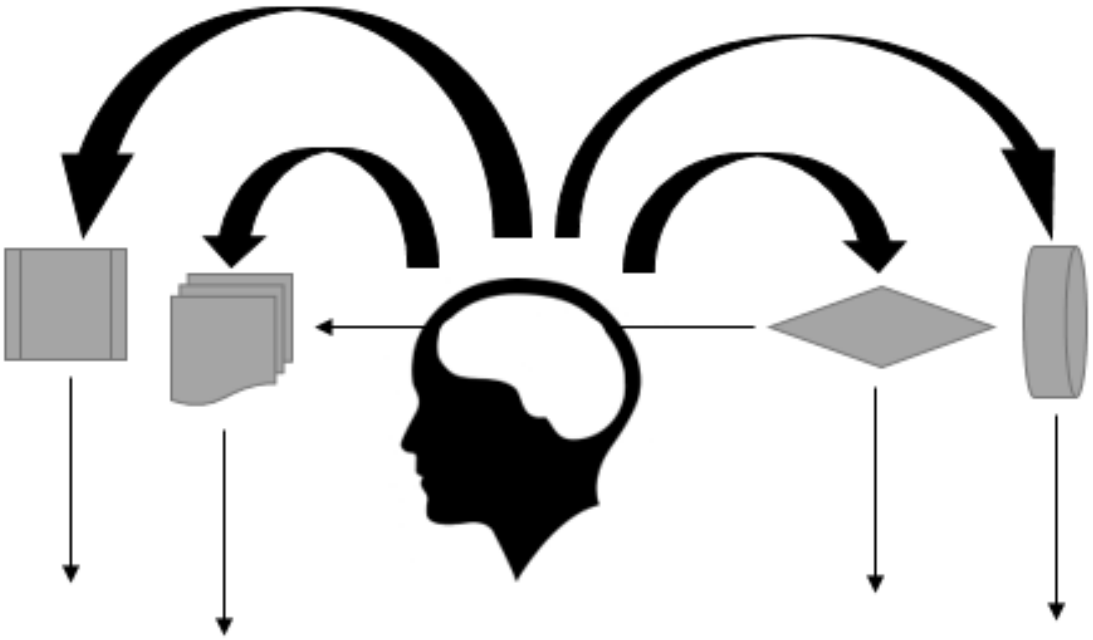
2017



UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

CONTROL DE UN PUENTE GRÚA MEDIANTE AUTÓMATA PROGRAMABLE BASADO EN UNA FUNCIÓN GENÉRICA DE EJE

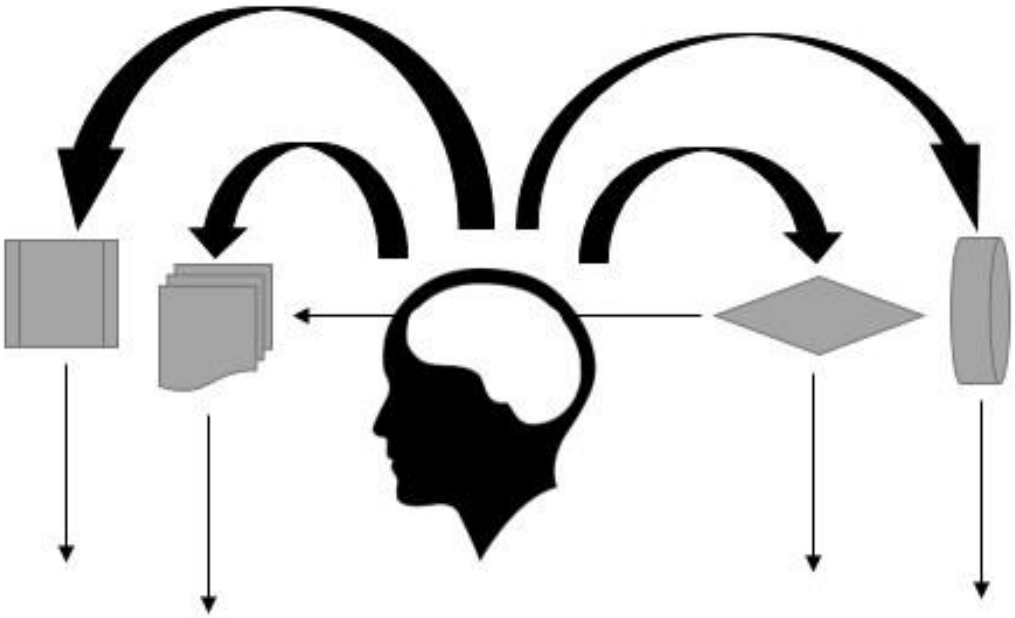
AUTOR DEL PROYECTO: CÉSAR ROMERO BARBERÁ.

DIRECTOR DEL PROYECTO: D. RUBÉN PUCHE PANADERO.

CO-DIRECTOR: D. ÁNGEL SAPENA BAÑÓ.

Septiembre de 2017

ÍNDICE GENERAL



CONTROL DE PUENTE GRÚA MEDIANTE AUTÓMATA PROGRAMABLE BASADO EN UNA FUNCIÓN GENÉRICA DE EJE.

MEMORIA

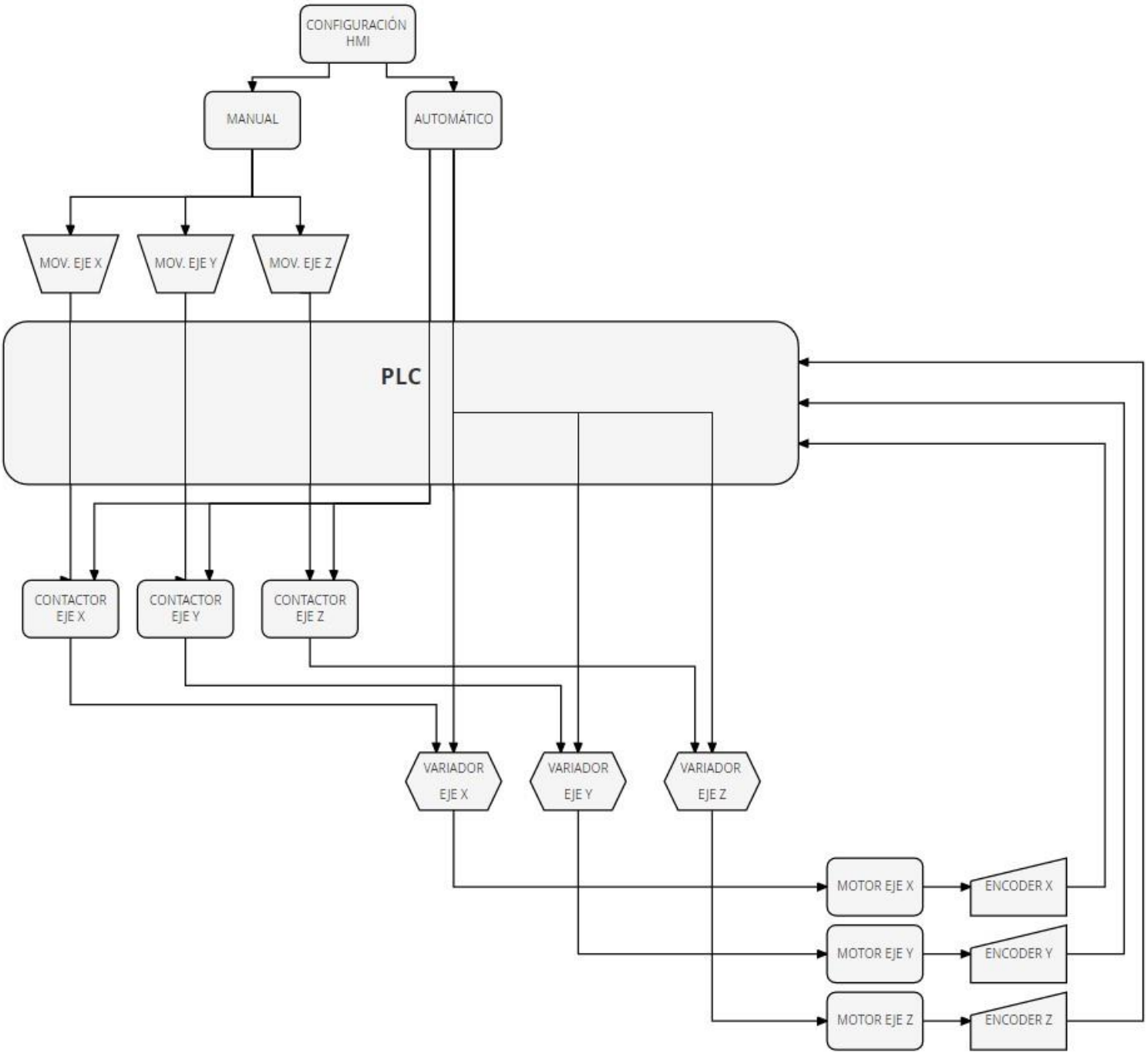
CÉSAR ROMERO BARBERÁ

MEMORIA

1. OBJETIVO DEL TRABAJO DE FIN DE GRADO

El objetivo del trabajo de fin de grado es el control y visualización de un puente grúa mediante la programación y parametrización de una función genérica que controle cualquier eje acoplado en una máquina industrial, de modo que solo haya que introducir ciertos parámetros configurables como la longitud del eje, la posición mínima, la máxima y otros datos, adaptándose así a las necesidades de cada aplicación y evitando errores de programación, siendo posible su reutilización para cada control que se requiera.

El sistema de control permite comunicarnos con la máquina a través del autómata programable, mediante un SCADA en una pantalla HMI con el que podremos controlar el proceso, el cual se ha representado mediante el flujograma en la siguiente página.


CONFIGURACIÓN HMI
MANUAL
AUTOMÁTICO
MOV. EJE X
MOV. EJE Y
MOV. EJE Z
PLC
CONTACTOR EJE X
CONTACTOR EJE Y
CONTACTOR EJE Z
VARIADOR EJE X
VARIADOR EJE Y
VARIADOR EJE Z
MOTOR EJE X
MOTOR EJE Y
MOTOR EJE Z
ENCODER X
ENCODER Y
ENCODER Z

1.1. ANTECEDENTES

Desde el punto de vista académico, el antecedente de este proyecto era la necesidad de crear una función genérica que pueda controlar cualquier tipo de eje. Ya que hay muchos programas que requieren de varias funciones, éste se centra en juntar toda la funcionalidad de éstas en una sola función, de modo que habría menos dificultades a tener en cuenta a la hora de configurar el eje.

Desde el punto de vista industrial, tenemos varios antecedentes a tener en cuenta. Desde hace años, las empresas dedicadas a la producción en cadena, han implementado lo que se ha denominado como "Automatización Industrial", lo cual ha traído consigo enormes ventajas en muchos aspectos: Repetición de tareas, mejor calidad, ahorro en costes, mejora del tiempo de producción, mayor seguridad para los operarios, una mejor comunicación entre distintos niveles, etc…

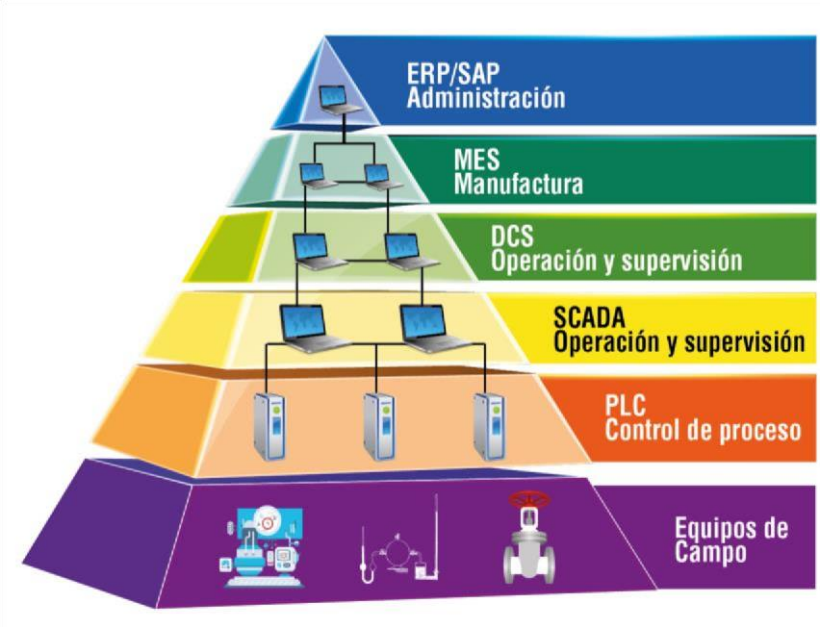
Para ello, la opción más extendida ha resultado ser el Autómata Programable, siendo barato, muy compacto, y por supuesto eficaz, ya que se basa en módulos que podemos instalar y configurar a nuestro antojo.

Al principio, los lenguajes de programación de los autómatas eran muy complejos, ya que los creaban los propios fabricantes. Con el tiempo, se empezó a unificar todos los lenguajes de programación, habiendo diferentes tipos: Diagramas de contactos (Ladder), lenguaje estructurado (ST), esquemas GRAFCET, diagrama de bloques funcionales, etc…

Mas tarde, y con la ayuda del sistema modular que poseían ya los autómatas, se empezaron a comunicar éstos con sus módulos mediante redes de comunicación, donde se intercambiaba información y se controlaba procesos a la vez.

Tanta evolución llevó consigo que el autómata fuera un componente indispensable, ya no solo en la automatización de grandes industrias, sino en otros campos como en logística, mecánica, domótica, etc…

Pero la industria tenía la necesidad de ver y controlar una máquina o proceso desde la distancia, fue ahí donde se crearon los **SCADAS** o **Supervisory control and data acquisition**, los cuales permiten el acceso a datos remotos de un proceso y permiten el control del mismo. No se trata de un sistema de control, sino de una utilidad software de monitorización o supervisión, el cual realiza la tarea de interface entre los niveles de control y los de gestión, justo a un nivel superior en la pirámide de la automatización industrial



1. PIRÁMIDE DE AUTOMATIZACIÓN

Desde el punto de vista industrial, y enfocado a la aplicación de este proyecto, debemos hacer un inciso en la historia y el desarrollo de las grúas. Desde la antigüedad se ha venido utilizando los distintos tipos de **grúas** para realizar muy diversas tareas.

Hasta la llegada de la revolución industrial, los principales materiales de construcción para **las grúas** eran la madera y la piedra. Desde la llegada de la revolución industrial los materiales más utilizados fueron el hierro fundido y el acero, además de utilizar como fuente energética máquinas de vapor en el s. XVIII.

Las grúas modernas de hoy en día utilizan generalmente motores de combustión interna o motores eléctricos e hidráulicos para proporcionar fuerzas mucho mayores debido a sus grandes prestaciones de par.

Hay distintos tipos de grúas, sin embargo, este proyecto se basará en el uso del **puente-grúa.**

2. JUSTIFICACIÓN

Podemos encarar la justificación de este proyecto desde dos perspectivas diferentes, y con diversidad de objetivos, los cuales se explicarán a continuación.

2.1. JUSTIFICACION ACADÉMICA.

La realización de este proyecto tiene como finalidad la obtención del título de Grado en Ingeniería Eléctrica, por la Escuela Técnica Superior de Ingeniería del Diseño (ETSID) en la Universitat Politècnica de València, así conforme con la norma vigente del Ministerio de Educación del Gobierno de España, incluido en el Real Decreto 1393/2007 de 29 de octubre por el que se establece la ordenación de las enseñanzas universitarias oficiales.

El proyecto se ha realizado en el departamento de Ingeniería Eléctrica y con la ayuda del profesor y director del proyecto D. Rubén Puche Panadero y el co-director del proyecto D. Ángel Sapena Bañó.

2.2. JUSTIFICACIÓN FUNCIONAL

Se detallan a continuación algunas justificaciones por las cuales ha sido necesario el desarrollo de este proyecto.

Un estudiante debe serlo para toda la vida, debe estar renovándose constantemente, explorar nuevos campos, y no estancarse. Con este proyecto, se pretende ampliar los conocimientos prácticos sobre el trabajo con autómatas programables, practicando y ampliando nociones teóricas sobre lo estudiado durante todo el grado universitario. Específicamente, lo estudiado en las siguientes asignaturas: "Regulación y protección de máquinas eléctricas", "Operación Remota de Sistemas Eléctricos", "Tecnología de Accionamientos Electromecánicos" y "Control de Máquinas y Accionamientos".

Seguidamente, y como aplicación de lo aprendido en las prácticas de empresa, hay que iniciarse en la adaptación de los sistemas de control y automatización de procesos, imprescindible si se quiere uno labrar un futuro profesional en este campo.

Hay que saber utilizar además las modernas redes industriales, de modo que sepamos como facilitarnos el trabajo en el futuro. Esto nos ayudará a tener procesos a gran distancia de subprocesos y poderlos comunicar fácilmente mediante redes de comunicación, sin necesidad de desplazamientos que cuesten tiempo y esfuerzos. Mediante un simple ordenador, podremos conectar entre si varios autómatas programables y establecer entre ellos una comunicación bidireccional.

Finalmente, y complementando los aprendido en la carrera con lo aprendido en la empresa de prácticas, hay que citar la importancia de conocer un mercado de sensores y actuadores en constante desarrollo para la mejora de la automatización industrial. También darnos cuenta que no todo es sencillo y rápido; habrá problemas que nos podemos encontrar en la industria siempre como interferencias en la red, errores de conexiones, problemas con los motores, etc…

3. ESTUDIO DE VIABILIDAD TÉCNICA

El proyecto es enteramente viable, ya que se trata de la automatización de un puente grúa óptimamente programado mediante una función muy genérica con la que se parametricen ciertos datos, lo que contribuye a un menor porcentaje de error, de tiempo y esfuerzos a la hora de instalarlo en la industria. Hay que tener en cuenta el tiempo que tardaremos en crear esta función, y el tiempo que tardaríamos en crear todo un programa entero especializado para cada eje y para cada puente grúa, se puede intuir que sería contraproducente llevarlo a cabo de esta manera. Sin embargo, vamos a centrarnos y mirar un poco más de cerca los diferentes puntos de vista que tenemos que contemplar.

Analizar la viabilidad de un proyecto es más importante que planificar y para poder concluirlo resulta imprescindible llevar a cabo una investigación amplia. A continuación, veremos distintos puntos de vista a tener en cuenta.

3.1. Limitaciones, restricciones y supuestos.

Debido a que trabajamos con un software, las limitaciones serán las que imponga éste. Sin embargo, y como veremos más adelante, utilizaremos un software potente, capaz de darnos muchísima libertad a la hora de programar, ya que nuestro programa será tan amplio y multifuncional como queramos.

3.2. Oportunidades.

Gracias al ingenio de muchos profesores y los muchos recursos de los que dispone la universidad, éste proyecto tiene la oportunidad de crearse con material más que de sobra.

3.3. Requisitos.

Para llevar a cabo este proyecto, habremos de necesitar un software de programación y un ordenador personal.

3.4. Alternativas.

Se ha hecho un estudio de alternativas antes de empezar este proyecto, más adelante veremos qué ha influido en la elección del PLC, del software de programación, y respecto a la parte de programación, veremos que objetivos tenemos marcados y por qué hemos llegado a ellos.

3.5. Línea de acción.

El camino a seguir para completar el presente proyecto ha sido el de completar la programación de la función genérica, junto a la creación de un SCADA para su aplicación. Una vez terminados ambos, se da inicio a la elaboración de la memoria.

La viabilidad humana es un factor también a considerar, debido a los costes de aprendizaje tanto a nivel de implementación física como de los diferentes soportes informáticos utilizados para el control de las aplicaciones realizadas.

4. FACTORES A CONSIDERAR: ESTUDIO DE NECESIDADES, LIMITACIONES Y CONDICIONANTES.

Hay que señalar que la función aquí descrita y creada se ha realizado de modo que puede servir para muchas aplicaciones diferentes, por tanto, cabe la necesidad de hacer dicha función versátil y multifuncional.

4.1. ESPECIFICACIONES DEL ENCARGO.

La función deberá permitir un control sobre la posición del eje, así como una monitorización en tiempo real de todos los parámetros que intervienen.

Como nos vamos a basar en una máquina de 3 ejes como aplicación de esta función, la función deberá ser capaz de las siguientes acciones:

- Modos de funcionamiento: 2 modos de funcionamiento diferentes: Modo MANUAL y AUTOMÁTICO.
- Software de Control: Una única función que debe de ser capaz de: ○ Configurar parámetros físicos del eje: Longitud, velocidad, posición de sensores inductivos,

finales de carrera. Mediante estos datos, la función ha de ser capaz de calcular las posiciones máximas y mínimas, así como la distancia que recorre el eje por cada pulso del encoder. Con el modo de funcionamiento MANUAL, el eje ha de poder moverse a un lado o a otro de forma manual.

  - Maniobra de "Puesta a cero": Mediante esta maniobra, la máquina debe llevar la plataforma u objeto que transporte a su puto de inicio. Para ello, el objeto deberá sobrepasar el sensor inductivo.
  - Control de Posición Y Velocidad: El objeto recorrerá el eje hasta una de las posiciones introducidas a través del SCADA, recorriendo el eje a la velocidad que indique el propio SCADA.

- Posible control de las maniobras desde un ordenador.

4.2. ESTUDIO DE NECESIDADES PROPIAS

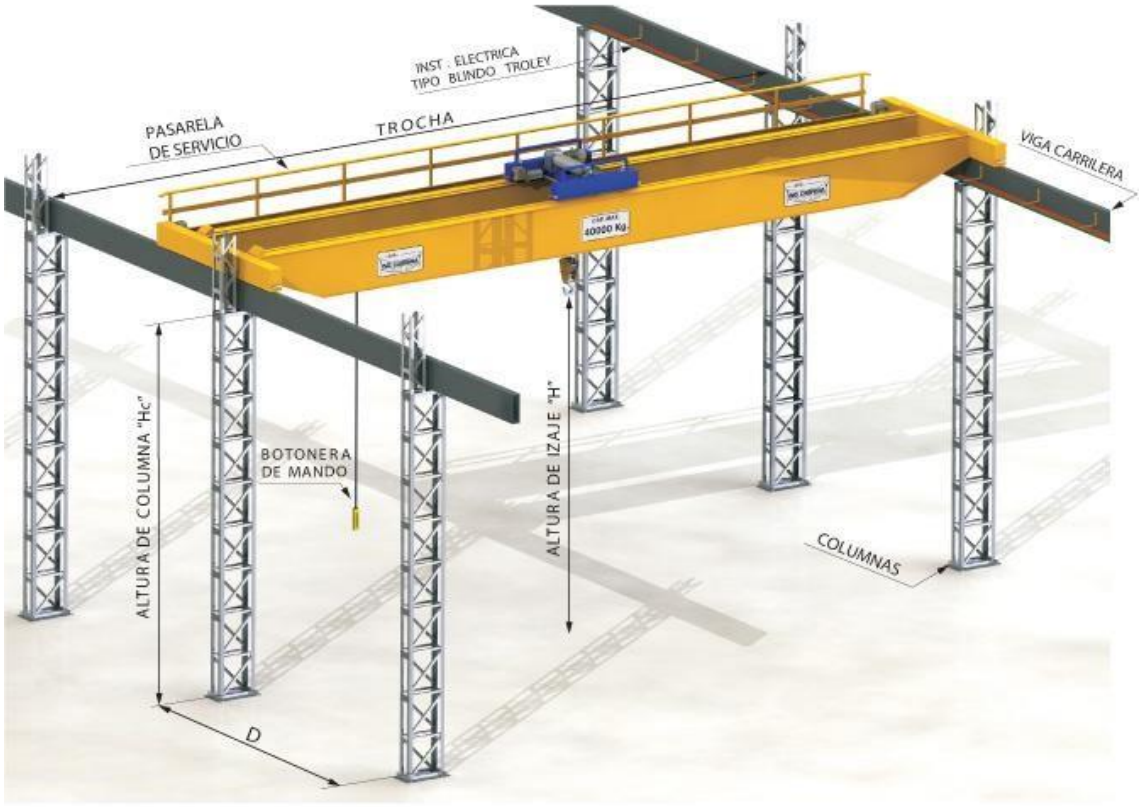
Se deberá de disponer de todo el equipamiento necesario para la programación; esto implica tener un ordenador en buen estado y un software de programación estable y fiable.

Además, siempre podemos confiar en la ayuda que nos puede proporcionar la asistencia técnica de Siemens, ya que trabajaremos en su entorno con el software TIA Portal. Incluso tenemos a nuestra disposición el foro oficial de esta empresa, en el que podremos preguntar a moderadores oficiales todas las dudas que tengamos.

En cuanto a la parte de programación, no ha habido problemas a la hora de llevar a cabo el proyecto. Eso si, en el programa se ha implementado unos márgenes de seguridad para las acciones de control que intervienen para garantizar la durabilidad de estos dispositivos al máximo.

5. DESCRIPCIÓN DEL PROCESO A CONTROLAR.

El proyecto trata de crear una función genérica para mover un eje cualquiera aplicado a un puente grúa, por lo que necesitaremos al menos dos ejes (X, Y) como muestra el siguiente esquema, y un tercer motor para subir y bajar el gancho a lo largo del izaje.



Para la prueba de este proyecto, dictaminamos las siguientes medidas:

- Trocha (Eje X): 5 m = 5.000 mm
- Viga carrilera (Eje Y): 13 m = 13.000 mm

El puente grúa llevará en su totalidad los siguientes componentes:

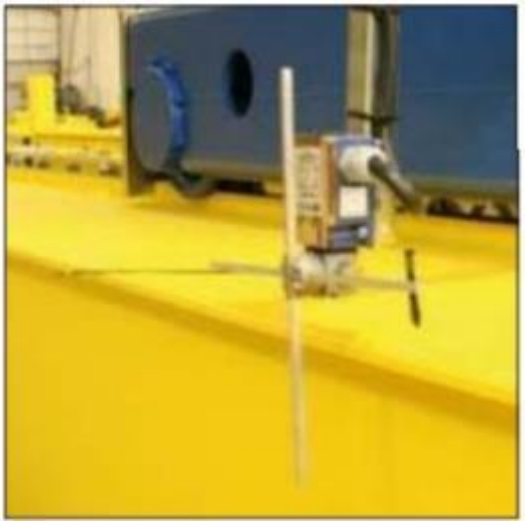
Un **polipasto**, el cual será el encargado de subir y bajar la carga.

Limitador de carga.

Un **encoder** por motor

Final de carrera para elevación.

Final de carrera para traslación de puente.



Final de carrera para traslación carro.

Un **variador de frecuencia** para la traslación del carro

Un **variador de frecuencia** para la elevación.

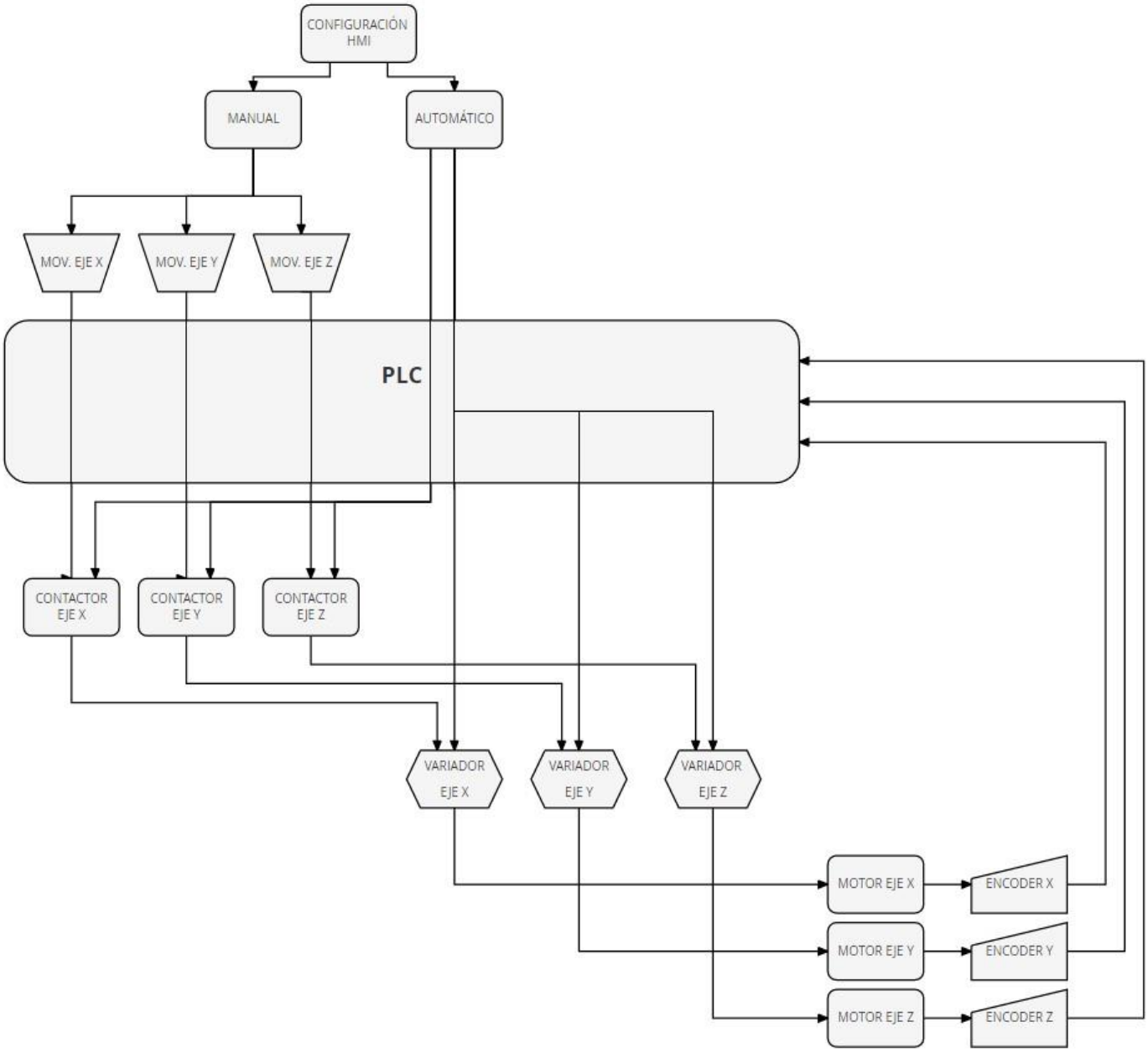
Un **variador de frecuencia** para la traslación del puente.

Todos estos componentes vienen impuestos por el cliente, así que no habrá que hacer ninguna selección entre todos ellos.

6. DESCRIPCIÓN DE LOS CRITERIOS DE SELECCIÓN Y JUSTIFICACIÓN DE LA SOLUCIÓN ADOPTADA.

El proceso de control y automatización de un sistema de estas magnitudes se puede hacer de muchas formas, ya que existen en el mercado multitud de componentes que nos pueden hacer el trabajo más fácil, así como software que tienen entre sí diferencias que pueden ayudar. A continuación, se explicará la solución adoptada.

Para empezar, necesitaremos un flujograma de cómo va a ser nuestro proceso:



Como se puede observar, desde la pantalla táctil o HMI configuraremos lo conveniente para el proceso: longitud del eje X, Y…, posición, límites, velocidad, etc…

El programa almacenará los datos necesarios y comenzará el proceso en caso de ser automático, o esperará a las órdenes convenientes en caso de ser manual. Esto hará actuar al variador de frecuencia y a los contactores respectivamente para el posterior movimiento del motor.

El encoder se encargará de situar el punto móvil en el eje a base de pulsos que enviará a la CPU donde convertirá esos pulsos en posición.

6.1. ELECCIÓN DEL PLC.

Un autómata programable o PLC es toda máquina electrónica, diseñada para controlar en tiempo real procesos secuenciales. Su manejo y programación puede ser realizada por personal eléctrico o electrónico sin conocimientos informáticos. Realiza funciones lógicas, temporizaciones, contajes, cálculos, regulaciones, etc. La función básica de los autómatas es la de reducir el trabajo del usuario para automatizar el proceso, es decir, la relación entre las señales de entrada que se tienen que cumplir para activar cada salida, puesto que los elementos tradicionales (como relés auxiliares, de enclavamiento, temporizadores, contadores...) son internos.

El autómata programable es el elemento principal de cualquier proceso de control, es el encargado de gestionar y controlar los sensores y actuadores en función del programa que se introduzca en él. Por lo tanto, la elección del autómata es importantísima para el éxito del proceso de control. En el mercado existe gran cantidad de autómatas diferentes de fabricantes distintos y con características muy variadas.

Al ser un proyecto más bien teórico, podemos elegir el PLC que queramos. Un autómata muy potente y con estructura modular nos permitirá una gran flexibilidad y será adaptable al control de diversos procesos totalmente diferentes y a las nuevas tecnologías que vayan surgiendo en un futuro, pero su precio es alto. Por el contrario, un autómata poco potente limitara la calidad del proceso de control. Por lo tanto, se debe elegir un autómata con estructura modular para asegurarse la flexibilidad y con una potencia y memoria un poco superior a la necesitada por si surgen imprevistos, pero no demasiado potente pues encarecería el proceso. Existen multitud de autómatas programables en la industria, de diversos fabricantes y con características muy variadas.

A continuación, expondremos una lista de autómatas que cumplen los requisitos establecidos en el anterior parágrafo.

Módulo de procesador M340:

Un autómata muy potente de la familia Schneider, con estructura modular y que permite la conexión a través de la red. Cuenta con una gran variedad de módulos que proporcionan una gran flexibilidad al autómata.

Es un autómata que cuenta con 1024 entradas y salidas digitales (Añadiendo módulos), y 256 entradas y salidas analógicas. Tiene varias funciones de comunicación:

- Gestión de ancho de banda, Ethernet TCP/IP
- Editor de Dados, Ethernet TCP/IP
- Mensajería TCP Modbus, Ethernet TCP/IP
- Rack Viewer, Ethernet TCP/IP
- Administrador de rede SNMP, Ethernet TCP/IP - Gestión de red (NMT) CANopen.

En lo que respecta a la programación, tiene estas funciones para controlar el programa: 1 tarea maestra cíclica/periódica, 1 tarea rápida periódica, 64 tareas de eventos y sin tarea auxiliar.

La alimentación es vía bastidor, por lo que habrá que procurarle una fuente de alimentación en el rack.

Como vemos, tiene un bus para comunicación CAN-OPEN, por lo que podríamos añadir repartidores CAN-OPEN para la distribución de los distintos sensores.

S7-1214C AC/DC/RLY

Un autómata de Siemens versátil, de estructura modular, y que permite la conexión a través de la red mediante puerto Ethernet.

Tiene un total de 10 salidas y 14 entradas digitales, y de ellas, 6 son válidas para contaje rápido (HSC or High Speed Counting).

Se alimenta a 24 VDC a partir de una fuente de alimentación externa, no se alimenta mediante bastidor.

Tiene una frecuencia máxima de contaje de 100 kHz, por lo que nos servirá para ciertos encoders que no superen esa resolución.

Hay que tener en cuenta que se le añade un potente software como es el TIA Portal, con el que podemos sacar el máximo provecho al autómata.

CP1L

La serie CP1L de Omron ofrece la compactibilidad de un micro PLC con la capacidad de un PLC modular. Proporciona toda la funcionalidad necesaria para controlar la máquina, incluida la extraordinaria capacidad para el desarrollo de aplicaciones "Motilón".

El CP1L dispone de 10, 14, 20,30, 40 o 60 E/S incorporadas y se puede ampliar con una extensa gama de unidades expansoras CP1W o CPM1A hasta un máximo de 180 puntos de E/S.

Utiliza un puerto USB estándar para la programación y motorización. Además, ofrece dos puertos de comunicaciones serie opcionales, de los que también puede utilizarse uno para una pantalla o Internet.

Tiene también entradas para un contador rápido, con una frecuencia máxima de 100 kHz.

A continuación, se expone qué PLC se ha elegido y sus razones.

Se elige el **S7-1200** de Siemens debido a dos razones principales:

- Es un autómata que, aunque de gama media-baja, es muy versátil y potente, y tiene lo necesario para llevar a cabo el proyecto.
- Una de las razones por la que se hizo este proyecto era para profundizar en la utilización del software TIA Portal de Siemens, muy presente en la industria. Es por ello que es una de las razones con más peso para elegir este autómata.
- Los autómatas son muy parecidos entre si respecto a sus características (nº de entradas, características comunes…) y además son todas marcas

comerciales, muy presentes en las industrias. Sin embargo, nos decantaremos por el SIEMENS por la necesidad propia de aprender a trabajar con TIA Portal.

6.2. LENGUAJE DE PROGRAMACIÓN

Una de partes más importantes antes de ponerse a programar es la elección del lenguaje con el que programaremos. Hay varios tipos, y cada uno tiene sus pros y sus contras. A continuación, explicaremos de donde vienen estos lenguajes y detallaremos los principales para poder elegir el que mejor convenga. Esta elección también dependerá de la experiencia del programador, por lo que cada uno entenderá a su modo qué es una ventaja y qué una desventaja.

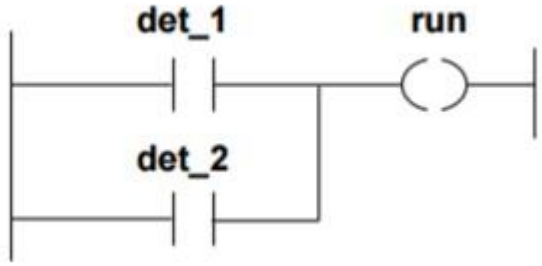
Primeramente, hay que saber que todos estos lenguajes están estandarizados en la norma IEC 61131, y esto ofrece muchas ventajas:

- Disminución de los costes de formación.
- Homogeneidad de la documentación de las aplicaciones ya que ésta contará con una estructura de programas idéntica y objetos de lenguaje predefinidos.
- Cada función de una aplicación puede programarse en el lenguaje que mejor se adapte para asegurar la coherencia final.

La norma lo que hace es definir los lenguajes de programación, la sintaxis y representación gráfica de los objetos. Además, define también la estructura de los programas y la declaración de variables.

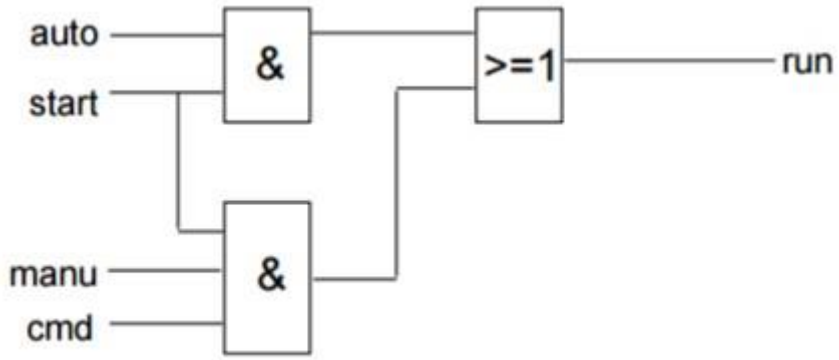
A continuación, se detallan los lenguajes que rige la norma.

- **Ladder Diagram (LD):** ○ Elementos gráficos organizados en redes conectadas por barras de alimentación. ○ Forma gráfica de los elementos impuesta por la norma. ○ Evaluación de la red por elementos interconectados.
    - ○ Elementos utilizados: Contactos, bobinas, funciones y bloques funcionales ○ Elementos de control de programa (salto, return…). Ejemplo:



- **Function Block Diagram (FBD):** ○ Representación de funciones por bloques enlazados uno a otro. ○ Ninguna conexión entre salidas de bloques de función.
    - ○ Evaluación de una red: De la salida de un bloque funcional a la entrada de otro bloque funcional.

  Ejemplo:



- **Instruction List (IL):** ○ Se encuentra formado por una serie de instrucciones: Cada una debe empezar en una línea nueva.
    - ○ Una instrucción está compuesta por un operador y uno ó más operandos separados por comas.
    - ○ Las etiquetas son opcionales y deben terminar en ":"
    - ○ Los comentarios son opcionales y debe ser el último elemento de una línea. El comienzo y el final de los comentarios está indicado mediante los símbolos (* *) Ejemplo:

```
CAL  C10(CU := %IX10, PV := 15)  es equivalente a :
     LD   15
     PV   C10
     LD   %IX10
     CU   C10
```

- **Structured Text (ST):** ○ Sintaxis similar a la de PASCAL, permitiendo la descripción de estructuras algorítmicas complejas.

- Sucesión de enunciados para la asignación de variables, el control de funciones y bloques de función, usando operadores, repeticiones y ejecuciones condicionales.
- Los enunciados deben terminar con ";" Ejemplo:

```
J:=1 ;
WHILE J<=100 AND X1< >X2 DO ;
J:=J+2 ;
END_WHILE ;
```

- **Sequential Function Chart (SFC)** o Muy útil para describir funciones de control secuencial.
  - Se basa en la norma GRAFCET IEC 848.
  - Etapas representadas gráficamente por un bloque o literalmente mediante una instrucción común a los leguajes IL y ST: STEP……END_STEP
  - Transiciones representadas gráficamente por una línea horizontal o literalmente mediante la instrucción: TRANSITION……END_TRANSITION.
  - Condición de transición programable en lenguaje LD, FBD, IL o ST.
  - Acciones asociadas a las etapas: Variables booleanas o un segmento de programa escrito en uno de los cinco lenguajes.
  - Asociación entre acciones y etapas de forma gráfica o literal.
  - Propiedades de acción que permiten temporizar la acción, crear pulsos, memorizar… Ejemplo:

En nuestro software, tenemos distintos lenguajes de programación, que entran en la norma IEC 61131 pero con otros nombres:

Ya hemos dado una pequeña revisión al lenguaje KOP, que el equivalente al lenguaje Ladder en Siemens, y tiene las mismas ventajas y desventajas que éste.

El lenguaje FUP, que es el equivalente al lenguaje FBD, se basa en funciones booleanas y matemáticas.

Finalmente, tenemos el lenguaje utilizado para nuestra función, que es el ST o en Siemens llamado SCL, el cual es el mismo que el ST con algunas funciones adicionales. Por tanto, a la hora de elegir el lenguaje en nuestro programa, optaremos por SCL.

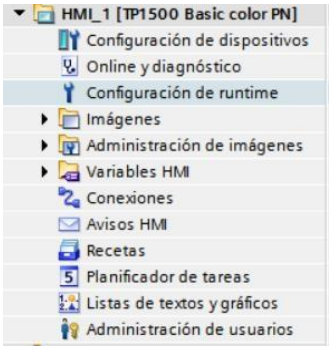
Una vez determinadas las ventajas y características de los diferentes tipos de lenguajes, se ha optado por el uso predominante del lenguaje ST o Texto Estructurado, ya que personalmente creo que permite una mayor versatilidad a la hora de resolver un problema, porque tenemos más variedad de posibilidades.

La función genérica, al poderse exportar a otros programas, podremos llamarla en el lenguaje que queramos. Veremos cómo en la página principal, donde llamamos a la función, el lenguaje principal es el KOP, que es para Siemens el lenguaje Ladder (LD).

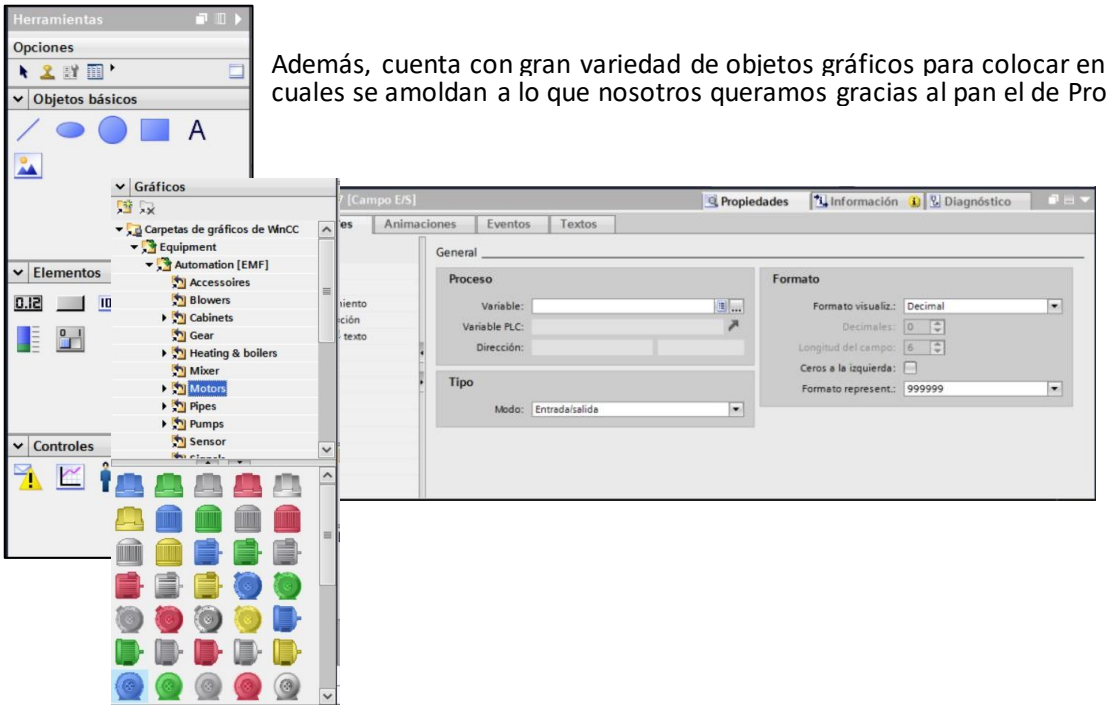
7. SELECCIÓN DEL SCADA.

En este apartado plantearemos cual sería un mejor SCADA para nuestra función y que marca comercial usaríamos para crearla.

Como el SCADA que vamos a crear es solo una propuesta para un cliente teórico, nos basaremos en las características de nuestra propia función. De modo que este SCADA deberá de contar con las entradas y salidas parametrizables de nuestra
Empezaremos con el winCC de TIA PORTAL:

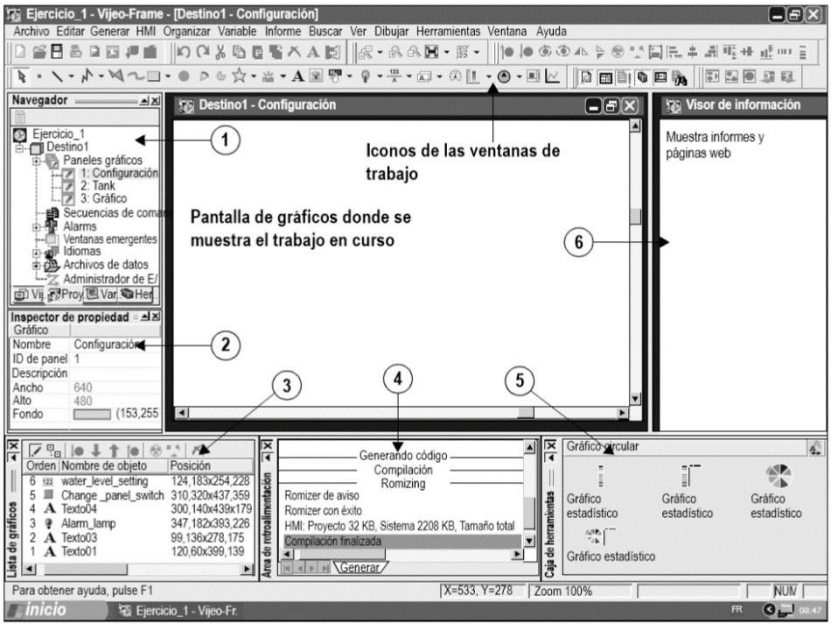


Una de las ventajas del TIA PORTAL es que es muy visual. Tiene todas las herramientas que necesitemos a la vista. Tenemos la configuración, las nes, variables, avisos, recetas, administración de usuarios, está todo al de un clic.



Además, cuenta con gran variedad de objetos gráficos para colocar en el canvas, los cuales se amoldan a lo que nosotros queramos gracias al pan el de Propiedades.

VIJEO DESIGNER de Telemecanique:



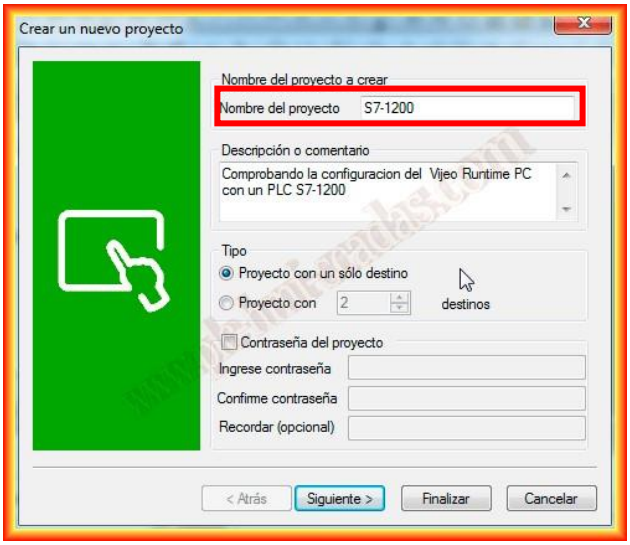
Este sería el aspecto habitual que tiene este editor. Su diseño es parecido al TIA Portal. Vamos a explicar un poco en que se basa cada zona señalada.

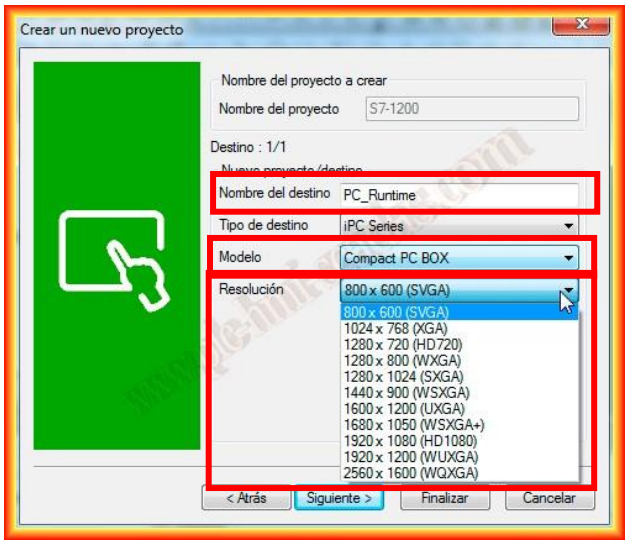
Como podemos observar, los gráficos son un poco peores que los del TIA Portal, y cuenta con menos opciones.

Elemento	Nombre de la pantalla/ icono	Descripción
1	Navegador	Utilizado para crear aplicaciones. La información relativa a cada proyecto se ordena de forma jerárquica en un explorador de documentos.
2	Inspector de propiedad	Muestra los parámetros del objeto seleccionado. Cuando se selecciona más de un objeto, sólo se muestran aquellos parámetros que son comunes a todos los objetos.
3	Lista de gráficos	Ofrece una lista con todos los objetos que figuran en la sinopsis e indica: • el orden de creación • el nombre • la posición • las animaciones • otras variables asociadas El objeto que aparece resaltado en la lista se selecciona en la sinopsis. La información se muestra de manera similar (esto es, orden, nombre y posición) para un grupo de objetos. Para visualizar la lista de objetos de un grupo, haga clic en +. Es posible seleccionar cada objeto de forma individual.
4	Área de retroalimentación	Muestra la progresión y los resultados de la comprobación de los errores, de la compilación y de la carga. Si se produce un error, el sistema muestra un mensaje de error o un mensaje de alerta. Para ver dónde se encuentra el error, haga doble clic en el mensaje de error.
5	Caja de herramientas	Biblioteca de componentes (gráfico de barras, cronómetros, etc.) que suministra el fabricante y/o que usted ha creado con anterioridad. Para colocar un componente en la sinopsis, selecciónelo en la Caja de herramientas y arrástrelo hacia la sinopsis. Puede exportar o importar sus propios componentes.
6	Visor de información	Muestra la ayuda en línea o el contenido de los informes.

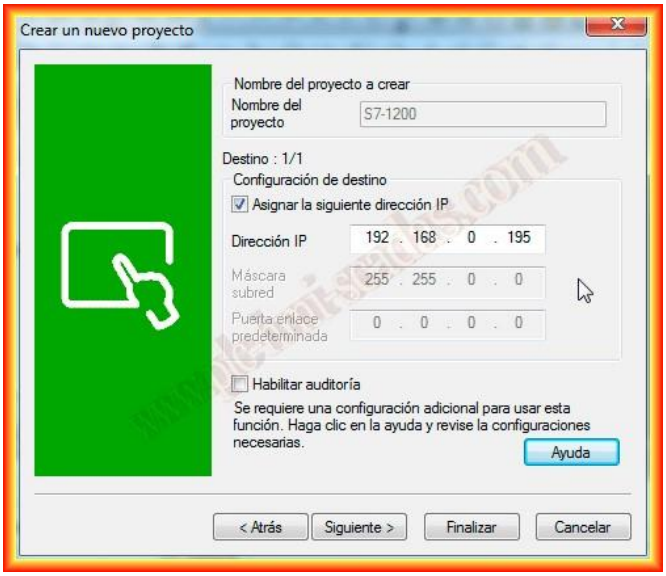
En todo caso, vamos a proceder a configurarlo para poder conectarlo a un SIEMENS s7-1200. (Las imágenes que voy a proceder a usar son propiedad de la página web http://plc-hmi-scadas.com, para más información, contactar con dicha página.)

Si iniciamos el software, abriremos un proyecto nuevo dándole un nombre. Una vez en el panel siguiente, habrá que darle un nombre al HMI, elegir el modelo deseado y elegir la resolución de nuestra pantalla táctil.

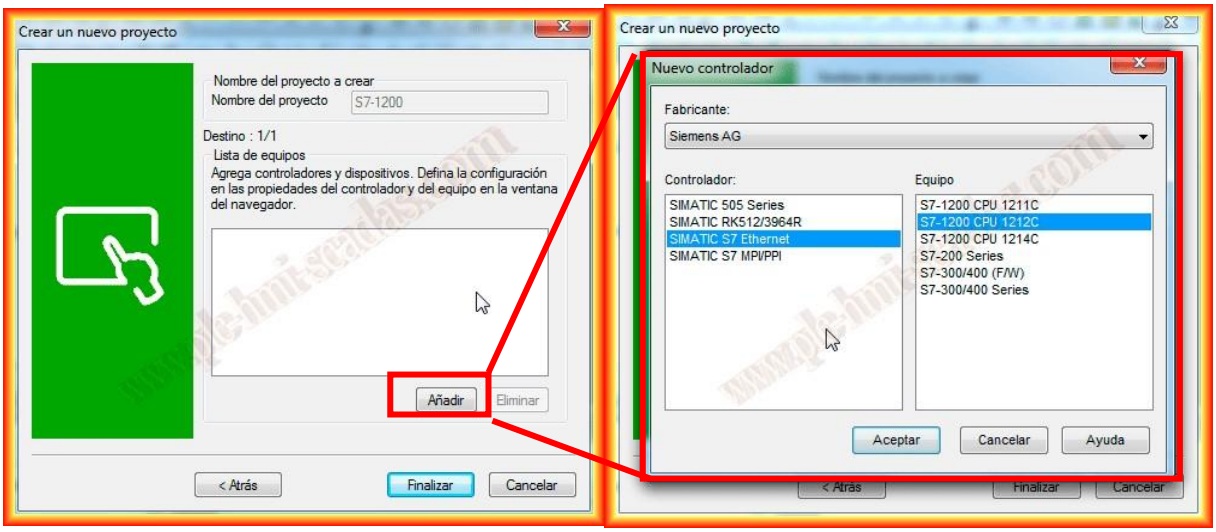




En la próxima pantalla, elegiremos la dirección IP de la propia pantalla, todo en función del protocolo de comunicación que queramos usar.



A continuación, habrá que definir el dispositivo que va a comunicarse con la pantalla, el cual es la CPU que vamos a conectar. En el siguiente panel, pulsaremos en AÑADIR:



Esto abrirá una pantalla donde elegiremos el controlador. Seleccionaremos el fabricante (SIEMENS) y escogeremos la CPU s7-1200 CPU 1212C.

Se nos habrá generado un nuevo driver en la zona de Administrador de E/S con nuestra CPU. Si pulsamos sobre él, deberemos configurar la IP de la CPU. Con esto ya podríamos trabajar con el software.

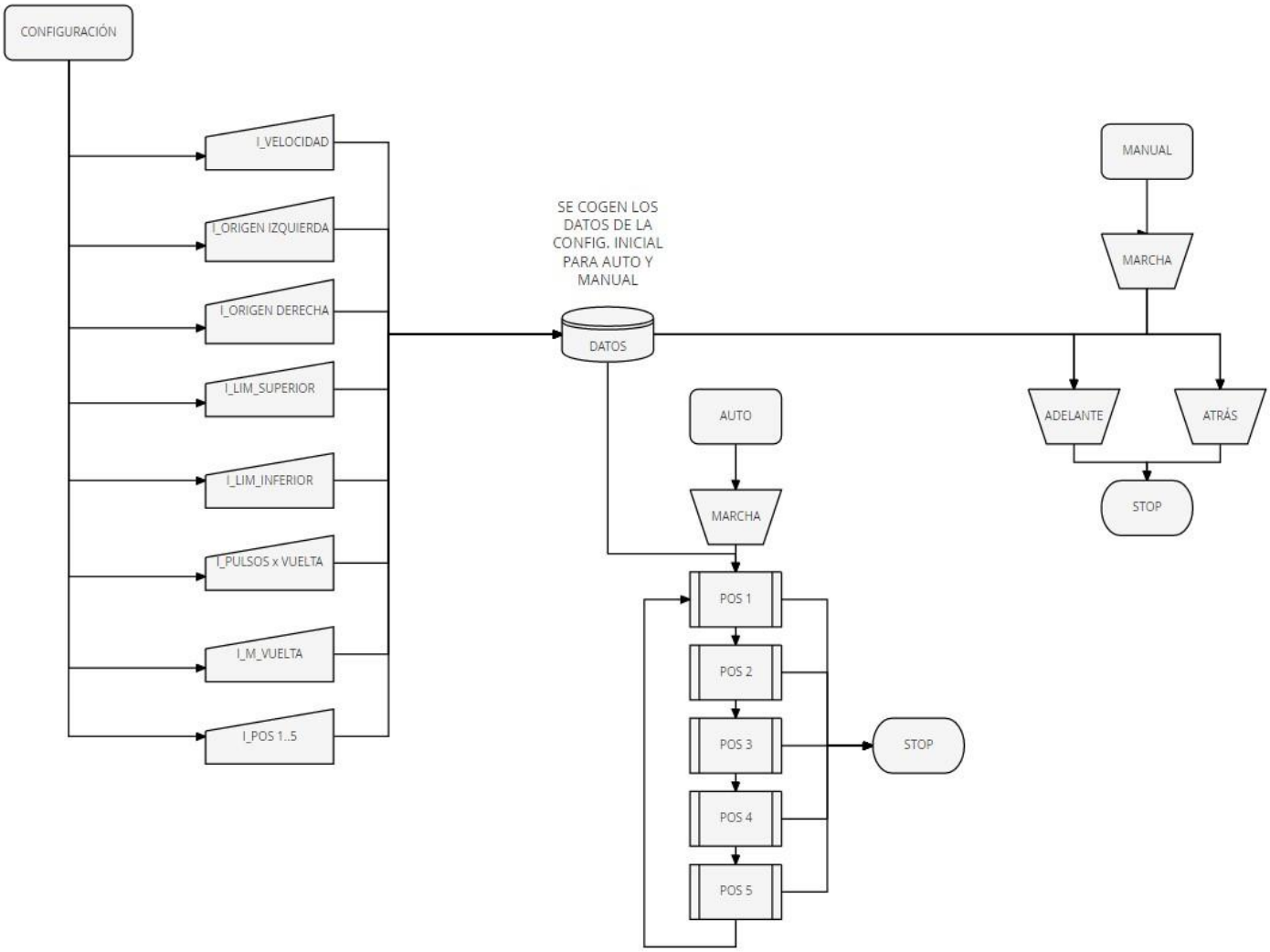
Para no complicar mucho el proyecto, vamos a elegir el **WinCC de TIA PORTAL**, ya que facilita mucho el tener el entorno de programación en el mismo software que el canvas para el SCADA. Además, SIEMENS facilita la tarea de la creación del SCADA mediante unas reglas que ayudan a posicionar los objetos gráficos además de otras opciones gráficas; podemos simular la pantalla sin mucho esfuerzo; y cuenta con muchas más herramienta s que el otro editor. Sin embargo, la razón principal es que queremos abarcar el uso del TIA Portal en su totalidad, y eso incluye el editor de SCADA's, su aplicación y su simulación.

8. DEFINICIÓN DE LA FUNCIÓN

La función que vamos a programar es el corazón del proyecto. A continuación, explicaremos como es su funcionamiento, qué necesitaremos y como programaremos las diferentes opciones que necesita.

Primeramente, construiremos un flujograma que explique el funcionamiento de la función:

8.1. FLUJOGRAMA DE LA FUNCIÓN



En la imagen anterior, observamos el flujograma de la función. Procedemos a explicarlo:

Primeramente, debemos introducir los diferentes parámetros necesarios para la correcta configuración de la función. Tenemos la velocidad, la elección del origen, la elección del límite, la resolución del encoder, el avance del eje, y las distintas posiciones del eje.

Estos datos serán necesarios para iniciar los movimientos manuales y automáticos, los cuales usan unos datos u otros, ya que, por ejemplo, el movimiento manual no usa los datos de las posiciones.

Estos datos serán almacenados en la función. A continuación, una vez hayamos elegido entre Manual o Automático, pulsaremos marcha para empezar el trabajo de la función.

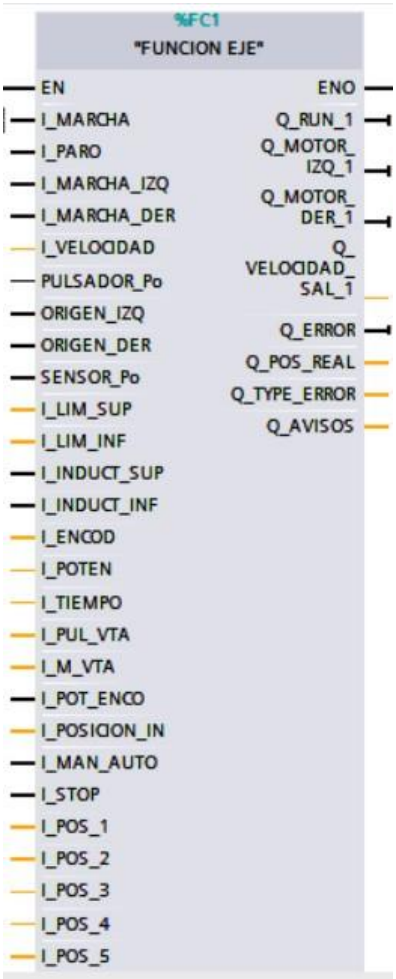
En el caso del Automático, tenemos varias subrutinas que son las distintas Posiciones que deberá tomar el eje. Éstas irán una detrás de otra hasta completar el ciclo, el cual volverá a empezar. Este ciclo podrá ser interrumpido mediante la entrada de STOP.

En el caso del movimiento Manual, podremos accionar el motor directamente mediante los botones Adelante o Atrás, que, en el caso de la función, son "Motor Izquierda" y "Motor Derecha".

Como vemos, es una función bastante genérica, y por ello se necesitan muchos parámetros a configurar. En el siguiente apartado veremos mejor la cantidad de detalles por parametrizar que posee la función.

8.2. INTERFAZ DE LA FUNCIÓN

En este apartado vamos a ver cómo será la función una vez hayamos declarado las variables necesarias para el funcionamiento completo:



En esta imagen se observa la interfaz de la función. Aquí conectaremos las variables de los bloques de datos a sus respectivas entradas y salidas. La función solo puede controlar un eje, por tanto, en un programa de varios ejes tendremos tantas interfaces como ejes haya, y, por tanto, en los bloques de datos, habrá una estructura por cada eje. Esto se verá más profundamente en los siguientes puntos.

Cada variable de entrada y de salida está coloreada con un color dependiendo del tipo de variable que haya que introducir, de modo que si están en negro serán de tipo BOOL, y si

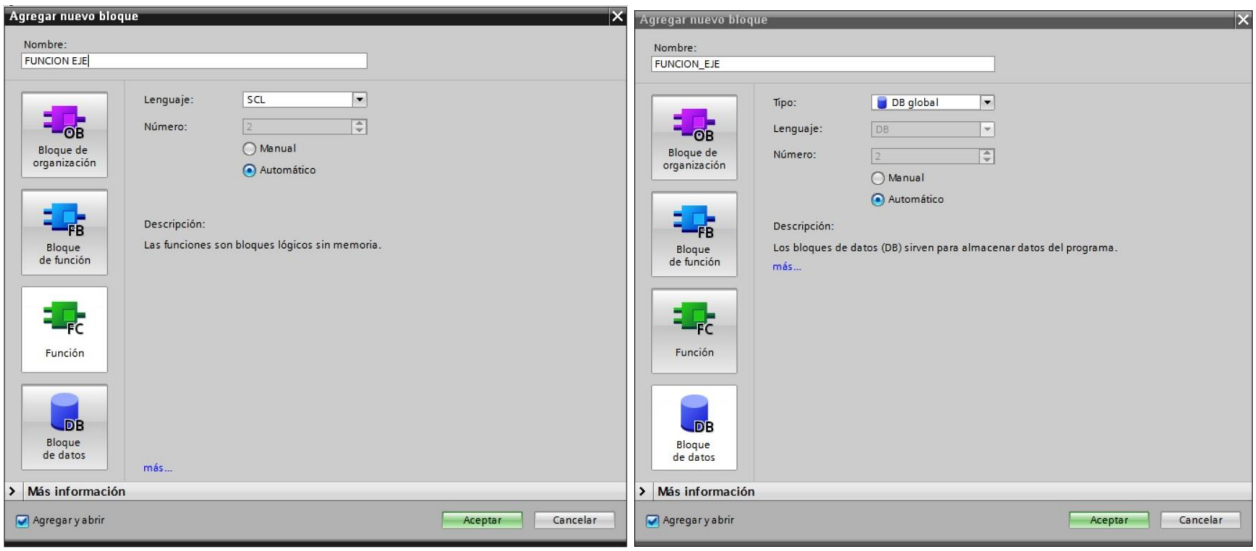
es de tipo INT estará anaranjado. Esto último es parte de la propia configuración del software de programación.

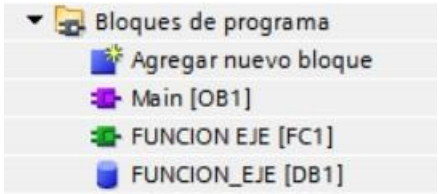
Las distintas entradas y salidas vienen explicadas en el Anexo III – Manual de usuario de la función genérica.

9. CREACIÓN DE LA FUNCIÓN

9.1. INTRODUCCIÓN

Empezamos abriendo el programa TIA PORTAL V14 y creamos nuestros dos primeros componentes del programa: La función y su DB (o Data Block). El primera contendrá la función genérica para controlar el eje, objetivo de este trabajo. El segundo será donde se encuentren todos nuestros datos, su almacenamiento.





9.2. VARIABLES

Siendo una función genérica, vamos a definir datos de 3 ejes distintos: Eje X, Eje Y y Eje Z. Esto nos permitirá independizar cada función relacionada a cada eje.

Para declarar estos datos en un solo bloque de datos, lo que haremos es construir una estructura (STRUCT) por cada eje. Así, tendremos una STRUCT para el eje X, otra para el eje Y, y otra para el eje Z.

FUNCION_EJE

	Nombre	Tipo de datos	Valor de arranq...	Remanen...	Accesible d...	Escrib...	Visible en ...	Valor de a...	Coment...
1	Static								
2	X	Struct							
3	Y	Struct							
4	Z	Struct							

Dentro de cada estructura, definiremos, pues, cada una de las variables, las cuales serán entradas y salidas en la función genérica. Así pues, coincidirán con las entradas y salidas a declarar en la función.

Por tanto, definiremos, para empezar, las siguientes variables de entrada:

- I_MARCHA
- I_PARO
- I_MARCHA_IZQ
- I_MARCHA_DER
- I_VELOCIDAD

Y las siguientes variables de salida:

- Q_RUN
- Q_MOTOR_IZQ
- Q_MOTOR_DER
- Q_VELOCIDAD_SAL

Además, tenemos que definir el tipo de dato que es cada variable, de modo que introduzcamos una breve explicación sobre qué son este tipo de datos, y por qué definimos cada variable con su correspondiente tipo:

FUNCION_EJE

	Nombre	Tipo de datos	Valor de arranq...	Remanen...	Accesible d...	Escrib...	Visible en ...	Valor de a...	Coment...
1	Static								
2	X	Struct							
3	I_MARCHA	Bool	false						
4	I_PARO	Bool	false						
5	I_MARCHA_IZQ	Bool	false						
6	I_MARCHA_DER	Bool	false						
7	I_VELOCIDAD	Int	0						
8	Q_RUN	Bool	false						
9	Q_MOTOR_IZQ	Bool	false						
10	Q_MOTOR_DER	Bool	false						
11	Q_VELOCIDAD_SAL	Int	0						

- Marcha: Un pulsador que nos permite encender el programa de la función. Solo tiene dos estados, encendido o apagado. En este caso, usamos un tipo de datos BOOL, el cual solo tiene dos estados: 0 o 1.

- Paro: Un pulsador que nos permite detener el programa en caso de necesitarlo. Es el mismo caso que la variable Marcha, dos estados, un BOOL.

- Marcha Izquierda y Marcha Derecha: Estas dos variables son las responsables de dirigir el eje hacia un lado u otro. Si la variable Izquierda está encendida, el eje se dirigirá hacia el lado que nosotros definimos como izquierda. En el caso de la variable Derecha, el eje se dirigirá hacia donde nosotros hemos definido como derecha. Por tanto, siendo variables de dos posibles estados, serán BOOL.

- Velocidad: Ésta es la variable de referencia para la velocidad que queramos darle al eje correspondiente. Esta velocidad puede adquirir cualquier valor real. Para hacerlo más sencillo, siendo éste el programa base, optamos por darle solo valores dentro del rango de números enteros. Por tanto, será de tipo INT o "entero".

- Run: Como primera de las variables de salida, es la responsable de avisarnos cuando la función está en marcha, y al tener únicamente dos estados posibles, la definiremos como BOOL.

- Motor Izquierda y Derecha: Estas dos variables de salida son las que mandan la orden de mover el motor de los ejes al variador de frecuencia cuando se cumplen los requisitos para moverlos en cada caso. Solamente hay dos estados posibles, por tanto, serán un BOOL cada una.

- Velocidad de Salida: Es la encargada de transmitir al variador de frecuencia el valor de velocidad que queremos dar al eje. Hay que tener en cuenta que, todas las variables que relacionen la velocidad (como veremos más adelante) deben ser el mismo tipo de datos. Por tanto, si la velocidad de referencia a la entrada es de tipo INT, ésta será del mismo tipo.

Habrá que tener en cuenta que estas variables no son definitivas, ya que esto es una primera vista del programa y habrá muchas más variables, aspectos, y configuraciones a tener en cuenta. Esto es lo básico que hará que los ejes se muevan.

Repetiremos la declaración de datos para cada eje al que le apliquemos la función.

FUNCION_EJE

	Nombre	Tipo de datos	Valor de arranq...	Remanen...	Accesible d...	Escrib...	Visible en ..	Valor de a..	Com...
1	Static								
2	X	Struct							
3	I_MARCHA	Bool	false						
4	I_PARO	Bool	false						
5	I_MARCHA_IZQ	Bool	false						
6	I_MARCHA_DER	Bool	false						
7	I_VELOCIDAD	Int	0						
8	Q_RUN	Bool	false						
9	Q_MOTOR_IZQ	Bool	false						
10	Q_MOTOR_DER	Bool	false						
11	Q_VELOCIDAD_SAL	Int	0						
12	Y	Struct							
13	I_MARCHA	Bool	false						
14	I_PARO	Bool	false						
15	I_MARCHA_IZQ	Bool	false						
16	I_MARCHA_DER	Bool	false						
17	I_VELOCIDAD	Int	0						
18	Q_RUN	Bool	false						
19	Q_MOTOR_IZQ	Bool	false						
20	Q_MOTOR_DER	Bool	false						
21	Q_VELOCIDAD_SAL	Int	0						
22	Z	Struct							
23	I_MARCHA	Bool	false						
24	I_PARO	Bool	false						
25	I_MARCHA_IZQ	Bool	false						
26	I_MARCHA_DER	Bool	false						

10. PROGRAMACIÓN DE LA FUNCIÓN

10.1. PRIMEROS PASOS: MARCHA Y PARO

Empezaremos con lo básico: Marcha y paro.

```
IF #I_MARCHA AND NOT #I_PARO THEN

END_IF;
```

La función entera, de momento, se encontrará dentro de la función IF de Marcha; de modo que para que la función se ponga en marcha, deberemos activar el pulsador homónimo. Para que la función trabaje, el botón paro debe estar desactivado, por ello lo negamos en las condiciones de la función IF.

```
1  IF #I_MARCHA AND NOT #I_PARO THEN
2
3      #Q_RUN_1 := TRUE; //RUN PARA VER CUANDO ESTÁ EN MARCHA
4
5      IF #I_MARCHA_IZQ AN ... THEN ... END_IF; //SE MUEVE A IZQUIERDA CUANDO PULSAMOS MARCHA IZQUIERDA
8
9      IF #I_MARCHA_DER AN ... THEN ... END_IF; //SE MUEVE A DERECHA CUANDO PULSAMOS MARCHA DERECHA
```

Dentro de esta función, optamos por poner una variable de salida “RUN” que permite indicar que la función está en funcionamiento.

A continuación, tenemos dos funciones IF para poner en marcha el motor, dependiendo de si queremos a izquierda o a derecha:

```
5   IF #I_MARCHA_IZQ AND NOT #I_MARCHA_DER THEN
6       #Q_MOTOR_IZQ_1 := TRUE;
7   END_IF; //SE MUEVE A IZQUIERDA CUANDO PULSAMOS MARCHA IZQUIERDA
8
9   IF #I_MARCHA_DER AND NOT #I_MARCHA_IZQ THEN
10      #Q_MOTOR_DER_1 := TRUE;
11  END_IF; //SE MUEVE A DERECHA CUANDO PULSAMOS MARCHA DERECHA
```

Estas dos funciones tienen como condiciones que solo podamos girar el motor hacia un lado, por lo que, si estamos a izquierdas, girará a derechas en caso de pulsar la marcha a derechas, y viceversa.

10.2. CLASIFICACIÓN Y ORGANIZACIÓN

Para ordenar mejor la estructura de la función, usaremos una herramienta del software llamada REGIONES, con la cual clasificaremos las partes del programa en subprogramas que desempeñan esa función. La función tendría este aspecto:

```
1   IF #I_MARCHA AND (NOT #I_PARO OR NOT #Q_ERROR) THEN
2
3       #Q_RUN_1 := TRUE; //RUN PARA VER CUANDO ESTÁ EN MARCHA
4       REGION MOVIMIENTOS//-----------------
16      REGION ERRORES//-----------------
34      REGION AVISOS//-----------------
44      REGION ORIGENES//-----------------
103     REGION CONTROL_POSICIÓN//-----------------
192     END_IF;
193
```

Como vemos, hemos definido varias partes, como son los MOVIMIENTOS, los ERRORES, los AVISOS, los ORÍGENES y el CONTROL DE POSICIÓN.

10.3. MOVIMIENTOS

Para la región de MOVIMIENTOS tenemos la siguiente estructura:

```
REGION MOVIMIENTOS
    IF #I_MAN_AUTO AN ... THEN ... END_IF; //SE MUEVE A IZQUIERDA CUANDO PULSAMOS MARCHA IZQUIERDA MANUAL

    IF #I_MAN_AUTO AN ... THEN ... END_IF; //SE MUEVE A DERECHA CUANDO PULSAMOS MARCHA DERECHA MANUAL

    //I_MAN_AUTO = 1 PARA MANUAL. = 0 PARA AUTOMÁTICO

END_REGION//-----------------
```

Esta región ya se ha explicado anteriormente, con lo que pasaremos a la siguiente.

10.4. ERRORES

Esta es la región de ERRORES, en la cual, dependiendo del error, tendremos un código de error

```
REGION ERRORES
    IF (#Q_MOTOR_DER_1 AND #Q_MOTOR_IZQ_1) THEN
        #Q_ERROR := TRUE;
        #Q_TYPE_ERROR := 100;
    END_IF;//ERROR DE MOVIMIENTO
    IF (#ORIGEN_DER AND #ORIGEN_IZQ) THEN
        #Q_ERROR := TRUE;
        #Q_TYPE_ERROR := 200;
    END_IF;//ERROR DE ORÍGENES
    IF #I_LIM_INF > #I_LIM_SUP THEN
        #Q_ERROR := TRUE;
        #Q_TYPE_ERROR := 300;
    END_IF;//ERROR DE LÍMITES
    IF #Q_ERROR THEN
        #Q_MOTOR_DER_1 := #Q_MOTOR_IZQ_1 := FALSE;
    END_IF;

END_REGION//-----------------
```

Tendremos:

- Errores de movimiento: en caso de que se activen las variables de motor izquierda y motor derecha al mismo tiempo.
- Errores de origen: En caso de que se activen las variables de origen de ambos lados al mismo tiempo.
- Errores de límites: En caso de que el límite inferior sea superior al límite superior.

10.5. AVISOS

La siguiente región es la de AVISOS, y es la encargada de mostrar el texto dependiendo del código de error que tengamos en la anterior región.

```
REGION AVISOS
    IF #Q_TYPE_ERROR = 100 THEN
        #Q_AVISOS := 'ERROR DE MOVIMIENTO';
    ELSIF #Q_TYPE_ERROR = 200 THEN
        #Q_AVISOS := 'ERROR DE ORIGENES';
    ELSIF #Q_TYPE_ERROR = 300 THEN
        #Q_AVISOS := 'ERROR DE LIMITES';
    END_IF;

END_REGION//-----------------
```

10.6. ORÍGENES

La siguiente región es la de ORÍGENES, la cual, dependiendo si elegimos para el origen o posición cero un lado u otro.

Como vemos, seleccionando el origen como izquierda, al pulsar el pulsador para posición cero (Pulsador_Po), se desencadenan una serie de órdenes para el movimiento a posición cero.
Esto moverá el motor a izquierda a la velocidad normal que hayamos definido. Al accionar el sensor de puesta a cero, nos moveremos un poco a la derecha a un 10% de la velocidad definida hasta que dejemos de accionar el sensor de puesta a cero.

```
REGION ORIGENES
        IF #ORIGEN_IZQ AND NOT #Q_ERROR THEN
            IF #PULSADOR_Po AND NOT #SENSOR_Po THEN
                #AUX_1 := TRUE;
            END_IF;

            IF #AUX_1 THEN
                #Q_MOTOR_IZQ_1 := TRUE;
                #Q_VELOCIDAD_SAL_1 := #I_VELOCIDAD;
            ELSE
                #Q_MOTOR_IZQ_1 := FALSE;
            END_IF;

            IF #SENSOR_Po THEN
                #AUX_1 := FALSE;
                #Q_MOTOR_DER_1 := TRUE;
                #Q_VELOCIDAD_SAL_1 := #I_VELOCIDAD * 10 / 100;
            ELSE
                #Q_MOTOR_DER_1 := FALSE;

            END_IF;

            IF NOT #SENSOR_Po AND NOT #AUX_1 THEN
                #Q_VELOCIDAD_SAL_1 := 0;
            END_IF;
        ELSE
            #Q_VELOCIDAD_SAL_1 := 0;
            #Q_MOTOR_IZQ_1 := FALSE;
            #AUX_1 := FALSE;
        END_IF;      //PARA ORIGEN IZQUIERDA
```

Como vemos, si elegimos como origen la derecha, ocurre exactamente el mismo proceso que el anterior, en sentido contrario.

```
IF #ORIGEN_DER AND NOT #Q_ERROR THEN

    IF #PULSADOR_Po AND NOT #SENSOR_Po THEN
        #AUX_2 := TRUE;
    END_IF;

    IF #AUX_2 THEN
        #Q_MOTOR_DER_1 := TRUE;
        #Q_VELOCIDAD_SAL_1 := #I_VELOCIDAD;
    END_IF;

    IF #SENSOR_Po THEN
        #AUX_1 := FALSE;
        #Q_MOTOR_IZQ_1 := TRUE;
        #Q_VELOCIDAD_SAL_1 := #I_VELOCIDAD * 10 / 100;
    ELSE
        #Q_MOTOR_IZQ_1 := FALSE;

    END_IF;

    IF NOT #SENSOR_Po AND NOT #AUX_2 THEN
        #Q_VELOCIDAD_SAL_1 := 0;
    END_IF;

ELSE
    #Q_MOTOR_DER_1 := FALSE;
END_IF;      //PARA ORIGEN DERECHA
END_REGION//-----------------
```

10.7. CONTROL DE POSICIÓN MEDIANTE ENCODER

A continuación, y, por último, tenemos la región de control de posición. Aquí mezclamos algunas partes del programa ya que tienen cierta relación entre sí, como se verá.

Primeramente, nos encontramos con la conversión de pulsos del encoder. Se trata de una serie de operaciones muy sencillas:
Necesitamos un par de variables de entrada para las operaciones, como son los pulsos por vuelta (I_PUL_VTA) y los mm por vuelta (I_M_VTA):

$$AUX_1 = \frac{\frac{pulsos}{\cancel{vuelta}}}{\frac{mm}{\cancel{vuelta}}} = \frac{pulsos}{mm};$$

Estas variables vienen dadas por el propio eje y por el encoder.

A continuación, con los pulsos recibidos del encoder, podremos hallar la distancia que recorre el motor.

$$AUX_2 = \frac{\cancel{pulsos_{encoder}}}{\frac{\cancel{pulsos}}{mm}} = mm\ recorridos$$

```
REGION CONTROL_POSICIÓN
    IF #I_POT_ENCO THEN //1 PARA ENCODER, 0 PARA POTENCIOMETRO
        #AUX_ENCOD_1 := #I_PUL_VTA / #I_M_VTA; //pulsos/vuelta dividido mm/vuelta = pulsos/mm
        #AUX_ENCOD_2 := #I_ENCOD / #AUX_ENCOD_1; //pulsos dividido pulsos/mm = mm
        #Q_POS_REAL := #AUX_ENCOD_2;
```

Así pues, la posición real se refleja en la variable auxiliar dos, y esta a su vez la refleja en la variable de salida (Q_POS_REAL).

A continuación, tenemos el movimiento automático, el cual se activará cuando (I_MAN_AUTO) sea 0 o FALSE.
Empezaremos con una variable de posición que irá sumando a medida que el eje llegue a la posición deseada.

Como se ve, por cada posición, hay una tanda de posibles movimientos dependientes de la posición de la parte móvil del eje: Si la posición del eje es menor que la posición [1] entonces habrá que mover el motor a izquierda, y si es mayor, a derecha. En caso de que no sea ni mayor ni menor (que sea igual) el motor no hará nada, y se sumará 1 a la variable de posición. Así se hará con todas las posiciones. Se ha hecho hasta 5 de ellas, con posibilidad de aumentarse en caso de requerirse.

```
IF NOT #I_MAN_AUTO THEN
    IF #INT_POS = 0 THEN
        IF #AUX_ENCOD_2 < #I_POS_1 THEN
            #Q_MOTOR_IZQ_1 := TRUE;
            #Q_MOTOR_DER_1 := FALSE;
            #VAR_POS_1 := TRUE;

        ELSIF #AUX_ENCOD_2 > #I_POS_1 THEN
            #Q_MOTOR_DER_1 := TRUE;
            #Q_MOTOR_IZQ_1 := FALSE;
            #VAR_POS_1 := TRUE;

        ELSE
            #INT_POS := #INT_POS + 1;
        END_IF;
    END_IF;

    IF #INT_POS = 1 THEN
        IF #AUX_ENCOD_2 < #I_POS_2 THEN
            #Q_MOTOR_IZQ_1 := TRUE;
            #Q_MOTOR_DER_1 := FALSE;

        ELSIF #AUX_ENCOD_2 > #I_POS_2 THEN
            #Q_MOTOR_DER_1 := TRUE;
            #Q_MOTOR_IZQ_1 := FALSE;

        ELSE
            #INT_POS := #INT_POS + 1;
        END_IF;
    END_IF;

    IF #INT_POS = 2 THEN
        IF #AUX_ENCOD_2 < #I_POS_3 THEN
            #Q_MOTOR_IZQ_1 := TRUE;
            #Q_MOTOR_DER_1 := FALSE;

        ELSIF #AUX_ENCOD_2 > #I_POS_3
        THEN
            #Q_MOTOR_DER_1 := TRUE;
            #Q_MOTOR_IZQ_1 := FALSE;

        ELSE
            #INT_POS := #INT_POS + 1;
        END_IF;
    END_IF;

    IF #INT_POS = 3 THEN
        IF #AUX_ENCOD_2 < #I_POS_4 THEN
            #Q_MOTOR_IZQ_1 := TRUE;
            #Q_MOTOR_DER_1 := FALSE;

        ELSIF #AUX_ENCOD_2 > #I_POS_4 THEN
            #Q_MOTOR_DER_1 := TRUE;
            #Q_MOTOR_IZQ_1 := FALSE;

        ELSE
            #INT_POS := #INT_POS + 1;
        END_IF;
    END_IF;

    IF #INT_POS = 4 THEN
        IF #AUX_ENCOD_2 < #I_POS_5 THEN
            #Q_MOTOR_IZQ_1 := TRUE;
            #Q_MOTOR_DER_1 := FALSE;

        ELSIF #AUX_ENCOD_2 > #I_POS_5 THEN
            #Q_MOTOR_DER_1 := TRUE;
            #Q_MOTOR_IZQ_1 := FALSE;

        ELSE
            #INT_POS := 0;
        END_IF;
    END_IF;
END_IF;

END_IF; //CONVERSIÓN ENCODER
```

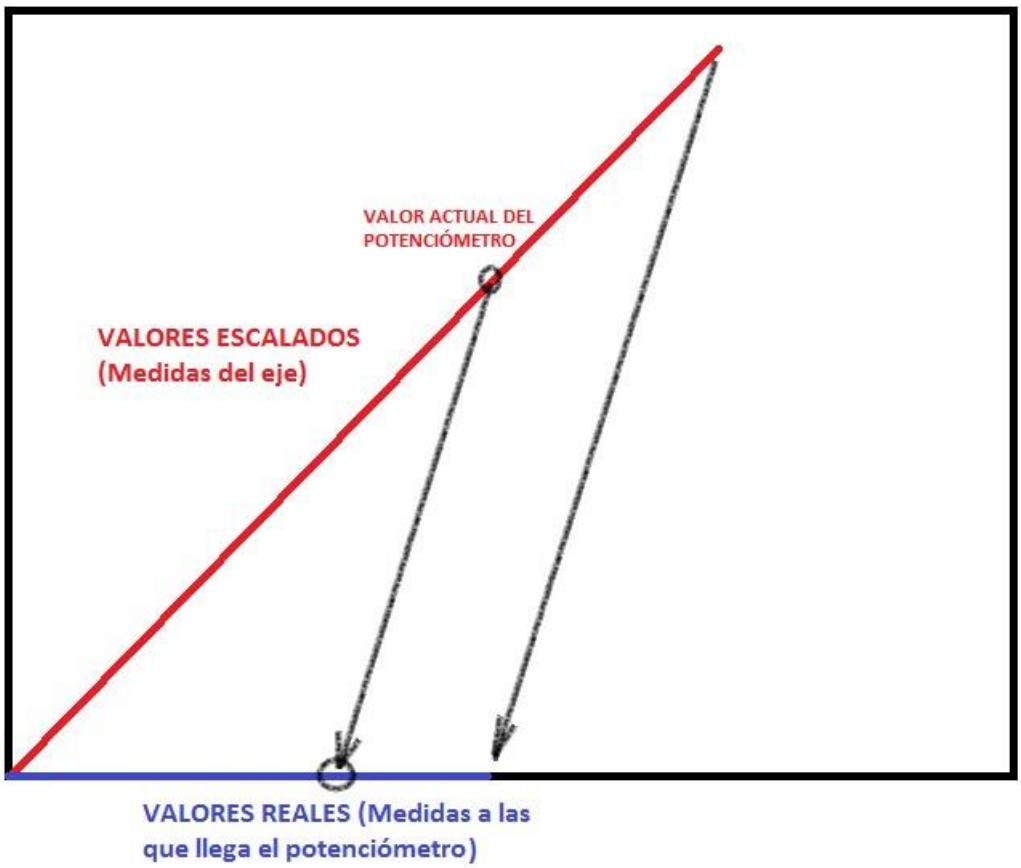
10.8. CONTROL DE POSICIÓN MEDIANTE POTENCIÓMETRO

Finalmente, en caso de que quisiéramos mover el eje con un potenciómetro, tendríamos esa opción como entrada, y ello se rige mediante la siguiente parte del programa.

Se basa en un escalado entre los límites superior e inferior del eje del valor que tome el potenciómetro en la entrada al autómata. Este escalado se guarda en la variable ESCALADO_POT, y ésta será la posición que tome el motor en el eje.

```
    IF NOT #I_POT_ENCO THEN
        #ESCALADO_POT := SCALE_X_INT(MAX := #I_LIM_SUP, MIN := #I_LIM_INF, VALUE := #I_POTEN);
        //ESCALA EL VALOR DEL POTENCIOMETRO ENTRE EL LIMITE SUPERIOR Y EL INFERIOR
        //EL RESULTADO SE GUARDA EN LA VARIABLE #ESCALADO_POT
        #Q_POS_REAL := #ESCALADO_POT;
    END_IF; //ESCALADO POTENCIOMETRO
END_REGION//-----------------
```

A continuación, tenemos un ejemplo gráfico de lo que haría la función provista por TIA Portal para el escalado.

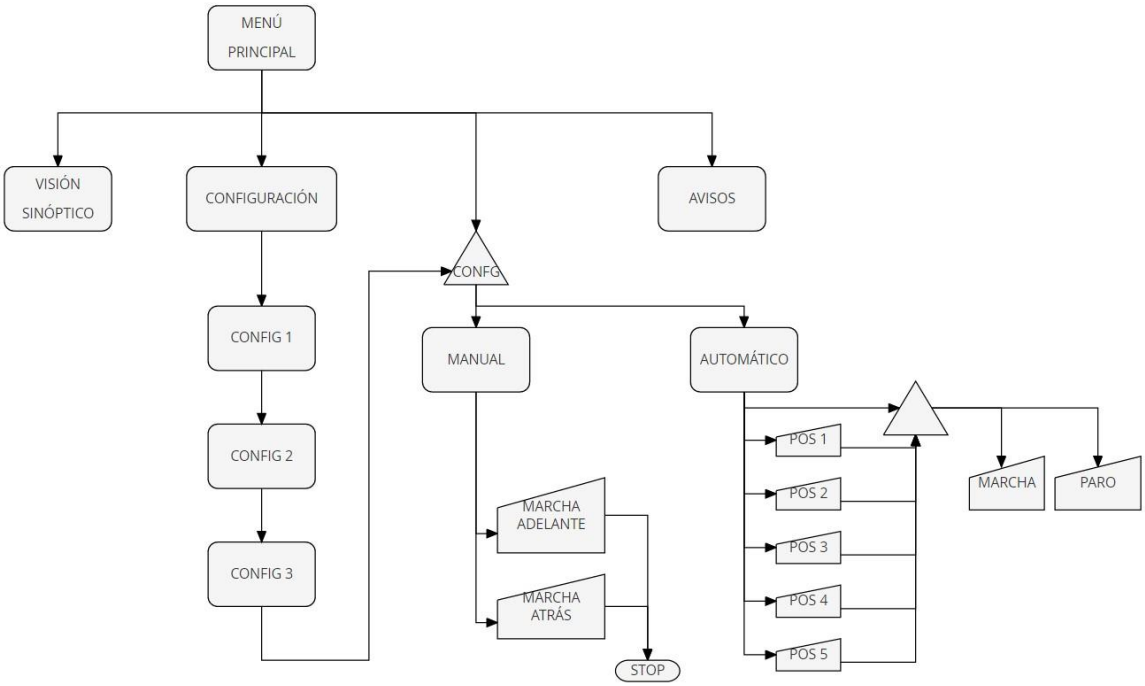


11.SCADA

El SCADA del programa se basa en una serie de paneles o "IMÁGENES" por las que nos moveremos para la maniobra del puente grúa, En este apartado vamos a describir de forma detallada cada una de las ventanas que contiene nuestra pantalla táctil para el correcto funcionamiento de nuestro control y para tener una correcta visualización del proceso. Se recalca que este **es solo una propuesta de SCADA**. Se han tenido en cuenta únicamente las entradas y salidas de la propia función. En caso de querer extender el proyecto más allá de la función, se podría

ampliar el número de paneles para la implementación de nuevas herramientas, habiendo antes diseñado un programa apto para ello.

11.1. FLUJOGRAMA SCADA



Este es el funcionamiento que tiene el SCADA. Primeramente, desde el menú principal podemos acceder a los demás paneles. Como vemos, tenemos el panel de "Visión sinóptico", el de "Configuración", el de "Manual", el de "Automático" y el de "Avisos".

Podemos observar que, para acceder a los paneles de Manual y Automático, como condición, debemos haber hecho la configuración inicial a través de todos los paneles de configuración.

Dentro de Manual, podremos accionar manualmente la marcha adelante y la marcha atrás. Éstas pararan con el botón de STOP.

Para el apartado de movimientos Automáticos, como en el caso de la configuración, deberemos definir las distintas posiciones, definidas como condición, para accionar los botones de marcha y paro.

Finalmente, podremos acceder libremente al panel de Avisos desde el Menú principal.

11.2. MENÚ PRINCIPAL

Este panel es el menú principal del SCADA. Desde éste podemos manejarnos a través de los diferentes paneles. Tenemos diferentes opciones, de arriba abajo:

- Visión sinóptico: Esta opción nos redirige al panel de visión del sinóptico.
- Configuración: Esta opción nos redirige al primer panel de Configuración.

Bajo el panel de Aviso, tenemos las dos opciones de movimientos diferentes:

- Movimientos manuales: Nos redirige al panel de Movimientos Manuales.
- Movimientos automáticos: Nos redirige al panel de movimientos automáticos.

En la base de este panel, nos encontramos la opción de Panel de Avisos, la cual nos redirige al panel con los diferentes avisos.

A la izquierda de éste, tenemos un botón que nos lleva al Menú Principal, sin ninguna función en este panel.

Si pulsamos sobre Configuración, observamos que éste está protegido por contraseña, debido a que son opciones delicadas, y solo puede acceder a ellas el Jefe del Equipo, o una autoridad mayor al operario:

El usuario y contraseña son los siguientes:

USER: JEFE_DE_EQUIPO

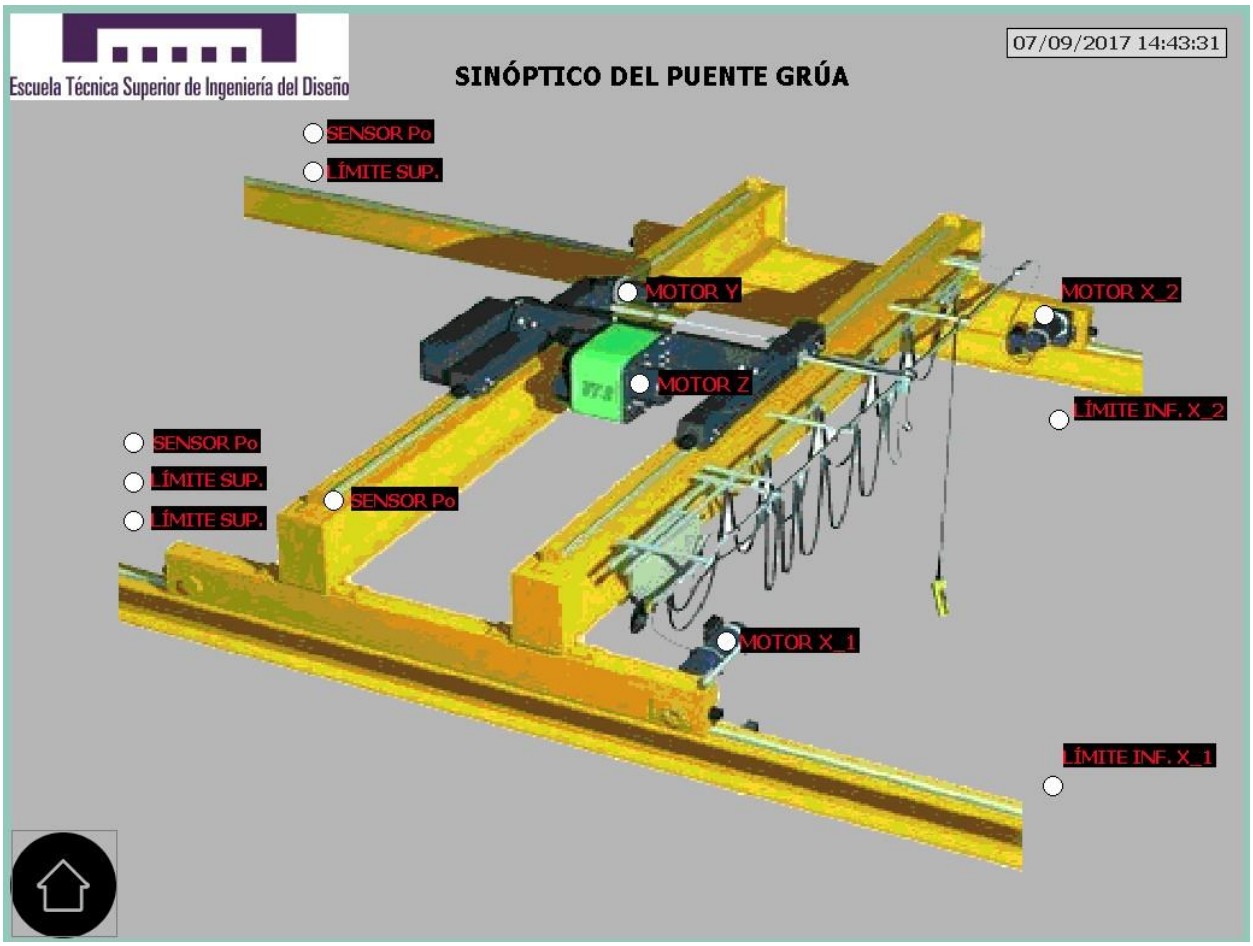
PASS: EQUIPO123

Una vez demos en ACEPTAR, nos dejará entrar en el apartado de Configuración.

Más adelante veremos como la ventana de AVISO desaparecerá una vez configurados los parámetros en Configuración.

Una vez se hayan configurado los parámetros, el menú principal aparecerá tal que así:

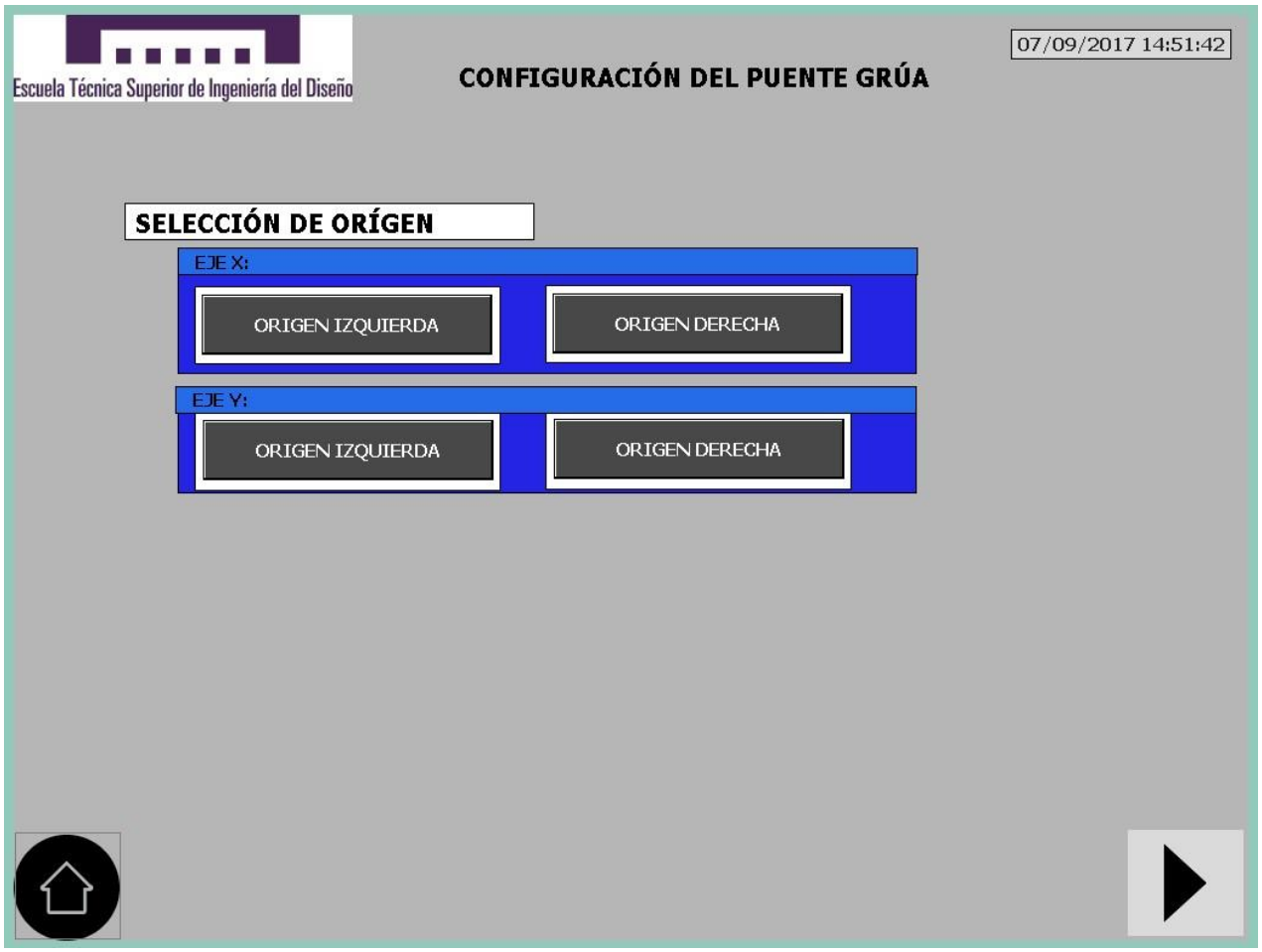
11.3. VISIÓN SINÓPTICO.



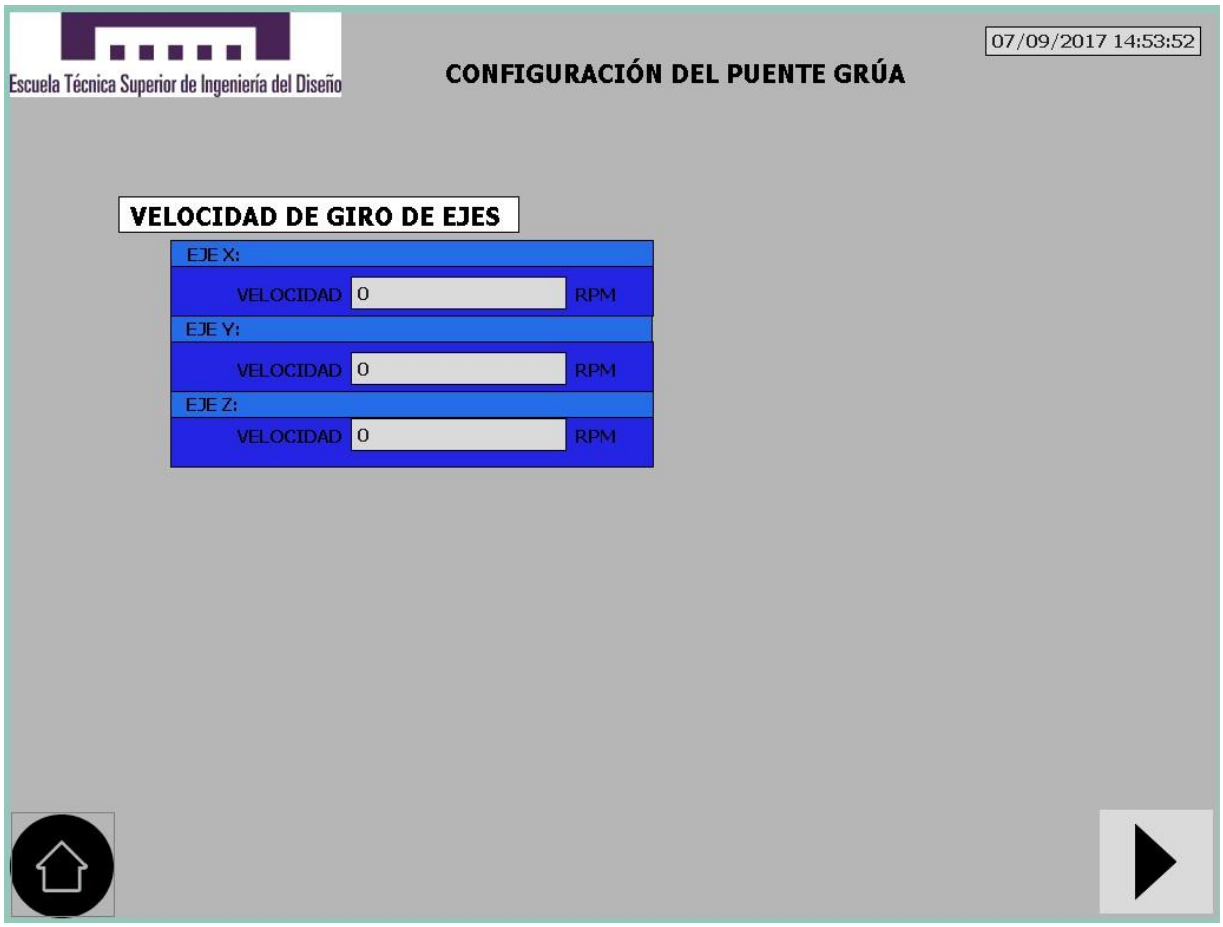
Este panel es el sinóptico del puente grúa. Tenemos una distribución gráfica de unos pilotos que representan las diferentes salidas y entradas del PLC. Se encenderán en verde cada vez que se activen. Este panel se puede usar como sistema de comprobación de Poka-Yokes (llamados así en la industria a las detecciones de errores, es una técnica de calidad que se aplica con el fin de evitar estos errores en la operación de un sistema.), aunque se podría crear un nuevo panel para la comprobación de éstos.

11.4. CONFIGURACIÓN

Este es el primer panel de Configuración. En éste definiremos los límites físicos de cada eje. El eje Z definirá la longitud desde el puente grúa al suelo. Hay que tener en cuenta que el límite superior ha de ser siempre superior al límite inferior, o tendremos un aviso de error.



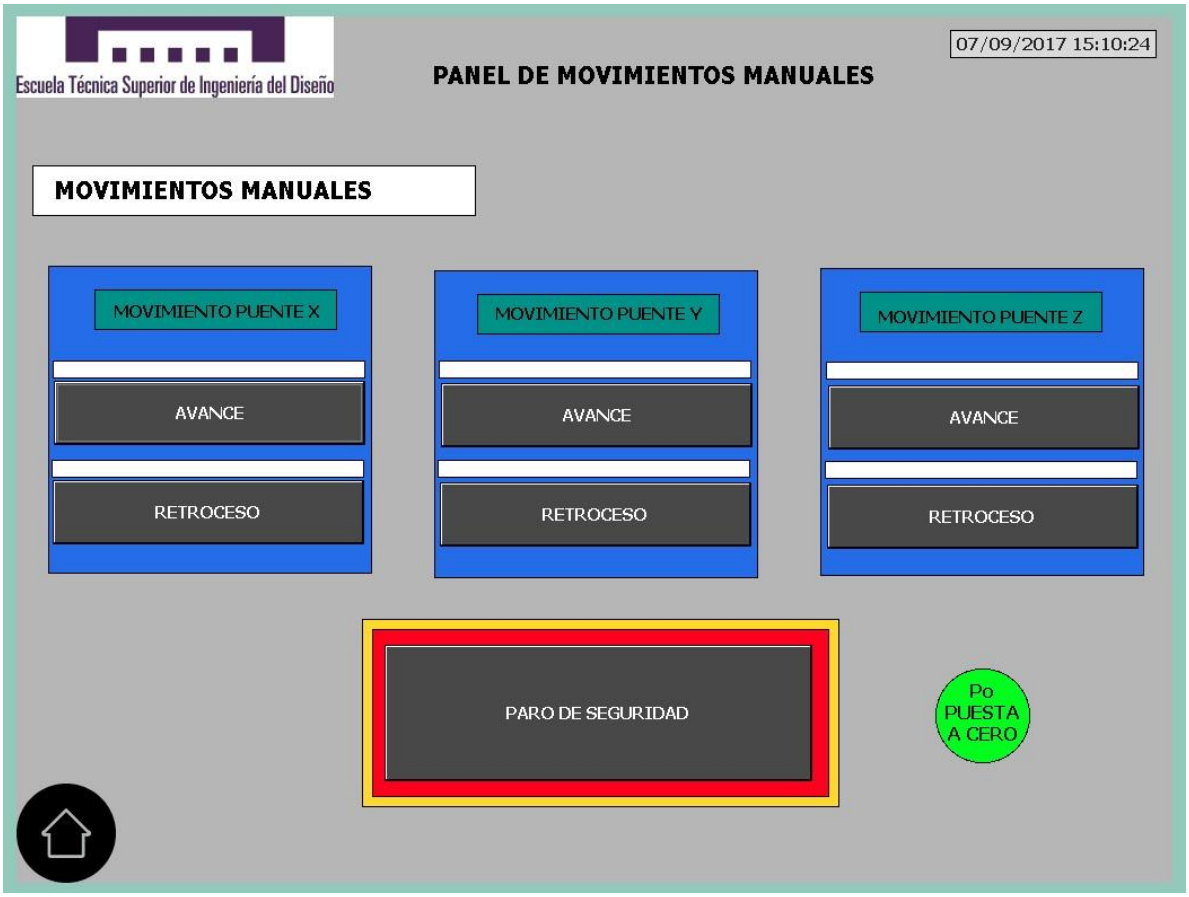
Este panel es en el que seleccionaremos el origen de cada eje. Se supone que el eje Z tiene el origen en el polipasto. Como se ha explicado, en el programa tendremos dos opciones por eje: origen izquierda u origen derecha. Dependiendo de los intereses de la empresa contratante, y de la estructura del edificio objetivo, se seleccionará un origen u otro en cada eje. Recalcar que son independientes, se puede seleccionar el mismo origen en ambos ejes.



El último panel de configuraciones es para definir la velocidad de cada eje en RPM. Se definirán unos valores de seguridad para evitar una velocidad excesiva.

Las velocidades dependerán del paso que tenga el eje, y por tanto dichos límites.

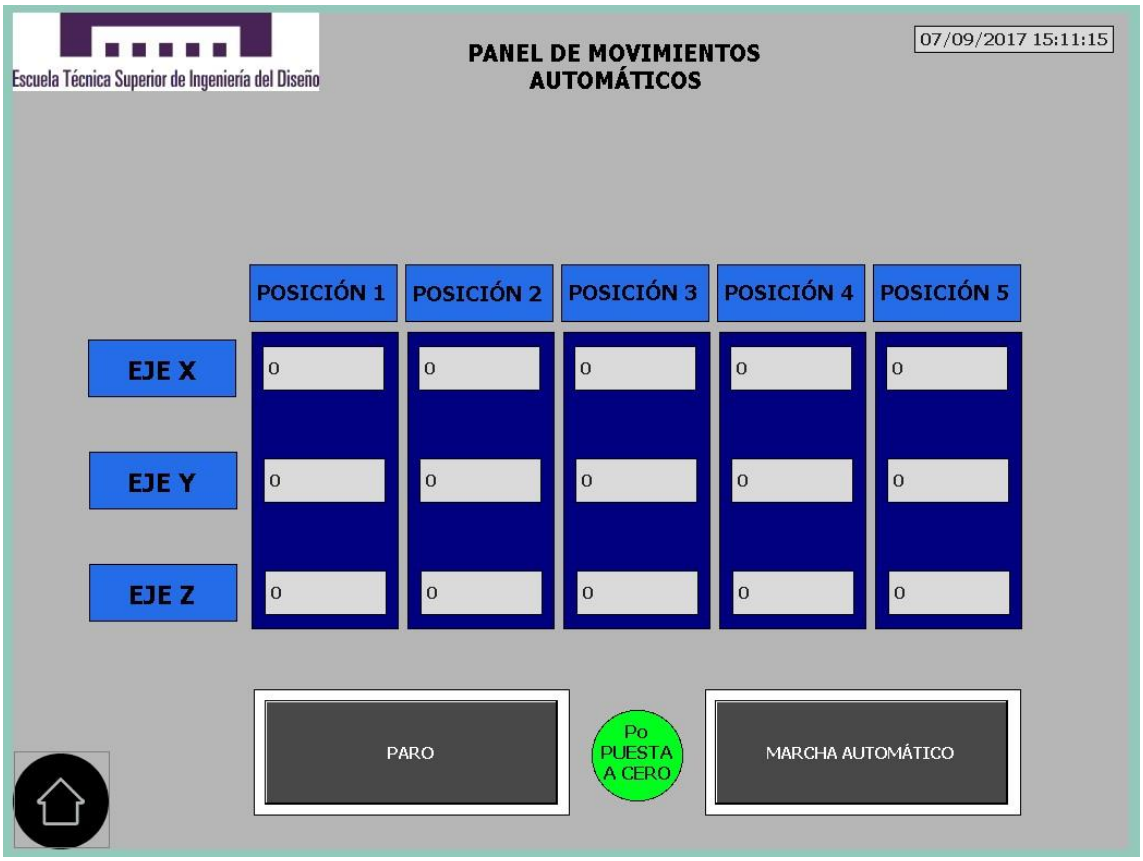
11.5. PANEL DE MOVIMIENTOS MANUALES



Este panel es el de movimientos manuales. Este panel da una salida directa a los diferentes variadores y contactores, y se podrá controlar los 3 ejes de manera manual. Se dispone además de un paro de seguridad. Éste, mientras esté activo, no dejará controlar el puente-grúa manualmente.

Tiene además un pulsador de PUESTA A CERO, que devolverá los ejes a su punto de origen establecido en la configuración.

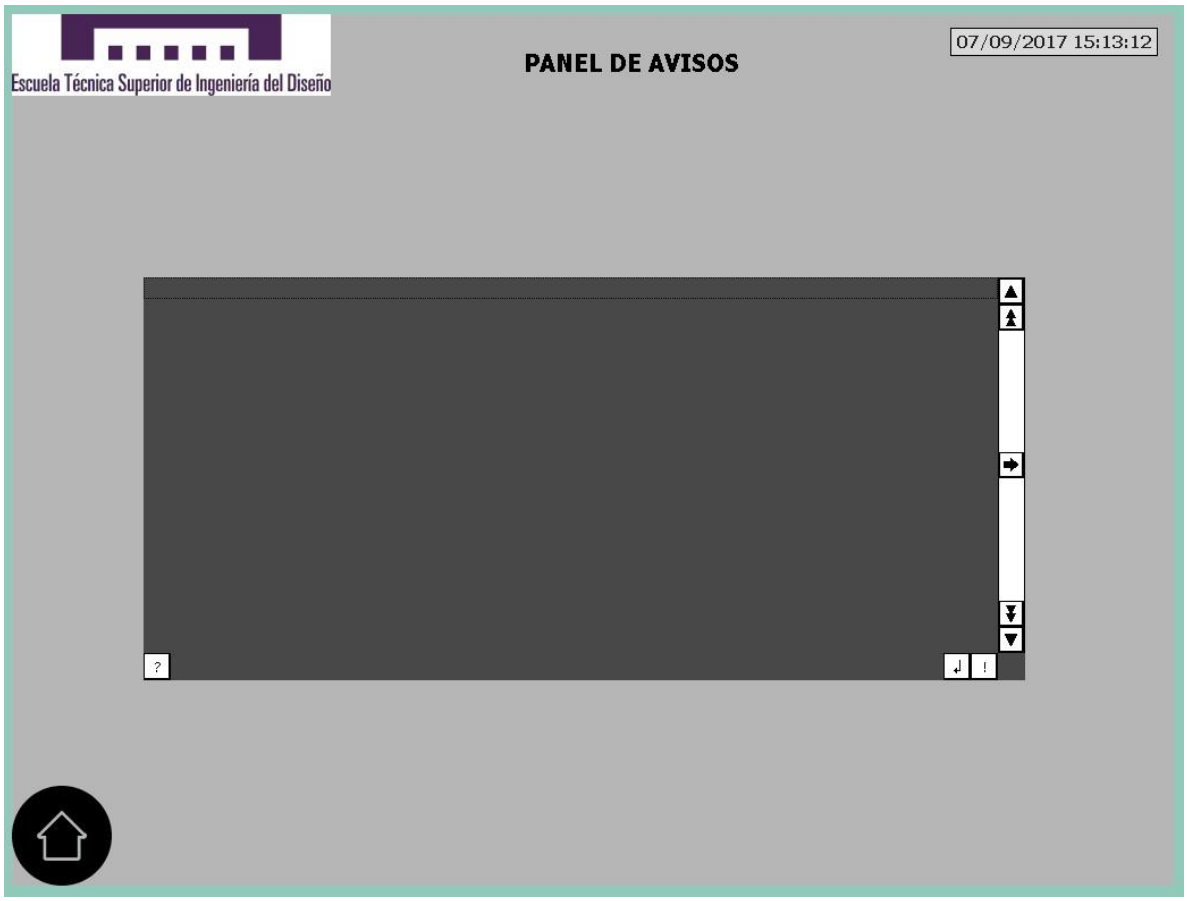
11.6. MOVIMIENTOS AUTOMÁTICOS



Este panel es el de movimientos automáticos. En éste, tras un estudio de los diferentes movimientos que se precisan en el lugar objetivo, se declararán las coordenadas en 5 diferentes posiciones por cada eje del que se haga uso. Una vez definidas las coordenadas, se dará a MARCHA AUTOMÁTICO para empezar el ciclo automático. Se parará cuando le demos a paro.

Tiene además un pulsador de PUESTA A CERO, que devolverá los ejes a su punto de origen establecido en la configuración.

11.7. PANEL DE AVISOS



En este panel, se observará los diferentes avisos debidos a errores de selección, como movimientos, orígenes, límites, etc…

12.CONCLUSIONES

Las conclusiones que podemos extraer de este trabajo son varias.

Podemos extraer de nuestra educación técnica, y es que nunca estaremos totalmente preparados para todo lo que venga, necesitamos estar en constante renovación en el mundo de los automatismos, ya que siempre van saliendo nuevas mejoras, nuevos componentes y tecnología que nos puede hacer las tareas mucho más sencillas. Un ejemplo de ello es el software que hemos utilizado, el cual dispone de muchas herramientas que nos han solucionado muchos problemas que ni siquiera nos planteábamos. Simplemente hemos visto esas herramientas y las hemos usado.

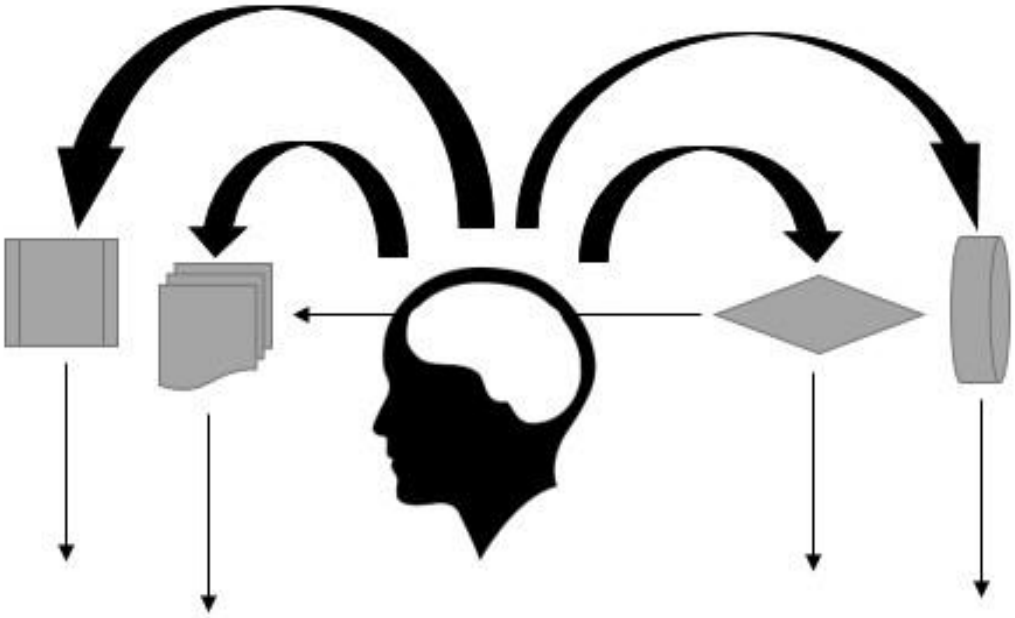
Desde el punto de vista de la programación, nos ha sido muy útil utilizar este software, ya que es una parte importante dentro de la automatización industrial, al ser utilizado por la mayoría de empresas. Nos ha ayudado a tener una mejor comprensión de los sistemas de lenguaje utilizados por SIEMENS, y además, a nivel genérico, tenemos una mejor organización de los programas que realicemos, acostumbrándonos a mejores hábitos de programación y evitando así fallos comunes.

Es apropiado destacar el gran manejo y la brevedad que impone el lenguaje estructurado. Lo vemos por ejemplo en programas hechos y otras funciones genéricas, en estándares de grandes empresas también, los cuales abogan por la sencillez de lectura de sus programas.

Me ha servido personalmente, además, para labrarme una pequeña formación en el ámbito de SIEMENS, y no solo en esta empresa, sino que también me ha ofrecido una visión más amplia de lo que supone un proyecto de envergadura mucho mayor a la que acostumbramos a encarar.

Por último, y como base de este proyecto, me gustaría recalcar la utilidad de usar funciones genéricas, las cuales usamos siempre en los softwares. Yo mismo he usado funciones para trabajos pequeños que habrían costado de hacer varias líneas más de código. En este caso, el crear esta función genérica ahorrará al que la use más de 100 líneas de código, y es que se convierten en herramientas muy útiles y eficientes.

De este proyecto, me llevo mejoras, si, pero también me llevo muchas cosas por mejorar, y que tendré presente en el futuro laboral.



CONTROL DE PUENTE GRÚA MEDIANTE AUTÓMATA PROGRAMABLE BASADO EN UNA FUNCIÓN GENÉRICA DE EJE.

PLIEGO DE CONDICIONES

CÉSAR ROMERO BARBERÁ

Contenido

1.- CONDICIONES GENERALES

1.1.- OBJETIVO DEL PLIEGO

El objetivo del pliego de condiciones es constituir un acuerdo entre propiedad y contratista para la ejecución del proyecto presentado.

Tenemos cuatro condiciones básicas para abarcar:

Condiciones técnicas: Son las condiciones de 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 realizara 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 del mismo. 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 graduado de Ingeniería Eléctrica, solamente se tendrá en cuenta los diferentes puntos, sin llegar a desarrollarlos enteramente, y citando la normativa y legislación que se debe seguir.

1.2. NORMATIVA VIGENTE.

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

En lo que respecta a la parte eléctrica y electrónica se ha utilizado la normativa expuesta en el Reglamento Electrotécnico para Baja Tensión, aprobado por el Real Decreto 842/2002, de 2 de agosto y publicado en el BOE nº224, de 18 de septiembre de 2002. Y de sus Instrucciones Técnicas Complementarias basadas en las normas UNE, en especial en la UNE 20 460. Cabe destacar las siguientes instrucciones:

- 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 para viviendas y edificios.

También se ha tenido en cuenta la normativa sobre Seguridad Industrial expuesta por el Ministerio de Industria, Turismo y Comercio. Así como la Directiva 2006/42/CE del parlamento europeo y del consejo sobre máquinas y la normativa para la seguridad de maquinaria EN418. Debido al carácter educacional de este proyecto no se tendrán en cuenta las medidas de seguridad que se especifican en la norma de operación remota sobre máquinas.

2. CONDICIONES A SATISFACER POR LOS COMPONENTES

2.1. COMPONENTES PASIVOS

2.1.1. CONDUCTORES

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

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

Los cables que soporten tensiones de 24V provenientes de la fuente de alimentación del autómata al igual que los que soporten 15V provenientes del polo común del variador de frecuencia se conectarán mediante cables de 0.5 mm² de sección. Estos cables podrán ser independientes o formar parte de mangueras. Los elementos conectados a corriente alterna se conectaran mediante mangueras de conductores de 1.5 mm^2 excepto la alimentación de los motores que se realizara con mangueras de tres conductores y toma de tierra de 2.5 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.

2.2. ORDENADOR PERSONAL

El ordenador utilizado deberá contar con una CPU, un monitor, un ratón y un teclado. Debido a su utilización en el ámbito industrial se deberán tener en cuenta las condiciones ambientales en las que dicho ordenador deberá trabajar como son la temperatura, la humedad, la suciedad, elementos corrosivos… La temperatura de trabajo de este elemento estará comprendida entre los 0ºC y los 60ºC. Nos basaremos sobretodo en las especificaciones que recomienda Siemens para la instalación y uso del software de programación TIA Portal. El PC utilizado deberá tener las siguientes características técnicas como mínimo:

- Procesador Intel® Core™ i5-3320M 3,3 GHz o superior
- Memoria RAM 8 GB o más
- Espacio libre en el disco duro de 2Gb. –
- Sistema operativo Windows 7 o más reciente.

2.3. AUTÓMATA PROGRAMABLE

El autómata programable estará alimentado por una fuente de alimentación de corriente alterna a 220V. Los módulos que quieran conectarse estarán conectados también a 220V antes de relé de emergencia. Se deberá respetar y dejar libre el 20% de las entradas y salidas totales de la CPU y módulos. También se deberá dejar libre el 20% de las entradas del switch en caso de que hubiera.

3. CONDICIONES CONSTRUCTIVAS

3.1. MONTAJE ELÉCTRICO

El montaje eléctrico se sustentará sobre un cuadro eléctrico en un armario para tal fin. En él se colocarán todos los elementos eléctricos necesarios como serán:

- Interruptores automáticos.
- Magnetotérmicos.
- Bases auxiliares schucko.
- Variadores de frecuencia.
- Borneros.
- CPU y rack junto a los respectivos módulos.
- Canaletas para conductores.
- Conductores.

Los conectores para los sensores y finales de carrera serán de 3 pines de M12. El ingeniero responsable llevará a cabo las modificaciones eléctricas necesarias para una buena optimización de la instalación.

4. PRUEBAS DE FUNCIONAMIENTO.

Antes de conectar a la red el proceso se realizarán unos ensayos para verificar la buena conexión de todos los componentes, basado en la normativa vigente en el Reglamento de Baja Tensión, en la ITC-bt-05: Verificaciones e inspecciones.

4.1. REVISION VISUAL Y DE CONTINUIDAD.

Esta revisión consiste en verificar que todos los componentes, elementos y sensores están bien conectados, y sus conexiones y cableado están bien sujetos, sin riesgo de cortocircuito y/o falta a tierra.

4.2. PRUEBAS EN TENSIÓN

Se conectará la máquina a la red de alimentación y entonces se comprobará primero los dispositivos de seguridad.

Una vez hecho esto, se probarán varias funciones y/o ciclos del programa cargado en la máquina para poder chequear el funcionamiento de todos los sensores y componentes.

4.3. PRUEBA FINAL

Se realizará una prueba de movimiento para la traslación de carro, para la traslación del puente y finalmente para el movimiento del polipasto.

5. CONTROL DE CALIDAD.

Mediante la compra de los diferentes componentes para la instalación, se supone una verificación de calidad por parte del fabricante, por tanto, no es competencia del contratista ni del proyectista. Sin embargo, éste último deberá encargarse de la revisión del buen funcionamiento de los componentes que usa para evitar riesgos y futuros desastres.

Habrá que hacer una revisión visual de los aislantes de los cables en busca de defectos de fábrica o debidos a complicaciones en la instalación, pues esto evitará cortocircuitos y derivaciones a tierra.

Habrá que comprobar que los componentes que sirven a la seguridad estén en perfecto funcionamiento, esto es: Seta de emergencia, relé de seguridad y en caso de que lo hubiera, relé de puertas y relé de barreras, así como los finales de carrera de puertas y las propias barreras.

Respecto a la parte de programación, el propio cliente juzgará junto al programador si el programa, o la función en este caso, hace lo que debe hacer y si se adapta a los objetivos del proyecto.

6. CONDICIONES DE EJECUCIÓN.

Las condiciones de ejecución de este proyecto serán asegurarse de tener todos los componentes y sensores que puedan llevar a cabo el funcionamiento de la función programada, como pueden ser los finales de carrera, los sensores inductivos, los motores y los encoders.

7. COMPRA DEL MATERIAL

Los distintos componentes del puente grúa son impuestos por la empresa contratista, y por lo tanto no serán tenidos en cuenta en este proyecto.

El único material a tener en cuenta será la compra de la CPU, el cual, como se especifica en la memoria del proyecto, se elige el más adecuado a las necesidades del programador y del proyecto.

Al tratarse de tan poco material, no es necesario recalcar las dificultades que habría en la compra de los distintos componentes, ya que solamente habría que hacer un pedido a algún distribuidor cercano. Estos son algunos de los distribuidores más cercanos en nuestro caso (VALENCIA):

- Valektra.
- Disai.
- INRA.

8. CONDICIONES ECONÓMICAS

8.1. MEJORAS DEL PROYECTO INICIAL

Como se especifica en la memoria, éste es un proyecto educacional. Esto implica que las mejoras se hacen constantemente junto a la ayuda del profesor. De modo que se empieza por una función básica y, a modo de aprendizaje, se investiga sobre el software de programación las diferentes mejoras que puede aportarnos a nuestra función genérica. Por lo que el proyecto está en constante renovación durante su proyección.

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 cinco 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 seis y treinta días naturales después de la recepción del producto definitivo por parte del contratante, el producto sufre un recargo del 2% sobre el precio inicial previsto.
- Si se efectúa el pago entre los treinta uno y sesenta días posteriores a la recepción definitiva, este sufre un recargo del 4% sobre el importe establecido.
- Para un aplazamiento del pago entre los sesenta y un y noventa y un días tras la recepción definitiva, el producto sufre un recargo del 6,5% sobre el precio final.
- Si se retrasase el pago entre los noventa uno y trescientos días naturales seguidos después de la entrega del producto, el precio de este sufriría un recargo del 30% sobre el presupuesto inicial.
- En caso de que el contratante optara 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 tendrá derecho a demandarle ante los tribunales.

9. CONDICIONES LEGALES

9.1. CONTRATO

El contrato deberá realizarse por escrito, cumplir todos los requisitos legales y estar firmado por todas las partes implicadas.

En este contrato se deberán especificar el precio inicial de fabricación del producto y su coste unitario. A este precio se le añadirán, si es necesario, las tarifas impuestas en el apartado anterior. Además, deberán aparecer todas aquellas cláusulas que deban cumplir alguna de las partes.

9.2. ADJUDICACIÓN DE LA CONTRATA

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

9.3. ARBITRAGE Y JURISDICCIÓN

Si en el caso de producirse algún desentendimiento entre las dos partes y estas no se pusieran por ellas mismo en acuerdo, se nombraría a un técnico por cada una de las partes para que llegaran a un acuerdo entre ellos.

9.4.- IMPUESTOS

Se exigirá 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 añadirá un impuesto sobre el precio del mismo correspondiente al impuesto sobre el valor añadido (I.V.A.) que está estipulado actualmente en un 18%. 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 será producida por el retraso sin previo aviso de la entrega del producto del contratista al contratante.

10.- 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 deberán 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 deberá cumplir con la normativa establecida en el Reglamento Electrotécnico de Baja Tensión.

Deberá 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 podrá 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 deberá abonar un importe del 10% del importe de fabricación.

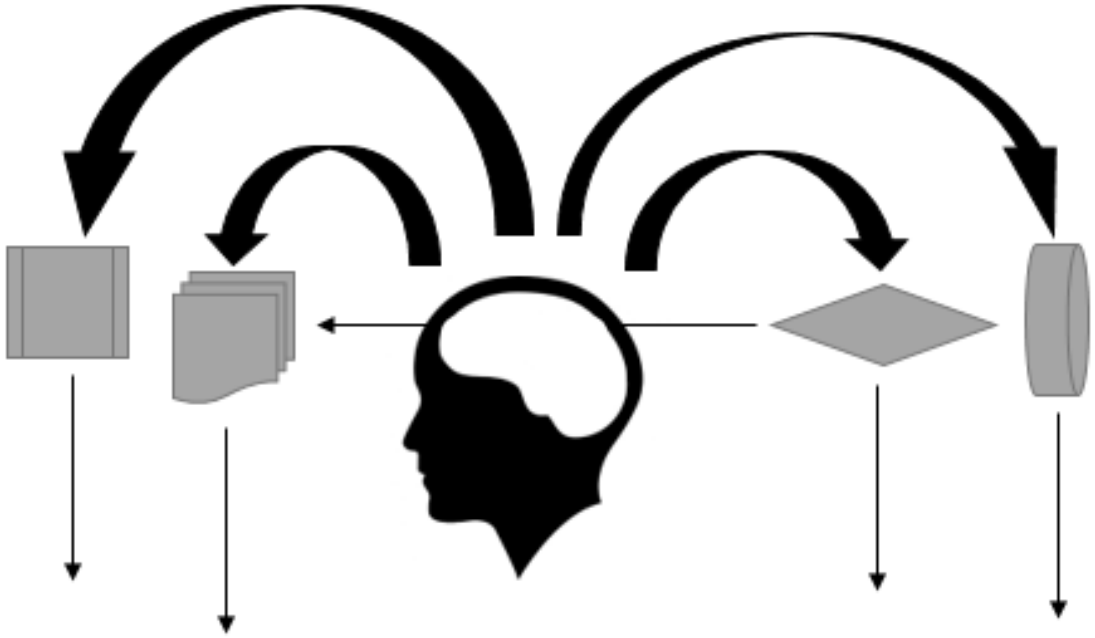
Los elementos fabricados cumplirán con los requisitos especificados en el proyecto y solo se podrán modificar con la aprobación del proyectista.

El contratante deberá facilitar al contratista todos los elementos e información necesaria para una realización eficiente y rápida del proceso de fabricación.

Además, deberá 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 será competencia del proyectista comprobar el cumplimiento de las comprobaciones especificadas por parte de la empresa instaladora.



CONTROL DE UN PUENTE GRÚA MEDIANTE AUTÓMATA PROGRAMABLE BASADO EN UNA FUNCIÓN GENÉRICA DE EJE

PRESUPUESTO

Contenido

1. COSTE HARDWARE

Dentro del hardware especificaremos todos los elementos de control, como son el ordenador para la programación del proyecto y la CPU elegida.

Primeramente, vamos a elegir el ordenador de programación mínimo que necesitaríamos, nos fijaremos en los requisitos mínimos para instalar el software:

Requisitos mínimos de hardware:

Artículo	Cantidad	Precio/ud.	Precio
Procesador: CoreTM i5-3320M 3.3 GHz o similar	1	192,49€	192,49€
Memoria principal: 8 GB de memoria (recomendado) o más	1	65€	65€
Disco duro: 300 GB SSD	1	130€	130€
Pantalla: 15,6" display de pantalla ancha (1920 x 1080)	1	160€	160€

Requisitos mínimos de software

Artículo	Cantidad	Precio/ud.	Precio
Windows 10 Profesional	1	279,00 €	279,00 €

Subtotal de precios (Hardware)
826 €

(Los precios estimados son medias de varias marcas y gamas, no se asegura que ese sea el precio final)

Éste sería el precio estimado de lo que necesitaríamos para correr el programa con facilidad y sin problemas.

Este precio subiría un poco más para poder abarcar los demás componentes del ordenador. Podríamos prever fácilmente **1200€.** Esto nos aseguraría tener un ordenador que no nos de problemas en unos cuantos años para programar proyectos más grandes que éste.

Respecto a la CPU, podemos encontrar una SIEMENS 1214C AC/DC/RLY con referencia 6AG1214-1BG40-5XB0 y con diferentes precios. El precio a continuación es un precio estimado, provisto por un proveedor de automatismos:

Artículo	Cantidad	Precio/ud.	Precio
SIEMENS 1214C AC/DC/RLY 6AG1214-1BG40-5XB0	1	471,89 €	471,89 €

2. COSTE SOFTWARE

A continuación, necesitaremos el precio del software que utilizaremos para la programación del proyecto, así como sus respectivos subprogramas de los cuales necesitaremos sus diferentes herramientas y funciones adicionales.

Artículo	Cantidad	Precio/ud.	Precio
SIMATIC S7, STEP 7 PROFESSIONAL V14 FLOATING LICENSE	1	2 191,13 € (IVA incl.)	2 191,13 €
SIMATIC WinCC Basic V14	1	117,37 € (IVA inc.)	117,37 €
SIMATIC STEP 7 PLCSIM V14 SP1 for STEP 7 Basic and STEP 7 Professional:	1	0€	0€

Subtotal de precios estimados (Software)	Precio total (horas de programación)
2308,5 €	(2308.5/1650h)*110h = 153.9 €

Éste sería el precio del software en caso de que quisiésemos la versión profesional, que es la versión que más herramientas y diferentes configuraciones nos ofrece. Como vemos, SIEMENS nos ofrece algunas de sus herramientas gratuitamente, pero inútiles sin el software principal.

3. DETALLE DE COMPONENTES ELÉCTRICOS PARA LA APLICACIÓN DEL PROYECTO.

Este apartado simplemente define que, por los trámites llevados a cabo para establecer los detalles del proyecto, los materiales y componentes eléctricos como son motores, variadores de frecuencia, encoders, finales de carrera, y todo componente alternativo para la construcción del proyecto, quedan totalmente adquiridos y abonados por la empresa cliente de este proyecto.

4. OFERTA ADICIONAL

Se oferta además una serie de ventajas para el cliente por un precio adicional:

- Mantenimiento técnico
  En caso de que hubiera problemas con el programa, o con la CPU, se podrá llamar al número de contacto de la ingeniería, o mediante el correo electrónico proporcionado por ésta, para resolver los problemas ocasionados. El horario de asistencia es de 08:00 a 17:30, de lunes a viernes, tanto en calendario laboral como en festivos, aprovechando los cierres de las industrias para el paro de las máquinas.

- Actualizaciones

Se ofrece tener el proyecto en constante renovación, teniendo en cuenta las diferentes sugerencias del cliente y las aportaciones de la propia ingeniería, después de llevarlas a consenso entre contratista y proyectista.

- Formación técnica.

Se ofrece un pequeño curso de formación de aproximadamente 2 horas para la implementación de la función, la resolución de problemas, y la configuración e instalación de hardware.

El precio de estos servicios estará basado en la complejidad del proyecto, en comparación con otros proyectos, en el número de componentes, y las horas de programación.

El precio final de estos servicios será de **800 €/año**

5. MANO DE OBRA

En este apartado calcularemos los costes que han sido necesarios debido a la mano de obra. Se expondrán los tiempos usados para las diferentes tareas como diseño, programación, etc…

El precio de la mano de obra se ha calculado suponiendo que el encargado de realizar todo el trabajo es el autor del proyecto que, una vez aprobado éste, será Graduado en Ingeniería Eléctrica. Por esto, hay que tener en cuenta el salario del empleado, la seguridad social y las posibles primas. Por tanto, el precio será de 12.68 €/hora.

En la siguiente tabla tendremos los costes de cada una de las tareas necesarias para la realización del proyecto:

Concepto	Horas	Importe
Estudio previo	15	190.2
Instalación y preparación del software	30	380.4
Programación del autómata	70	887.6
Programación del SCADA	40	507.2
Redacción del proyecto	70	887.6
MANO DE OBRA TOTAL		2853 €

6. GASTOS GENERALES

Estos gastos se corresponden con los gastos de la empresa debidos al consumo de energía eléctrica, impuestos, agua, teléfono. Estos gastos se representan en función del porcentaje sobre la mano de obra directa. En nuestro caso tomaremos un 6%.

CONCEPTO	IMPORTE
Coste mano de obra	2853 €
Gastos 6%	171.18 €

7. COSTE TOTAL Y BENEFICIO

Finalmente, vamos a calcular el coste total, el cual sería la inversión inicial que necesitaríamos para llevar a cabo desde cero el proyecto.

Precio estimado hardware	1200 €
Precio estimado CPU	471,89 €
Precio estimado software	153.9 €
Precio mano de obra	2853 €
Precio gastos generales	171.18 €
Precio total	4849,97 €

A este precio se sumaría el precio del mantenimiento ofertado, más las actualizaciones, más el curso de formación. Éste es de 800 €/año.

A continuación, tendremos en cuenta el beneficio industrial. Este concepto depende de muchos factores tales como el mercado, la competencia o la demanda, por tanto es un valor bastante arbitrario.

Para un beneficio industrial del 30% sobre el total, sacaremos el precio a pagar por el cliente:

Precio	Beneficio industrial	Precio total
4849,97 €	25%	6062,46 €

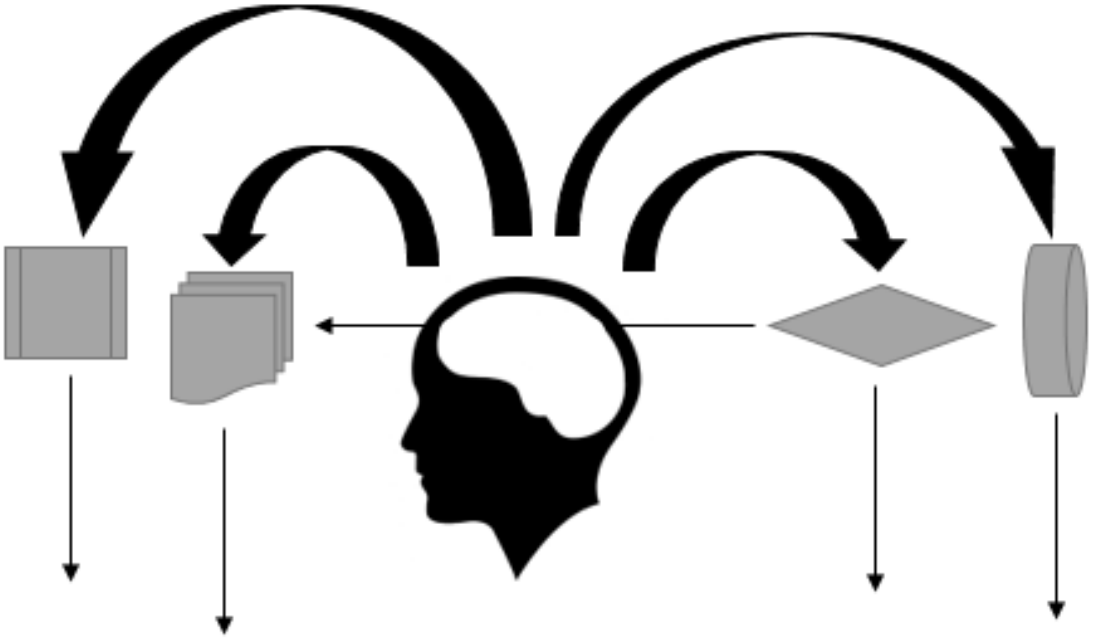
Por lo tanto, el precio final para el primer año y para los siguientes años serán los siguientes:

Precio año 1	6062,46 € + 800 €/año 1	6862,46 €*
Precio año 2 y siguientes	800 €/año	800 €/año

*La parte del mantenimiento (800€) se pagaría a lo largo del año contractual establecido para pagar dicha parte.

El coste total de este proyecto para el año 1 será entonces de 6862,46 €

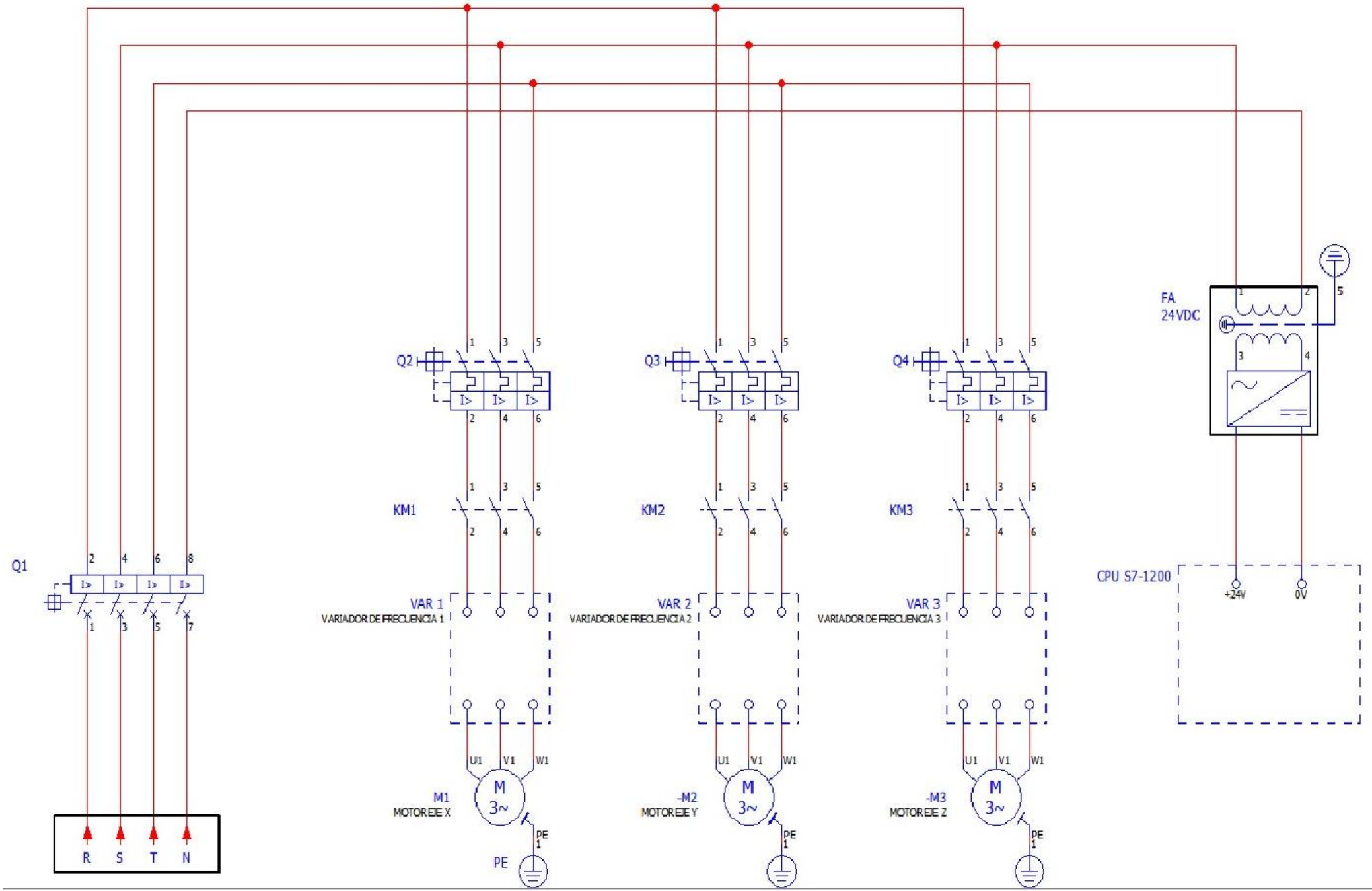
(Seis mil ochocientos sesenta y dos euros con cuarenta y seis céntimos.)

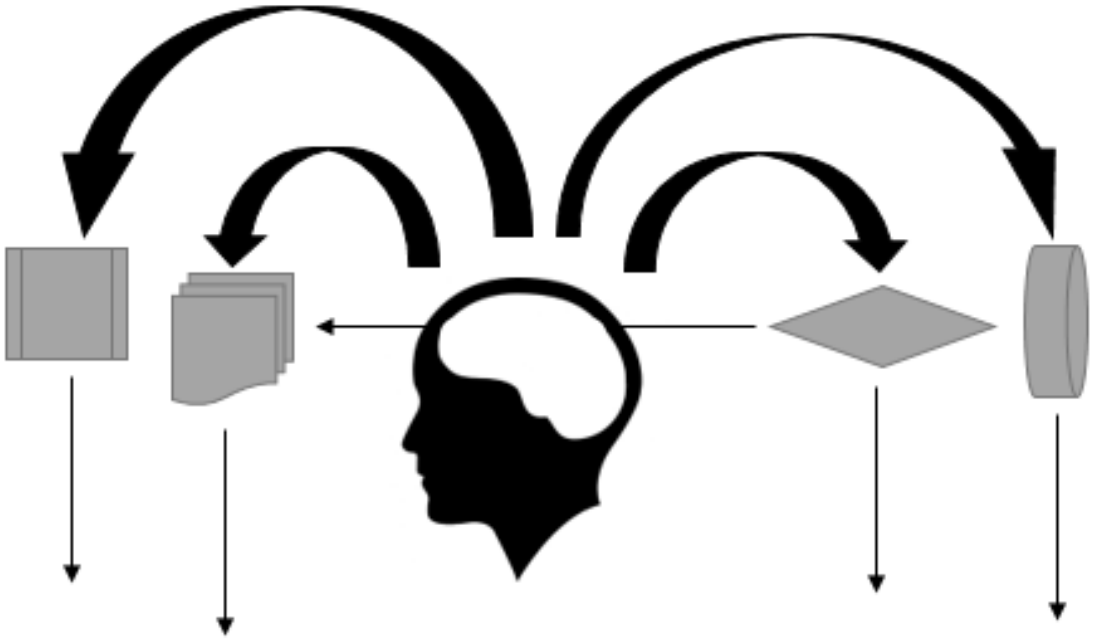


CONTROL DE UN PUENTE GRÚA MEDIANTE AUTÓMATA PROGRAMABLE BASADO EN UNA FUNCIÓN GENÉRICA DE EJE

CÉSAR ROMERO BARBERÁ

ESQUEMA ELÉCTRICO DE POTENCIA


Q1
2 4 6 8
I> I> I> I>
1 3 5 7
R S T N
Q2
Q3
Q4
1 3 5
I> I> I>
2 4 6
KM1
KM2
KM3
VAR 1
VARIADOR DE FRECUENCIA 1
VAR 2
VARIADOR DE FRECUENCIA 2
VAR 3
VARIADOR DE FRECUENCIA 3
U1 V1 W1
M
3~
M1
MOTOR EJE X
-M2
MOTOR EJE Y
-M3
MOTOR EJE Z
PE
1
FA
24 VDC
1 2
3 4
5
CPU S7-1200
+24V
0V



UNIVERSITAT POLITÈCNICA DE VALÈNCIA

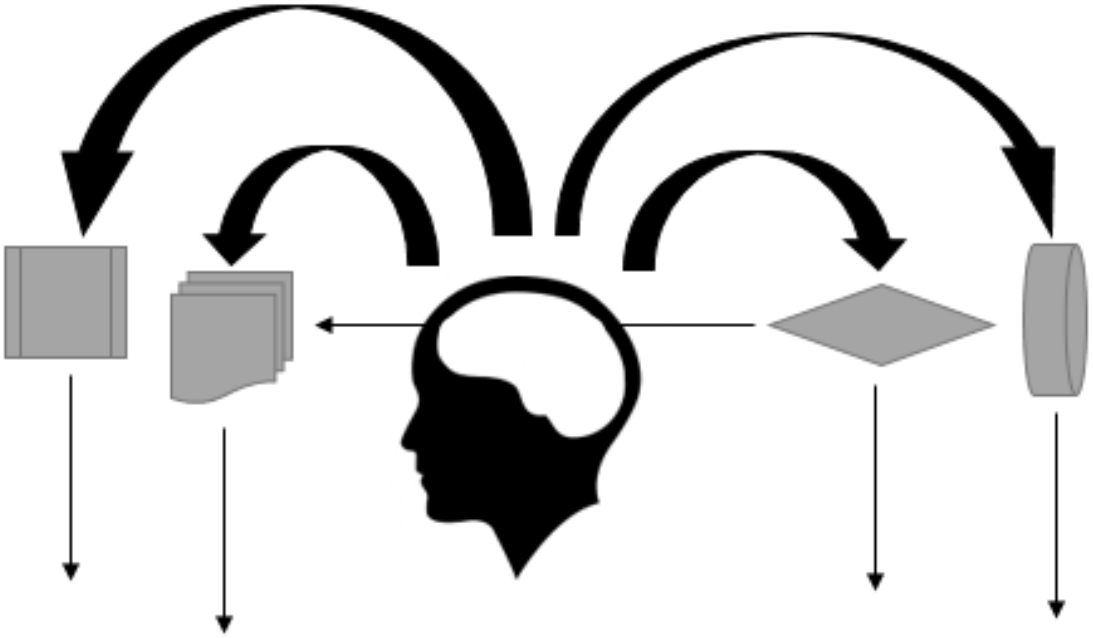
Escuela Técnica Superior de Ingeniería del Diseño

CONTROL DE UN PUENTE GRÚA MEDIANTE AUTÓMATA PROGRAMABLE BASADO EN UNA FUNCIÓN GENÉRICA DE EJE

ANEXOS

CÉSAR ROMERO BARBERÁ

ÍNDICE ANEXOS



CONTROL DE UN PUENTE GRÚA MEDIANTE AUTÓMATA PROGRAMABLE BASADO EN UNA FUNCIÓN GENÉRICA DE EJE

MANUAL DE USUARIO DE TIA PORTAL V14

CÉSAR ROMERO BARBERÁ

Contenido

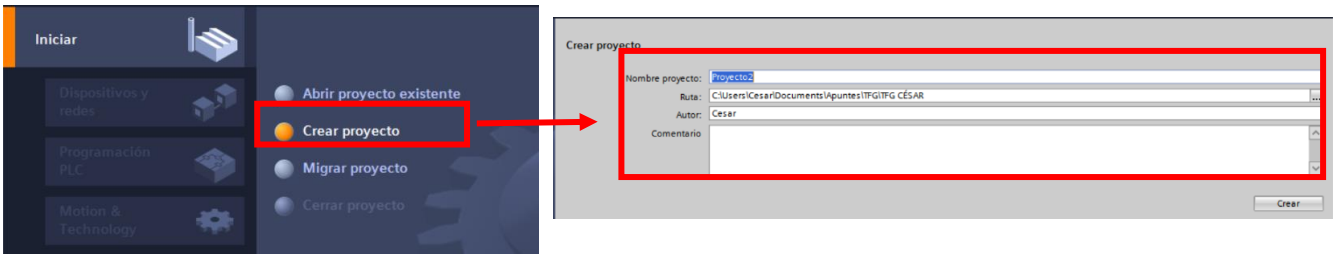
1. CONFIGURACIÓN INICIAL

El software que se ha utilizado para la programación es el TIA Portal V14, de la marca SIEMENS. Este programa tiene un gran abanico de autómatas a elegir de la propia marca, y nos permite programar en diferentes lenguajes que, si bien no son exactamente los mismos que hemos visto anteriormente con la norma, son similares y con distinto nombre.

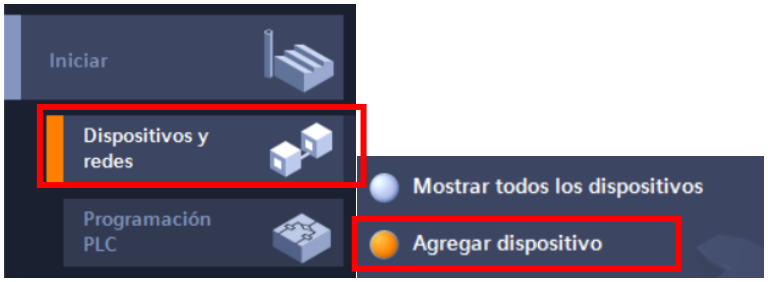
Veremos estos lenguajes junto a la forma de cómo crear un programa en TIA Portal:

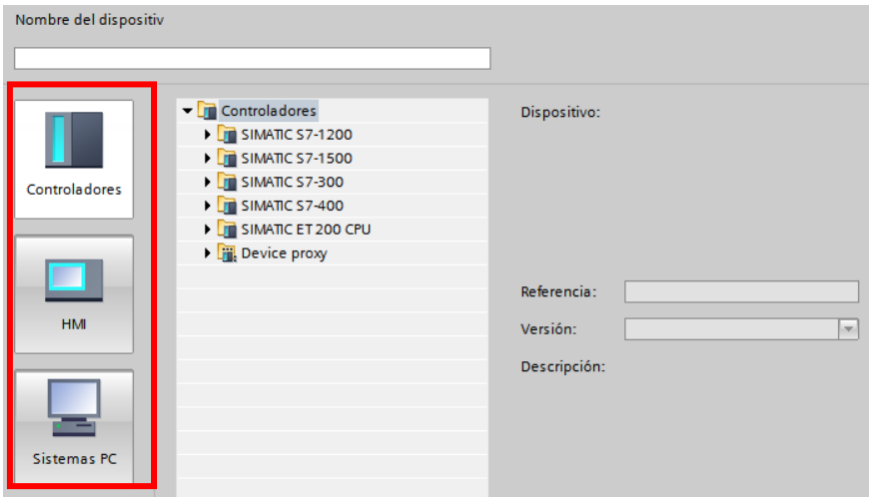
- Creación de un nuevo proyecto:

Al iniciar el programa, nos encontramos ante un menú bastante intuitivo. Nos dirigiremos a la opción "Crear proyecto" en la pestaña "Iniciar", donde podremos darle nombre, ruta, nombre del autor y algún comentario a nuestro nuevo proyecto. Una vez completados estos campos, pulsaremos sobre "Crear"



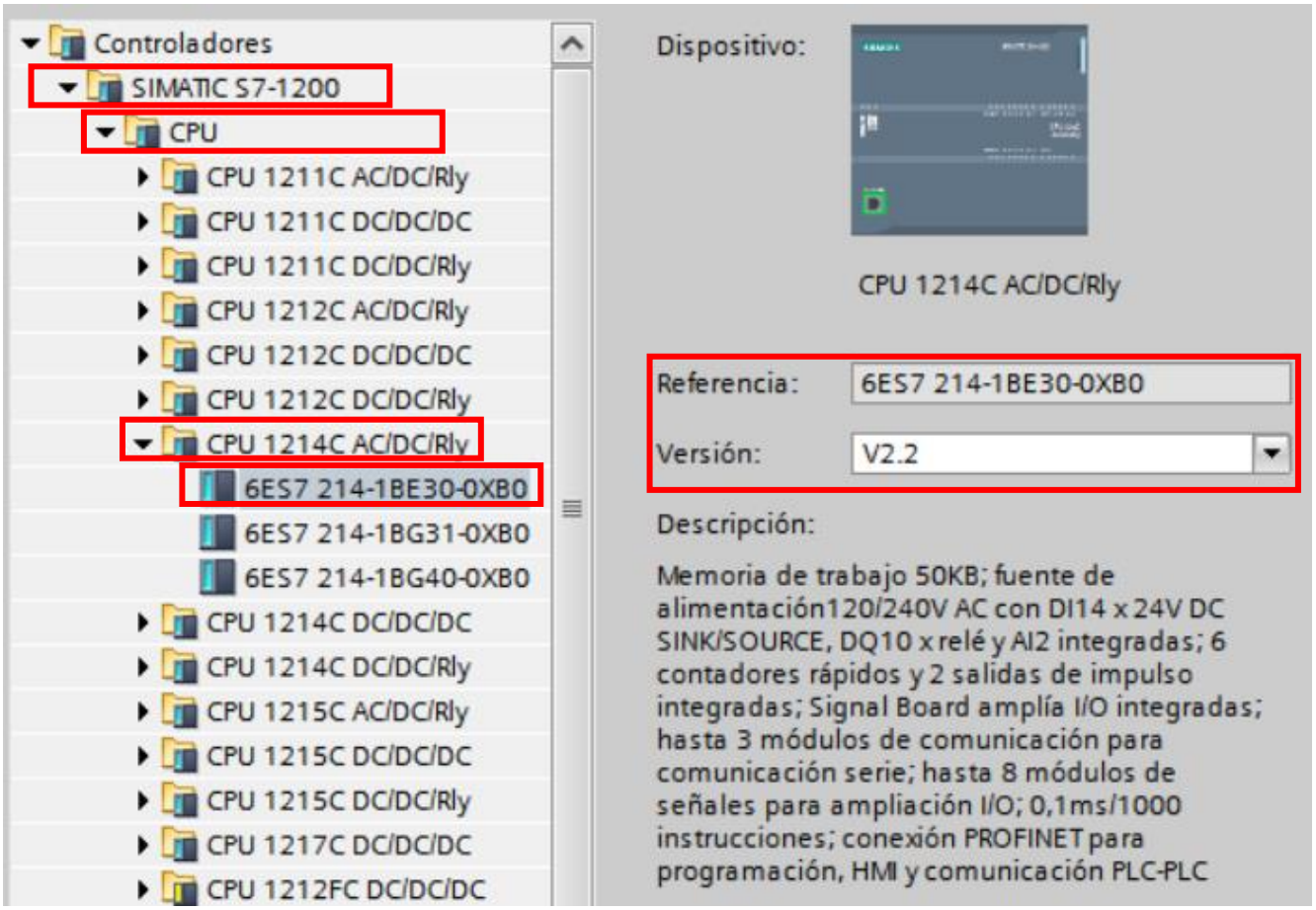
Una vez hecho esto, el programa nos llevará a la siguiente pestaña "Dispositivos y redes", donde podremos añadir los diferentes dispositivos que conforman nuestra comunicación:



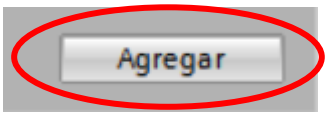

Nombre del dispositiv
Controladores
HMI
Sistemas PC
Controladores
SIMATIC S7-1200
SIMATIC S7-1500
SIMATIC S7-300
SIMATIC S7-400
SIMATIC ET 200 CPU
Device proxy
Dispositivo:
Referencia:
Versión:
Descripción:

En la pestaña "Dispositivos y redes", debemos pulsar en "Agregar dispositivo" donde podremos elegir de una larga lista de controladores, pantallas táctiles (HMI) y sistemas PC, dependiendo de las características que deseemos o las características de los componentes que tengamos.

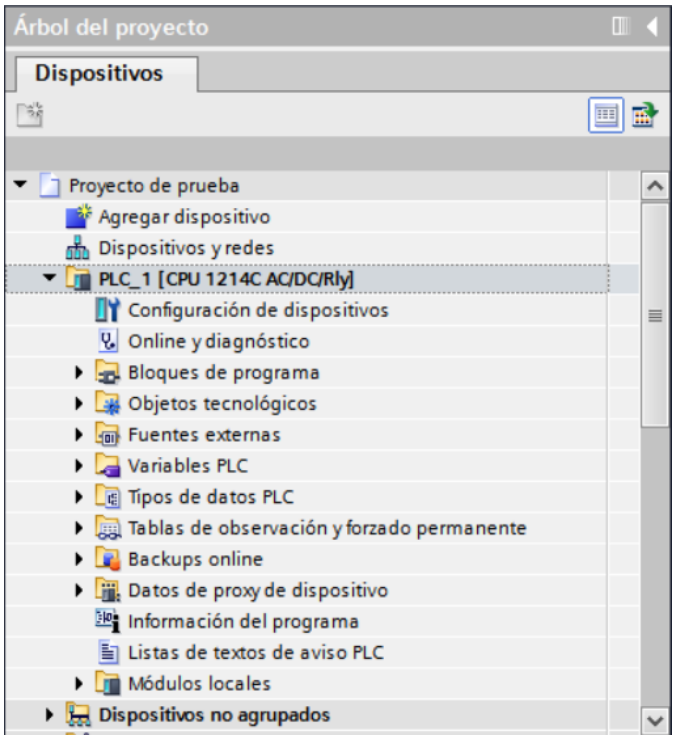
Como vemos, en la carpeta de "Controladores", tenemos los diferentes SIMATIC de Siemens. Elegiremos los de la gama SIMATIC S7-1200, y dentro de esta carpeta, en el único apartado CPU, elegiremos la CPU S7 1214C AC/DC/RLY. A continuación, elegiremos la versión que necesitemos dependiendo de la memoria de trabajo que necesitemos. Para ello, usaremos la versión 2.2, con 50 kB de memoria.



Para agregar esta CPU, pulsaremos sobre el botón homónimo, situado más abajo.



2. COMPOSICIÓN DEL PROGRAMA

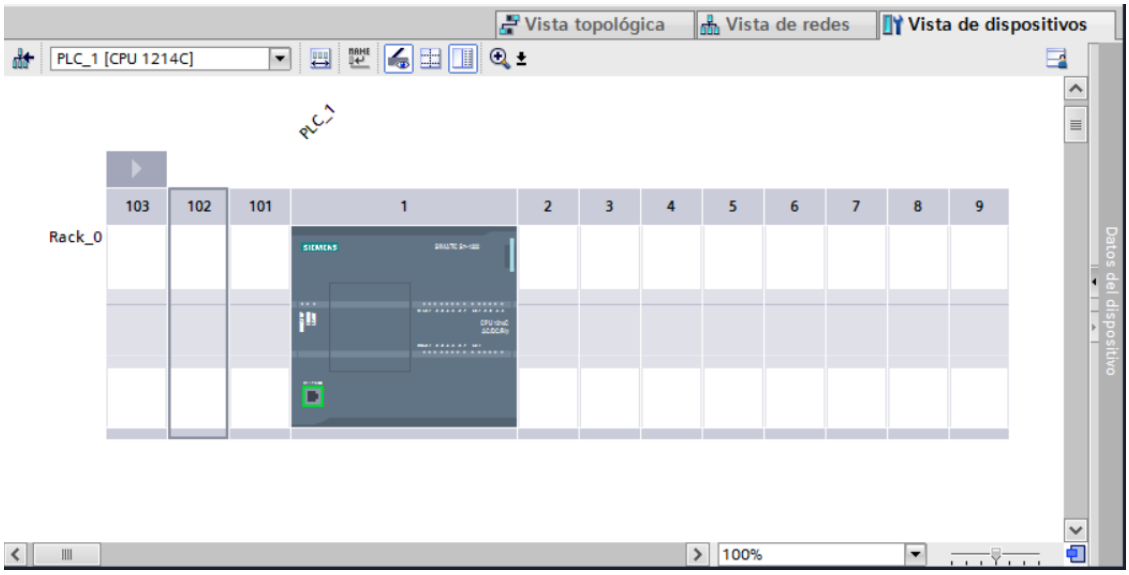


1.1. Árbol del proyecto:

Es el esquema general del programa. Está distribuido por dispositivos, de modo que si elegimos una CPU aparecerán las carpetas que componen el programa. Éstas a su vez contienen los distintos elementos: Bloques de función, variables, tablas de observación, etc…

Más adelante volveremos a la carpeta de Bloques de programa para empezar la programación.

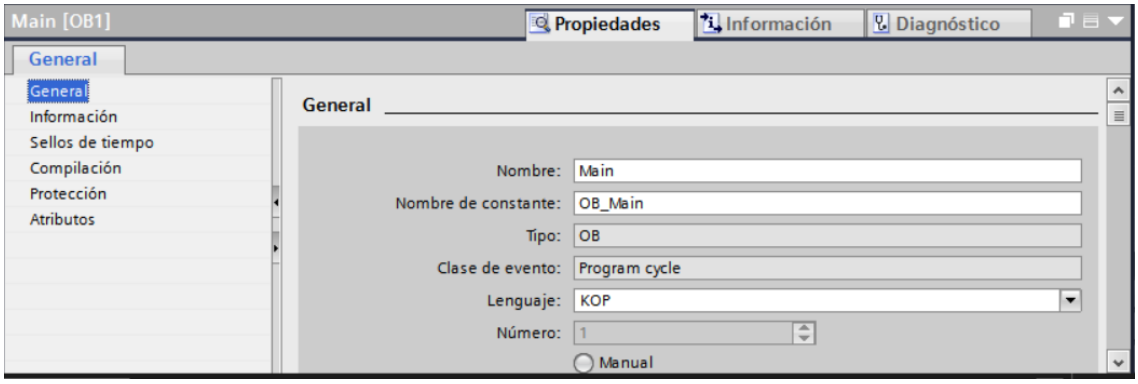
En caso de que eligiéramos una pantalla HMI, saldrán más carpetas, en una de las cuales tendremos opciones para crear paneles para nuestro SCADA.



2.1. Vista de dispositivos:

Esto pretende ser un esquema de los componentes físicos que pongamos en nuestro bastidor o RACK, por lo que podríamos instalar módulos de comunicación, de expansión de entradas, de salidas, y de otro tipo de comunicaciones si la CPU lo permite.

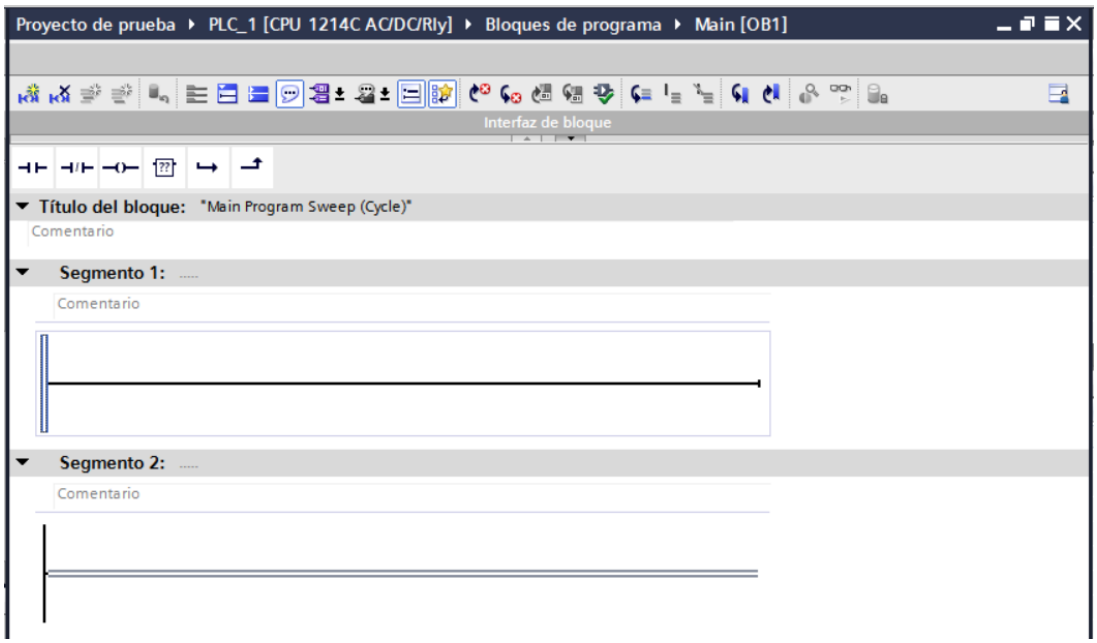
A la derecha de esta pantalla tendríamos un catálogo de Tarjetas de comunicación, placas con baterías, módulos, expansiones para entradas analógicas y digitales, también para salidas, etc… De modo que podemos elegir desde ahí lo que necesitemos.



2.2. Información y Propiedades:

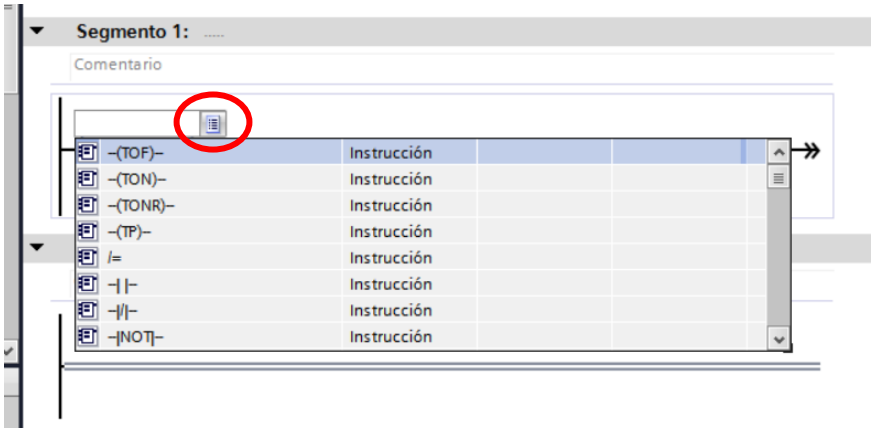
Ésta es la pantalla de información, diagnosis y propiedades. Dependiendo de lo que seleccionemos, aparecerán las propiedades o los avisos, para el programa, para las imágenes de la pantalla táctil, para la simulación, etc…

Como vemos, se pueden seleccionar las opciones más generales el nombre del proyecto, nombre de la función, el lenguaje en el que trabajaremos, etc… En otras opciones tenemos la posibilidad de editar información, opciones de compilación, proteger bloques, y acceder a otras opciones avanzadas en Atributos.



2.3. Programa del proyecto:

Esta es la vista en la que trabajaremos. La estética de ésta dependerá del lenguaje de programación que hayamos elegido en Propiedades. Como vemos, en lenguaje KOP o Ladder, se separa por segmentos, lo que nos ofrece una organización entre líneas de programación muy recomendable de aprovechar, ya que se puede escribir comentarios en cada segmento y definir un poco el uso de cada uno.



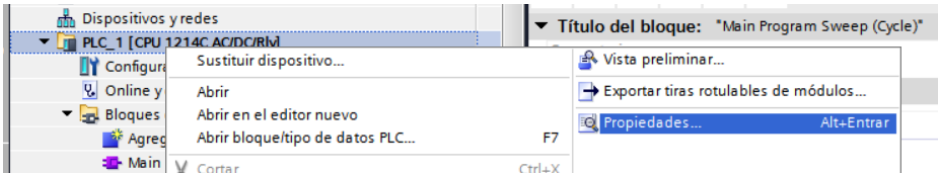
Una vez hayamos creado nuestra función, la deberemos declarar en el bloque MAIN. Si pulsamos sobre el segmento y seleccionamos la opción de CUADRO VACÍO, podremos introducir la función. Además, el TIA Portal te ofrece una biblioteca extensa de funciones que podremos ver clicando en el botón al lado del recuadro de texto.

3. Contadores rápidos

A continuación, entrando un poco más en el apartado de configuración, vamos a activar un HSC (High Speed Counter) o Contador Rápido para el conteo de pulsos del encoder. Cuando tenemos que utilizar un encoder y trabajar con el de manera precisa debemos hacerlo con Rutina de Interrupción.

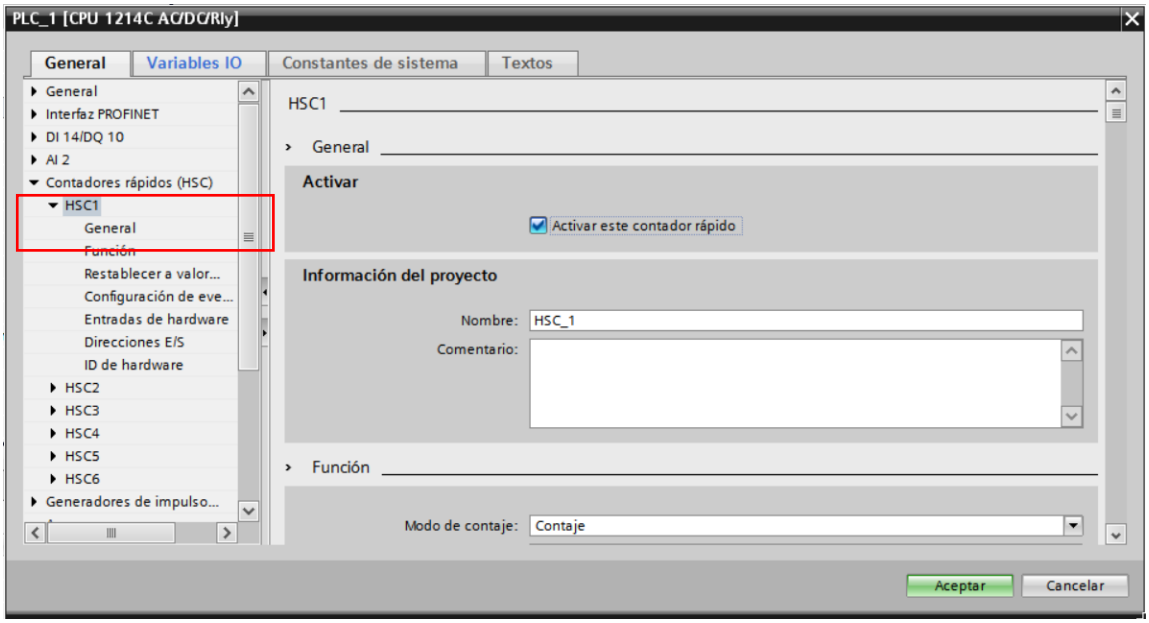
Una rutina de interrupción de un Contador de Alta Velocdad (HSC - High Speed Counter) lo que hace es controlar el contaje de pulsos del encoder con un valor de referencia. Cuando el valor de contaje del encoder es igual al de referencia, el programa principal se detiene y se ejecuta una subrutina y actúa directamente sin esperar a que termine el ciclo de SCAN

3.1. Configuración inicial

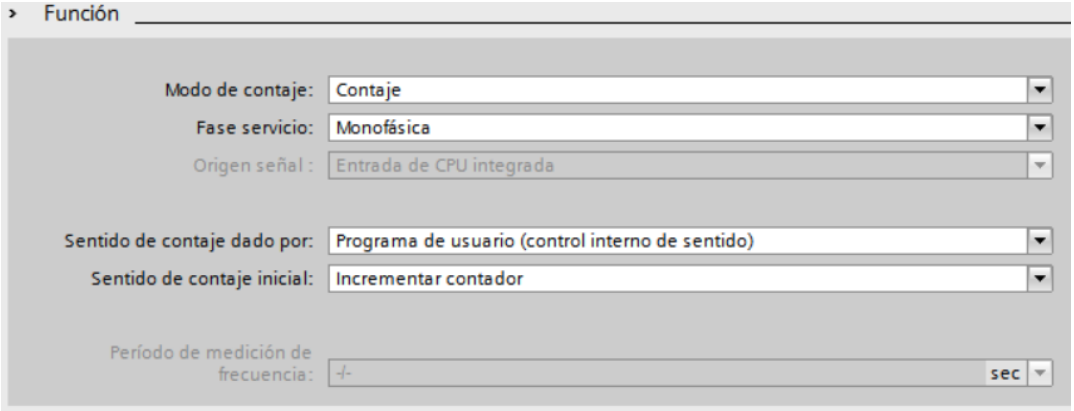


1º clicamos con el botón derecho sobre la CPU y seleccionamos “Propiedades”.

En la siguiente pantalla deberemos elegir la opción de “Contadores rápidos” y seleccionar el HSC1.

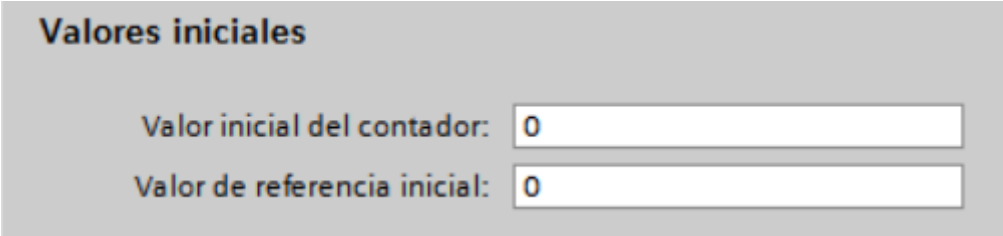


A continuación, activaremos la casilla de “Activar este contador rápido”. Más adelante, en el apartado Función tendremos las siguientes opciones:

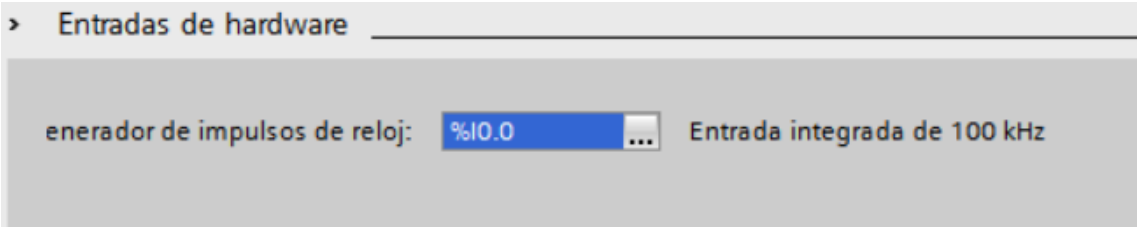


- En el **MODO DE CONTAJE** seleccionaremos la opción de contaje, ya que nuestro objetivo es que se cuenten los pulsos del encoder.
- Para **FASE DE SERVICIO,** seleccionaremos Contador A/B Cuádruple, ya que queremos que gestione las señales del encoder.
- En el apartado de **SENTIDO DE CONTAJE INICIAL** le daremos el valor de Incrementar contador.

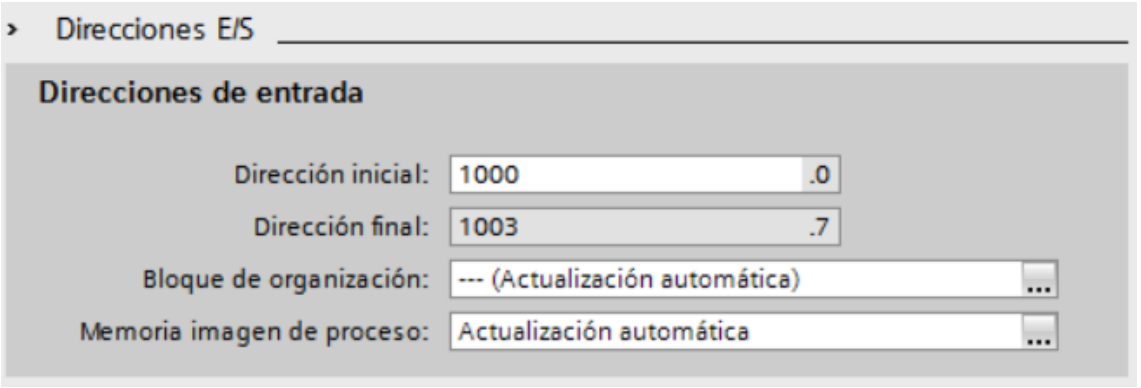
Podemos pre-fijar unos valores iniciales del contador y del valor de referencia (consigna).



En el apartado "Entradas de Hardware" indicaremos la entrada física a la que está conectado el encoder.



Más adelante, podremos asociar la dirección de memoria al contador.



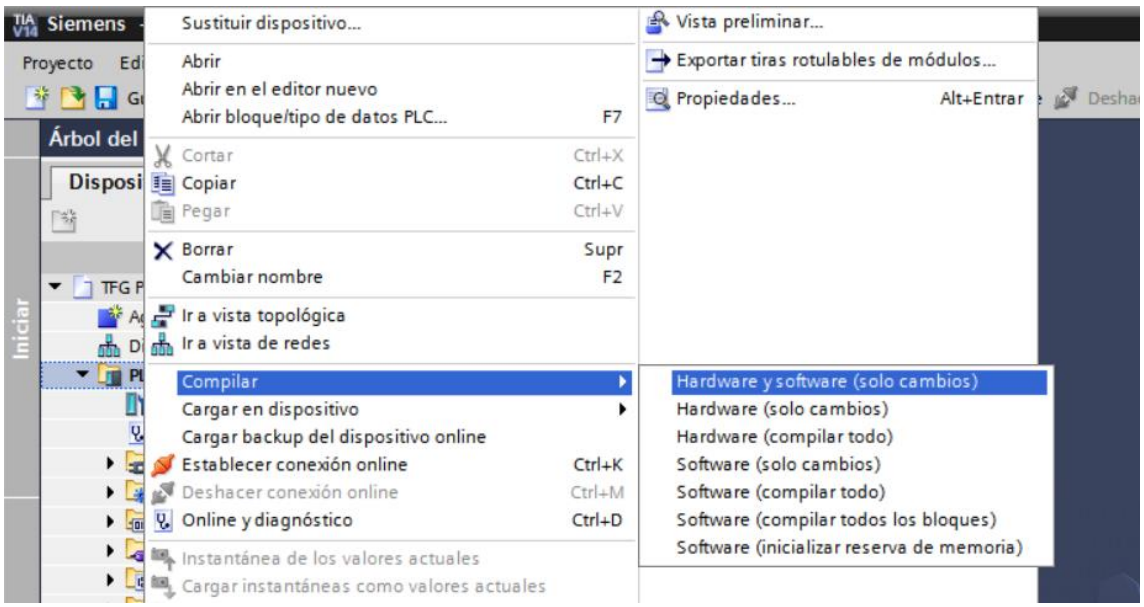
Cada HSC almacena por defecto en una determinada dirección:

HSC	Tipo de datos	Dirección predeterminada
HSC1	DInt	ID1000
HSC2	DInt	ID1004
HSC3	DInt	ID1008
HSC4	DInt	ID1012
HSC5	DInt	ID1016
HSC6	DInt	ID1020

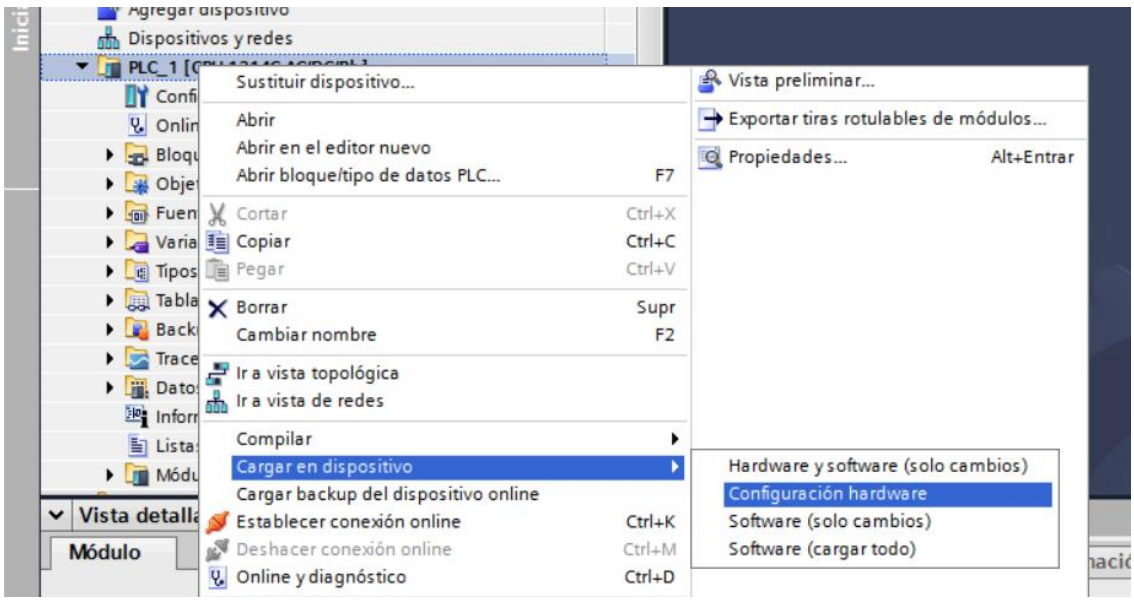
3.2. Cargar la configuración en la CPU

Para que la CPU interprete la instrucción HSC deberemos configurar su hardware de manera interna además de compilarla.

Para ello, haciendo click derecho sobre nuestra CPU, clicaremos en Compilar, y compilaremos tanto Hardware como Software.

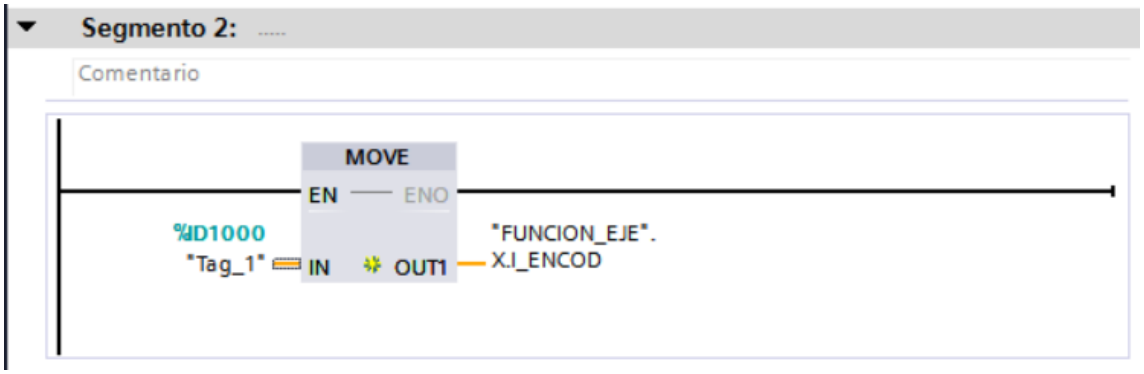


Además, para que surja efecto y se cargue satisfactoriamente en nuestra CPU, deberemos cargar esta configuración en ésta, mediante la función "Configuración hardware":



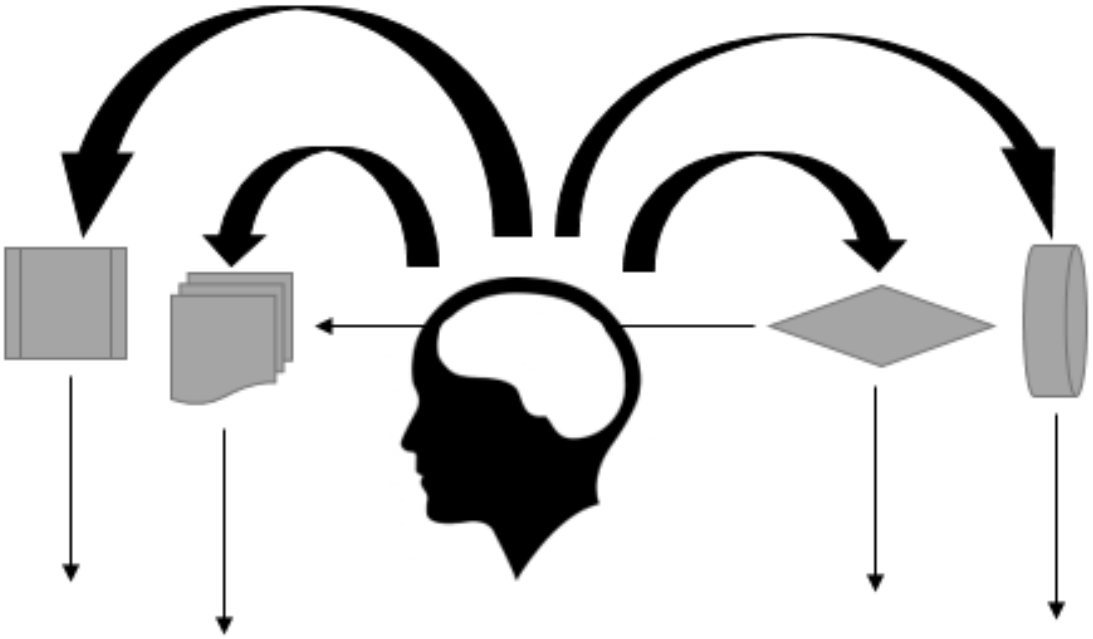
3.3. Final de configuración y puesta en marcha

Podemos ver una configuración final de cómo vamos a contar los pulsos del encoder:



De modo que moveremos lo que el contador lea a la memoria ID1000, y mediante la instrucción MOVE, lo moveremos al apartado del DB de Función Eje, I_ENCOD, el cual se encarga de leer los pulsos del encoder.

Así, la configuración del HSC está terminada, ya que la función genérica leerá lo que haya en esa memoria "FUNCIÓN_EJE".X.I_ENCOD.



CONTROL DE UN PUENTE GRÚA MEDIANTE AUTÓMATA PROGRAMABLE BASADO EN UNA FUNCIÓN GENÉRICA DE EJE

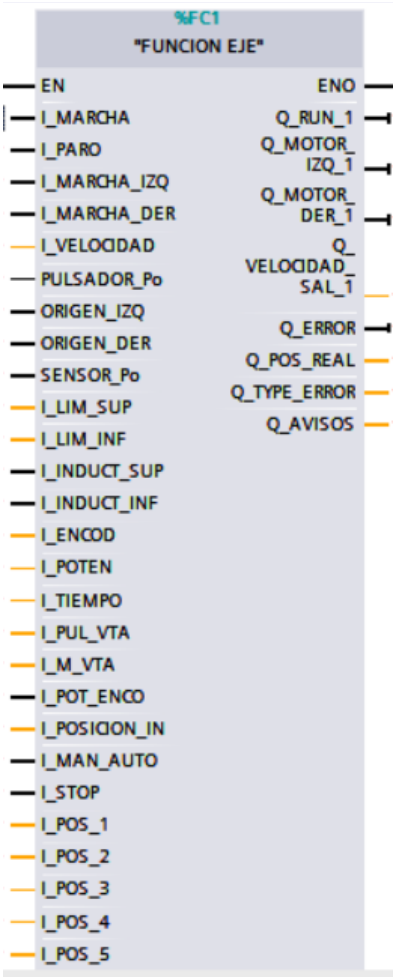
MANUAL DE USUARIO DE LA FUNCIÓN GENÉRICA

CÉSAR ROMERO BARBERÁ

Contenido

1. INTRODUCCIÓN

En este documento se expone el manual de usuario con las instrucciones necesarias para el control de esta función. Se explicará cada variable, tanto de salida como de entrada, y la implementación de alguna función extra para el correcto funcionamiento, como será la configuración del contador rápido.



Así es la función vista desde la función MAIN, donde se observa todas las variables de entrada y de salida. Cada una tiene un conector de color distinto, en función del tipo de variable que sea.

2. VARIABLES DE ENTRADA.

I_MARCHA	De tipo BOOL, es la encargada de encender la variable y poner el proceso en movimiento.
I_PARO	De tipo BOOL, se encarga de parar el proceso en cuanto es activada.
I_MARCHA_IZQ	Al activarse, manda la señal de mover el motor a la izquierda a la salida de la función.
I_MARCHA_DER	De igual manera, manda la señal de mover el motor a la derecha.
I_VELOCIDAD	De tipo INT, es el valor de referencia para la velocidad a la que queremos desplazarnos por el eje.
PULSADOR_Po	De tipo BOOL. Al activarse, devuelve el carro móvil a la posición inicial.
ORIGEN_IZQ	De tipo BOOL. Si activamos esta variable, determinaremos el origen en el lado izquierdo.
ORIGEN_DER	De tipo BOOL. Si activamos esta variable, determinaremos el origen en el lado derecho.
SENSOR_Po	De tipo BOOL. Será la entrada física al sensor de final de carrera de posición inicial. Una vez el carro llegue a este sensor, parará en dicha posición.
I_LIM_SUP	De tipo INT. En esta variable se designará el límite superior del eje. Se deberá poner un límite inferior al límite superior físico del propio eje.
I_LIM_INF	De tipo INT. En esta variable se designará el límite inferior del eje. Se deberá poner un límite superior al límite inferior físico del propio eje.
I_INDUCT_SUP	De tipo BOOL. Será la entrada física al sensor inductivo para el límite superior.
I_INDUCT_INF	De tipo BOOL. Será la entrada física al sensor inductivo para el límite inferior.
I_ENCOD	De tipo INT. Será la entrada física al encoder del motor.
I_POTEN	De tipo INT. Será la entrada física a un potenciómetro o resistencia variable para el control manual del eje.

I_PUL_VTA	De tipo INT. Esta variable servirá para definir la resolución del encoder del motor.
I_M_VTA	De tipo INT. Esta variable servirá para definir la cantidad de milímetros por vuelta que dará el eje. Esta configuración dependerá del propio eje.
I_POT_ENCO	De tipo BOOL. Esta variable servirá para elegir internamente con qué queremos controlar el motor. Tenemos como opción el potenciómetro o el encoder. Para elegir el potenciómetro, deberemos tener la variable a FALSE. Si elegimos el encoder, deberemos tener la variable en TRUE.
I_POSICION_IN:	De tipo INT. Es la variable de entrada de posición para determinar una posición.
I_MAN_AUTO	De tipo BOOL. Es la variable de elección entre los modos MANUAL y AUTOMÁTICO. Para elegir MANUAL, la variable será TRUE. Para elegir AUTOM., la variable será FALSE.
I_STOP	De tipo BOOL. Es una variable para el control manual en el que podemos asegurarnos de no mover el motor en caso de que esté activada.
I_POS	De tipo Int. En estas variables configuraremos las posiciones de un ciclo automático en caso de que se necesite uno.

3. VARIABLES DE SALIDA.

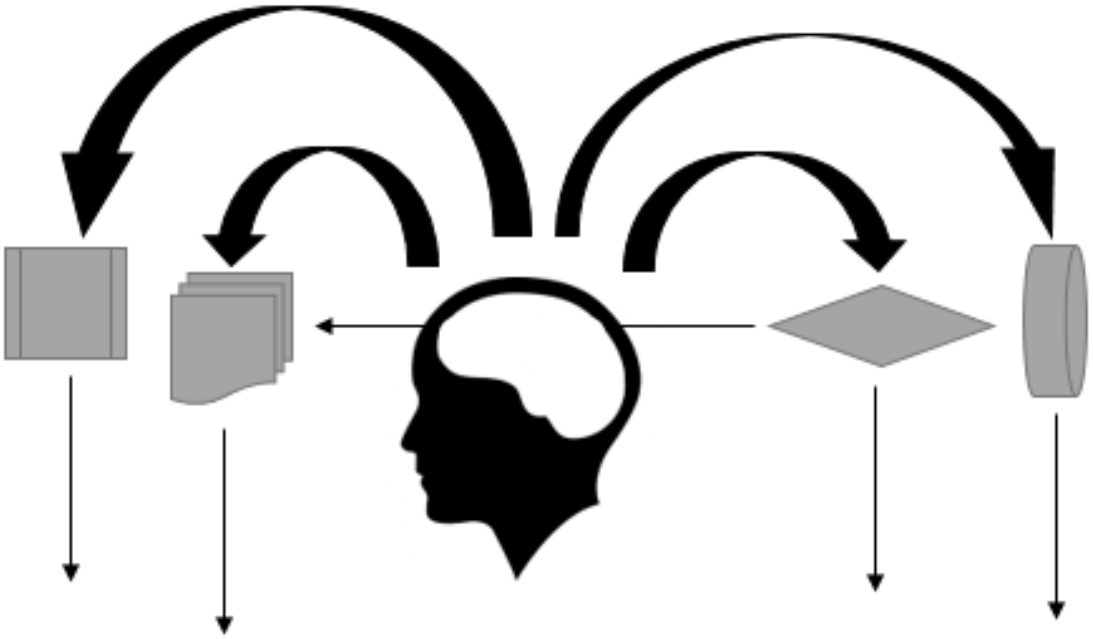
Q_RUN_1	De tipo BOOL. Nos indicará el estado en el que se encuentra la función. Si es FALSE, la función no está en marcha. Si es TRUE, la función está encendida.
Q_MOTOR_IZQ_1	De tipo BOOL. Nos indica que la función está mandando la señal de mover el motor a la izquierda.
Q_MOTOR_DER_1	De tipo BOOL. Nos indica que la función está mandando la señal de mover el motor a la derecha.
Q_VELOCIDAD_SAL_1	De tipo INT. Es la velocidad en estado real del motor. En base a unos cálculos internos y la señal del encoder.
Q_ERROR	De tipo INT. Se enciende o apaga dependiendo de que haya algún error.

Q_POS_REAL	De tipo INT. Esta variabe refleja la posición en tiempo real del motor a lo largo del eje.
Q_TYPE_ERROR	De tipo INT. Esta variable refleja, dependiendo del error, un código. Más adelante se exponen los diferentes códigos.
Q_AVISOS	De tipo String. Esta variable, dependiendo del código de error, facilita un mensaje para ayudar a solucionar el error.

4. EXPLICACIÓN DE FUNCIÓN.

Esta es la función genérica para el control de un eje. Esta función tiene como objetivo el control de posición y velocidad de cualquier eje al que se le asigne mediante la programación en un autómata programable.

La función deberá parametrizarse primero con algunos datos como son los límites del eje, el origen para la puesta a cero del eje, los pulsos por vuelta del eje, los milímetros por vuelta del eje, etc… todos ellos recogidos en las variables anteriormente explicadas.



CONTROL DE UN PUENTE GRÚA MEDIANTE AUTÓMATA PROGRAMABLE BASADO EN UNA FUNCIÓN GENÉRICA DE EJE

CÉSAR ROMERO BARBERÁ

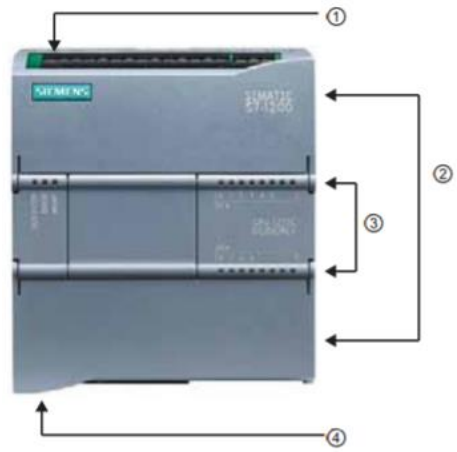
HOJA TÉCNICA S7-1214 AC/DC/RLY

Contenido

1. INTRODUCCIÓN

El controlador lógico programable (PLC) S7-1200 ofrece la flexibilidad y capacidad de controlar una gran variedad de dispositivos para las distintas tareas de automatización. Gracias a su diseño compacto, configuración flexible y amplio juego de instrucciones, el S7-1200 es idóneo para controlar una gran variedad de aplicaciones.
La CPU incorpora un microprocesador, una fuente de alimentación integrada, así como circuitos de entrada y salida en una carcasa compacta, conformando así un potente PLC. Una vez cargado el programa en la CPU, ésta contiene la lógica necesaria para vigilar y controlar los dispositivos de la aplicación. La CPU vigila las entradas y cambia el estado de las salidas según la lógica del programa de usuario, que puede incluir lógica booleana, instrucciones de contaje y temporización, funciones matemáticas complejas, así como comunicación con otros dispositivos inteligentes.
Numerosas funciones de seguridad protegen el acceso tanto a la CPU como al programa de control:

- Toda CPU ofrece protección por contraseña que permite configurar el acceso a sus funciones.
- Es posible utilizar la "protección de know-how" para ocultar el código de un bloque específico.



① Conector de corriente

② Conectores extraíbles para el cableado de usuario (detrás de las tapas)

② Ranura para Memory Card (debajo de la tapa superior)

③ LEDs de estado para las E/S integradas

④ Conector PROFINET (en el lado inferior de la CPU)

Los diferentes modelos de CPUs ofrecen una gran variedad de funciones y prestaciones que permiten crear soluciones efectivas destinadas a numerosas aplicaciones.

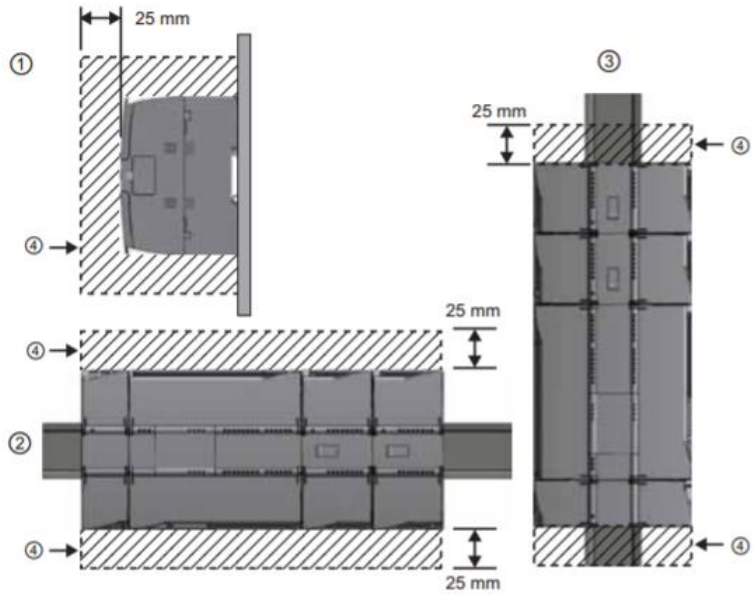
2. CARACTERÍSTICAS

A continuación, las diferentes características del PLC 1214C:

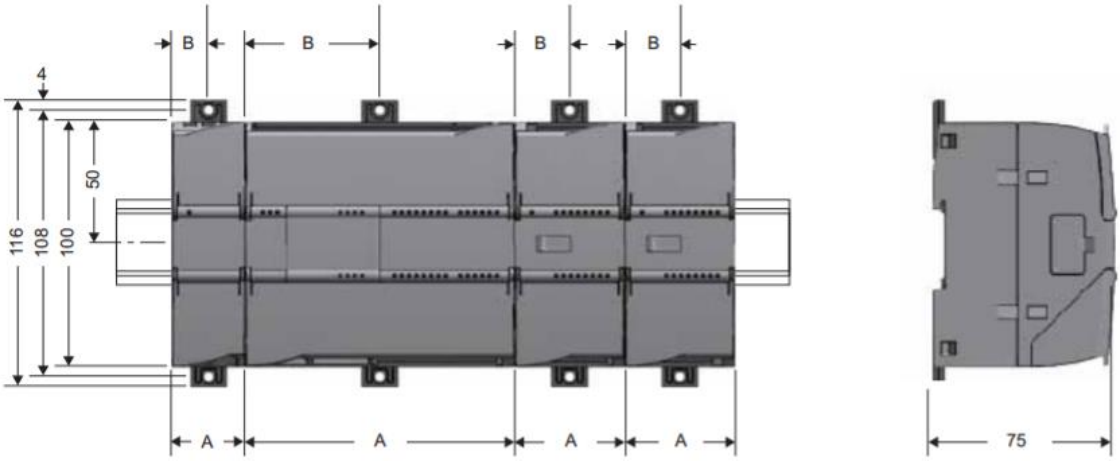
FUNCIÓN	CPU 1214C
Dimensiones físicas (mm)	110 x 100 x 75
Memoria de usuario **• Memoria de trabajo** **• Memoria de carga** **• Memoria remanente**	 50 kB 2 MB 2 kB
E/S integradas locales **• Digitales** **• Analógicas**	 14 entradas/10 salidas 2 entradas
Tamaño de la memoria imagen de proceso	1024 bytes para entradas (I) y 1024 bytes para salidas (Q)
Área de marcas (M)	8192 bytes
Ampliación con módulos de señales	8
Signal Board	1
Módulos de comunicación	3 (ampliación en el lado izquierdo)
Contadores rápidos **• Fase simple** **• Fase en cuadratura**	6 3 a 100 kHz 3 a 30 kHz 3 a 80 kHz 3 a 20 kHz
Salidas de impulsos	2
Memory Card	SIMATIC Memory Card (opcional)
Tiempo de respaldo del reloj de tiempo real	Típico: 10 días / Mínimo: 6 días a 40 °C
PROFINET	1 puerto de comunicación Ethernet
Velocidad de ejecución de funciones matemáticas con números reales	18 µs/instrucción
Velocidad de ejecución booleana	0,1 µs/instrucción

3. MONTAJE

Los equipos S7-1200 son fáciles de montar. El S7-1200 puede montarse en un panel o en un raíl DIN, bien sea horizontal o verticalmente. El tamaño pequeño del S7-1200 permite ahorrar espacio. La refrigeración de los dispositivos S71200 se realiza por convección natural. Para la refrigeración correcta es preciso dejar un espacio mínimo de 25 mm por encima y por debajo de los dispositivos. Asimismo, se deben prever como mínimo 25 mm de profundidad entre el frente de los módulos y el interior de la carcasa



Dimensiones de montaje (mm)



DISPOSITIVO S7-1214C		ANCHO A	ANCHO B
CPU-1214C		110 mm	55 mm
Módulos de señales:	8&16 E/S, DC, relé.	45 mm	22,5mm
	16I/16Q relé	70 mm	35mm
Mod. Comunicación:	CM 1241 RS232	30 mm	15mm

La CPU se puede montar en un panel o en un perfil DIN. Para montar la CPU en un panel, proceda del siguiente modo:

- Posicione y taladre los orificios de montaje (M4 o estándar americano n.º 8) según las dimensiones de montaje indicadas en la tabla.
- Extienda los clips de fijación del módulo. Asegúrese que los clips de fijación al perfil DIN en los lados superior e inferior de la CPU están en posición extendida.
- Atornille el módulo al panel utilizando tornillos dispuestos en los clips.

Para montar la CPU en un perfil DIN, proceda del siguiente modo:

- Monte el perfil DIN. Atornille el perfil al panel de montaje dejando un espacio de 75 mm entre tornillo y tornillo.
- Enganche la CPU por el lado superior del perfil.
- Extraiga el clip de fijación en el lado inferior de la CPU de manera que asome por encima del perfil.
- Gire la CPU hacia abajo para posicionarla correctamente en el perfil.
- Oprima los clips hasta que la CPU encaje en el perfil.

4. DIRECTRICES DE CABLEADO

La puesta a tierra y el cableado correctos de todos los equipos eléctricos es importante para garantizar el funcionamiento óptimo del sistema y aumentar la protección contra interferencias de la aplicación y del S7-1200.

a. AISLAMIENTO GALVÁNICO

Los límites de la alimentación AC del S7-1200 y de las E/S a los circuitos AC se han diseñado y aprobado para proveer un aislamiento galvánico seguro entre las tensiones de línea AC y los circuitos de baja tensión. Estos límites incluyen un aislamiento doble o reforzado, o bien un aislamiento básico más uno adicional, según las distintas normas. Los componentes que cruzan estos límites, tales como optoacopladores, condensadores, transformadores y relés se han aprobado, ya que proveen un aislamiento galvánico seguro. Los límites de aislamiento que cumplen estos requisitos se identifican en las hojas de datos de los productos S7-1200, indicando que tienen un aislamiento de 1500 V AC o superior. Esta indicación se basa en una prueba de fábrica rutinaria de (2Ue + 1000 V AC) o equivalente, según los métodos aprobados. Los límites de aislamiento galvánico seguro del S7-1200 se han comprobado hasta 4242 V DC.

b. PUESTA A TIERRA

La mejor forma de poner a tierra la aplicación es garantizar que todos los conductores neutros y de masa del S7-1200 y de los equipos conectados se pongan a tierra en un mismo punto. Este punto debería conectarse directamente a la toma de tierra del sistema. Todos los cables de puesta a tierra deberían tener la menor longitud posible y una sección grande, p. ej. 2 mm2 (14

AWG). Al definir físicamente las tierras es necesario considerar los requisitos de puesta a tierra de protección y el funcionamiento correcto de los dispositivos protectores.

c. CABLEADO DEL S7-1200

Al diseñar el cableado del S7-1200, prevea un interruptor unipolar para cortar simultáneamente la alimentación de la CPU S7-1200, de todos los circuitos de entrada y de todos los circuitos de salida. Prevea dispositivos de protección contra sobreintensidad (p. ej. fusibles o cortacircuitos) para limitar las corrientes de fallo en el cableado de alimentación. Para mayor protección es posible disponer un fusible u otro limitador de sobreintensidad en todos los circuitos de salida.

Utilice dispositivos de supresión de sobretensiones apropiados en el cableado sujeto a perturbaciones por descargas atmosféricas.

Evite colocar las líneas de señales de baja tensión y los cables de comunicación en una misma canalización junto con los cables AC y los cables DC de alta energía y conmutación rápida. El cableado deberá efectuarse por pares; con el cable de neutro o común combinado con el hilo caliente o de señal.

Utilice el cable más corto posible y vigile que tenga una sección suficiente para conducir la corriente necesaria. El conector acepta cables con una sección de 2 mm2 a 0,3 mm2 (14 AWG a 22 AWG). Utilice cables apantallados para obtener una protección óptima contra interferencias. Por lo general, los mejores resultados se obtienen poniendo a tierra la pantalla del S7-1200.

Al cablear circuitos de entrada alimentados por una fuente externa, prevea dispositivos protectores contra sobrecorriente en estos circuitos. La protección externa no se requiere en los circuitos alimentados por la alimentación de sensores de 24 V DC del S7-1200, puesto que la alimentación de sensores ya está protegida contra sobrecorriente.

Todos los módulos S7-1200 incorporan conectores extraíbles para el cableado de usuario. Para evitar conexiones flojas, asegúrese que el conector está encajado correctamente y que el cable está insertado de forma segura en el conector. No apriete excesivamente los tornillos para impedir que se deteriore el conector. El par máximo de apriete de los tornillos del conector es de 0,56 Nm.

Para impedir flujos de corriente indeseados en la instalación, el S7-1200 provee límites de aislamiento galvánico en ciertos puntos. Tenga en cuenta estos límites de aislamiento al planificar el cableado del sistema. En los datos técnicos encontrará más información acerca de la ubicación de los puntos de aislamiento galvánico y la capacidad que ofrecen. Los aislamientos con valores nominales inferiores a 1500 V AC no deben tomarse para definir barreras de seguridad.

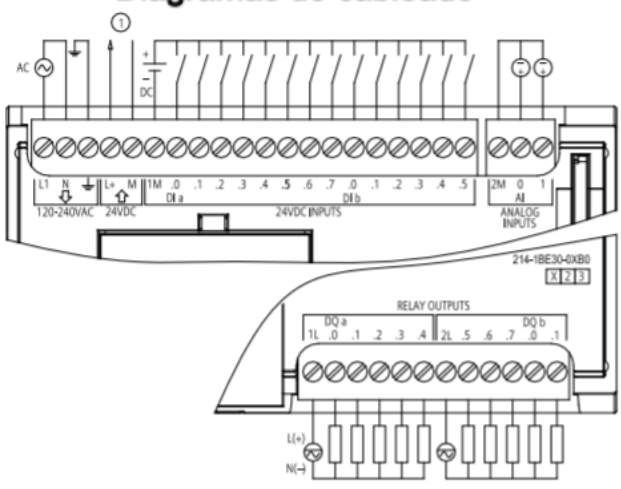
5. DATOS TÉCNICOS

TENSIÓN NOMINAL: 24V DC
120/230V AC
TOLERANCIA: 20,4V DC a 28,8V DC
85V AC a 264V a AC, 47 a 63 Hz

a. DATOS TÉCNICOS CPU 1214C AC/DC/RLY

REFERENCIA	6ES7 214-1BE30-0XB0
DIMENSIONES A x A x P (mm)	110 x 100 x 75
PESO	475 gramos
DISIPACIÓN DE POTENCIA	14 W
INTENSIDAD DISPONIBLE (SM y bus CM)	1600 mA máx. (5 V DC)
INTENSIDAD DISPONIBLE (24 V DC)	400 mA máx. (alimentación de sensores)
CONSUMO DE CORRIENTE DE LAS ENTRADAS DIGITALES (24 V DC)	4 mA/entrada utilizada
MEMORIA DE USUARIO	50 KB de memoria de trabajo / 2 MB de memoria de carga / 2 KB de memoria remanente
Conexiones	• 3 para HMI • 1 para la programadora • 8 para instrucciones Ethernet en el programa de usuario • 3 para CPU a CPU
Transferencia de datos	10/100 Mb/s
Aislamiento (señal externa a lógica del PLC)	Aislado por transformador, 1500 V DC

Diagramas de cableado



① Alimentación de sensores 24 V DC

Figura A-7 CPU 1214C AC/DC/relé (6ES7 214-1BE30-0XB0)

b. DATOS TÉCNICOS DE LAS ENTRADAS DIGITALES 1214C AC/DC/RLY

Número de entradas	14
Tipo	Sumidero/fuente (tipo 1 IEC sumidero)
Tensión nominal	24 V DC a 4 mA, nominal
Tensión continua admisible	30 V DC, máx.
Sobretensión transitoria	35 V DC durante 0,5 seg.
Señal 1 lógica (mín.)	15 V DC a 2,5 mA
Señal 0 lógica (máx.)	5 V DC a 1 mA
Aislamiento (campo a lógica)	500 V AC durante 1 minuto
Grupos de aislamiento	1
Tiempos de filtro	0,2, 0,4, 0,8, 1,6, 3,2, 6,4 y 12,8 ms (seleccionable en grupos de 4)
Frecuencias de entrada de reloj HSC (máx.) (señal 1 lógica = 15 a 26 V DC)	Fase simple: 100 KHz (Ia.0 a Ia.5) y 30 KHz (Ia.6 a Ib.5) Fase en cuadratura: 80 KHz (Ia.0 a Ia.5) y 20 KHz (Ia.6 a Ib.5)
Número de entradas ON simultáneamente	14
Longitud de cable (metros)	500 apantallado, 300 no apantallado, 50 apantallado para entradas HSC

c. DATOS TÉCNICOS DE LAS ENTRADAS ANALÓGICAS 1214C AC/DC/RLY

Número de entradas	2
Tipo	Tensión (asimétrica)
Rango	0 a 10 V
Rango total (palabra de datos)	0 a 27648
Rango de sobreimpulso (palabra de datos)	27.649 a 32.511
Desbordamiento (palabra de datos)	32.512 a 32767
Resolución	10 bits
Tensión de resistencia al choque máxima	35 V DC
Alisamiento	Ninguno, débil, medio o fuerte
Rechazo de interferencias	10, 50 ó 60 Hz (consulte las frecuencias de muestreo en Tiempos de respuesta de las entradas analógicas (Página 346))
Impedancia	≥100 KΩ
Aislamiento (campo a lógica)	Ninguno
Precisión (25°C / 0 a 55°C)	3,0% / 3,5% de rango máximo
Rechazo en modo común	40 dB, DC a 60 Hz
Rango de señales operativo	La tensión de señal más la tensión en modo común debe ser menor que +12 V y mayor que -12 V
Longitud de cable (metros)	10 trenzado y apantallado

d. DATOS TÉCNICOS DE LAS SALIDAS DIGITALES 1214C AC/DC/RLY

Número de salidas	10
Tipo	Relé
Rango de tensión	5 a 30 V DC ó 5 a 250 V AC
Señal 1 lógica a I.máx	---
Señal 0 lógica con carga de 10 kΩ	---
Intensidad máx.	2.0 A
Carga de lamparas	30W DC / 200W AC
Resistencia en estado ON	Máx. 0.2 Ω
Corriente de fuga por salida	---
Sobrecorriente momentánea	7 A si están cerrados los contactos
Protección contra sobrecargas	No
Aislamiento (campo a lógica)	1500 VAC durante 1 minuto (boina a contacto) ó Ninguno (bobina a lógica)
Resistencia de aislamiento	100 MΩ
Aislamiento entre contactos abiertos	750 V AC durante 1 minuto
Grupos de aislamiento	2
Tensión de bloqueo inductiva	---
Retardo de conmutación	10ms máx
Grupos de aislamiento	2
Frecuencia de tren de impulsos	No recomendado
Vida útil mecánica (sin carga)	10.000.000 ciclos abiertos/cerrados
Vida útil de los contactos bajo carga nominal	100.000 ciclos abiertos/cerrados
Reacción al cambiar de RUN a STOP	Último valor o valor sustitutivo (Valor predet.: 0
Número de salidas ON simultáneamente	10
Longitud de cable (metros)	500 apantallado, 150 no apantallado