	Sensor de pistas (señal 2, arriba)	A2
11	Codificador horizontal impulso 1	B1
12	Codificador horizontal impulso 2	B2
13	Codificador vertical impulso 1	B3
14	Codificador vertical impulso 2	B4
15	Pulsador de referencia brazo giratorio delante	I5
16	Pulsador de referencia brazo giratorio atrás	I6
17	Motor cinta transportadora hacia delante	Q1 (M1)
18	Motor cinta transportadora hacia atrás	Q2 (M1)
19	Motor horizontal hacia el estante	Q3 (M2)
20	Motor horizontal hacia la cinta transportadora	Q4 (M2)
21	Motor vertical hacia abajo	Q5 (M3)
22	Motor vertical hacia arriba	Q6 (M3)
23	Motor brazo giratorio hacia delante	Q7 (M4)
24	Motor brazo giratorio hacia atrás	Q8 (M4)



configuración de entrada y de salida (PLC)

	entrada	salida
tipo	sinking	sourcing
traspuesta		

Datos técnicos



Motor de codificador:

El accionamiento del dispositivo de control del almacén elevado se realiza con tres motores de codificador. Se trata de máquinas de corriente continua e imanes permanentes que, con ayuda de sensores Hall, posibilitan una medición angular incremental. Los motores de codificador presentan una tensión nominal de 24 V y una potencia máxima de 2,03 W con una velocidad de 214 r.p.m. El consumo de corriente a la máxima potencia es de 320 mA. El engranaje integrado tiene una transmisión de 25:1, es decir que el codificador genera tres impulsos por revolución del árbol motor o 75 impulsos por revolución del árbol de salida del engranaje. Como se registran dos impulsos desfasados, el codificador utilizado puede distinguir en qué sentido gira el motor.

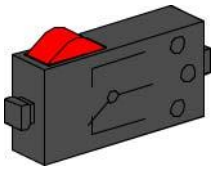
La conexión se realiza mediante un cable de cuatro conductores: el conductor rojo debe conectarse a una salida de 24 V y el conductor verde, a masa. Los cables negro y amarillo transmiten los impulsos (salida push-pull, máx. 1 kHz, máx. 10 mA).

Fototransistor:

Los fototransistores se utilizan en el almacén elevado automatizado como barreras de luz. A la vez se aprovecha que, a partir de una cierta luminosidad, el fototransistor conduce corriente. No obstante, si este umbral de luminosidad no se alcanza, el fototransistor pierde su conductividad. Junto con una lámpara de lente, que se contrapone al fototransistor, este último conduce normalmente corriente y, con ello, se puede utilizar como barrera de luz. Para reducir la influencia de la luz ambiente, se puede usar una cubierta contra luz parásita.

Atención: Al conectar el fototransistor a la alimentación de corriente, debes observar la polaridad correcta. El polo positivo debe conectarse a la marca roja en el fototransistor.

Minipulsador:



En el manipulador de aspiración al vacío se emplean minipulsadores como interruptores de referencia. Al aplicar métodos de medición incremental, un interruptor de referencia sirve para determinar la posición absoluta o el ángulo absoluto. El minipulsador allí utilizado se puede usar tanto como contacto normalmente cerrado como normalmente abierto. Cuando se acciona el pulsador, existe una conexión conductora entre el contacto 1 y el contacto 3, mientras que la conexión entre el contacto 1 y el contacto 2 se interrumpe. En la figura 1 se muestra el esquema de conexiones del minipulsador.

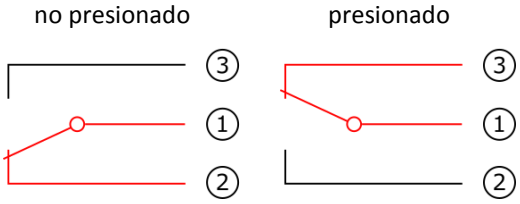
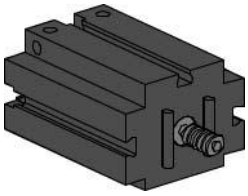


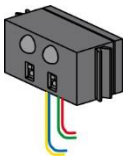
Fig. 1: Esquema de conexiones del minipulsador

Motor S de 24 V:



El brazo giratorio del dispositivo de control del almacén elevado se acciona con un motor S. Este motor compacto es una máquina de corriente continua e imanes permanentes, que se puede utilizar junto con un engranaje reductor insertable. El motor funciona con una tensión nominal de 24 V CC, y el consumo de corriente es de 300 mA como máximo. De ello resultan un par de giro máximo de 5 mNm y una velocidad en vacío de 10700 r.p.m. El engranaje reductor dispone de una transmisión de 64,8:1 y una salida lateral.

Sensor de pistas IR:



El sensor de pistas IR es un sensor digital infrarrojo para detectar una pista negra sobre un fondo blanco a una distancia de 5-30 mm. Está constituido de dos elementos de transmisión y dos de recepción. Las señales se efectúan como salidas push-pull. La conexión se realiza con cuatro cables. El cable rojo debe conectarse a la salida 9 V CC y el cable verde, a masa. Los cables negro y amarillo transmiten las señales. La placa de conexión asume la transformación de la tensión y la adaptación de nivel de 24 V CC a 9 V CC.

¿Qué es un almacén elevado?

Un almacén elevado es un almacén ahorrador de superficie, que permite colocar o retirar mercancías con la asistencia de ordenadores. En la mayoría de los casos, los almacenes elevados están diseñados como estanterías de paletas. Esta estandarización posibilita un alto grado de automatización y la conexión a un sistema de planificación de recursos empresariales. Los almacenes elevados se caracterizan por un gran aprovechamiento del espacio y una alta necesidad de inversión.

La colocación y retirada de mercancías se realiza con dispositivos de control de estantes, que se mueven por pasillos situados entre dos filas de estantes. Este sector es parte de la zona previa, donde también se realiza la identificación de la mercancía. Mediante la técnica de transporte (p. ej. transportadores de cadena, vías de rodillos o transportadores verticales), la mercancía se pone a disposición y se traspasa a los dispositivos de control de estantes. Si los dispositivos de control de estantes están automatizados, ninguna persona debe estar presente en ese sector. En el caso de un almacén elevado automatizado, la mercancía se pone a disposición con ayuda de una cinta transportadora. La mercancía se identifica mediante un código de barras, que se lee utilizando un sensor de pistas.

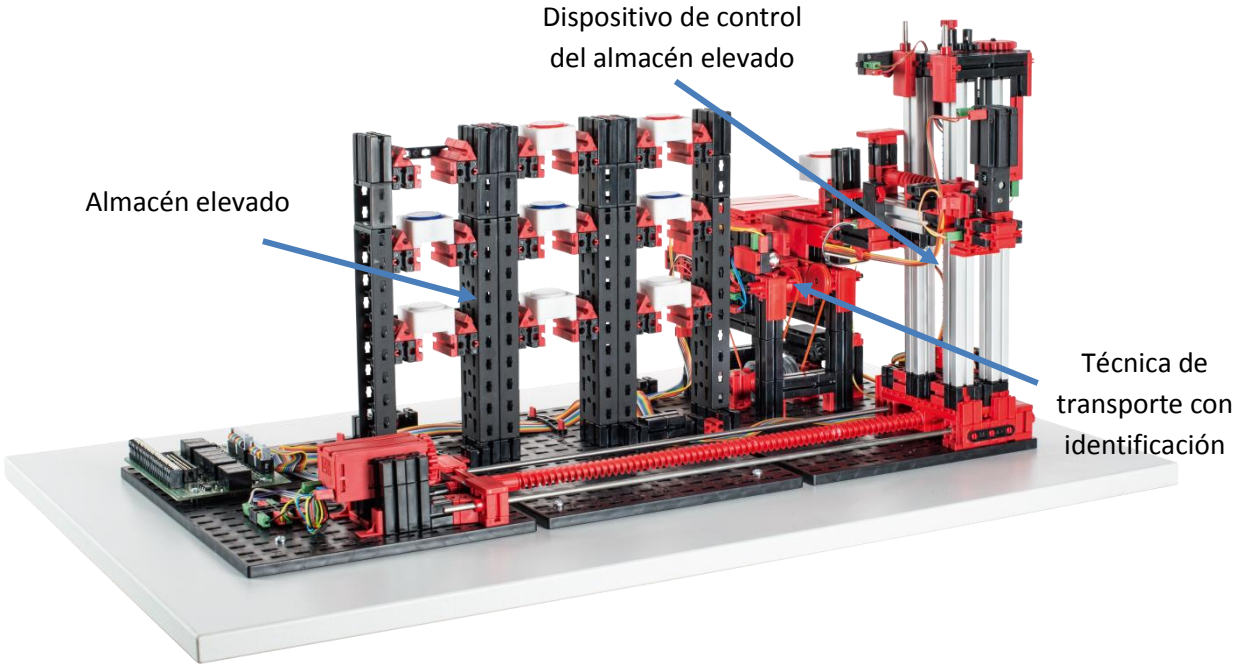


Fig. 1: Sectores del almacén elevado

La colocación en almacén se realiza con frecuencia según el principio de almacenamiento dinámico. En este caso, la asignación entre la posición de almacenamiento y la mercancía se abandona, lo que lleva a que la mercancía a almacenar se deposite a discreción en una posición desocupada. Con ello se espera optimizar los recorridos. El sistema de gestión de stocks guarda la posición de la mercancía depositada, poniéndola así a disposición. Una identificación (parcialmente) automatizada de la mercancía mediante chips RFID o códigos de barras mayormente en un punto central —el llamado punto de identificación— y una estandarización de los puestos de almacenamiento (iguales medidas exteriores, igual peso de las piezas) son imprescindibles. La estrategia ABC, con la que se divide el almacén en tres zonas situadas a distinta distancia del puesto de colocación y retirada, sirve para continuar optimizando los recorridos. La mercancía frecuentemente requerida se coloca en la

llamada zona A, que se encuentra en la proximidad inmediata del puesto de colocación y retirada. La mercancía raras veces requerida se almacena, por consiguiente, en la llamada zona C, situada a mayor distancia del puesto de colocación y retirada.

En el caso del almacén elevado automatizado, el almacenamiento estático y dinámico se pueden demostrar gráficamente. En el almacenamiento estático, a cada fila se le asigna, por ejemplo, un color. Así, la fila superior tiene asignado el color blanco, la fila del medio el color rojo y la fila inferior el color azul. La reposición de cada fila de color se efectúa desde la posición de almacenamiento más cercana a la zona previa hasta la posición de almacenamiento más alejada de la zona previa. En el almacenamiento dinámico, la asignación fija entre las filas de estantes y los colores se suprime. Esto conlleva que el dispositivo de control del almacén elevado deposite la pieza a discreción en una posición desocupada. La asignación entre un color y la posición de almacenamiento seleccionada debe guardarse en el sistema de gestión de estantes.

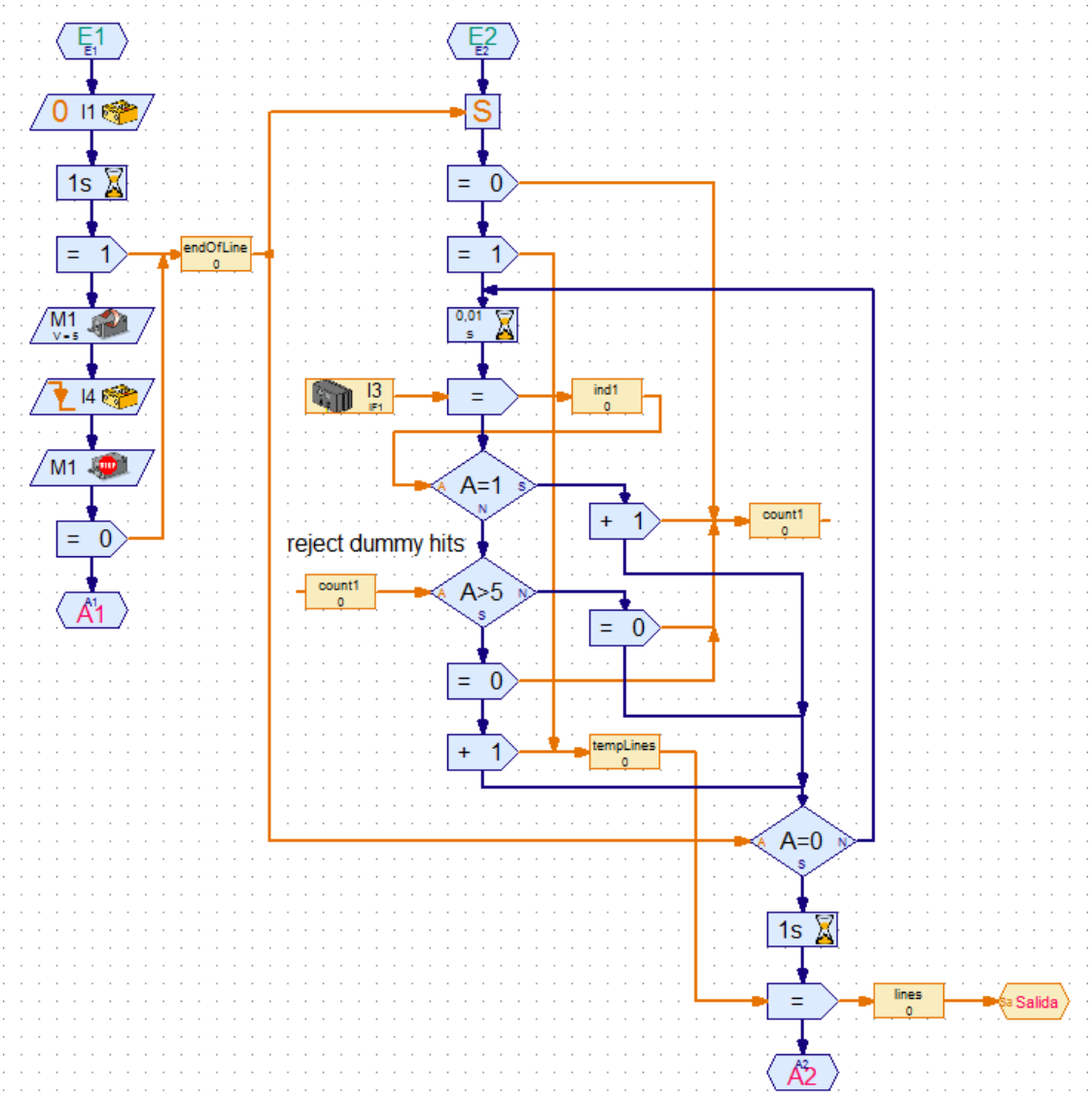


Fig. 2: Algoritmo para la identificación de códigos de barras en ROBO Pro

La identificación de la pieza en el almacén elevado automatizado se realiza con ayuda de un simple código de barras. Para ello, los portapiezas se proveen de un código al que se le asignan los colores blanco, rojo y azul. Dicho código se analiza con un sensor de pistas, que registra las diferencias claro-oscuro y las evalúa en función de la anchura como marcación o como reflexión. Las reflexiones aparecen a menudo en los bordes de los portapiezas, y deben descartarse para eliminar interpretaciones erróneas. La diferenciación se realiza por la anchura de las zonas oscuras o por el número de pasos de tiempo consecutivos que se evalúan como oscuros. Las zonas oscuras, que abarcan más de cinco pasos de tiempo consecutivos, se evalúan como marcaciones. En la figura 2 se muestra la implementación de este algoritmo para la identificación de códigos de barras en ROBO Pro. La anchura mínima definida en este proceso limita el número de patrones a diferenciar que pueden utilizarse para identificar portapiezas, pero es suficiente para codificar los tres colores.

Fig. 3: Codificación cromática

En la figura 3 se muestra la asignación entre los códigos utilizados y los colores respectivos. Estas marcaciones se colocan en la cara del portapiezas que señala hacia el sensor de pistas, y permiten así asignar un portapiezas a una pieza de color.

Calibración

Las posiciones que alcanza el dispositivo de control del almacén elevado automatizado están colocadas en el subprograma "Calibración". Estas posiciones describen la ubicación de los compartimentos del almacén elevado, así como la ubicación de la cinta transportadora respecto de la posición cero. Solo se tienen en cuenta las posiciones x e y que se alcanzan con los motores de codificador. Las posiciones z, a las que se llega con un motor S, se alcanzan con ayuda de pulsadores y, por tanto, no requieren calibración. Las diez posiciones (nueve lugares de almacenamiento + cinta transportadora) se describen con la ayuda de ocho variables. Para los lugares de almacenamiento se colocan a tal fin los niveles (tres posiciones x) y filas de estantes (tres posiciones y) como variables. En el caso de la cinta transportadora, se coloca tanto la posición x como la y.

Tabla 1: Posiciones preestablecidas y modificadas del almacén elevado

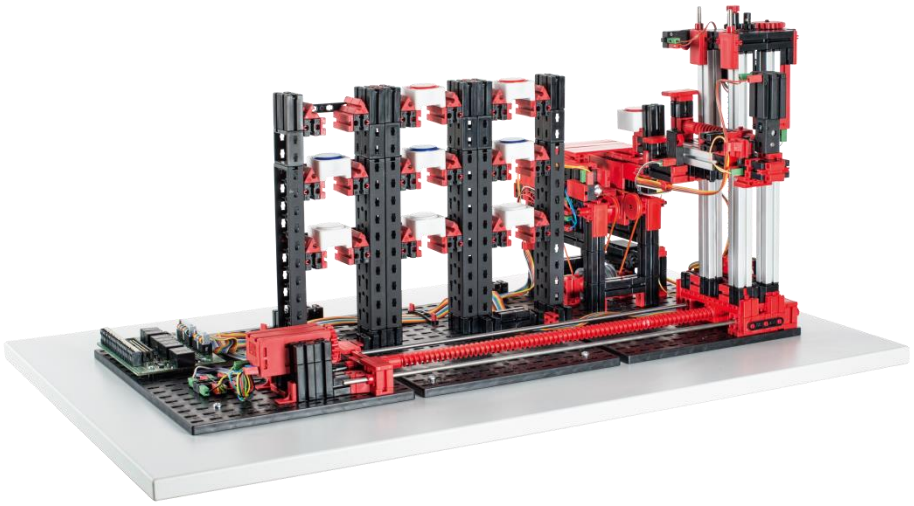
Posición	**Nombre de la variable**	**Valor preestablecido**	**Valor adaptado**
Cinta transportadora (posición x)	X_0	10	
Cinta transportadora (posición y)	Y_0	729	
Primera fila	X_1	760	
Segunda fila	X_2	1365	
Tercera fila	X_3	1972	
Nivel superior	Y_1	85	
Nivel medio	Y_2	460	
Nivel inferior	Y_3	850	

Almacén elevado: definición y propiedades

¿Qué es un almacén elevado?

¿De qué trata la zona previa?

Marque los sectores fundamentales del almacén elevado automatizado y nómbrelos.



Almacén elevado: definición y propiedades SOLUCIÓN

¿Qué es un almacén elevado?

Un almacén elevado es un almacén ahorrador de superficie, que permite colocar o retirar mercancías

con la asistencia de ordenadores y que, con su alta estandarización, posibilita un alto

grado de automatización.

¿De qué trata la zona previa?

La zona previa es el sector de un almacén elevado, en el que la mercancía se pone a disposición y

se identifica. La zona previa comprende también el dispositivo de control del almacén elevado y la

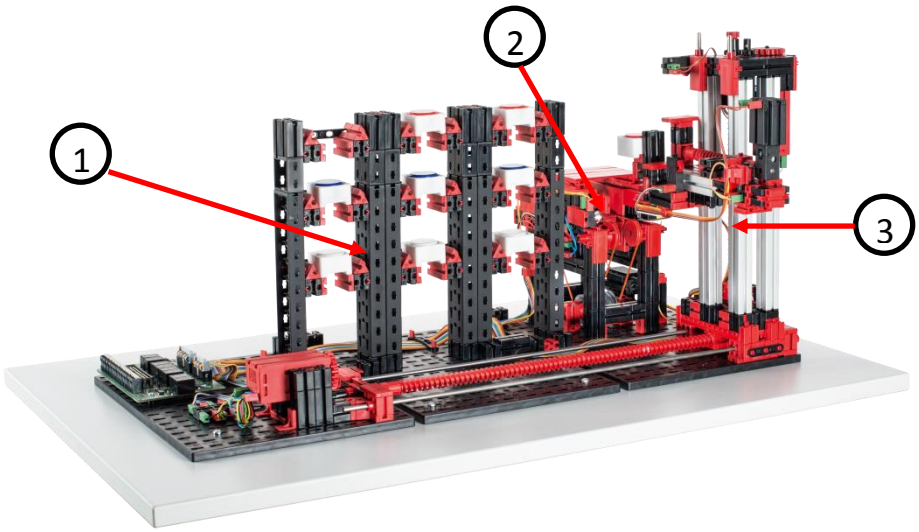
técnica de transporte.

Marque los sectores fundamentales del almacén elevado automatizado y nómbrelos.

1 Almacén elevado

2 Técnica de transporte con identificación

3 Dispositivo de control del almacén elevado



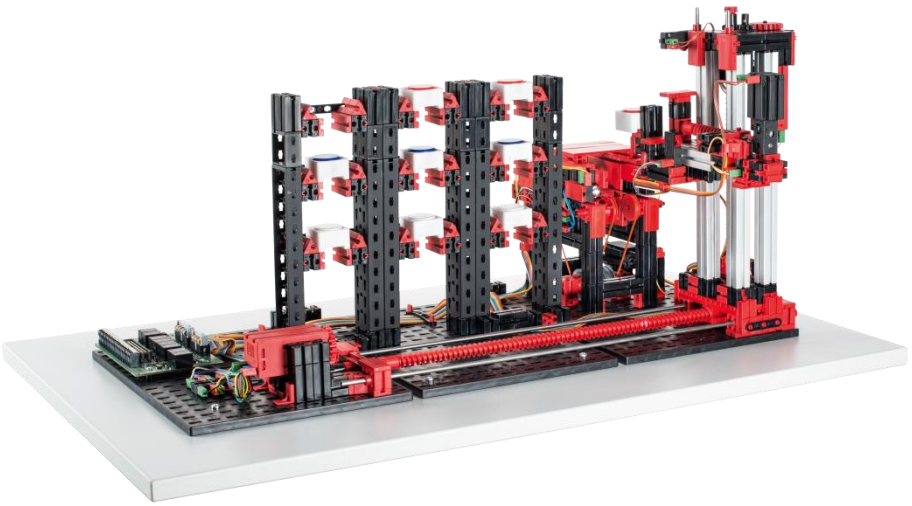
Almacenamiento dinámico

¿Cuáles son los dos requisitos previos en la utilización del almacenamiento dinámico?

¿Qué es de esperar en el almacenamiento dinámico?

¿Cómo se puede optimizar aún más el almacenamiento dinámico?

Aplique la estrategia ABC en el almacén elevado automatizado.



Almacenamiento dinámico **SOLUCIÓN**

¿Cuáles son los dos requisitos previos en la utilización del almacenamiento dinámico?

- *Identificación (parcialmente) automatizada de la mercancía*
- *Estandarización de los puestos de almacenamiento*

¿Qué es de esperar en el almacenamiento dinámico?

- *Optimización de los recorridos*
- *Optimización del uso de la superficie de almacenamiento*

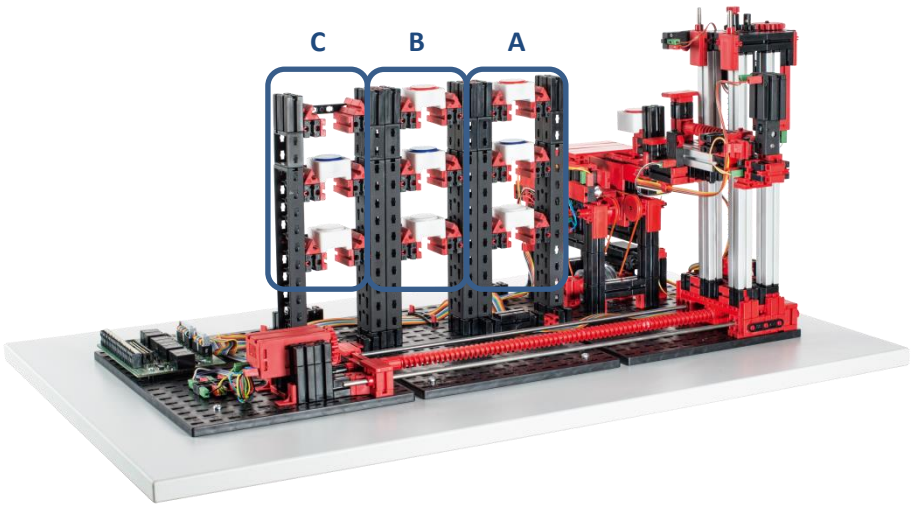
¿Cómo se puede optimizar aún más el almacenamiento dinámico?

Aplicando la estrategia ABC, en la que la mercancía frecuentemente requerida se coloca cerca del puesto de colocación

y retirada y la mercancía raras veces requerida, muy lejos de

la zona de almacenamiento.

Aplique la estrategia ABC en el almacén elevado automatizado.



Mantenimiento y búsqueda de errores

El almacén elevado automatizado no requiere, en general, ningún mantenimiento. En caso necesario, se pueden reengrasar los tornillos sin fin o sus tuercas. Tenga en cuenta al respecto que la aplicación de una película de grasa en determinados sitios puede impedir una unión en arrastre de fuerza.

Problema: **Uno de los tres motores/ejes ha dejado de moverse.**
Solución: Realice una inspección visual del robot. Controle especialmente el cableado del motor que falla. Compruebe, dado el caso, si hay cables rotos usando un multímetro.

Problema: **Uno de los tres motores/ejes sobrepasa la posición preestablecida y no se detiene de forma autónoma.**
Solución: Controle si los tres conductores del cable del codificador están correctamente conectados al TXT Controller. A tal fin puede ser útil la ventana "Prueba de interfaz".

Problema: **Uno de los tres motores/ejes no llega correctamente a las posiciones, y queda detenido poco antes de la posición deseada.**
Solución: Controle que las pinzas de sujeción y las tuercas de las pinzas del robot estén bien apretadas. De lo contrario, existe la posibilidad de que se produzca un resbalamiento entre las piezas en arrastre de fuerza.

Problema: **La cinta transportadora no se desplaza o no se desplaza lo suficiente, aunque hay una pieza sobre la cinta.**
Solución: Una de las dos barreras de luz de la cinta transportadora no funciona. Compruebe el cableado de las barreras de luz, y asegúrese de que estas no estén cubiertas por componentes desplazados. A tal fin puede ser útil la ventana "Prueba de interfaz".

Problema: **El dispositivo de control roza contra el almacén elevado o no recoge correctamente el contenedor.**
Solución: Adecue las posiciones del programa en la subfunción "Configuración".

Problema: **El dispositivo de control queda detenido en el almacén elevado.**
Solución: La posición del almacén elevado está mal ajustada. Al recoger el portapiezas, el dispositivo de control debe desplazarse hacia arriba. Si el eje correspondiente no se desplaza contra el tope, la rutina permanece en una repetición ilimitada. Para evitar esto, debe ajustar las posiciones de ese eje de modo que la rutina de recogida del portapiezas no se extienda más allá de los límites.