Universidad de Valladolid

ESCUELA DE INGENIERÍAS INDUSTRIALES

UNIVERSIDAD DE VALLADOLID

ESCUELA DE INGENIERIAS INDUSTRIALES

Grado en Ingeniería Electrónica y Automática Industrial

Título del TFG

Creación de una célula robotizada para formación basada en equipos Fanuc y Siemens.

Autor: Delgado García, José

Tutor: Herreros Lopez, Alberto.

Departamento: Ingeniería de Sistemas y Automática

Valladolid, 18 de Mayo de 2020

Resumen

Este TFG tiene como objetivo crear un aula de formación, basada en los buses de campo Profinet y Profisafe para la creación de señales seguras, de autómatas Siemens por medio de programación en Step 7 y programación de robots de la marca Fanuc. La creación de esta aula de formación, irá desde el cableado eléctrico de los distintos elementos de la instalación; como armarios eléctricos, hasta el diseño mecánico y construcción de los diversos útiles.

El aula de formación, está concebida como una zona de carga de operario, el cual está protegido por medio de software en caso de una intervención no segura, donde se cargarán las piezas diseñadas. Dichas piezas serán cogidas por nuestro robot siempre y cuando el operario se encuentre en una zona segura y dejadas en el puesto anexo.

Palabras Clave

STEP 7.

FANUC.

Profinet.

Comunicación.

Programación.

Abstract

The Project speaks about the creation of an education center, based on fieldbus Profinet and Profisafe for the develop of safety signals, of Siemens PLC programed by Step7 and programing of Fanuc branded robots. The develop of this education center, will go from the electric wired of the different elements of the facility like electricity cupboard, to the mechanical design and built of the assembly.

This programming center, is conceived as a operator charging zone, this one its protected by software in case of an unsafe intervention, where the different parts will charged. That parts, will be taken by the robot, always in safety conditions for the operator, and will be leave it in the next assembly.

Keywords

STEP 7.

FANUC.

Profinet.

Comunication.

Programing.

Contenido

Objetivo

En el año 2015, la empresa francesa Icare Systems, estableció su sede española en Valladolid. A lo largo de los años, ya en 2019 se independizó de la filial francesa dando lugar a Icare Automation. Debido a la creciente demanda de gente formada en automatismos y robótica, una de mis tareas en esta empresa, ha sido la de elaborar un aula de formación con el objetivo de poder ofrecer una educación profesional y enfocada a nuestros clientes.

Para la realización del proyecto no contábamos con absolutamente nada, salvo un pequeño autómata, concretamente el 315-2 PN/DP. La idea de éste aula de formación era la de establecer las bases del estándar de programación del grupo Renault.

Para conseguir este objetivo, los paso a seguir serán en primer lugar hacer acopio de los materiales necesarios. Debido a los estándares de programación de dicha empresa, será necesario realizar dos armarios eléctricos específicos. La lista de los materiales a usar son estándar por lo que no podremos realizar ningún tipo de modificación sobre ella. Cuando tengamos los materiales realizaré el cableado y la instalación de los armarios dentro de nuestra aula de formación.

Con los armarios ya cableados en su lugar, el siguiente paso que realizaré será hacer buscar los materiales disponibles en el taller para poder crear un puesto de operario lo más completa posible, para ello contamos con una serie de elementos que enumerase explicarán más a delante, contamos con electroválvulas de accionamiento externo e interno, cartas de entradas y de salidas de la marca Festo, cartas de seguridad Festo, puertas de seguridad Euchner, barreras inmateriales Sick, imanes neumáticos Goudsmit, pilotos neumáticos de la marca Destaco, y distintos elementos eléctricos como balizas para el operario, botoneras etc.. de la marca Schneider.

Mi idea es la de diseñar con los elementos que tengo un puesto de carga operario, el operario tendrá un montaje en el que depositará una pieza, saldrá de la zona de seguridad creada con las barreras inmateriales Sick, y validará esa pieza por medio de un pulsador fuera de la zona de carga, si la pieza cargada está bien posicionada se encenderá un luz roja que prohíbe el trabajo, si ha sido mal cargada, la luz verde permanecerá encendida a la espera de intervención por parte del operario. Según la disponibilidad de elementos mecánicos dicha pieza quedará sujeta hasta que robot entre a tomar esa pieza, y dejarla en otro montaje de distinta geometría.

El cableado de los elementos de la instalación será el siguiente paso, tendremos un juego de electroválvulas para la garra del robot, otro juego para el montaje del operario, un montaje con detectores para dejar la pieza, una puerta de seguridad y un pupitre de validación del operario.

Con todo ello cableado, el siguiente paso que debo realizar, es cablear las señales de los elementos de la instalación y los 24 V de los mismos, de acuerdo a lo

que indica el estándar de la empresa. Una vez hecho esto deberé realizar el cableado de las comunicaciones industriales Profinet.

Debido a la necesidad de comprar un robot, contactaremos con un comercial de la empresa Fanuc, ya que es el robot más demandado actualmente en la industria por encima de ABB y Kuka, ya que son muy fiables y considerablemente más barato. El robot debido a nuestras necesidades no será el empleado en las fábricas actualmente si no uno más pequeño.

Otra de las necesidades de éste aula de formación será la creación de los montajes físicos es decir, la mecánica. Para ello utilizaré las herramientas de la oficina y restos de canaleta de otras obras así como soportes de acero de los que disponemos.

Un vez tenga todos los elementos de la instalación en su posición y fijados al suelo en el caso del robot y los montajes, comenzaré a realizar las comunicaciones y reconocimiento por parte del autómata de los distintos elementos de la instalación. Con el robot, realizaré las trayectorias directamente sobre los montajes, ya que es el modo en el que se programa en las fábricas del grupo mencionado anteriormente.

Por otro lado, realizaré la programación del autómata de manera off line, y posteriormente la cargaré corrigiendo in situ los problemas que vaya teniendo.

Estado del Arte

Automatización Industrial

En primer lugar, vamos a definir a grandes rasgos qué es la automatización industrial. Es la aplicación de distintas tecnologías para controlar un proceso o una máquina que desarrolla tareas repetitivas, haciendo que ésta opere de modo automático reduciendo al máximo la intervención humana. En definitiva lo que la automatización industrial busca es la productividad, reduciendo tiempos de producción y manteniendo la calidad producto mientras se reducen sus costes.

Hoy en día, la automatización se encuentra en todos los lugares, desde la fabricación de alimentos, productos sanitarios o químicos, hasta las industrias de materias primas o producción de automóviles.

Historia de la Automatización Industrial

Si buscamos sobre la historia de la automatización en general, veremos que está desde sus inicios muy ligada a la robótica. Desde los griegos se creaban brazos mecánicos para sus templos, para así captar la atención de la gente. Durante los siglos XVII y XVIII se construyeron muñecos que hacían movimientos de manera autónoma.

Entrando ya en el siglo XIX, encontramos las tarjetas perforadas [1], no eran más que láminas de cartulina con perforaciones. Pero dichas perforaciones no se hacían al azar, si no que empleaban código binario, de este modo podían crear distintas instrucciones, como si de un programa actual se tratase. Estas tarjetas, fueron de gran uso en la industria textil. Cabe mentar a James Watt, inventor de la máquina de vapor, gracias a él se pudo dar músculo a la industria del siglo XIX, dando lugar a la revolución industrial, gracias él se podría producir de manera más barata y más segura.

Si bien ya teníamos el músculo y un primitivo cerebro, a mediados del siglo XX, en torno a 1968, surgiría la necesidad de crear procesos industriales mucho más autónomos, más flexibles y más rápidos. En respuesta a esta necesidad surgieron los PLCs, (Programmable Logic Controller) de la mano de Bedford Associates, creando un controlador modular digital. Con ellos buscaban una manera sencilla de programar procesos industriales para que todo el mundo fuera capaz de usarlos, desde los ingenieros hasta los empleados de mantenimiento. Se diseñaron también pensando que debían ser sistemas robustos, capaces de aguantar durante años, pero también flexibles, capaces de adaptarse a cualquier proceso industrial o modificación del mismo [2].

Autómatas programables

Los autómatas programables, o PLC son el cerebro de la automatización industrial, se basa a grandes rasgos de un pequeño ordenador que es capaz de automatizar procesos de producción complejos. Se usan desde hace unos 20 años, sustituyendo a los sistemas cableados de relés. EL PLC proporcional es más sencillo, así como más flexible en caso de necesitar ampliar el proceso industrial. Gracias a su programación interna de contadores, temporizaciones etc…hace que el control del proceso sea más preciso. Si buscamos una definición, la más extendida es la que nos proporciona la NEMA (Asociación Nacional de Fabricantes Eléctricos)[3].

"Instrumento electrónico, el cual utiliza una memoria programable en l que guarda instrucciones de determinadas funciones, como operaciones lógicas, secuencias de acciones, especificaciones temporales, contadores y cálculos para el control mediante módulos de E/S analógicos o digitales sobre diferentes tipos de máquinas y de procesos".

Los PLCs son sistemas muy especializados en los procesos industriales. Son capaces de tomar y analizar la información de su entorno y utilizarla para controlar el proceso industrial que lleva a cabo. Como se usan para controlar procesos industriales, una de sus características es que son muy rápidos, recibe la información y la sintetiza.

Debido a la flexibilidad de estos sistemas, debido a la gran cantidad de detectores disponibles, son capaces de adecuarse a cualquier tipo de proceso industrial. Otra de las cualidades de los PLC es que es capaz de adaptarse a cualquier modificación a un precio competitivo [4]. Sin embargo, constan con la desventaja de necesitar mano de obra especializada para la programación del proceso industrial, aumentando los costes del mismo.

Estructura del PLC

A continuación, vamos a hablar sobre la estructura de los PLCs.[5]

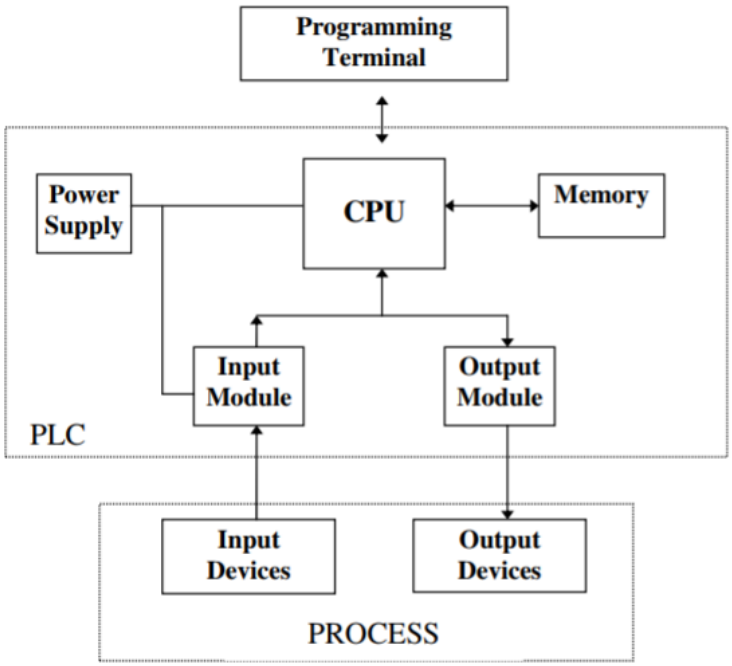


Ilustración 1: Estructura del PLC

- **CPU**, Unidad Central de Proceso es la parte que se encarga de interpretar las instrucciones del programa y monitorizar el estado de las entradas del sistema por medio de sensores. Mediante un bus de datos se transmite y se envía la información de la CPU al resto de partes del sistema [7].
- **Fuente de alimentación**, como todo dispositivo electrónico, necesita de energía para poder funcionar, la alimentación en general suele ser de 24 V que provienen de una fuente de alimentación. En algún tipo de autómata, se incluye algún tipo de suministro de energía auxiliar como pilas, para que en caso de que la fuente de alimentación falle se pueda mantener el proceso durante unos minutos.
- **Memoria** [18]. Todos los PLCs constan de memoria RAM y memoria ROM, variando su cantidad según el tamaño del proceso. Todas las memorias de los PLC se pueden sustituir en al menos cinco:
    - **Memoria Ejecutiva**. Es el propio sistema operativo del PLC, es el que realiza el escaneo de las funciones del PLC. Ayuda al microprocesador a entender las instrucciones programadas por el usuario.
    - **Memoria del Sistema**. Mientras la memoria executiva realiza sus funciones, en algunas ocasiones necesita almacenar datos de las mismas. Esta memoria solo puede ser usada por la memoria ejecutiva.
    - **Memoria de las Entradas/Salidas**. Cada entrada o salida del sistema, tiene asignada una parte de memoria, éstas tienen una única dirección. Durante la ejecución del programa, el microprocesador escanea el programa que ha creado el usuario e interpreta los comandos, así como lee el estado de las entradas y altera el de las salidas.

  - **Memoria de Datos.** En el momento que temporizadores, contadores u operaciones matemáticas son utilizadas en el programa, parte de la memoria debe almacenar el resultado para poder ser tenido en cuenta en cada escaneo del microprocesador.
  - **Memoria del Programa.** Esta memoria está destinada a almacenar el programa creado por el usuario.
- **Módulos de Entrada/Salida.** Hacen de interconexión entre los módulos de gran potencia a los circuitos de baja potencia. Son el nexo entre el PLC y el proceso industrial a controlar. Permiten conectar al PLC directamente con los actuadores del proceso. Lo más normal es hablar de señales de 12V-24V.

Por otro lado, un PLC no puede ser programado directamente, necesita de una interface, normalmente, se conecta a un ordenador el cual ejecuta el programa del fabricante Siemens, Schneider, Omron etc… Y por medio del propio programa el usuario es capaz de modificar el sistema y programar el sistema de control del proceso industrial.

Tipos de Autómatas Programables

Los autómatas programables se categorizan normalmente de acuerdo a la siguiente clasificación [6]:

- PLC Nano. Son autómatas que integran su propia fuente de alimentación, las entradas, las salidas y la CPU dentro del mismo encapsulado. Al tener un tamaño reducido el tamaño de las entradas y las salidas que puede manejar se ve reducido, normalmente no más de 100.
- PLC Compacto. Al igual que los anteriores tienen fuente de alimentación, CPU, entradas y salidas en el mismo encapsulado, sin embargo, cuenta como ventaje que es capaza de aguantar más entradas/salidas y de más tipos, e incluso interfaces operador.
- PLC Modular. Este tipo de PLC es el empleado en la industria como tal. Consta de un rack, o bastidor donde se irán colocando los distintos elementos del sistema. Podremos conectar infinidad de elementos.

Comunicaciones Industriales

Los elementos que se utilizan en un proceso industrial son de todo tipo, desde detectores láser, actuadores neumáticos, motores etc… por lo que se necesita un sistema que englobe la heterogeneidad de dicho proceso. Los medios que empleamos para dar coherencia a esa disparidad de datos se denomina comunicación industrial. Entendemos por comunicación industrial al conjunto de métodos, sistemas y herramientas que nos permiten un intercambio de información coherente entre los distintos elementos del proceso industrial.

Las comunicaciones industriales, se emplean en los procesos industriales para poder comandar, supervisar y mantener un proceso industrial en tiempo real.

Historia de las comunicaciones industriales

A lo largo del siglo XX, la automatización industrial tiene un gran desarrollo, potenciado en parte por sistemas electromagnéticos de la mitad de siglo. La aparición en la segunda mitad de siglo de los autómatas programables crea un punto de inflexión en el control de sistemas. Gracias a los PLC, los procesos industriales se podían controlar de manera mucho más precisa y eficiente.

Debido a la implantación de los PLCs y su capacidad para controlar procesos industriales, surgió la necesidad de crear un medio de comunicación entre el proceso y el autómata a tiempo real, para permitir un mejor control, es el surgir de las comunicaciones industriales.

Morfología de los Sistemas Industriales de Control

La manera de estructurar un proceso industrial depende de varios factores, entre ellos la posibilidad de subdividir el proceso, la seguridad y la importancia de los tiempos de respuesta por ello, se distinguen tres morfologías distintas de un sistema de control [12].

Control Centralizado

Este tipo de sistema de control se emplea en procesos industriales poco complejos, donde la totalidad del proceso puede ser controlado por un solo autómata. Según el tamaño del proceso se empleará un autómata con mayor o menor capacidad para la gestión del proceso, ya que al controlar todo el proceso debe de ser capaz de gestionar todas las señales y salidas con la menor pérdida de tiempo posible.

Una de las ventajas de este tipo de sistemas es que no es necesario planificar intercomunicaciones entre los procesos, el propio autómata gestionas sus propias señales, el coste por tanto es más reducido al usar un solo autómata y no necesitar de un estudio de comunicaciones extensivo. Como desventajas tenemos que si la programación del proceso falla, toda la instalación queda paralizada, y debido a la gran cantidad de líneas de código para el control del proceso, hace que el sistema sea mucho más complejo. El cableado de las señales de los detectores o de los actuadores debido a las grandes distancias a abordar implica una mayor complejidad en los cuadros eléctricos y en la búsqueda de fallos de los mismos.

Control Distribuido

A cada unidad del proceso industrial, se le asignará un autómata, dimensionado eso sí de manera coherente al tamaño del proceso. Debido a la individualidad de las operaciones debe existir un medio para comunicarse entre sí mediante una red de comunicaciones de datos. Por lo que todos estos autómatas que se encuentran en el proceso industrial deben de ser compatibles a dicho protocolo de comunicación.

Utilizando este modo de distribución del proceso de control, se crean procesos muy simples, reduciendo la posibilidad de errores, creando programas mas depurados y de fácil comprensión. Otra de las ventajas es que la existencia de un fallo en una parte del proceso no afecta al resto en contra, el poder individualizar los procesos de manera eficiente conlleva un proceso de estudio largo y costoso, ya que se debe diseñar el proceso para garantizar la independencia entre el resto y diseñar un modelo de comunicación adecuado.

Control Híbrido

En muchos casos, el individualizar los procesos de control no es sencillo, por lo que se debe recurrir a un sistema a medio camino entre distribuido y centralizado, un control híbrido.

Se busca reducir e individualizar el proceso lo máximo posible pero estructurada con elementos de control de nivel superior que son los que supervisan todas las partes de proceso, que son los encargados de gestionar la información común.

Pirámide CIM

CIM es el anagrama en ingles de *Computer Integrated Manufacturing,* es un modelo de automatización que busca la eficiencia del proceso industrial mediante la jerarquización de los elementos que lo componen. Busca aumentar la flexibilidad, mejorar la calidad del producto reducir los costes y tiempos de fabricación, es decir aumentar la competitividad de las empresas.

Ilustración 2: Pirámide CIM

El modelo CIM, busca relacionar todo el proceso industrial mediante un todo, analizando las partes que lo componen e interrelacionarlo todo. En la pirámide de la ilustración, vemos como cada nivel se asocia a una serie de trabajo específicos

asociado a un tipo de información distinto. Cada red gobierna las redes de nivel inferior, el flujo de información es tanto horizontal como vertical. En algunos casos cabe decir que la pirámide se divide en 5 niveles.

Nivel E/S

Es el nivel en el que se sustenta la pirámide CIM, está formada por los sensores que nos permiten recoger información del proceso industrial y por los actuadores que nos permiten tomar parte en el mismo y modificarlo. Son los elementos que están en contacto con el proceso industrial, son elementos como; sensores de presencia de pieza, finales de carrera, detectores de presión y temperatura, actuadores como motores, pistones y aprietes neumáticos etc…

Nivel de Campo y Proceso

En este nivel encontramos los sistemas que adquieren la información de los detectores y gestionan las acciones por parte de los actuadores, hablamos del nivel en el que encontramos a los autómatas programables, equipos como los robots o controladores de los motores.

Combinando el nivel anterior y ésta ya tenemos un nivel de control suficiente para un proceso industrial. De hecho en los procesos industriales de pequeñas dimensiones o de empresas de pequeño tamaño con estos dos niveles es suficiente para controlar el proceso. Aunque sean procesos simples, no implica que carezcan de la implantación de buses de comunicación debido a la gran cantidad de sensores y actuadores de los que recoger informaciones.

Nivel de Control

Todos los dispositivos del nivel de campo y proceso pueden ser monitoreados si se aplica un sistema de comunicación adecuado, que sea capaz de interconectar los distintos procesos con los sistemas de gestión y supervisión.

Este nivel se encarga de controlar lo que sucede en todos los procesos de la planta, si ponemos el ejemplo de una línea de producción de vehículos contralaría la producción del chasis del coche y de la integración de los motores al mismo tiempo. Mediante SCADA (supervisión, control y adquisición de datos) podemos obtener una imagen virtual de nuestra fábrica, pudiendo recorrer desde un mismo ordenador el estado completo del proceso industrial, pudiendo ver los fallos que ha habido y las paradas del proceso de producción. Para ello es necesario establecer una comunicación entre los autómatas de la planta, y los sistemas de control, normalmente se emplea una red LAN mediante cable ethernet, no como el que tenemos en casa, con prestaciones mucho más altas claro está, permitiendo conectar gran cantidad de sistemas sin pérdidas de información.

Nivel de Gestión

Éste es el nivel más alejado del proceso productivo, se emplean ordenadores de maneral general. Gracias al nivel de control antes explicado, podemos obtener información de todos los niveles inferiores, con toda esta información el nivel de gestión de la empresa crea estadísticas sobre el proceso industrial como costes, rendimiento productividad etc… en general, datos para la toma de decisión de los altos cargos y facilitar la toma de decisiones.

El tipo de comunicaciones que se emplean aquí ya dejan de ser tan robustos y de corto tiempo de respuesta como en los procesos industriales, sino que lo importante ahora son los datos que se transmiten. Normalmente, se empleas métodos de comunicaciones más baratos y estándar como el ethernet que estamos acostumbrados en nuestras casas.

Redes de Comunicación

Las comunicaciones empleadas en un proceso industrial tienen que tener una serie de características en concreto, deben de ser muy robustas, ya que van a estar expuestas a condiciones poco favorables. El tipo de cable que se emplee debe de estar apantallado para evitar que los cables de alta potencia de la instalación afecten a las comunicaciones, también deben de ser resistentes ya que es posible, dependiendo del tipo de instalación que estén sometidos a estrés de calor, como posibles cortes. Al nivel de éste TFG, lo que más nos interesa es conocer los buses de campo.

Un bus de campo o “field bus”, es un sistema de transmisión de datos que lo que hace es interconectar dispositivos industriales y elementos de control de forma mucho más barata y efectiva que los métodos analógicos. Son sistemas bidireccionales sirven tanto como enviar como recibir señales, otra de las grandes ventajas es que son muy flexibles, es decir si nuestro proceso industrial se ve ampliado el poder controlar las señales de la nueva parte es mucho más simple y económicamente viable. Por otro lado, los buses de campo se caracterizan por ser muy seguros ya que permiten el diagnóstico de las señales y comprobar el estado de la instalación. Por medio de esta comprobación es posible saber si un instrumento está actuando de la manera que debería, facilitando las tareas de mantenimiento, localizando los errores de forma concreta dentro de una gran instalación. Como decíamos, los buses de campo son sustitutos del cableado analógico, ello conlleva la pérdida de grandes armarios de conexiones de control, reducción del número de PLCS ya que el número de cartas a integrar es menor.

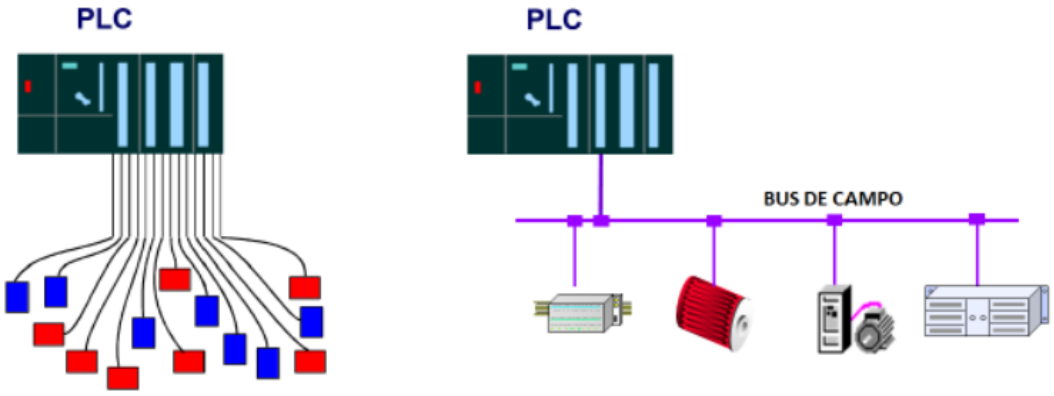


Ilustración 3: Cableado convencional vs bus de campo.

El gran problema de los buses de campo es que no están normalizados, por lo que existe una gran variedad de ellos con distintas características y aplicaciones. Otros de los problemas es que la rotura de un solo cable de comunicación conlleva la caída de toda la instalación.

Profibus y Profinet

Si buscamos tipos de buses de campo, encontraríamos una lista sin fin de ellos, ya que cada marca ha creado su propio bus de campo y otros han surgido de manera libre. Por lo que la cantidad de buses es enorme. Sin embargo, para no extendernos, hablaremos de los que no que nos incumben en este proyecto de fin de grado.

Profibus

PROFIBUS (PROcess Field BUS) es el más usado en Europa, impulsado por la industria alemana y grandes fabricantes como ABB, Siemens, AEG etc... Es un bus de campo abierto, es decir, son buses de campo que no están creados por empresas, si no que tienen especificaciones públicas y disponibles a un precio mucho más razonable. Está estandarizado por la norma DIN 1245, lo que proporciona confianza al usuario, así como la posibilidad de comunicar equipo de distintos fabricantes sin ningún tipo de problema. Dentro de profibus, encontramos varios tipos:

- **Profibus-DP.** Son redes que tienen un maestro y un esclavo. El maestro sabe qué tipo de esclavos tiene conectados, inicializa la red y verifica la correcta configuración de los esclavos. Entre ellos se envían información de manera continua. Un ejemplo de ello son los sistemas de aplicación de resina mástica en el proceso de fabricación de vehículos, que son esclavos de los robots.
- **Profibus-PA.** Las siglas PA significan Process Automation, mediante esta tecnología se permite su aplicación en industrias con riesgos de explosión.

- **Profibus-FMS.** Se emplea en la comunicación de los PLCs con procesos pequeños, en este caso es más importante la funcionalidad que la información.

Profinet

Este es el estándar que vamos a emplear en nuestra isla robotizada. Este tipo de red permite la conexión de equipos desde el nivel de campo (actuadores PLCs…) hasta los sistemas informáticos e internet, que sería el nivel de gestión que hablábamos antes. Se basa en Ethernet, por lo que utiliza los protocolos TCP/IP para la interconexión y transferencia de datos. Como decimos, se basa en Ethernet, pero no tiene nada que ver con el que tenemos en nuestra casa. Hablamos de Ethernet Industrial, permite la creación de redes de comunicación de gran número de dispositivos. Es un sistema que ofrece lo mismo que el Ethernet de casa pero más seguro, más potente y que permite el control de acceso mediante usuarios etc.. Una de las cosas más importantes de la red de profinet, es que la caída de la red puede ser peligrosa en el proceso industrial, por lo que se emplea comunicación extremo a extremo, es decir se envía un paquete de datos y se recibe una respuestas del mismo.

Como características del profinet tenemos; ofrece intercambio de información a tiempo real, se pueden utilizar switches , valen de sistemas maestro esclavo, es un protocolo abierto, utiliza la clavija estándar rj45, sin embargo no permite conexiones de más de 100 metros.

Robótica Industrial

Historia de la Robótica

En primer lugar, antes de hablar sobre la evolución e historia de la robótica, debemos de establecer el origen de la palabra robot.

Origen de la Palabra Robot

El origen de la palabra robot, proviene del eslavo, concretamente de la palabra robota, que significa “labor forzada”.

La primera vez que podemos encontrar la palabra robot en la historia, se remonta a 1920, cuando el dramaturgo Karel Čapek, de origen Checo, presenta su obra Rossum´s Universal Robots (R.U.R) en el Teatro Nacional de Praga. En la obra de Karel, aparece la figura de robot como la invención de una empresa para aligerar la carga de trabajo de los humanos. Sin embargo, esta palabra se popularizó por el conocido escritor de ciencia ficción Isaac Asimov en 1942 con su relato Runaround, en el que expone las tres leyes básicas de la robótica:

- Un robot no hará daño a un ser humano o, por inacción, permitirá que un ser humano sufra daño.

- Un robot debe cumplir las órdenes dadas por los seres humanos, a excepción de aquellas que entrasen en conflicto con la primera ley.
- Un robot debe proteger su propia existencia en la medida en que esta protección no entre en conflicto con la primera o con la segunda ley.

Origen y Evolución de la Robótica

Una vez que hemos establecido el origen de la palabra robot, llega el momento de hablar de lo que es un robot industrial, sin embargo, no existe una definición universal, ya que todas las federaciones nacionales e internacionales tienen la suya propia, aquí exponemos algunas de ellas:

- Robot Institute of America. (RIA): Un robot industrial es un manipulador multifuncional reprogramable, capaz de mover piezas, materias, herramientas o dispositivos especiales, según trayectorias variables, programadas para realizar tareas diversas.
- Asociación Internacional de Estándares (ISO): Un robot industrial es un manipulador de 3 o más ejes, con control automático, reprogramable, multiplicación móvil o no, destinado a ser utilizado en aplicaciones de automatización industrial. Incluye al manipulador (sistema mecánico y accionadores) y al sistema de control (Software y hardware de control y potencia).

El ser humano siempre ha querido construir máquinas para facilitar su trabajo o simplemente para su propio entretenimiento. Desde el 400 a.c ya encontramos escritos que hablan de objetos autónomos, como la paloma de Arquitas de Tarento, el cual había creado una paloma de madera que se movía con vapor de agua. Posteriormente, en la edad Media y el Renacimiento se siguieron construyendo autómatas, entre ellos, el gallo de Estrasburgo (1350), el cual al marcar las horas movía el pico y las alas, es actualmente el autómata más antiguo que se conserva. Durante los siglos XVII y XVIII, se construyeron autómatas de mayor complejidad, en el gremio relojero se popularizó la creación de autómatas, Jaquet-Droz destaca por haber creado el escribano autómata, el cual era un niño sentado en un pupitre que podía realizar hasta cuatro dibujos distintos imitando el comportamiento humano. Jaques de Vaucanson (1709-1782) construyó varios autómatas, el más conocido es el pato mecánica, capaz de comer, beber e incluso de digerir la comida por disolución para después evacuarlo.

Cabe destacar, a James Watt (1736-1819) que influyó de manera decisiva al desarrollo de las máquinas autónomas, con su regulador de Watt, que es un regulador centrífugo de acción proporcional, con él apareció el concepto de realimentación y regulación automática, base de los actuales robots industriales.

Ya en el siglo XX, es donde se empiezan a sentar las bases de lo que es la robótica industrial como la conocemos. En 1954, George Devol comienza con la construcción de un brazo articulado que realizaba una serie de movimientos programados por computador, él lo denominaba " dispositivo de transferencia programada de

artículos". En 1958 Devol y Joseph Engelberger se conocieron, y empezaros a trabajar en el desarrollo de sus máquinas para el ámbito industrial, fundando en 1960 Unimation.

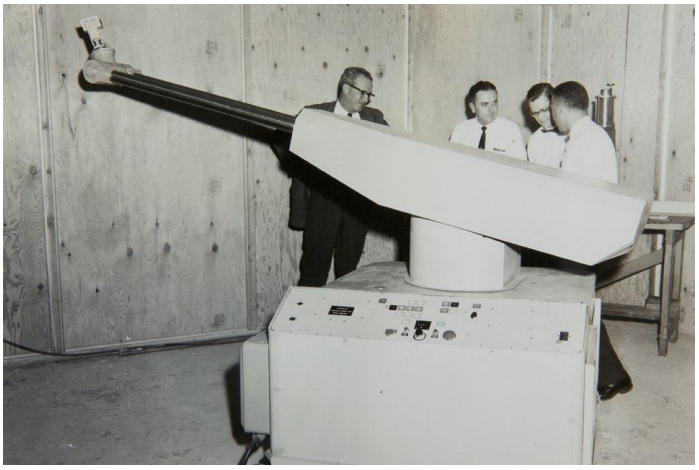


Ilustración 4: Robot Unimate

Años más tarde, instalaron el primer robot de la empresa en General Motors, dicho robot el "Unimate" estaba controlado por interruptores y finales de carrera y motores hidráulicos. El robot se empleó para la carga y descarga de piezas en una máquina de fundición por inyección. Ya en 1972, Nissan crea la primera asociación de robótica del mundo JIRA (Japanese Industrial Robot Association) dos años más tarde, Estados Unidos ante el gran crecimiento del país nipón en la industria, crea Robot Institute of America. (RIA). En Europa la fiebre por la robótica tardaría en llegar, en 1973 la empresa sueca ASEA, construye el primer robot de accionamiento completamente eléctrico el IRB6. Ese mismo año, se desarrolla el primer lenguaje de programación de robots, WAVE. Posteriormente, en 1974, surge el lenguaje de programación AL. En ese mismo año, la empresa Milacron crea el robot T3 el cual se caracterizará por su control por ordenador.

En 1978, la empresa Unimation, sorprendió al mundo creando el robot PUMA, acrónimo de Programmable Universal Machine for Assambly. El robot fue desarrollado por Victor Scheinman basándose de su experiencia en General Motors, este robot fue clave en las tareas de montaje, actualmente, el prototipo original se encuentra en el Museo Nacional de Historia Americana.

Unos años más tarde ya en 1982, en Japón Hiroshi Makino, desarrollo el robot SCARA, un acrónimo de Selective Compilance Assembly Robot Arm. Era un pequeño robot de 4 grados de libertad de posicionamiento horizontal. Se caracterizan por que su tiempo de ciclo de trabajo era muy reducido, haciendo tareas de gran repetitividad. En 1989, la japonesa Kato Corporation, crea el primer robot bípedo, el WL12RIII. Éste robot era capaz de andar y de incluso subir escaleras.

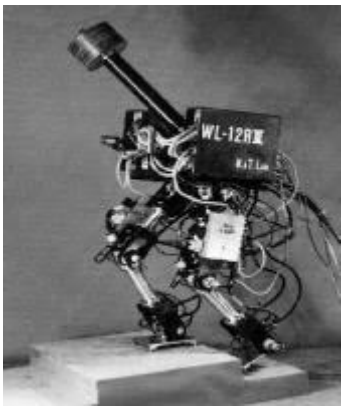


Ilustración 5: Robot bípedo WL12RIII.

En los años posteriores, siguió evolucionado la robótica, hasta lo que es hoy en día, pero antes de finalizar este recorrido por la historia de la robótica, se debe mencionar en 2001 la creación de ASIMO, diseñado como un asistente personal, capaz de reconocer la cara, voz y nombre de su propietario.

Definición y Clasificación de los Robots

Antes de comenzar a clasificar los distintos tipos de robots, vamos a hablar de las distintas generaciones. Agrupamos a los robots en distintas generaciones de acuerdo a su desarrollo tecnológico, si buscamos información sobre las distintas generaciones, encontraremos que se dividen de 3 a 5 generaciones dependiendo del autor.

- **Primera generación:** Son aquellos robots que desarrollan actividades de forma secuencial, su interacción con el ambiente es prácticamente nula.
- **Segunda generación:** Interactúan en cierto modo con su entorno, adquieren información por medio de sensores y actúan en consecuencia a ellos.
- **Tercera generación:** Poseen capacidad de planificación, son robots capaces de adquirir información de su entorno e interactuar con el de diferentes maneras. Aquí es donde comienza la creación de robots inteligentes, los cuales usan ordenadores para su control y toma de datos del entorno.
- **Cuarta generación:** En este punto hablamos de la compenetración de robot tal y como lo conocemos con una inteligencia artificial, la cual permite al robot aprender de las situaciones y tomar decisiones respecto a ella.

A continuación, pasamos a clasificar a los manipuladores desde el punto que nos concierne, el industrial. Entre ellos podemos encontrar:

- **Robots SCARA** [9].Los robots SCARA surgieron en Japón como comentamos anteriormente, son robots que surgieron para solucionar el elevado coste del resto de alternativas para realizar tareas que demandaban bajo tiempo de ciclo. A grandes rasgos se parecen a un brazo humano, con la salvedad de que el movimiento de muñeca es rígido a diferencia de otros robots que veremos luego. Este robot sacrifica la flexibilidad por velocidad dentro de su

rango de acción, por ello son la opción por excelencia para las funciones de pick and place.

Ilustración 6: Fanuc SR-3iA

- **Robots cartesianos** [10]. Los robots cartesianos son aquellos que operan exclusivamente de forma lineal, el caso más representativo es el CNC. Su movimiento siempre es recto, no pueden realizar giros, emplean para los movimientos cadenas y servos que permiten realizar los movimientos de forma precisa y rápida. Son empleados en la mayoría de tareas de paletización de cargas de tamaño considerable, sobre todo en líneas de envasado con tareas monótonas. Normalmente tienen tres actuadores que les permiten moverse en los tres ejes X, Y y Z, aunque en algunos casos se añada un eje de rotación al extremo de la herramienta.

Ilustración 7: Gesku Robot Cartesiano

- **Robots paralelos** [11]. Este tipo de robots, está formado por bases, una fija y una móvil, las cuales están unidas por cadenas cinemáticas seriales. Son utilizados en aplicaciones que requieran movimientos rápidos y precisos de una línea de producción. Se caracterizan por que su relación carga potencia es alta, son capaces de manipular cargas de peso superior a sí mismos. La

mayor desventaja, es que no existen ecuaciones que modelen la configuración del robot, es decir cada para cada robot se debe construir su modelo dinámico.

Ilustración 8: IRB 360

- **Robots antropomórficos** [13]**.** La palabra antropomórfico, proviene del griego «anthrōpos», «hombre», y μορφή, «morfē», «forma» . Esta palabra se aplica al parecido entre algo y al cuerpo humano, por ejemplo los robots humanoides son antropomórficos. Si nos referimos más al ámbito industrial, los robots antropomórficos se refieren a los manipuladores industriales típicos, los cuales se asemejan al brazo de un ser humano, en la manera de posicionarse y orientarse en el espacio. Esta configuración de robots se emplea para realizar tareas que requieren de mover grandes pesos, y alta repetibilidad.

Ilustración 9: Kuka KR 500.

Dentro de los robots antropomórficos, está surgiendo una nueva categoría que se está implantando de manera rápida en el ámbito industrial, los robots o robots colaborativos. Se define como robot colaborativo aquel que está destinado a trabajar con un ser humano, el robot aporta la fuerza y resistencia mientras que el humano aporta su capacidad de resolver problemas y

destreza. Dentro del rango de robots [14], podemos encontrar robots similares a los manipuladores industriales, como los UR3 de Universal Robots, o incluso exoesqueletos que se acoplan al operario ayudándolo en sus tareas en la línea de fabricación.

Ilustración 10: Robot Colaborativo UR3.

Manipuladores Industriales.

Antes de comenzar a hablar más en detalle de los manipuladores industriales, debemos hablar de una serie de términos empleados por los fabricantes de robots.

- Grados de Libertad: Es cada uno de los movimientos que puede realizar cada articulación respecto con la anterior. Un robot será más flexible cuántos más grados de libertad tenga.
- Zona de Trabajo: Es la zona en el espacio en la que el robot puede trabajar, viene dada por las limitaciones físicas del robot.
- Capacidad de Carga: Es la máxima carga que un robot puede cargar a una velocidad determina considerando su configuración más desfavorable.
- Resolución: Es la variación mínima en el desplazamiento que puede realizar el robot.
- Precisión: Es la diferencia entre el punto programado y el punto que se desea.
- Repetibilidad: Es la capacidad de reproducir el mismo punto de forma sucesiva sin variación.

Un manipulador industrial consta de dos partes, el armario de control y el brazo robot.

El armario de control es el que regula y controla las acciones del robot, realizando los cálculos necesarios y procesando la información que recibe del manipulador. Dentro del armario de control, o controlador como lo llamaremos a partir de ahora encontraremos:

Panel de control. En el panel de control será donde encontremos los mandos de anulación y rearme de sistema, puesta en manual o automático de la bahía y por último la parada de emergencia del controlador.

Computadora principal. Es la parte encargada de realizar los cálculos de movimiento, control de la alimentación suministrada y hace las veces de memoria del sistema ya que será donde se guarden los distintos programas que vayamos a realizar. Es el encargado de enviar y recibir las señales del sistema, calcula las trayectorias y movimientos de los servos para evitar las singularidades para alcanzar el punto deseado, así como la detección de posibles colisiones entre los ejes o superficies externas. A esta computadora por medio de las distintas tarjetas que monte, conectaremos nuestra red de profinet para poder establecer la comunicación y la relación maestro esclavo.

Fuente de alimentación. Para que todo el sistema del robot funcione, es necesario una alimentación, normalmente hablamos de trifásica 400V para poder mover los servos, también dentro de la bahía del robot podemos encontrar salidas de 24V para el control de la garra que haya sido instalada.

El brazo de un robot industrial está compuesto por una serie de elementos rígidos unidos entre sí por unas articulaciones. Si buscamos la analogía con el brazo humano los elementos de unión serían nuestro codo mientras que los elementos rígidos de los que hablamos que dan la fuerza estructural al manipulador corresponderían a lo que es el antebrazo. En el brazo del robot encontraremos todo el cableado de potencia que viene desde la bahía y los distintos motores, así como los encoders que nos permiten saber cuánto ha girado dicho motor. Actualmente como elementos de potencia se emplean motores de corriente alterna sin escobillas ya que nos permiten regular la velocidad y el par del manipulador variando solamente la frecuencia de la corriente. Para controlar la posición como hemos dicho antes, se utilizan encoders ópticos seguidos por resolvers al girar el motor el disco que forma el encoder también gira generando una serie de pulsos por medio de un láser al emisor creando una relación posición número de pulsos recibidos. La última parte a destacar dentro del manipulador industrial es la garra, según el tipo de aplicación a la que esté destinada será de un tipo u otra ya sean aprietes y pilotos para agarrar metales, o ventosas para agarrar lunas de coches etc. según sea la aplicación también dependerá el tipo de accionamiento de la misma garra, ya que para trabajos delicados podemos usar simples imanes o incluso la fuerza hidráulica para tareas que requieran mayor esfuerzo por parte del manipulador.

Desarrollo de la Instalación.

Elementos de la Instalación.

Para la realización de la instalación se procederá por etapas, la primera de ella será el cableado de los armarios eléctricos que utilizaremos en la instalación. Posteriormente se cablearán los elementos que usaremos, electroválvulas,

elementos de comunicación robot autómata etc… Una vez tengamos nuestra instalación física conforme, procederemos a la programación del autómata, estableciendo la distintas comunicaciones industriales y por último la programación del robot.

Armarios Eléctricos.

En éste aula de formación, encontramos dos armarios eléctricos, estos dos armarios cumplen el estándar creado por Renault para sus propias factorías, por lo que son unos armarios que vienen prediseñados por lo que no intervenimos en la fase de diseño de los mismos. Sin embargo, el cableado de los armarios y la interconexión de los mismos con el resto de elementos de la instalación no depende del estándar de Renault, si no de las propiedades de los distintos elementos a conexionar.

En la instalación distinguiremos dos tipos de armarios, ZB (armario de zona) y OPB (armario del operador).

Ilustración 11: ZB (izqda.) OPB (dcha.).

EL ZB es un armario que está enfocado al cableado de la instalación en sí, sin tener encuentra los lugares de intervención del operario ya que éstos cuentan con una seguridad extra. Los elementos que se cablean a este armario son los actuadores neumáticos fuera de la zona operario, elementos de seguridad progresiva neumática, conexionado de puertas de seguridad de la instalación etc… Por seguridad, es el propio ZB el que alimenta el armario del operador, por lo que el ZB será el que cuente con una fuente de 24V para alimentar a todos los componentes de la instalación. Dentro del armario ZB encontraremos 5 relés, 1 relé de reserva, 1 relé para el control de masa, y 3 relés para el funcionamiento de la

instalación y la obtención de un retorno eléctrico. Dichos relés son, el relé AU, para las paradas de emergencia, el relé ES para la puesta en servicio de la instalación y por último el relé de FZ que es el de zona cerrada. Todos ellos son accionados por variables de seguridad del autómata y obtenemos por su parte un retorno en las tarjetas de entradas.

En el OPB conectaremos los elementos propios del operador, como puede ser el pupitre de la instalación donde el operador selecciona la instalación en automático, las setas de seguridad y los puestos de carga propios del operador, por ello el OPB es un armario significativamente más pequeño que el ZB sobre el que recae toda la instalación. EL OPB es un armario que consta solamente de 2 relés uno que es para la seguridad del operario y otro de reserva.

Sin embargo, si hablamos de elementos Siemens propiamente, estos armarios son más similares de lo que parece.

Ilustración 12: Cartas Siemens ZB.

Ilustración 13: Cartas Siemens OPB.

Si observamos las Cartas Siemens ZB. y Cartas Siemens OPB. empezando por la izquierda encontramos una ET200s éste elemento es muy importante para la industria actual, ya que permite descentralizar los procesos y comunicar con la CPU de una manera muy simple. Este módulo permite el acople de múltiples tipos de cartas Siemens como las de entradas y salidas normales, de seguridad, arrancadores de motores, convertidores de frecuencia etc... La comunicación se realiza a través de Profinet lo que nos permite que con un solo cable seamos capaces de comunicar, enviar y recibir datos y conseguir diagnósticos de la propia ET200s. Esta tarjeta

cuenta con dos tomas de RJ45 para Profinet por lo que también nos sirve a su vez de pasarela para conectar otros elementos de la red, reduciendo el tamaño de cable y a su vez las posibles interferencias de la instalación.

Una vez tenemos nuestra ET200s que nos permite comunicar con el autómata, es momento de empezar a instalar las cartas, en nuestro caso al ser armarios prefabricados, se instalan una serie de cartas de alimentación, sin embargo no sería necesario ya que la propia ET200s permite el conexionado de 63 cartas siempre y cuando la longitud del bus no exceda los 2 metros.

Como hemos dicho, la siguiente carta es de alimentación de 24V en nuestros armarios encontramos 2 en el ZB y 3 en el OPB. La carta que se utiliza en cuestión es la siguiente: 6ES7138-4CA01-0AA0. La función de esta carta es impedir el paso de corriente a las cartas sucesivas si no tiene 24V entre su conexionado, la esencia de esto es por ejemplo inhabilitar una carta hasta que no tengamos cierto nivel de seguridad.

Las siguientes cartas que vamos a mencionar son las más comunes en toda instalación, son las propias cartas de entrada y salida de datos, en nuestro caso son entradas y salidas digitales y cada una acepta 8, sus referencias respectivamente son; 6ES7131-4BF00 -0AA0 y 6ES7131-4BF00-0AA0. En estas cartas recibiremos las informaciones de la instalación directamente sobre el autómata, y éste podrá accionar sus salidas directamente sobre la instalación.

Si observamos las fotografías encontramos unas cartas con un etiquetado amarillo, éstas cartas son de seguridad y pueden ser tanto de entradas como de salidas, a priori podríamos pensar que cuál es la diferencia entonces, éstas cartas a parte de tener el doble de tamaño y usar un slot más grande, son las encargadas de la seguridad de la instalación, es decir a estas cartas cablearemos las setas de emergencia, los contactores de seguridad etc…Las referencias de estas cartas son las siguientes; 6ES7138-4FB02-0AB0 y 6ES7138-4FA03-0AB0. Sin embargo, no sólo se diferencian de las tarjetas normales en tamaño y referencia, si no que el cableado interno es distinto, en una carta normal de entradas o salidas, una señal de 24V equivale a un borne de la tarjeta, pero con éstas tarjetas de seguridad no. Dos señales físicas de 24V son empleadas como una sola variables en el autómata. Que estas tarjetas usen esta gestión es propia seguridad, por ejemplo si una seta de emergencia fuera cableada con un solo cable y ese cable no funcionara, en caso de una emergencia sería inservible, por lo que se usa un cableado doble. Pero estas tarjetas son más versátiles, ya que esas señales, deben de ser configuradas dentro del propio autómata, si por ejemplo nuestra seta de emergencia esta cableada de modo que nos da dos señales a 1, en la configuración de la carta se debe definir también, es decir sería una configuración equivalente, por otro lado también se puede tener una señal si y otra no como son los selectores de seguridad, lo que sería una configuración no equivalente.

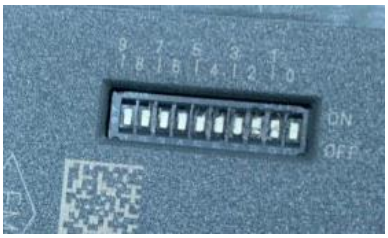


Ilustración 14: Safety Number.

Como hemos mencionado antes, usamos Porfisafe, por lo que todos los módulos de seguridad tienen un código único dado por el programa autómata que equivale a un código analógico que introducimos en el elemento de seguridad. De este modo se establece una asociación inequívoca entre la configuración del autómata y la configuración física de las tarjetas de seguridad. La configuración de ese número físico se hace por medio de unas pletinas con la configuración binaria, de éste modo en las tarjetas safety que empleamos podemos tener 2^8 combinaciones de números. Si observamos la ilustración Safety Number. El número que se ha puesto es 6, dicho número debe corresponder en el programa.

Autómata

Para la realización de este proyecto, vamos a usar la CPU 315-2 PN/DP. Ésta CPU es de un tamaño medio aplicada normalmente a sistemas distribuidos. Puede ser usada con un sistema de comunicación Profinet IO, lo cual hace que el cableado de señales y los armarios a emplear sean de pequeño tamaño.

Ésta CPU, nos permite bloquear por contraseña el programa, de tal modo que se añade un nivel de seguridad a la instalación. La CPU si existe un sobrevoltaje (superior a 28.8V), crea una backup del programa antes de interrumpir el proceso.

Actuadores neumáticos.

Para otorgar movimiento a nuestro puesto operario, hemos añadido una serie de actuadores neumáticos. Un actuador neumático es un elemento que por medio de aire a presión realizará un movimiento, que puede ser o lineales o de giro.

Si hablamos de actuadores mecánicos, la empresa que más destaca es la alemana Festo, fue fundada en 1925 es una multinacional focalizada en los suministros industriales, más en concreto en los actuadores neumáticos.

En primer lugar, vamos a explicar cómo funciona un poco un actuador neumático. Consta de dos partes diferenciadas; la interconexión de señales y comunicación con el autómata y la segunda la parte de accionamientos neumáticos propiamente.

Empezando por la parte eléctrica propiamente dicha, el primer módulo que encontramos es el llamado bus node con referencia CPX-(M)-FB33, es el encargado de realizar la comunicación con el autómata, la recepción y envío de los datos así

como el diagnóstico del módulo. Al igual que la ET200s consta de dos entradas de profinet pero sin embargo no es con enganche de rj45 si no con enganche de M12, al tener éstas dos tomas también cumple la función de pasarela para conexionar otros elementos después de él.

Ilustración 15: Actuador neumático garra.

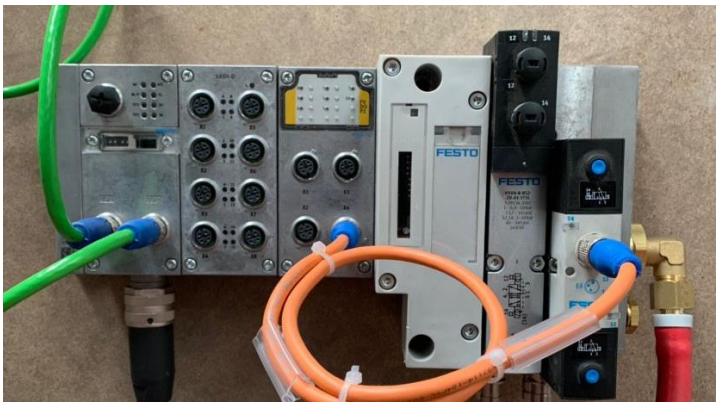


Ilustración 16: Actuador neumático operador.

El segundo módulo es el de entradas, con referencia CPX-16DI-D es el responsable de recibir las señales referentes al módulo Festo en su conjunto, en nuestro caso, será el responsable de recibir las señales de los detectores y el estado de los actuadores mecánicos del montaje. Es un módulo que cuenta con 8 entradas dobles, lo que es un total de 16 bits. El tipo de conexionado de las señales al igual que la señal de comunicación es de formato M12, con la peculiaridad de que el conexionado de profinet tiene 4 pines, mientras que el conexionado de entradas o salidas a un Festo consta de 5 pines ya que se añade una toma de tierra a la conexión de entradas.

Los cables de M12 que se emplean para las señales de entradas tienen 5 colores, blanco, negro, azul, marrón y tierra. Como hemos dicho el módulo de entradas consta de 8 conexiones M12, pero cada conexión tiene dos señales, por lo que según sea la interconexión de los cables recibiremos una señal u otra, de modo

que si conectamos entre si el marrón y el blanco recibiremos una señal y si es el azul y el negro la otra.

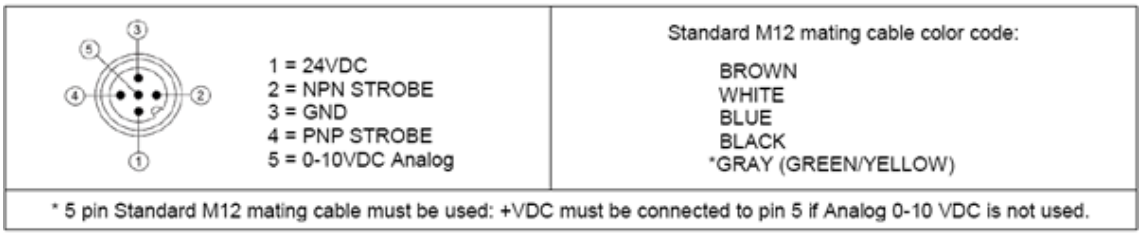


Ilustración 17: Conexionado M12.

Si observamos la Ilustración 15: Actuador neumático garra. y la Ilustración 16: Actuador neumático operador. Observamos que su configuración deja de ser igual ya que en el módulo del operador encontramos un módulo amarillo, al igual que en los armarios eléctricos. Este módulo de referencia CPX-FVDA-P2, es un módulo de seguridad, dedicado exclusivamente a la seguridad del operador. Al igual que los módulos de seguridad explicados anteriormente en los armarios eléctricos este módulo también consta de pines para otorgarle de un código de seguridad propio que debe ser el mismo al que se haya configurado en el programa.

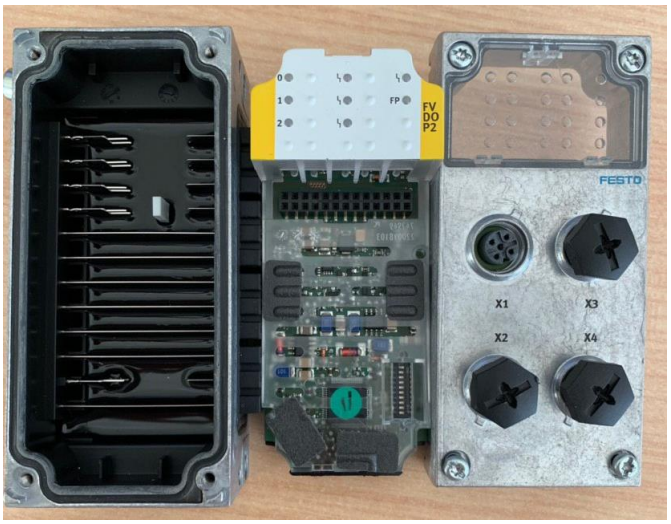


Ilustración 18: Módulo de Seguridad Festo.

Antes de hablar del porqué de este módulo es necesario explicar la conexión de 24V del conjunto y del módulo de eventos. En primer lugar, el módulo de comunicaciones CPX-(M)-FB33 también es el módulo por el que se alimenta el

conjunto, es bastante lógica esta configuración ya que para poder comunicar con un módulo es necesario que éste esté alimentado.

El conexionado de 24V se hace por medio de un conector de 7/8" que consta de 5 pines, de éstos pines 2 deben de otorgarnos 24V, 2 deben ser la masa (AT) y otro debe de hacer de tierra, tal y como se indica en la Ilustración 19: Conexionado 24V 7/8".

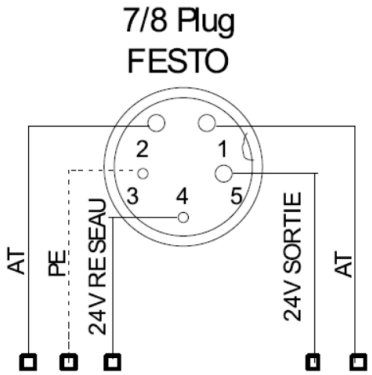


Ilustración 19: Conexionado 24V 7/8".

Por otro lado, el módulo de eventos con referencia VABA-S6-1-X1, es el encargado de permitir el paso de aire a las electroválvulas. Es decir que aunque tengamos aire, para poder actuar las electroválvulas de forma remota, éste módulo debe de estar activo. Dentro de un Festo distinguimos 24V eléctricos y 24V electrónicos, los primeros son los que alimentan las señales de entradas y el módulo de comunicación del Festo y los segundos son los que nos permiten actuar las electroválvulas, activando el módulo de eventos. Normalmente en una instalación para un montaje normal, los 24V eléctricos son constantes y los 24V electrónicos son otorgados por la zona cerrada de la instalación con un retorno por relé de modo que hasta que no se haya cerrado la zona las electroválvulas no podrán accionarse. Sin embargo, en el caso de un montaje de operador, se añade un nivel más de seguridad al añadirse el módulo CPX-FVDA-P2, la conexión eléctrica de los 24V eléctricos y electrónicos será fija, y será el propio módulo de seguridad quien permita o no el accionamiento de las electroválvulas. El control que se realiza es por medio del programa y creando una entrada electrónica del retorno eléctrico que hay dentro del propio módulo de seguridad de éste modo podemos accionar las electroválvulas de modo seguro, es decir es el propio módulo de seguridad el que alimenta las válvulas mientras que en el caso de la garra por ejemplo, es el propio módulo de comunicaciones quien hace la alimentación. Otra de las características de este módulo es que nos permite alimentar de manera externa electroválvulas u otros módulos pero siempre que el retorno de seguridad que hayamos programado esté activado. Por otro lado, al ser un módulo de seguridad, en el momento en el que se pierde comunicación con él la puesta en servicio de la instalación caerá por completo liberando el aire de la instalación.

Los últimos módulos que encontramos, son las propias electroválvulas son propiamente los elementos que otorgan acción al montaje por medio del guiado del aire en su interior, en las electroválvulas conectaremos los aprietes o los pilotos que necesitemos. Las válvulas que vamos a emplear son 5/2 de accionamiento eléctrico por lo que permanecerán activadas hasta que se dé el impulso contrario. Dentro de nuestra instalación, constamos de dos tipos de electroválvulas, unas de accionamiento interno, es decir el Festo recibe una señal para actuar dicha electroválvula, mientras que otras como podemos observar, cuentan con un cable naranja que sale del medio de la válvula, éstas son de accionamiento externo pueden ser accionadas por una carta de salidas del Festo, o por una señal externa.

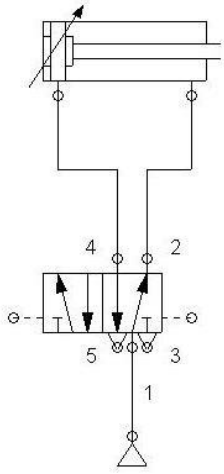


Ilustración 20: Electroválvula 5/2.

Puertas de seguridad.

Otro de los elementos de nuestra instalación son las puertas de seguridad, la marca que vamos a emplear es la alamana Euchner con referencia MGB-L0B-PNA-L-113615.

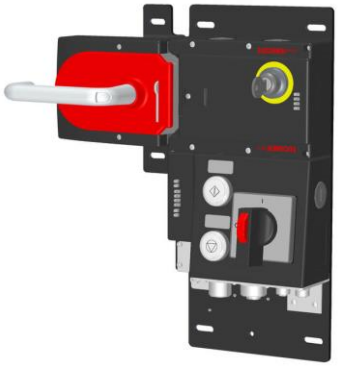


Ilustración 21. Puerta Euchner.

Este tipo de elementos son empleados para la seguridad de la instalación, no se podrá considerar una zona cerrada, y por tanto con posibilidad de poner la instalación en automático hasta que todas las puertas den su señal de cerrado. Cada puerta consta de dos módulos, uno el de comunicaciones y alimentación, que será donde conectemos nuestros 24V con un conector de 7/8" como el de los Festo, pero con la particularidad de que solo necesita 24V y una AT. Y la parte propia del picaporte, en el momento que vamos a iniciar una instalación, se deben de sincronizar picaporte y módulo de comunicaciones, para ello es necesario alimentar el módulo de comunicaciones, cerrar el picaporte y pagar el módulo de comunicaciones, de este modo conseguiremos que queden sincronizados. Al ser un elemento tan importante dentro de la instalación, también tendrá safety number, y deberemos configurar dicho número tanto físicamente como en el programa.

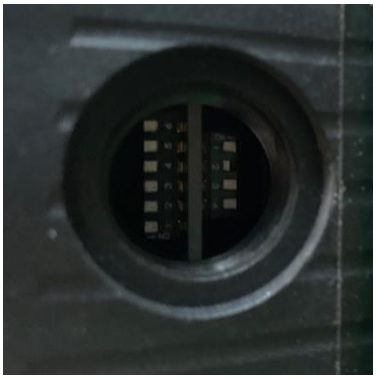


Ilustración 22: Configuración Safety Puertas.

Una de las características de éstas puertas es que tienen un bloqueo, y es posible que si al intentar configurarlas por primera vez con el picaporte dentro se pueden llegar a bloquear. Por lo que deberemos de forzar dentro del programa la variable del bloqueo para poder desbloquearla y realizar la configuración.

Robot

El robot que vamos a usar pertenece a la japonesa Fanuc, es un robot de pequeño tamaño pero muy versátil. El modelo en concreto es el M-10iA/12. Es un robot antropomórfico con una capacidad de carga de hasta 12Kg, en el caso más favorable, es por tanto un robot destinado a pequeñas tareas, pero más que suficiente para nuestra aula de formación.

Ilustración 23: Fanuc M10iA/12

Otros componentes.

Otros elementos dentro de nuestra instalación son las ET200 ECO [15] o las llamadas sistemas de periferia descentralizada, su objetivo es el tener que evitar cablear las señales de dentro de la instalación hacia el propio autómata, son sistemas que gracias a la aplicación de profinet nos permiten ahorrar grandes cantidades de cable, simplemente son elementos que deben de ser configurados en el programa otorgándoles una IP y una zona de memoria del autómata, son elementos con 8 entradas de M12 que posibilitan las gestión de 16 entradas, es decir es un elemento que en nuestro programa nos ocupará 2 bytes de memoria al igual que los modulas de entradas de Festo. Es un claro ejemplo de la ventaja y flexibilidad de los sistemas descentralizados. Al no tratarse de un elemento de seguridad no es necesario establecer una configuración de seguridad.

Ilustración 24. ET200 ECO.

Son elementos con dos entradas de profinet con conexión de m12, sin embargo, su conexionado de alimentación es distinto a lo que hemos visto hasta ahora, ya que emplean conexionado de ¾".

Mecánica de la Instalación.

En general para la realización de éste aula de formación, hemos empleado elementos disponibles en el taller, entre ello chatarra y elementos reciclados de otras instalaciones.

Manipulador del Robot

Para la realización de la garra del robot, debemos de tener en cuenta que el modelo Fanuc M10-iA no se caracteriza por su gran capacidad de carga, por lo que los elementos que usaremos serán en mayor parte aluminio, salvo para los elementos de sujeción de nuestros elementos móviles.

Para la realización de la garra nos ayudaremos de una placa de aluminio de 32cmX22cmX2cm. Ésta placa será la que insertemos directamente en el eje 6 del robot. Como elementos mecánicos disponemos de un piloto de un solo vástago de la marca Destaco. Por otro lado, disponemos de un imán neumático de la marca Goudsmit. Para la implantación de estos elementos sobre la placa de aluminio usaremos unas escuadras de hierro, ya que es de lo que disponía en el taller. Por otro lado, dentro del manipulador de la misma, colocaremos un detector inductivo de la marca Senstronic. Una de las características de este Fanuc M10-iA es que cuenta con un eje 4 hueco, por lo que no será necesario usar elementos auxiliares para pasar el cableado de nuestros elementos.

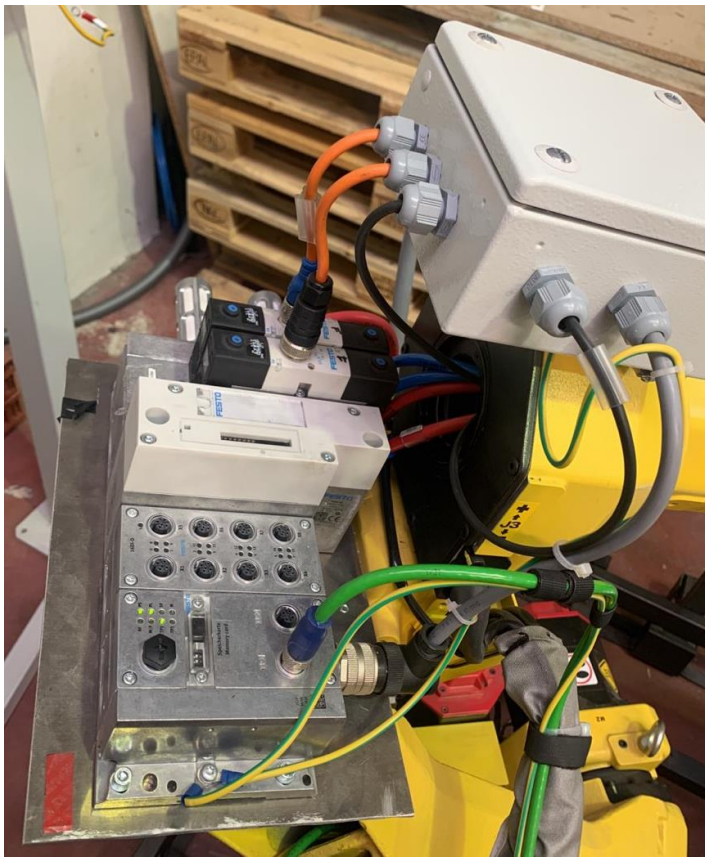


Ilustración 25: Conexionado Señales Garra.

En la intersección del eje 3 y 4, como podemos ver en la Ilustración 25: Conexionado Señales Garra. Hemos instalado el Festo que comandará nuestros elementos móviles. Nuestro módulo Festo estará formado por un módulo de comunicación, un módulo de entradas y dos electroválvulas de accionamiento externo, una para el piloto y otra para el imán neumático. Si observamos más arriba, encima del eje 4, podemos encontrar un cofre en el cual hemos cableado directamente las señales de nuestra garra, avance piloto, retroceso piloto, detector inductivo, electroválvulas, etc. directamente al robot. En el apartado de la programación Fanuc, explicaremos con más detenimiento el cableado y la configuración de éste cofre.

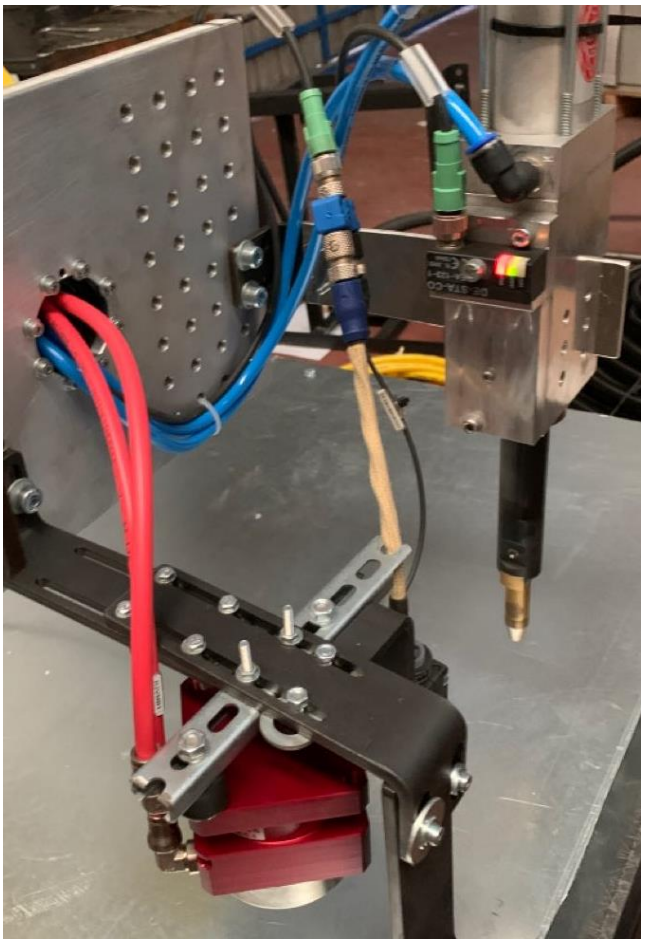


Ilustración 26: Garra Robot.

En la Ilustración 26: Garra Robot. Podemos observar el ensamblaje de la garra, los elementos de fijación negros a la placa de aluminio, tienen carriles para poder ser ajustada de maneras correcta. Por otro lado, en la parte frontal de la ventosa, hemos creado una pieza de metal para guiar a la pieza durante la cogida.

Puesto Operario

Para la realización del puesto operario, aprovecharemos la estructura de hierro en la que vino embalado el robot, y el armazón donde venía la bahía del controlador R-30iB. Anclaremos el armazón dentro de la plataforma del robot, para que así en caso de tener que mover el robot, no perdamos la repetibilidad y debamos de crear de nuevo las trayectorias. Para la realización del soporte emplearemos una canaleta de acero con la que cubrimos el hueco donde venía encajada la bahía. Una vez que tenemos la estructura de la mesa, la anclamos al soporte del robot.

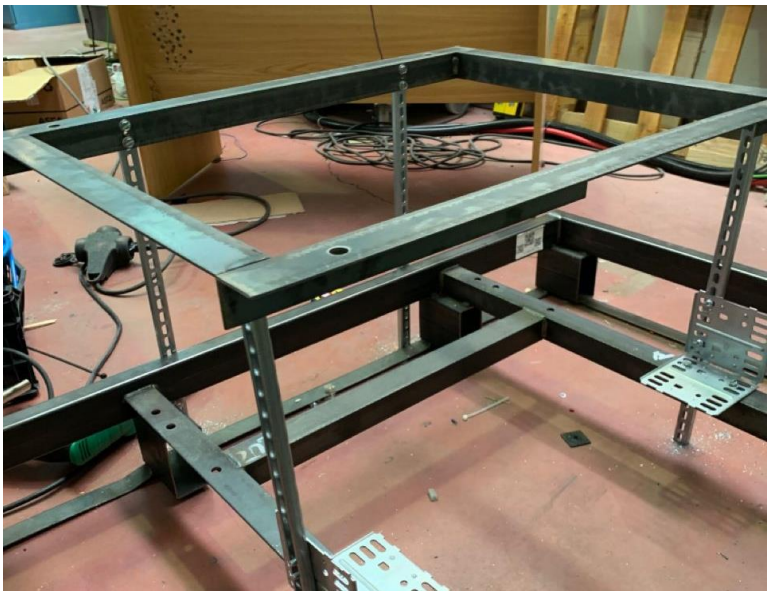


Ilustración 27: Anclaje Armazón.

Ilustración 28: Montaje Operador.

Una vez que ya tenemos el útil operador, es el momento de hacer el diseño de los elementos de la misma. Cómo elementos neumáticos, contamos con otro imán de accionamiento neumático y con un detector inductivo al igual que la garra. Nuestro módulo Festo, como hemos explicado anteriormente, constará de un módulo de seguridad, para aportar mayor protección al operario.

Como pieza a manipular por parte del operador, contamos con una canaleta de chapa de 11cmX45cm con dos perforaciones. En el montaje operador, crearemos unas guías de acero separadas 45 cm para poder encajar la pieza a lo largo, por otro

lado, dentro de las mismas guías tenemos unos cortes transversales, que nos permiten ajustar el ancho de donde reposará la pieza, de este modo, la pieza quedará encajada siempre de la misma manera permitiendo garantiza la repitibilidad de la instalación y evitar el retoque de trayectorias. Debemos tener en cuenta que al ser una canaleta, puede ser colocada de varias maneras, sin embargo el piloto embarcado de la garra se encuentra a un extremo, por lo que existe un riesgo de colocar mal la pieza y sufrir una colisión. Por ello uno de los agujeros de la canaleta encajará en un des-equivocador que hemos colocado en el mismo.

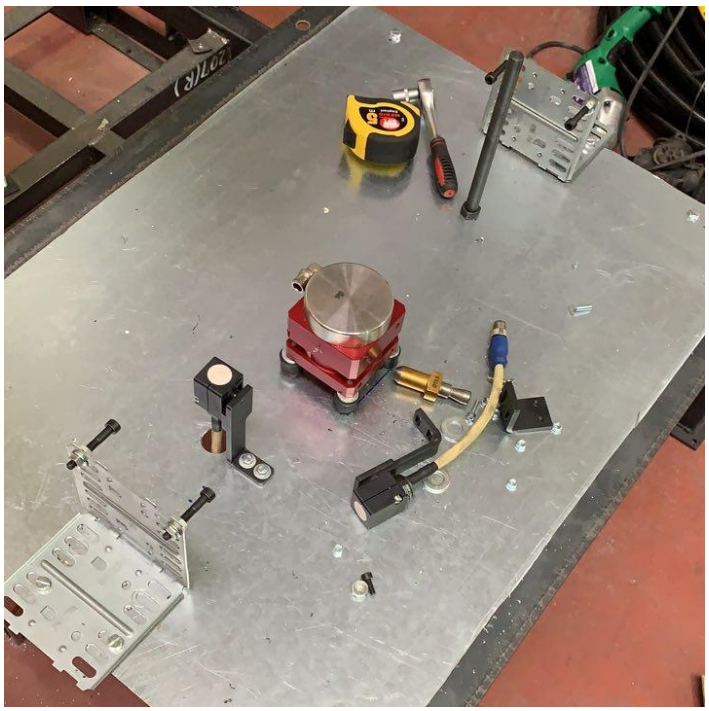


Ilustración 29: Montaje Operador.

Para añadir una seguridad para el operador respecto del robot, hemos instalado unas barreras inmateriales, es decir unas barreras que lanzan un campo láser desde un emisor hasta un receptor. Estas barreras deben de estar perfectamente alineadas, las mismas irán cableadas directamente a las tarjetas de entrada de seguridad del armario del, al igual que la seta de emergencia. Al ser de seguridad, siempre nos mandarán un bit a 1, en el momento de que alguien atraviese dicha barrera, el robot se detendrá siempre y cuando no esté en una zona de seguridad.

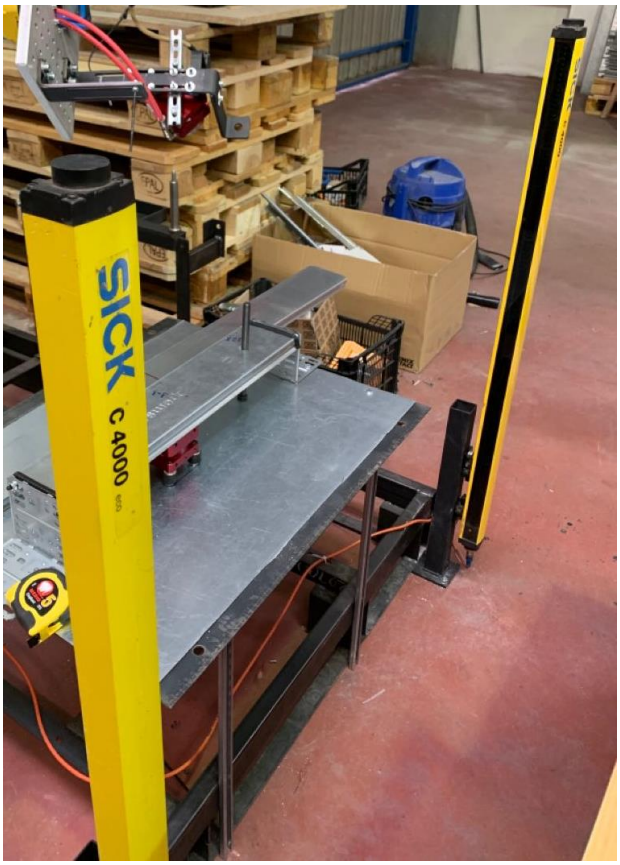


Ilustración 30: Barreras de Seguridad SICK.

Puesto de descarga.

Para aportar más dinamismo a la instalación, he diseñado un nuevo puesto para dejar la pieza. De este modo, mientras el robot se retira con la pieza para ir a dejar a este puesto, el operario puede cargar otra pieza en condiciones de seguridad. Para la elaboración de este puesto empleé un soporte del propio robot de su transporte. Añadí una varilla roscada y un tramo de canaleta para que cuando el robot dejara la pieza ésta no se pudiera desliza. Por otro lado, al ser un montaje, he añadido un detector para confirmar que la pieza ha sido dejada de la manera correcta. Dicho detector va cableado a nuestra ET200 ECO.

Ilustración 31: Puesto de Salida y ET200 ECO.

Programación en Step 7

Como hemos dicho anteriormente, el PLC es un ordenador industrial, sin embargo, es necesario para su programador un PC convencional y un programa específico para ello. En nuestro caso ya que usamos un PLC de la casa Siemens, el programa que vamos a emplear es Step7. Sin embargo, debido a que los ordenadores que vamos a usar corren Windows 10, y el programa para su correcto funcionamiento y la licencia usaremos una máquina virtual con Windows 7.

La máquina virtual que vamos a usar en concreto es VMware, en la cual crearemos una imagen de un ordenador con Windows 7 y el programa Step 7 ya instalado. Sin embargo al emplear una máquina virtual vamos a sufrir una serie de complicaciones en lo que a la comunicación con el PLC se refiere, los PLCs es posible que tengan asignada una IP de serie que debe ser conocida por el programador si no se conoce la IP del propio PLC lo mejor normalmente es formatear el dispositivo para que se asigne la IP de fábrica 0.0.0.0.

Dentro de la propia máquina virtual debemos elegir el tipo de conexión de red que queremos. A grandes rasgos hay dos modos de conexión el bridge y el NAT. El modo de conexión bridge es el empleado en las redes locales ya que duplica el estado físico de la configuración, de este modo se puede asociar una IP automática o una IP que nosotros deseemos concretamente. El otro modo disponible es el modo NAT con el no somos capaces de asignar la IP que deseamos concretamente.

En nuestro caso, como queremos asignar una IP al PC similar a la IP del PLC, debemos de seleccionar el modo Bridge.

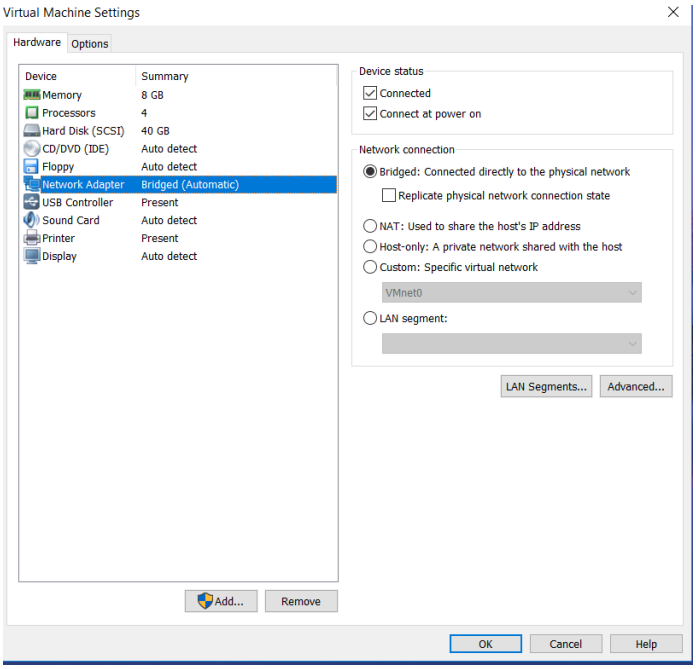


Ilustración 32: Configuración VMware.

La IP que tiene nuestro PLC, puesto que es un PLC ya usado en otros proyectos es 172.25.214.2, por lo que debemos de asignar una IP similar al PC. Para ello abriremos el menú de *redes y recursos compartidos* de nuestro ordenador, *conexiones de red local*, *propiedades* y seleccionaremos *protocolo de internet versión 4 (TCP/IPv4)*. Este es el protocolo que se emplea normalmente en todas las casas, está basado en la capa de red de OSI y nos proporciona una manera de identificar de manera única todos los componentes de la red. Por otro lado, la máscara que se empleará será la 255.255.252.0.

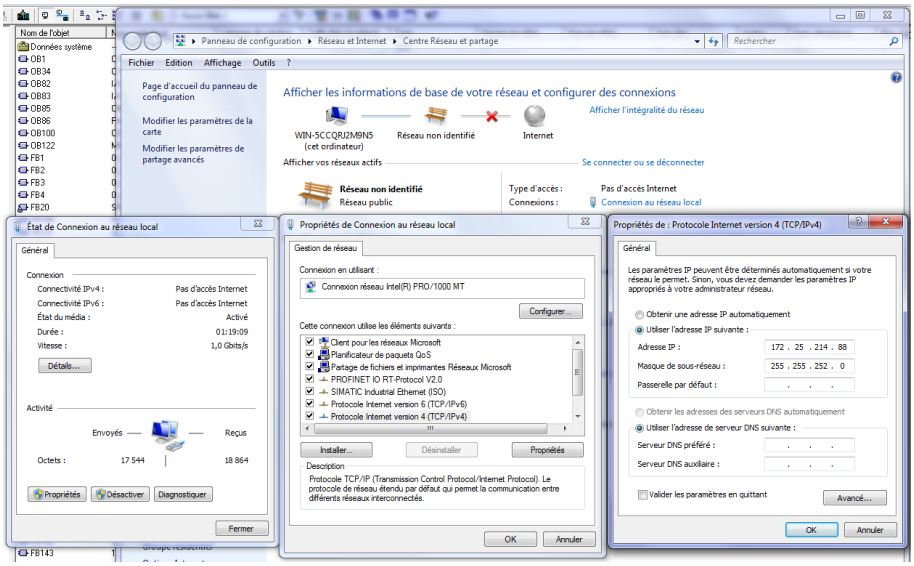


Ilustración 33: Configuración IP.

Cuando creamos un proyecto nuevo en Step7 nos encontramos con una hoja en blanco, así pues en el menú *Insertiom/Station* deberemos de seleccionar el tipo de Instalación que vamos a crear, en nuestro caso como nuestra CPU pertenece a la familia 315F deberemos de seleccionar la *Station Simatic 300*.

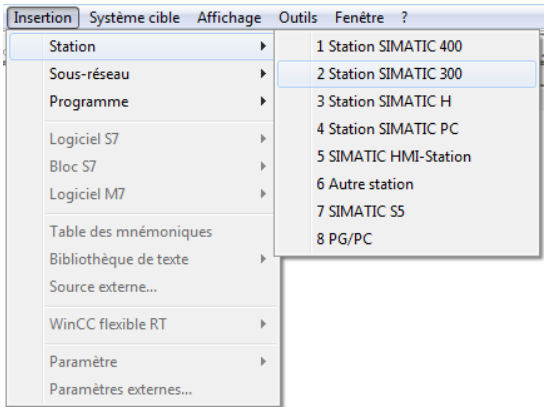


Ilustración 34: Creación de la Estación.

Una vez que ya hemos configurado nuestra red en la máquina virtual y hemos creado la estación, procederemos a establecer comunicación entre el PC y el PLC, para ello dentro del programa Step7 que es el propio de la casa Siemens, debemos de seleccionar que tipo de configuración queremos.

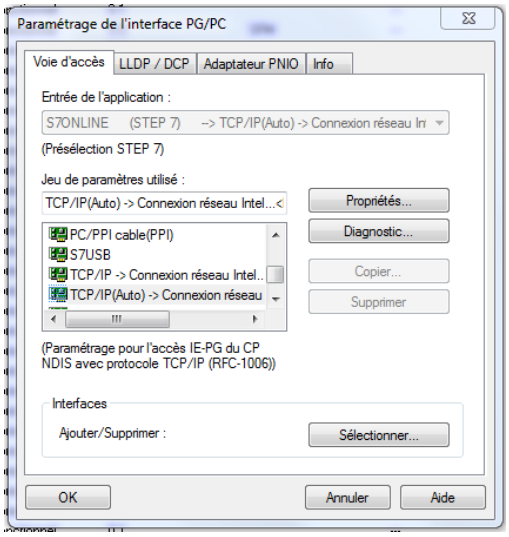


Ilustración 35:Configuración PC/PG.

Como hemos dicho anteriormente, vamos a emplear el protocolo de comunicaciones TCP/IP, por lo que dentro de nuestro programa deberemos de seleccionar la misma configuración para la correcta comunicación con el autómata.

Dentro del programa de Step7 se distinguen dos partes, la primera la configuración material en la que seleccionaremos los componentes que tenemos en nuestra instalación real, y la parte propia de programación. Una vez que la configuración de la máquina virtual ha sido realiza y del programa, procederemos a la configuración de material de la instalación.

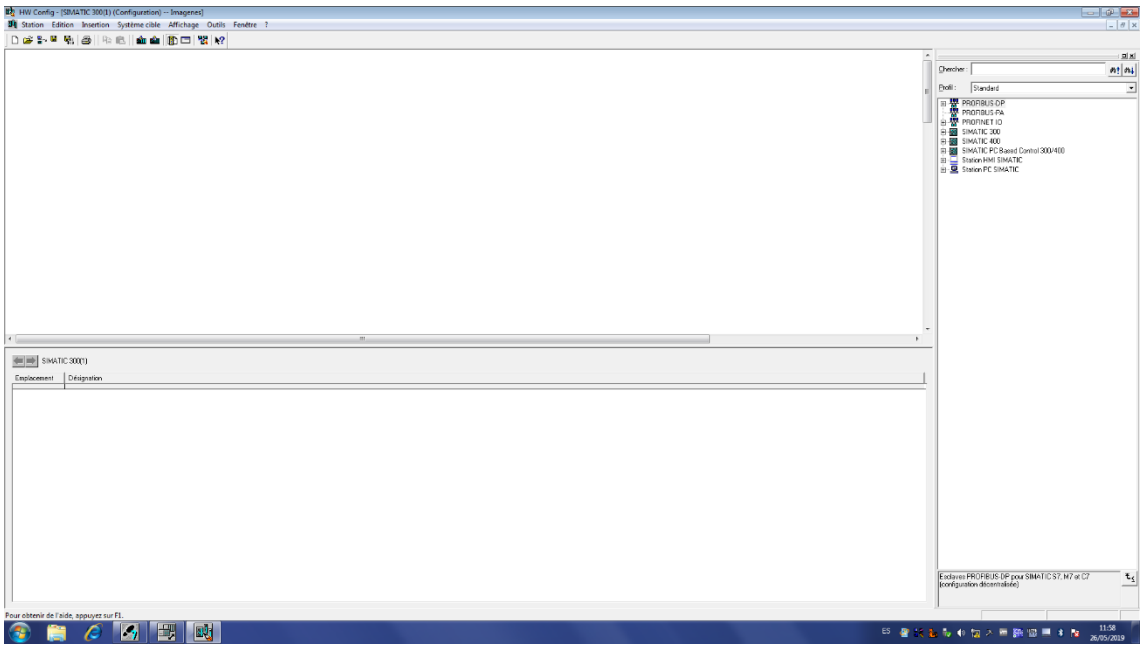


Ilustración 36: Configuración material.

Si observamos en la Ilustración 36: Configuración material. a la derecha hay un buscador, en él debemos de buscar los elementos de nuestra instalación por el número de referencia. En el momento en el que empezamos a realizar una configuración material lo primero que debemos de hacer es añadir el bastidor donde insertaremos nuestra CPU.

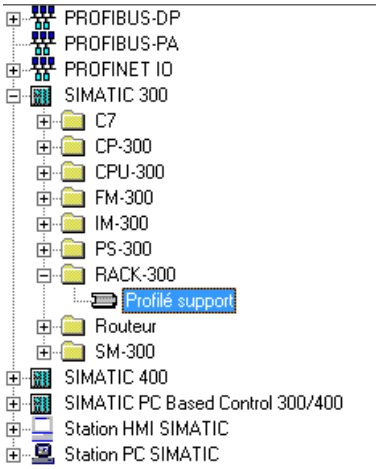


Ilustración 37: Inserción del bastidor.

En el momento que tenemos seleccionado el bastidor podemos añadir a nuestra configuración material la CPU que vamos a emplear, para ello debemos de buscar dentro del menú Simatic 300, CPU-300 y seleccionar nuestra CPU sin embargo un detalle a tener en cuenta es que dentro de cada CPU hay distintas versiones por lo que deberemos de seleccionar la versión concreta de CPU que tenemos, dicha versión viene señalada normalmente al lado del número de referencia.

En el momento que añadimos la CPU al bastidor, un menú aparecerá en el que nos solicitan que insertemos la dirección IP y la máscara de subred de la CPU, dicha IP y sub máscara tienen que estar en correlación las que hemos establecido en nuestro PC, por lo que la CPU será 172.25.214.2 y las submáscara de red 255.255.252.0.

Una vez que ya tenemos nuestra CPU en su bastidor, debemos de crear una red de profinet en nuestra configuración material, ya que será donde se Inter conexionen el resto de elementos de la instalación. Para ello con el botón derecho del ratón en la parte de la CPU dedicada a la conexión de entradas y salidas seleccionaremos la inserción de una red de profinet.

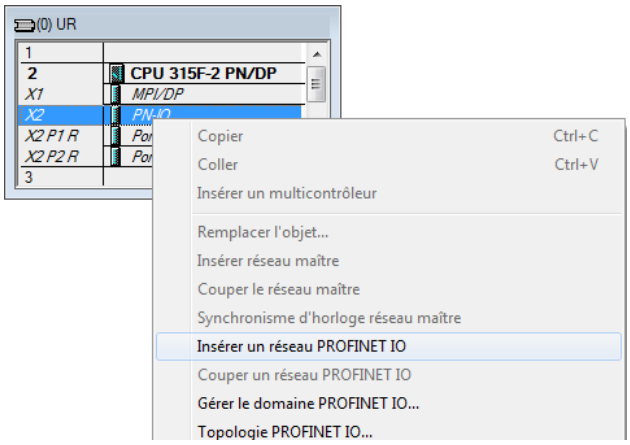


Ilustración 38: Red Profinet.

Una vez que hemos añadido la CPU y creado la red de Profinet, realizaremos una compilación de la configuración material, de este modo el programa Step7 de manera automática nos creará una tabla de nemónicos vacía, así como el OB1 que explicaremos más adelante. Continuamos añadiendo los elementos de nuestra instalación comenzando por el armario de zona ZB, para ello debemos tener en cuenta que se deben de colocar los bloques en el mismo orden que en la realidad ya que si no tendremos una configuración incoherente y la instalación nunca funcionará. En el caso de que no encontremos la referencia que queremos, es necesario que actualicemos nuestra librería o que nos pongamos en contacto con el fabricante del elemento que deseamos incorporar para que nos proporcione el bloque de Step7.

Cada vez que insertamos un elemento, si clicamos en él podemos asignar la IP y el nombre que deseamos. Es conveniente tener un cierto orden con las IPs que se asignan ya que debido al protocolo empleado no es posible duplicar la IP de ningún elemento. El número de aparato para evitar complicaciones, es recomendable que sea el mismo que el último número de la IP.

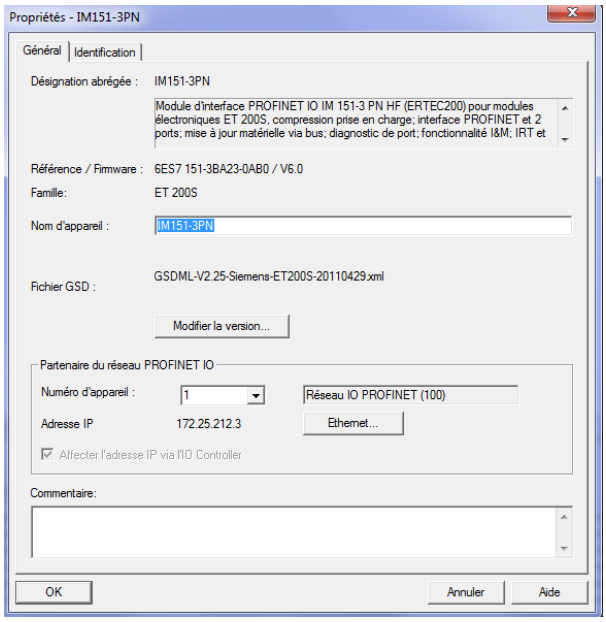


Ilustración 39: Asignación nombre e IP ET200s

Una vez que hemos añadido nuestra ET200s del ZB, debemos añadir el bloque perteneciente a las tarjetas de alimentación de 24V ya que al ser unas tarjetas físicas de siemens deben de ser contempladas en la configuración. Posteriormente añadiremos el resto de tarjetas de acuerdo a como están instaladas en nuestra configuración material y siguiendo la referencia ya que, pese a existir varias tarjetas que hacen lo mismo si la referencia no corresponde tendremos de nuevo incoherencias. En el momento que comenzamos a añadir las tarjetas propias de entrada y salida, podremos asignar direcciones de entrada y salida de las que luego crearemos nuestros nemónicos para el programa.

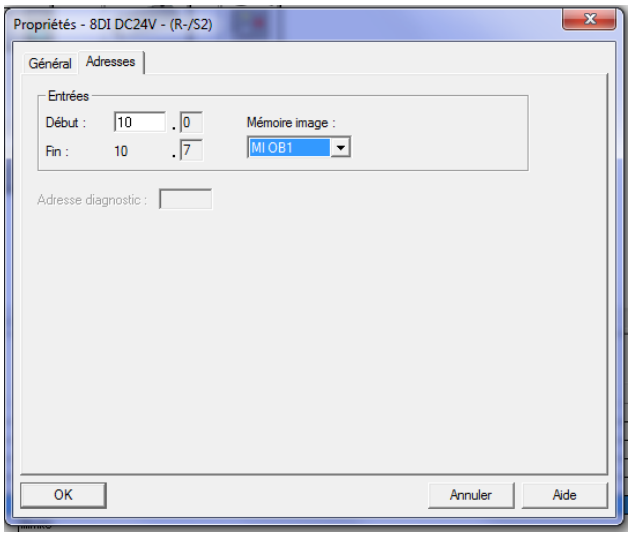


Ilustración 40: Asignación direcciones tarjeta de entradas.

Continuaremos añadiendo nuestros componentes de modo que finalmente nuestra configuración material quede de esta manera:

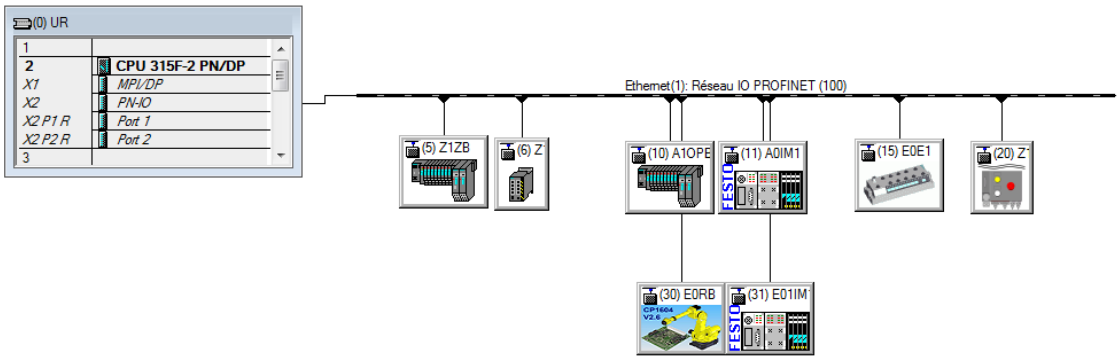


Ilustración 41: Configuración Material Instalación.

Antes de realizar una compilación de nuestra configuración material, es necesario configurar nuestra CPU. Dentro de las múltiples opciones que nos ofrece Step7 las que nos interesa son las referentes a la protección de la CPU. Debido a que vamos a usar el estándar Profisafe debemos de asignar que la CPU va a tener un programa de seguridad por otro lado, también añadiremos una contraseña a todo el programa para añadir un nivel más de seguridad.

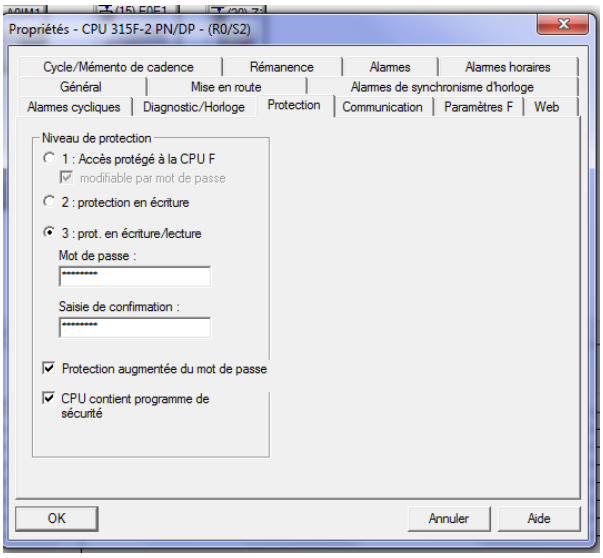


Ilustración 42: Protección CPU.

Antes de comenzar más a fondo con la programación del autómata y la generación del programa de seguridad, debemos hablar del lenguaje de programación de Step7.

En primer lugar, vamos a hablar de la estructura del programa. Si observamos la Ilustración 43: Estructura de un programa Step7. Podemos ver el esquema conceptual que forma este tipo de programación.

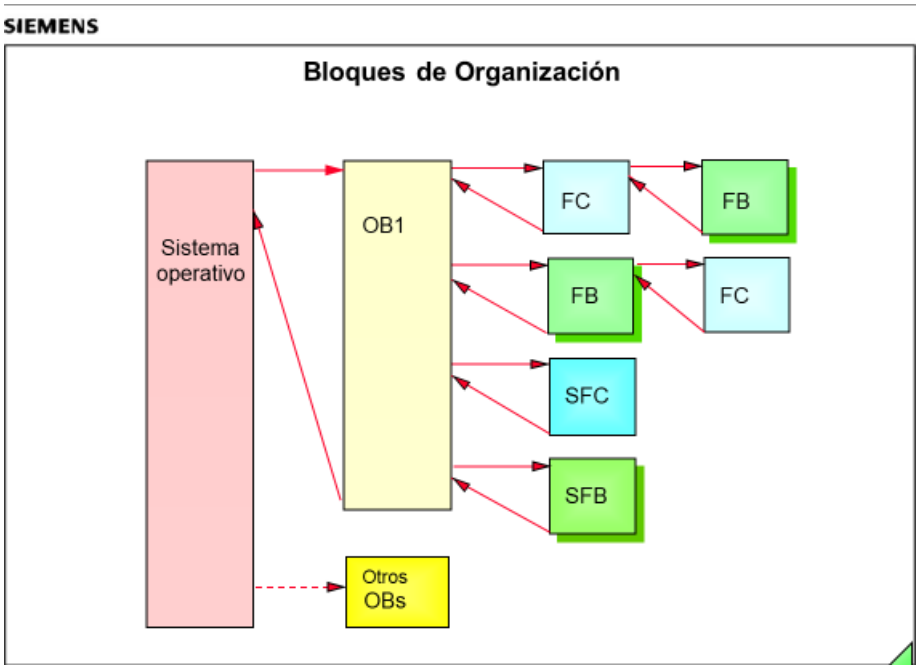


Ilustración 43: Estructura de un programa Step7.

Dentro de la CPU, como vemos en el esquema se ejecuta el sistema operativo, y una serie de OBs, que no son más que el propio programa de usuario.

El SO tiene como objetivo organizar las distintas funciones y los procesos de la CPU ajenas a las propias de las tareas de control, a grandes rasgos las funciones de la CPU son:

- Gestionar el arranque en caliente y normal de la CPU, el arranque en caliente es el momento en el que la CPU pasa a stop teniendo el selector de la misma en Run con este modo de arranque podemos forzar el rearranque sin tener que tocar el selector. Un arranque normal sería que nuestro selector se encuentra en stop y lo pasamos a run.
- Hace de pasarela entre las entradas y las salidas del sistema, actualizándolas a cada ciclo de reloj. Así como de elemento de comunicación con el resto de la instalación.
- Una de las funciones del SO es la del trato de errores y detección, el SO hace de wacthdog para la detección de alarmas y posibles problemas.

El programa de usuario, que es lo que más nos incumbe en el desarrollo de este proyecto es la base de un sistema automatizado, ya que en él programaremos las distintas situaciones de la instalación y las distintas repuestas del sistema. Step 7 se basa en una programación por bloques, lo que nos permite hacer un programa estructurado de modo que si es para una gran instalación, dentro de lo que cabe será entendible.

Dentro de un programa de Step 7 encontramos los siguientes bloques;

Bloques de organización (OB). Este tipo de bloque tiene la estructura de programa, son el nexo entre el SO y el programa de usuario. El propio SO es el que llama a este tipo de bloques ya que controlan de forma cíclica las alarmas del programa, el comportamiento de la instalación y nos permite realizar el tratamiento de los errores. Los bloques de organización son los encargados de decidir qué partes del programa se ejecutan y cuáles no. Un bloque de organización puede verse interrumpido por la llamada de otro bloque, esto dependerá de la prioridad que tenga un sobre el otro. Cada OB tiene destinada una función por ejemplo; los OB del 10 al 17 son los bloques destinados a las alarmas horarias, del OB 30 al 38 son los destinados a las alarmas cíclicas, son capaces de interrumpir la ejecución del programa, el más importante para nosotros es el OB34, ya que será en el que ejecutemos la rutina de seguridad, el más importante de todos es el OB1 donde se realiza la ejecución del programa este bloque es llamado de forma continua por el sistema operativo en este boque se ejecuta todo el programa, dentro de él se hará la llamada a todas las funciones que tengamos programadas, una vez que ha sido ejecutado vuelve a ser ejecutado de nuevo el tiempo máximo de ejecución de este bloque es de 150ms que será lo que denominaremos scan. El objetivo de que éste OB se ejecute de forma continua es la de poder actualizar constantemente el estado del sistema y actuar en consecuencia. La ejecución de este OB comienza en el momento en el que la CPU se encuentra con el selector en la posición de run. Ésta ejecución cíclica e puede ver interrumpida los otros OBs ya sea por una alarma, por una orden de stop o porque haya habido un corte de alimentación.

Bloques de función del sistema (SFBs) y funciones de sistema (SFCs). Son funciones que están dentro del propio sistema, a efectos prácticos de programación es una función que no es necesario de ser cargada. Son a groso modo atajos de programación. Son funciones que no constan de memoria, en el mismo ciclo de scan pasan sus variables por cero.

Bloques de función (FBs). Son la parte programable de step7 en este tipo de bloques el usuario puede crear su programa son bloques con memoria por lo que no perderemos los datos en cada ciclo de scan. Como hemos dicho son bloques con memoria, cada vez que se crea un FB se asigna un espacio de memoria en el que se almacenarán los datos de la función, a este bloque de datos se le denomina de instancia ya que está vinculado a la FB. El programa creado en una FB solamente se ejecutará cuando sea llamado por otro bloque lógico.

Funciones (FC). Las funciones son estructuras programables al igual que los FBs, con la diferencia de que no tienen memoria sus datos temporales se almacenan en una pila de datos locales que luego se borra. En el caso de que necesitemos mantener una memoria podemos usar memorias de un bloque de datos global. Al igual que las FBs las variables que usemos deben de ser inicializadas a un valor aunque no se vaya a guardar su valor. Este tipo de funciones se ejecuta cuando es llamada por otro bloque lógico.

Bloques de datos de instancia (DBs de instancia). Cada vez que una FB es llamada a ejecutar se transfieren los datos en ese momento a su DB de instancia asociado. Antes de crear un DB de instancia es necesario de haber creado un FB

para poder referenciarse. Pueden crearse varios DBs y ser asociados a un FB esto se denomina multi-instancia, pero en nuestro caso no será aplicable.

Bloques de datos (DB). Este tipo de datos nos permite almacenar el valor de las variables del programa, a diferencia de un bloque de instancia las variables almacenadas pueden ser usadas en todo el programa. El tamaño máximo de los DBs depende del tipo de CPU

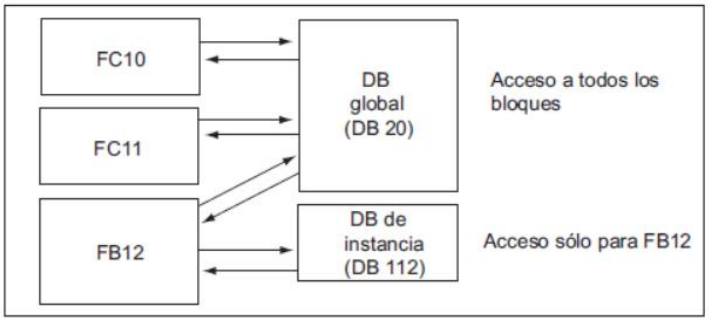


Ilustración 44: Diferencias DB global y DB de instancia.

A continuación dejo una tabla con los tipos de datos que usaremos a la hora de realizar nuestro programa:

Tipo y descripción	Tamaño en Bits	Formato-Opciones	Rango y notación numérica (Valores máximo y mínimo)	Ejemplo
BOOL (Bit)	1	Texto Booleano	TRUE/FALSE	TRUE
BYTE (Byte)	8	Número Hexadecimal	B#16#0 a B#16#FF	B#16#10
WORD (Palabra)	16	Número Binario	2#0 a 2#1111_1111_1111_1111	2#0001_0000_0000_0000
		Número Hexadecimal	W#16#0 a W#16#FFFF	W#16#1000
		BCD	C#0 a C#999	C#998
		Número Decimal sin signo	B#(0,0) a B#(255,255)	B#(10,20)
DWORD (Doble Palabra)	32	Número Binario	2#0 a 2#1111_1111_1111_1111_1111_1111_1111_1111	2#1000_0001_0001_1000_1011_1011_0111_1111
		Número Hexadecimal	DW#16#0000_0000 a DW#16#FFFF_FFFF	DW#16#00A2_1234
		Número Decimal sin signo	B#(0,0,0,0) a B#(255,255,255,255)	B#(1,14,100,120)
INT (Entero)	16	Número Decimal con signo	-32768 a 32767	1

Ilustración 45: Tipos de datos en step7.

Dentro de Step7, podemos elegir 3 lenguajes de programación:

- KOP, que es el esquema de contactos al que estamos acostumbrados, es lo que denominamos de forma más corriente lenguaje Ladder. Al programar con este lenguaje podemos ver la circulación de la corriente a través de los contactos de modo que es mucho más visual que el resto.
- AWL o lista de instrucciones es un lenguaje en base a texto orientado al lenguaje máquina, es mucho más complejo de comprender, ya que no es visual y requiere un grado de especialización alto. Por ello para facilitar su

programación se han creado estructuras de programa de lenguaje de alto nivel.

- FUP, es un lenguaje de programación gráfico, emplea diagrama de funciones usando álgebra booleana para representar la lógica.

Una vez que ya tenemos unas nociones básicas de Step7, podemos comenzar a realizar la programación en sí. Nuestra zona de programación una vez hemos insertado nuestra Estación Simatic 300 es la que podemos ver en la Ilustración 46: Zona de trabajo Step 7.

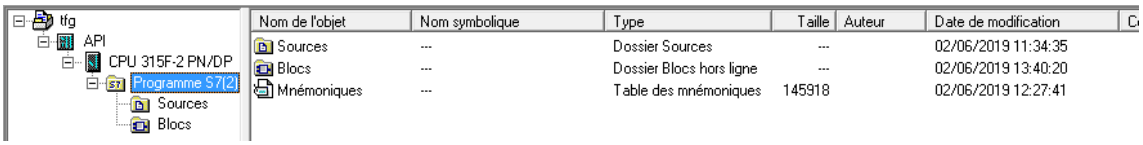


Ilustración 46: Zona de trabajo Step 7.

En la carpeta Sources podemos cargar programas ya escritos que pueden ser generados por macros de Excel, dichos programas se generan en lenguaje AWL. Al crear dichos programas por medio de una macro no es posible modificar el código directamente AWL si no que es necesario modificar la macro y volver a generarla para que la modificación sea efectiva.

Dentro de la carpeta bloques, será donde creemos nuestras distintas funciones, bloques de organización etc.... Ahí tendremos acceso a todo el programa y será desde donde trabajemos.

El apartado Mnémoniques será donde creemos las variables de nuestro programa, es decir donde declararemos las entradas, las salidas y las memorias. A la hora de crear nemónicos se puede hacer de dos maneras; desde la configuración material editando elemento a elemento con su correspondiente dirección o desde la tabla de nemónicos dando nombre y dirección a la variable, la tabla de nemónicos es más visual y nos posibilita la inserción de memorias, sin embargo al crear las entradas y las salidas, debemos de tener claro a que módulo corresponde cada uno, ya que es posible declarar variables en direcciones que no se usan en nuestra configuración material.

Si nos metemos dentro de la carpeta de bloques, veremos que solamente tenemos generado el OB1, el bloque de organización principal que habíamos hablado en su momento. Si clicamos dentro en el OB1, como es el inicio del proyecto, debemos de seleccionar que tipo de lenguaje y que tipo de programa va a ser, en nuestro caso tipo KOP, pero debido a que nuestra máquina virtual es francesa, el nombre que aparecerá es CONT. También como hemos seleccionado en nuestra CPU que contiene un programa de seguridad, se nos generarán de forma automática al realizar la compilación de la configuración material distintos FB y DB de seguridad. Sabemos que son de seguridad por que aparecen sombreados de color amarillo,

también son bloques que no podemos modificar ya que vienen con un candado instalado. Posteriormente cuando volvamos a compilar el programa con nuestros elementos colocados en la configuración material, debido a que muchos de ellos tienen elementos de seguridad, se generará de forma automática una serie de DBs con datos de seguridad que podemos leer pero nunca podemos modificarlos, salvo que modifiquemos la configuración material y volvamos ca compilar.

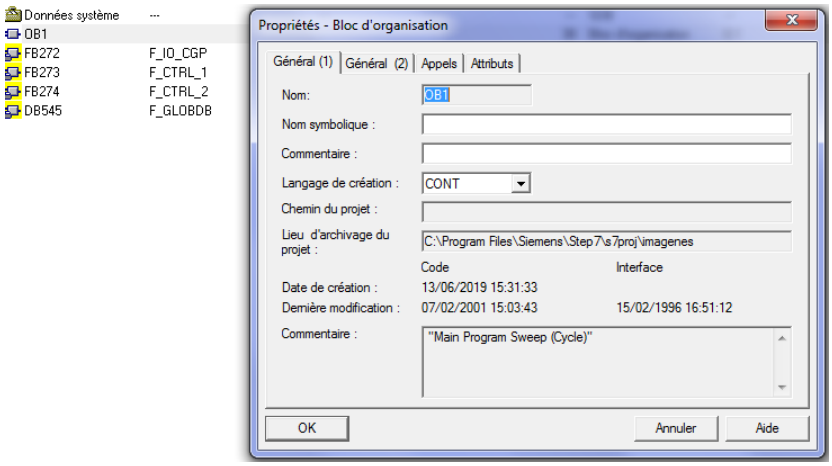


Ilustración 47: Selección de formato OB1.

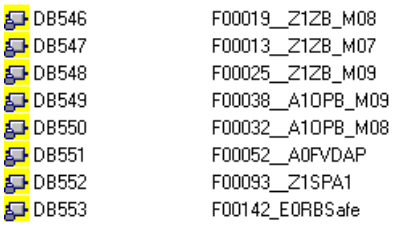


Ilustración 48: DBs de seguridad generados por los elementos de Safety.

Antes de continuar programando, vamos a generar un programa de seguridad, en él se establecerán todas las condiciones referentes a la seguridad de la instalación sin alguna de ellas la instalación nunca podrá funcionar de modo automático. Dentro de los modos de una instalación distinguimos 6:

- Parada de emergencia, en el contemplamos todas las paradas de emergencia de la instalación, en el momento en el que tenemos al menos una parada de emergencia accionada, la instalación entrará en stop es el más importante de todos, sin este modo en funcionamiento el resto de modos jamás podrá darse.
- Puesta en servicio, hablamos de puesta en servicio cuando tenemos una instalación en la cual tenemos una entrada de aire activada, es decir que

tenemos potencia para poder hacer funcionar nuestros distintos mecanismos.

- Zona cerrada, anteriormente hemos hablado de los puertas Euchner, dichas puertas envían una señal de cierre al autómata, y éste lo gestiona para declarar una instalación completamente cerrada. En el momento que tenemos una instalación cerrada todos los elementos están completamente funcionales y a la espera del modo automático. Como comentamos anteriormente en la conexión de 7/8 de los módulos Festo, en el momento que tenemos la zona cerrada habilitamos los 24V electrónicos dejando los Festos operativos.
- Modo manual, para tener un modo manual operativo nos basta con la zona cerrada, sin embargo un modo manual operativo también puede ser con los robots en manual o automático, en éste modo podemos operar los módulos Festo de modo remoto desde un pupitre específico para ello, nosotros en nuestro proyecto no constaremos de él.
- Modo automático, en este modo debemos tener los tres primeros y además todos los robots con sus bahías en automático, de esta manera la instalación estará lista para funcionar.

Para la generación del programa de seguridad, debemos generar en primer lugar un OB de alarma, que será el que haga de watchdog de nuestro programa y será donde ejecutemos toda la rutina de seguridad. En nuestro caso emplearemos el OB 34 como bloque de organización de seguridad, para ello debemos crearlo en nuestra carpeta de bloques, daremos a botón derecho *insertar nuevo objeto*, *bloque de organización.*

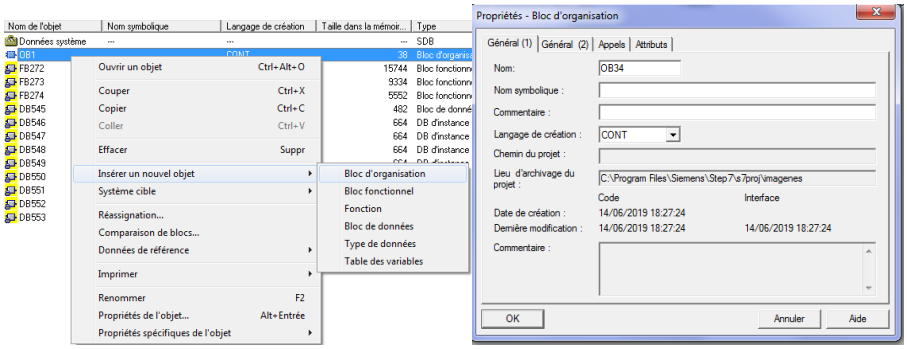


Ilustración 49: Creación del OB34.

Un menú aparecerá en el que debemos de seleccionar el nombre del OB y el tipo de lenguaje que vamos a usar. En el momento que lo generemos, pese a no agregar ningún nombre simbólico, el propio programa lo detecta como un OB de alarma. Dentro del OB veremos como se ha generado automáticamente el comentario de interrupción cíclica.

Una vez que hemos creado nuestro OB para hacer las llamadas a las rutinas de seguridad, es necesario crear las funciones y los bloques de función que albergarán nuestro programa de seguridad. Dentro de Step 7, encontraremos un botón amarillo que si ponemos el cursor encima pone *Editor de programa de seguridad.*

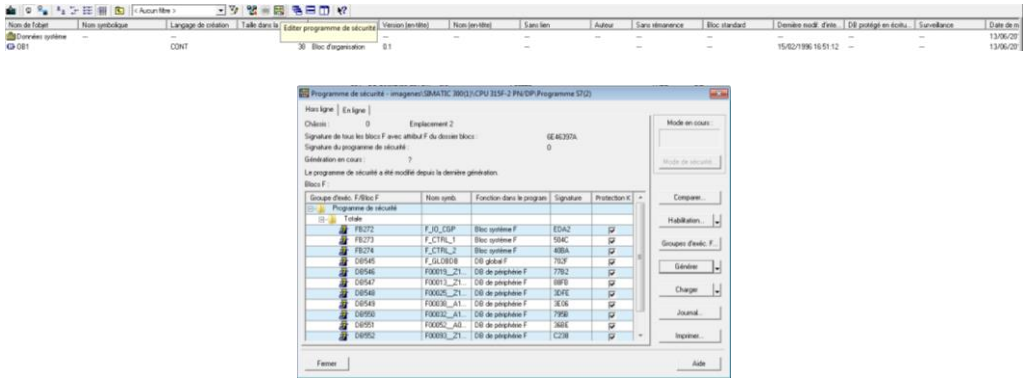


Ilustración 50: Editor de seguridad.

Si clicamos dentro de *Grupos de ejecución F*, que es la denominación que se da a los grupos de Safety, veremos que está vacío ya que aún no hemos seleccionado cuál será nuestro grupo de seguridad. Antes de continuar, debemos pensar nuestra estructura de seguridad, y cómo ha de ser. En primer lugar debemos de generar un FC al que llamaremos a todas las FBs de seguridad que vayamos a crear. En nuestro caso lo denominaremos FC1101, en él tomaremos todas las variables referentes a las cartas de seguridad. Por otro lado, como FBs en nuestro caso tenemos que diferenciar entre seguridad de la configuración materia que es la referente a las cartas de seguridad instaladas y su correcto funcionamiento, la seguridad de la zona, que es la referente a toda la instalación y la seguridad del operador, ya que al fin y al cabo es el que tiene más riesgo de sufrir un accidente por parte de la instalación y por ello crearemos un bloque de seguridad específico para él.

En primer lugar, debemos de crear el FC1101, para ello procederemos de la misma manera que para crear nuestro OB34, sin embargo, a la hora de elegir el lenguaje de programación, para que el propio programa lo detecte como una función de seguridad seleccionaremos CONT F.

Volvemos a clicar en nuestro *editor de seguridad* y dentro de *grupos de ejecución F* seleccionaremos nuevo, como vemos en *bloque de programa F* auto selecciona la función que acabamos de crear de forma automática. Dentro de *bloque de llamada F* que seria el bloque de llamada de safety, debemos de escribir la FC que deseemos crear, debe de ser una que no exista ya, dicha FC aparecerá en nuestra lista de bloques con un candado, ya que no es posible editarla. Si nos damos cuenta en la Ilustración 50: Editor de seguridad. vemos una firma de seguridad del conjunto de seguridasd, es un código alfa numérico y es único, para saber si un programa tiene un programa de seguridad perfectamente compilado y correcto o si

ha habido algún tipo de moficicación debe de coincidir con el numero de firma del programa. Una vez que hemos generado nuestra rutina de seguridad, empezaremos a realizar la programación de la misma.

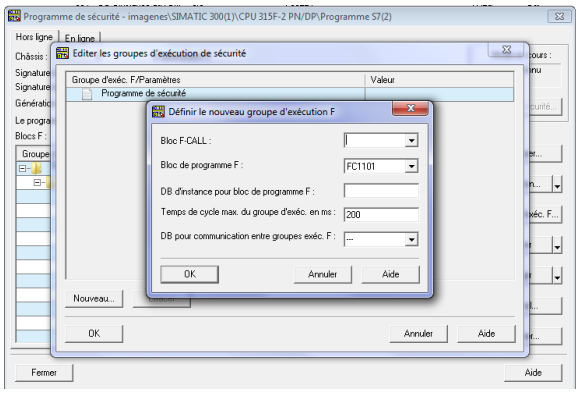


Ilustración 51: Generación de grupos de ejecución F.

Debemos crear los FBs antes nombrados, empezaremos por el FB de seguridad de la configuración material. A éste FB lo llamaremos FB1100 (cualquier otro nombre valdría) si recordamos, cuando compilamos la configuración material por primera vez, se nos crearon una serie de DBs de seguridad con candado, éstos son los propios de las cartas de seguridad son denominados DBs de periferia. Estos DBs nos indican si hay algún tipo de fallo dentro de dicha carta, en el caso de que existiera algún defecto en alguna de ellas no sería responsable dejar a la instalación funcionar, ya que podría suceder que una parada de emergencia por defecto de dichas cartas estuviera inhabilitada. La creación de éste FB es igual que la del OB 34, clicando simplemente en *bloque de funcionamiento* al igual que todo programa de seguridad debemos de seleccionar lenguaje CONT F.

Una vez que hemos creado nuestros FBs debemos crear los DBs de instancia asociados a cada una de ellas, para ello dentro de nuestra función de seguridad FC1101 haremos las llamadas a cada uno de los FBs, y en el momento de realizar la llamada se generará automáticamente un DB de instancia.

Ilustración 52: Llamada de FB y creación DB seguridad.

Hemos creado nuestros DBs de instancia de seguridad, asignado a una FB específica, pero también crearemos un DB global de seguridad, en el agruparemos las variables de seguridad que necesitaremos para poder activar los distintos elementos de la instalación.

Volviendo a la programación de la seguridad de la configuración material, debemos hablar de nuevo de los DBs de seguridad asociados, ya que usaremos una variable en concreto para generar una salida global de buen funcionamiento.

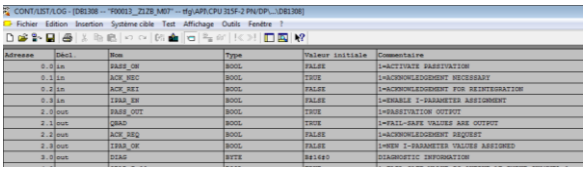


Ilustración 53: DB de periferia F.

Dentro de las variables de periferia F encontramos *PASS_ON* es una variable normalmente a 0, que permite la pasivación es decir la desactivación de la tarjeta de seguridad con independencia del programa de seguridad, es decir si encuentra dentro de su configuración algo incoherente pasaría a 1. Ésta podría ser nuestra variable de seguridad, sin embargo nos regiremos por la regla de que una variable, o una entrada de seguridad siempre debe de estar a uno, así en caso de rotura de hilo o mal funcionamiento del programa no podremos recibir señales y por tanto la instalación estará bloqueada. Por ello dentro de los DBs de periferia, usaremos la variable QBAD que siempre está a 1 salvo en el caso de que se haya pasivado un canal que pasa a valer 0. Para tomar esa variable, debemos de tomar el nombre de nuestro DB y poner el nombre de la variable en concreto de modo que si seguimos la Ilustración 53: DB de periferia F. nuestra variable será *"F00013_Z1ZB_M07".QBAD*, cabe mencionar que existe un DB de periferia F por cada carta de seguridad.

Una vez que ya hemos creado nuestra FBs de seguridad de las cartas, continuaremos creando nuestra FB de seguridad de la zona. En ella contemplaremos todos los elementos de seguridad referentes a la zona, cabe decir que hay elementos que constan de FB específicas que podemos encontrar en internet para el control de la seguridad de las mismas, nosotros usaremos algunas de ellas, como por ejemplo para los puertas y para el robot.

Antes de comenzar a definir nuestro programa de seguridad de zona, debemos explicar la FB1710. Dentro de nuestros armarios ya sea el ZB o el OPB, tenemos una serie de relés dichos relés actúan por una entrada de 24V y nos retornan otra salida de 24V. En la programación que estamos empleando, utilizamos dicho retorno para activar nuestros elementos y una de las salidas del relé como entrada de nuestro sistema para comprobar el correcto funcionamiento del relé. Aquí entre en juego el FB1710 [16].

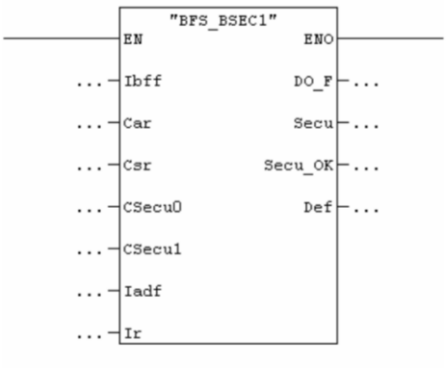


Ilustración 54: FB1710.

Como entradas de la FB encontramos las siguientes:

- **IBFF.** Que es la información de buen funcionamiento de la instalación, con la que podremos activar la FB. Si las IBFF no es uno nunca podrá trabajar la FB 1710.
- **CAR.** Condiciones con rearme. Si perdemos las CAR será obligatorio rearmar la instalación desde el pupitre principal.
- **CSR.** Condiciones sin rearme. Si perdemos estas condiciones es posible rearmar desde cualquier parte de la instalación, ya sean las puertas, pupitres secundarios etc…
- **CSecu0.** Condición de seguridad a 0. Es una realimentación a cero de nuestra variable de entrada del relé.
- **CSecu1.** Condición de seguridad a 1. Es una realimentación a uno de nuestra variable de entrada del relé.
- **Iadf.** Se usa para anular algún defecto dentro de la instalación, esta variable será empleada en la mayoría de las FBs.
- **Ir.** Se una para rearmar la instalación y hacer que vuelva a su estado funcional.

Como salidas de la FB encontramos:

- **DO_F.** Con esta variable activamos la salida de seguridad que deseamos.
- **Secu.** Nos genera una variable de seguridad.
- **Secu_OK.** Nos genera la variable de seguridad anterior comprobando el retorno del relé, por lo que será la variable que empleemos ya que es más segura.
- **Def.** En caso de que exista un defecto en la FB, por ejemplo, tener las CSecu0 y CSecu1 ambas a uno, nos impedirá trabajar con la FB y las variables de salida no se actuarán.

Dentro de nuestra FB de seguridad, definiremos una *información de buen funcionamiento de la instalación* para generar esta información tendremos en cuenta

no tener ningún defecto en ninguno de las FB 1710 que vayamos a emplear, así como tener el retorno de nuestra FB de las cartas de seguridad.

Una vez que ya hemos creado nuestra IBFF vamos a programar la FB 1710 para la parada de emergencia. En primer lugar, cabe decir que como estamos hablando de paradas de emergencia las condiciones sin rearme será siempre un bit a 1, para las condiciones con rearme de la instalación, deberemos de tener en cuenta no tener ninguna parada de emergencia pulsada, en el caso de que pulsemos una parada de emergencia, será necesario rearmar la instalación desde el pupitre operador principal. Para las condiciones de seguridad a 0 y a 1 pondremos el nemónico de retorno del relé.

Antes de explicar la programación del relé de puesta en servicio y de zona cerrada, es necesario explicar el funcionamiento de la FB para nuestras puertas Euchner, la FB que empleamos es las FB 1712 [16].

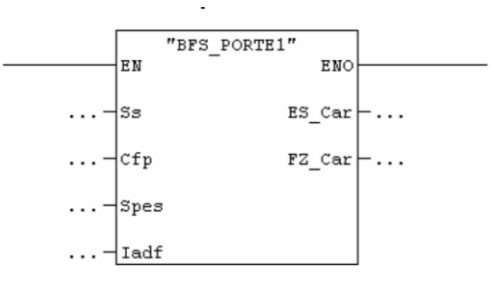


Ilustración 55: FB 1712.

En general esta FB funciona de forma similar a las FB1710 [16]. Las entradas son las siguientes:

- **SS.** El selector de seguridad de la puerta.
- **Cfp.** Control de cierre de la puerta, es una variable interna que detecta el cierre de la puerta enviando una señal vía profinet al autómata.
- **Spes.** Selector de puesta en servicio de las puertas nos indica si la puerta está en condiciones de cerrado o está abierta. Hace las veces de cerrojo.
- **Iadf.** Es un botón que sirve para anular el defecto de la puerta. En nuestro caso también será usado para anular el defecto de la instalación.

Como salidas encontramos dos, que son las que nos marcarán la diferencia entre puesta en servicio y zona cerrada.

- **ES_Car.** En este caso el selector Spes debe estar abierto, pero el control de cierre puede estar o a uno o a cero. Es como tener una cerradura abierta.

- **FZ_Car.** En este otro caso, el selector debe de estar cerrado con el control activado, es como si cerramos una puerta normal con llave.

Para la generación de las condiciones de puesta en servicio y zona cerrada, tendremos en cuenta el estado de las puertas, de modo que las condiciones con rearme de la instalación será tener el retorno ES_Car de la FB de las puertas, lo que significa que la zona está abierta. Por otro lado, como condiciones con rearme de la zona cerrada, pondremos el retorno FZ_Car lo que nos indicará que las puertas de la instalación están cerradas de forma correcta.

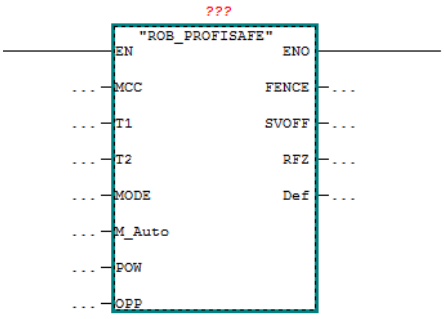


Ilustración 56: FB robot.

La FB 1722 [16] es la FB del robot, para nuestro caso de la marca Fanuc. Como señales de entrada de la FB tenemos:

- **MCC.** Entrada de la palanca del hombre muerto.
- **T1.** Selector del robot en manual.
- **T2.** Selector del robot en automático.
- **MODE.** Tipo de funcionamiento del robot que explicaremos más adelante.
- **M_Auto.** Modo automático.
- **POW.** Variable para la asignación de potencia del robot para poder ser movido en manual. En nuestro caso para poder moverlo en manual, usaremos el tener la puesta en servicio.
- **OPP.** Condiciones para poder mover en automático el robot, en nuestro caso la zona cerrada de la instalación.

Como salidas contamos con:

- **Fence.** Fence significa valla para nuestro caso significa que el robot está en condiciones de seguridad para poder moverse en automático.
- **SVOFF.** Esta variable nos retorna un 1 cuando se recibe una parada externa del robot, ya sea la perdida en auto de la instalación o una seta de emergencia.
- **RFZ.** Esta variable retorna un 1 cuando se cumplen las condiciones de seguridad deseadas, permitiendo al robot activar sus 24V de seguridad.

- **Def.** Indica si hay algún tipo de defecto en el robot.

Una vez que ya tenemos nuestra FB de seguridad de zona, procederemos a configurar nuestra FB de seguridad de operario.

Como habíamos comentado anteriormente, dentro del Festo de nuestro operario constamos de un módulo de seguridad. Dicho módulo de seguridad, hará las veces de nuestro relé como en el ZB. Vamos a emplear al igual que en el FB de seguridad de zona la FB 1710 de la misma manera. Tendremos unas condiciones de buen funcionamiento de la instalación, que reciclaremos la IBBF que habíamos creado para el FB de seguridad de zona. Ésta variable se puede usar sin problemas ya que habíamos dicho anteriormente que es una variable de un DB global de seguridad. Como condiciones sin rearme (que es la especialmente importante) usaremos el no tener cortada la barrera inmaterial que tenemos instalada y tener la zona de la instalación cerrada. Como Csecu0 y Csecu1 tenemos una variable digital que hemos creado, que corresponde a la entrada 0 del Festo de seguridad. Como botones de anulación de defecto y rearme de la instalación usaremos los mismos que para el FB de seguridad de zona. Como salida del bloque es decir, DO_F activaremos la señal homóloga a la entra de seguridad del Festo que comentábamos antes. El módulo Festo con el correcto funcionamiento de nuestra FB 1710 nos retornará un 1, dando alimentación 24V a las tarjetas de eventos de la instalación. De éste modo se podrá accionar de manera automática las electroválvulas y crear los distintos movimientos y secuencias que deseemos.

Ya tenemos nuestro programa de seguridad creado en su totalidad. Ahora pasaremos a la creación del programa en sí. Crearemos en primer lugar una función nueva, que la llamaremos FC100, será nuestra función de zona, al igual que nuestra FC1100 pero sin ser de seguridad, por lo que a la hora de crearla usaremos el lenguaje CONT. En ella haremos la llamada a las distintas FBs que creemos, en nuestro caso vamos a crear varias FBs muy simples.

- **FB101.** Esta FB será la referente al operador donde programaremos la gestión de trabajos del operador.
- **FB102.** Ésta la usaremos para realizar el control y comando del útil de trabajo del operador.
- **FB103.** Aquí realizaremos la gestión del robot con sus códigos de programa y eventos.
- **FB104.** Éste será nuestro último FB, hará referencia a la garra del robot y al control de sus elementos neumáticos.

En el momento que hagamos la llamada en nuestra función FC100 de las FBs crearemos los DBs de instancia asociados a ellas, estos DBs no serán de seguridad. Primer lugar vamos a explicar el puesto operario.

Nuestro puesto de operario consta de un montaje en el que el operario cargará las piezas, un pupitre que consta de una parada de emergencia, un botón de validación y un selector de modo automático o manual. Por otro lado consta, de una serie de balizas de 3 colores, color verde para indicar al operario que puede cargar,

color rojo para la prohibición de trabajo y color amarillo para señalar la existencia de algún defecto en el puesto de carga. A nivel de seguridad, consta de unas barreras láser que nos indicarán la presencia o no del operario.

Dentro del trabajo del operario se distinguen a nivel autómata de tres fases, dichas fases:

- Autorización de trabajo, éste evento se programa dentro de la FB del montaje, es éste el que habilita al operario a trabajar, y para ello tiene que cumplir varias condiciones. Un operario puede trabajar siempre y cuando no haya trabajado ya, tenga las condiciones mecánicas apropiadas para ello y que no haya una pieza ya cargada.
- Fin de trabajo, este evento es programado en el operario, ya que es él el que comunica al montaje la finalización de su trabajo. En nuestro caso, lo hemos programado teniendo en cuenta que el operario ha entrado y salido de la zona de trabajo y que ha realizado la validación desde su pupitre. Para controlar ese cruce del operario, se ha generado una memoria para detectar un cambio de 0 a 1 de nuestra barrera inmaterial.
- Registro de trabajo, la pieza cargada está correctamente cargada y se ha generado una memoria que lo confirma, dicha memoria será la que nos inhabilite el poder cargar de nuevo pieza.

En la FB del montaje de carga, programaremos los distintos movimientos del mismo así como la autorización a trabajar del operario y las distintas condiciones mecánicas del mismo.

A continuación, vamos a explicar la función que vamos a emplear para realizar el control de nuestro montaje. La función a usar tiene gran cantidad de entradas y salidas así que sólo voy a comentar las que empleamos en nuestro programa. La FB a usar es la FB 2042.

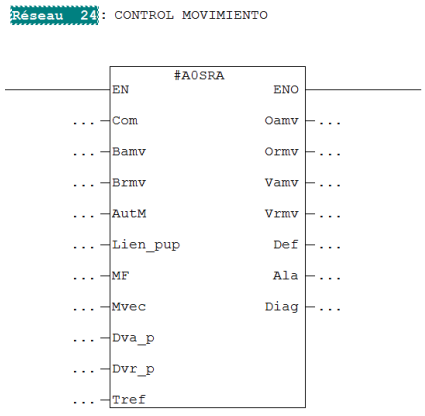


Ilustración 57: FB 2042.

Como entradas tenemos:

- **COM.** Al igual que tenemos un común de seguridad para poder activar las electroválvulas, también existe un común dentro del programa como memoria para permitir a la FB funcionar, y por tanto trabajar. En nuestro caso la parte eléctrica coincide con la parte programa, y será la seguridad del operario *"TSM".AOSOP_Secu_OK*.
- **Lien_pup**, aquí recogemos las informaciones de nuestra instalación, es una estructura en la que encontramos datos como la marcha en automático, la anulación y rearme del defecto etc..., en nuestro caso emplearemos la variable *lien_pup.Z_mauto* para decir a nuestra FB que estamos trabajando en modo automático.
- **MF.** Al igual que la variable anterior, es una estructura que guarda información general de la FB, para nosotros las más importantes son las siguientes:
  - **SA** y **SR.** Se refieren a las seguridades de avance y de retroceso, pero a tener nuestra variable **COM** es necesario que se tengan las seguridades mecánicas para poder avanzar o retroceder.
  - **AA** y **AR.** Autorizaciones de avance y de retroceso, cuando éstos bits están a uno, y tenemos la seguridad mecánica podremos realizar la acción.
  - **IA** e **IR.** Hacen referencia a la información de avance y de retroceso.
- **Tref.** Es el tiempo de referencia para generar un defecto.

Como salidas útiles para nosotros tenemos:

- **Oamv**. Es la orden de avance, es decir cuando cumplimos todas las condiciones necesarias, el común las seguridad, etc... la FB nos retorna un uno en ésta variable, nosotros lo que haremos será poner directamente nuestra variable del actuador de FB, en éste caso para avanzar o cerrar el elemento mecánico.
- **Ormv**. Procederemos de la misma manera que para el avance pero en éste caso para el retroceso.
- **Def**. Ésta variable nos indica que existe algún defecto en la gestión de nuestro movimiento siempre y cuando el tiempo referenciado se haya cumplido. El defecto significa que estamos comandando el avance o el retroceso pero no obtenemos información de que se haya producido dicho movimiento por parte de los detectores instalados.

Una vez explicada la FB a usar, podemos entrar dentro de la programación en sí. En lo que a las seguridades se refiere, podremos avanzar o retroceder nuestro elemento móvil siempre y cuando tanto el robot como el operario estén fuera de la zona, o en caso de que sea necesario por las condiciones mecánicas, si el robot está en su posición de cogida. Dentro de nuestra FB de montaje también programaremos las autorizaciones para que nuestro robot pueda coger dejar etc...

Como hemos comentado anteriormente, al cargar una pieza en el puesto de carga de nuestro operario, generamos una memoria asociada al mismo. En el caso de que generemos una memoria asociada a una pieza de tipo distinto al querido, se

creará un defecto y la baliza naranja se encenderá, pidiendo el realizar una carga de nuevo. Cuando una pieza tiene su memoria asociada de forma correcta, se creará lo que denominamos *información de pieza presente* llamado de forma abreviada IPP, la parte opuesta la *información pieza ausente* consiste en no tener una pieza y no tener detectores es decir, estar vacío. Explicamos esto porque nos va a ser muy útil a lo largo del trabajo. En primero lugar, para crear las ordenes de avance y retroceso de nuestros elementos móviles del montaje, nosotros comandaremos un retroceso de los mismos cuando exista una IPA (información pieza ausente) para que el operario pueda cargar libremente y cuando nuestro robot esté en el fin de cogida, retrocederemos nuestros elementos móviles. En el caso de que tengamos una IPP en el montaje (información pieza presente) lo que haremos será avanzar los elementos móviles para que el robot pueda coger la pieza.

Dentro de nuestra FB del montaje, tenemos programadas las distintas condiciones mecánicas, ya que el montaje tendrá unas condiciones para poder cargar la pieza que será todos los elementos retrocedidos, unas condiciones de descarga y unas condiciones de retroceso de descarga en caso de que sea necesario varios movimientos para la evacuación de la pieza.

En la FB 103, que es la que hace referencia a nuestro robot, programaremos lo relevante al mismo. Al igual que para los movimientos de dos posiciones usamos una FB específica, para nuestro robot, también usamos una FB específica. En nuestro caso, la FB a usar es la FB 1654.

Al igual que otras FBs consta con gran número de entradas y de salidas, por lo que solo explicaremos las que nos incumben.

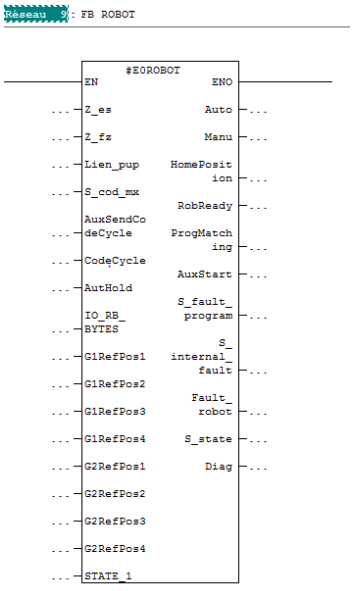


Ilustración 58: FB 1654 ROBOT.

En primer lugar comenzamos por las entradas y las salidas de la FB específica del robot y posteriormente, explicaremos la FB general del robot. Como entradas útiles tenemos:

- **Z_es.** Puesta en servicio, ésta variable nos permite el manejo del robot con la puerta abierta.
- **Z_fz.** En ésta variable diremos cuando estamos con la zona cerrada para poder así trabajar en automático.
- **S_cod_mx.** Ésta entrada de tipo entero nos limitará el número de códigos a usar, en nuestro caso no afectará ya que usaremos no más de tres códigos.
- **CodeCycle.** El código de ciclo de un robot, es el número de programa que éste leerá. Es decir según el código de ciclo que le enviemos el robot ejecutará una trayectoria u otra, en el caso de una instalación más grande, dependerá del tipo de pieza a coger etc…

Como salidas útiles de la FB tenemos:

- **Auto.** Es una variable que nos indica que el robot está en automático.
- **Manu.** Por el contrario, esta variable nos indica que el robot está en manual.
- **HomePosition.** Por medio de esta variable, sabemos si el robot se encuentra en una zona segura.
- **RobReady.** El robot está listo y a la espera de código para ejecutar su programa.

Programación Fanuc.

En primer lugar, en el momento en el que recibimos un robot, lo que debemos de hacer es realizar un payload del mismo. Esto consiste en realizar un calibrado de la carga que va a tener el robot. El realizar un buen payload [17] conlleva que los movimientos ejecutados por el robot serán más fluidos. En nuestro caso al usar una garra de apenas 5 kilos no nos influirá mucho ya que el robot aguanta hasta 12 Kg. En primer lugar, deberemos de hacer el payload sin haber montado la garra, es decir en vacío.

Para la calibración en vacío solamente se mueven los ejes 5 y 6. El resto de los ejes se quedarán fijos en su posición. Una vez hemos realizado el payload en vacío será el momento de realizarlo con la garra ya montada. Con la realización del payload determinaremos la masa de la garra, el centro de gravedad de la misma, de éste modo el robot propio robot podrá calcular los esfuerzos. Existen dos maneras de realizarlo, poniendo los datos directamente como vemos en Ilustración 59: Payload Manual.

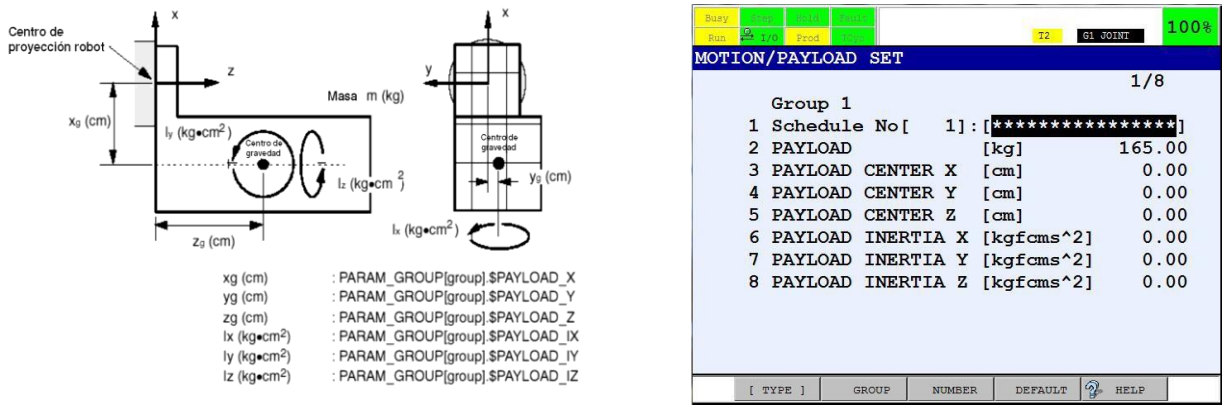


Ilustración 59: Payload Manual.

O que el robot haga sus propios cálculos, que será el método que empleemos. Para éste método aun así es aconsejable conocer el peso de la garra, pero se puede realizar de todos modos sin saberlo. El robot lo que hará será dos maniobras, una a alta velocidad, y otra a baja velocidad. Debemos de tener en cuenta que si la geometría de la garra puede colisionar con los brazos del robot, es necesario establecer unos límites dentro del giro de los ejes 5 y 6. Una vez se haya realizado nos aparecerá el peso de la garra y el centro de gravedad de la misma. Una vez que la tengamos, a la hora de realizar las trayectorias, podremos seleccionar nuestro payload garantizando unas trayectorias suaves y refinadas, evitando sobrecargar los servos.

Como comentábamos antes, el módulo Festo de la garra, ha sido configurado como un esclavo del robot, por lo que a diferencia del módulo Festo del operador, que es el autómata quién controla las señales de entradas y salidas, para la garra, será el robot quien lo haga. Para ello hemos realizado una configuración especial, dentro de la caja que hay en encima del eje 4, como vemos en la Ilustración 60: Conexionado Señales Garra.

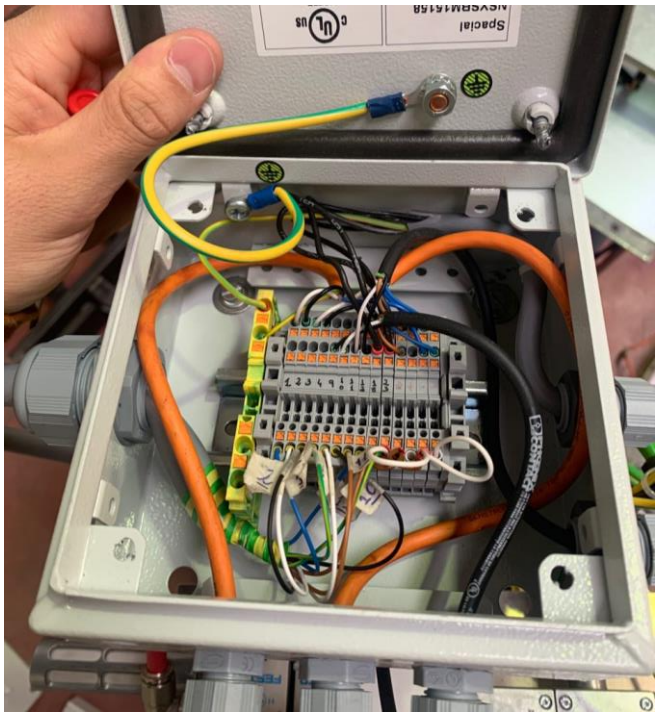


Ilustración 60: Conexionado Señales Garra.

Dichas señales van cableadas directamente a la controladora del robot, las variables serán denominadas como Robot In y Robot Out el conexionado se hace por medio del conector EE (End Effector). El conector EE es conector hembra que tiene montado 24 pines configurados de la siguiente manera:

- Señales de la herramienta 8RI (Robot IN) y 8RO (Robot Out).
- Una señal de detección de rotura de la garra (contacto normalmente cerrado).
- Una señal de neumática correcta (contacto normalmente cerrado).
- Consta de 4 tomas de 24 V y 2 tomas de 0V.

Con la compra del robot se nos ha suministrado el conector macho, posteriormente tuve que seguir el esquema adjunto para cablear las señales. Debido al tamaño hemos cableado 4 señales de entrada, 4 señales de salida, una de 24 V y una de 0 V que luego con la ayuda del repartidor podremos prolongar en caso necesario.

Dentro de las Robot In, tendremos las detecciones de avance, retroceso del piloto, el retroceso del imán neumático y la detección de la pieza. Como Robot Out, si vemos la Ilustración 26: Garra Robot. Observamos que las electroválvulas son de accionamiento externo, por lo que nuestras salidas del robot serán los eventos de avance y retroceso del piloto y del imán neumático. Al fin y al cabo, el Festo ha sido instalado en la garra, solamente para poder realizar un circuito neumático estanco.

Una vez que hemos realizado una comprobación de las señales, pasaremos a realizar un tool [17]. Un tool es la declaración de una herramienta, con él podremos

mover el TCP del robot que inicialmente está en el centro del eje 6 a donde haya sido calculado. Es muy importante realizar un buen tool ya que nos permitirá crear un punto de pivotaje para luego movernos con el robot de forma más precisa. En robótica, se distinguen dos tipos de herramientas, la simple que es en la que el eje Z es paralelo al eje Z del TCP inicial., o compleja en la que el eje Z de nuestra herramienta no es paralelo al del TCP inicial.

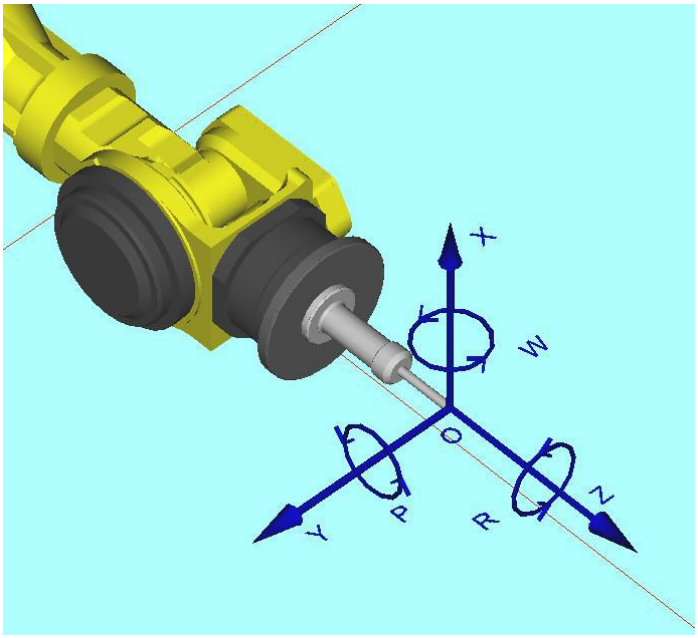


Ilustración 61: Ejemplo de Herramienta Simple.

Para la creación de una herramienta, existen muchas maneras. Si se ha realizado un buen estudio mecánico podremos al igual que en el payload introducir las variables directamente, esta técnica se denomina “entrada directa de valores”. Mediante este método introduciremos las coordenadas y la orientación de la herramienta directamente.

En nuestro caso es mucho más práctico usar el “método de los 3 puntos”, para ello lo que haremos será elegir un punto de la garra que sea interesante para realizar los movimientos. Yo he elegido el extremo del piloto ya que a la hora de realizar la cogida, me será mucho más fácil orientar la garra desde ese punto. Para ello elegimos un punto fijo de la estructura, y posicionamos el piloto en ese punto con 3 orientaciones distintas, con éste método, la controladora podrá calcular el TCP de la herramienta que hemos instalado, como comentábamos antes, nuestra herramienta es simple por lo que el sentido de la ordenada Z de nuestro nuevo TCP es el mimo que el del TCP inicial.

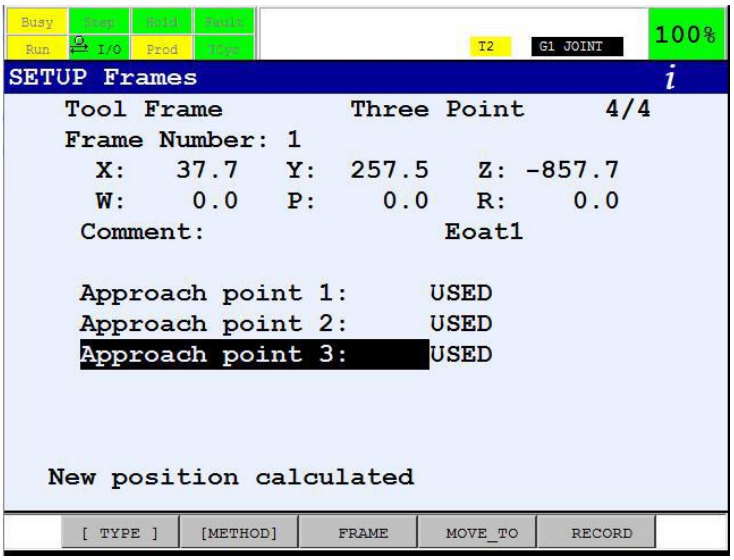


Ilustración 62: Creación Herramienta.

Posteriormente, una vez tenemos nuestro tool, puede suceder que la zona de trabajo donde nuestro manipulador debe actuar no tenga un sistema de coordenadas solidario al tool creado anteriormente. En nuestro caso debido a que el montaje operario es paralelo al suelo y por tanto solidario a la garra, no sería necesario crear un frame. Sin embargo, para ser más puristas realizaremos uno.

Al igual que cuando creamos la herramienta del robot, lo que estamos haciendo es crear un plano. Por lo que los métodos a realizar son los mismos que para la herramienta.

Una vez tenemos la mecánica del robot, nos faltaría el empezar a comunicar con el autómata. Para ello podemos configurar de distintas maneras la relación autómata/robot. En nuestro caso la relación será del robot siendo esclavo del autómata, para ello debemos de establecer la siguiente configuración en las cartas de comunicación del robot.

Al igual que cuando configuramos nuestros módulos Festo en la configuración material, añadiendo las tarjetas, y el orden de la realidad, en nuestro caso con el robot debemos de hacer lo mismo. En la ilustración siguiente, la configuración hace referencia a que la primera carta, será la de seguridad con un tamaño de 11 bytes, y posteriormente una carta de proceso que será con la que intercambiemos las variables no seguras de 32 bytes.

Emplacement	Module ...	Référence	Adresse E	Adresse S	Adresse de diagnostic
1	*CORB*	*A05B-2600-J930*			*1000**
X1	*PN-IO*				*1004**
X1 P1	*Port 1 - RJ45*				*1003**
X1 P2	*Port 2 - RJ45*				*1002**
X1 P3	*Port 3 - RJ45*				*1001**
X1 P4	*Port 4 - RJ45*				*1005**
2	CORBSafe		182...193	182...193	
3	CORBProcess		150...181	150...181	

Ilustración 63: Configuración Cartas Robots Step7.

Cabría esperar, que si el robot trae esas tarjetas, se configurarían automáticamente, sin embargo eso no es así. En primer lugar, para establecer la relación maestro esclavo, debemos de meternos en la configuración profinet del robot, y deshabilitar la tarjeta I/O Controller.

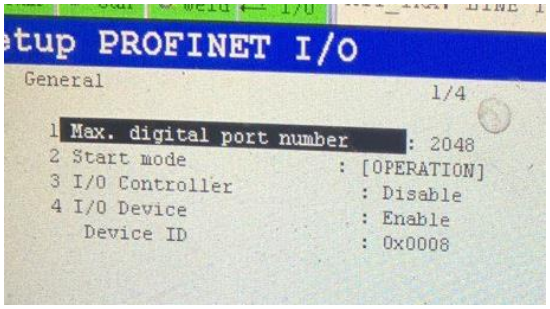


Ilustración 64: Configuración Profinet Fanuc.

Con ella deshabilitada, ignoraremos la configuración de la misma, si se hubiera declarado el robot como maestro, aquí deberíamos configurar las tarjetas que hemos asociado al robot.

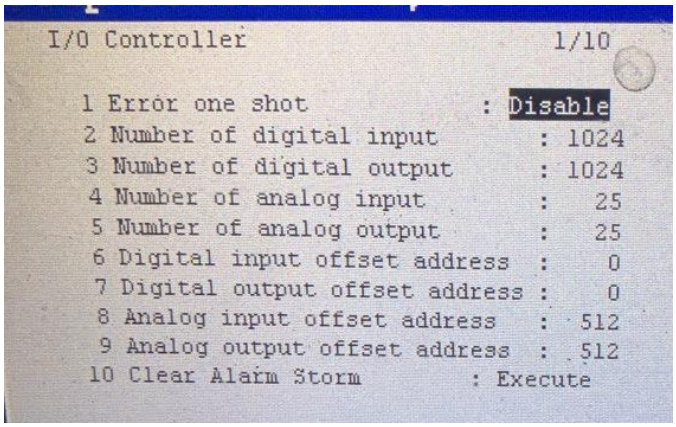


Ilustración 65: Configuración Tarjeta Controller.

En la configuración del IO será donde configuremos el robot de acuerdo a nuestra configuración material, ésta carta, hace refererencia a la carta de procesos que tenemos puesta nosotros. En la Ilustración 63: Configuración Cartas Robots Step7. Podemos ver que la tarjeta de proceso empieza desde la señal 150 a la 181, y éstamos a hablado de bytes u octetos, lo que corresponde a 32 bytes, que son 256 bits que son los que hemos onfigurado en la imagen de más abajo.

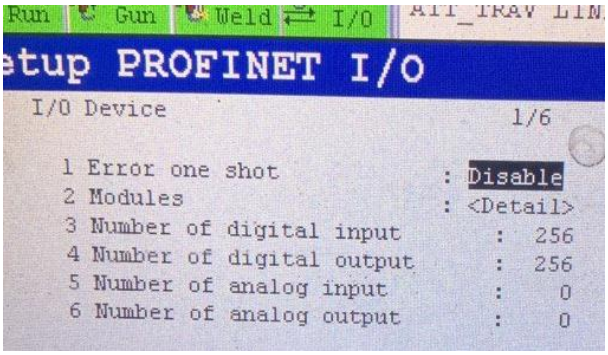


Ilustración 66: Device I/O.

Por otro lado, debemos de configurar el robot diéndole que cartas hay en cada slot y configurándolas. Si observamos de nuevo la Ilustración 63: Configuración Cartas Robots Step7. tenemos las en primer lugar el bloque CP1604 que hace referencia al controlador que tenemos instalado, posteriormente la tarjeta que controlará las variables de seguridad (8 bytes) y la tarjeta de proceso de la que hemos hablado anteirormente (32 bytes).

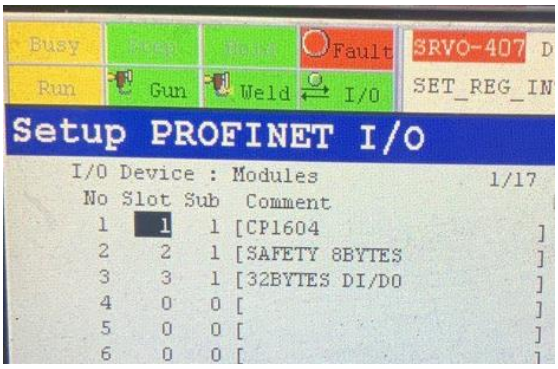


Ilustración 67: Configuración Profinet.

Al finalizar la configuración de las tarjetas será necesario reiniciar el robot, una vez esté encendido debemos de cargar la configuración de Step 7 dentro del robot, para ello si nos fijamos en la Ilustración 64: Configuración Profinet Fanuc. la segunda opción "Start Mode" tiene 3 modos de funcionamiento; "suspend" en la que se habilita al robot a recibir la configuración deseada, "read In" en la que el robot lee la configuración que ha sido cargada y determina si hay o no defectos y por último, "operation" que es el modo de funcionamiento del robot. Para poder cargar la configuración material desde Step7 ponemos el robot en "Suspend" y nos pedirá reiniciar, una vez ha sido reiniciado podemos cargar la configuración y asignarle una IP y un nombre desde el menú de Step 7, al igual que hicimos con los módulos Festo. Una vez ha sido cargada, pasaremos al modo "Read In", nuestro Fanuc leerá la configuración, y si ésta es coherente, se podrá poner en modo "operation". Una vez hemos pasado a "operation" será necesario volver a reiniciar el robot.

Aún no hemos acabado con la configuración de nuestro robot, si recordamos más arriba hablamos de que todos los elementos de seguridad de ésta instalación contaban con un número inequívoco que hace que todas las variables de seguridad sean específicas de cada elemento. En el caso del robot también existe un número de seguridad, que debe ser el mismo en nuestra configuración de Step 7 y en el robot. La pregunta de por qué un robot es un elemento de seguridad, se responde que al ser un elemento capaz de realizar movimientos a grandes velocidades, no debemos de permitirle moverse si existe algún defecto de seguridad en la instalación.

Para configurar este número, debemos ir al menú DCS que es el acrónimo de Dual Check Safety, la posición y la velocidad, se calculan con dos CPUs distintas por lo que si existe un error, se corta de manera inmediata la tensión hacia los motores. Dentro de éste menú también se configura nuestro número de profisafe. En nuestro caso el numero a poner es el 197 que es el que hemos configurado en el autómata, una vez hemos cambiado dicho número, debemos aplicar los cambios, nos pedirán la contraseña que está de serie y posteriormente aplicar los cambios y reiniciar. Una vez se ha reiniciado, dentro de nuestro autómata volveremos a cargar la configuración material y el programa de seguridad. De este modo nuestra instalación está totalmente funcional.

Después de configurar las catas etc.. el robot ya es completamente funcional de cara al autómata, es decir ya podemos controlar las señales enviarle condiciones de trabajo etc... Anteriormente comentábamos que íbamos a usar una FB, más concretamente la FB 1654, sin embargo, a lo largo del proyecto, sólo la vamos a poder usar para poder controlar algunas variables del robot. Normalmente, a nivel industrial, se usan distintos ciclos, ya sea para un tipo de pieza, u otra para realizar una trayectoria de coger/ dejar, o cualquier otra cosa, normalmente, se envía un valor entero por medio de la FB al robot y él ejecuta dentro de su lista la trayectoria correspondiente al ciclo dado, sin embargo, para poder usar ese tipo de configuración los robots tienen cargado un programa específico que les permite funcionar de esa manera. Nosotros emplearemos la opción que nos permite Fanuc de fábrica, que es el uso de las UI.

Ya a lo largo del proyecto, hemos explicado varios tipos de variables, las DI/DO que son las que configuramos en la carta de Process mencionada antes, las RI/RO que son las que hemos cableado directamente al controlador del robot, y ahora debemos hablar de UI/UO. Ésta variables permiten comandar el robot de manera externa, nos permiten arrancar, anular defecto del robot etc.. de modo remoto. Para nosotros, podremos comandar el robot con variables escritas desde el PLC. En total, son 18 entradas y 20 salidas que están ya definidas, nosotros decidimos si serán señales cableadas físicamente o señales digitales, que será nuestro caso.

De todas esas señales predefinidas, no usaremos todas, solamente las necesarias para poder realizar nuestro ciclo. Como salidas tenemos:

- **CMDENBL.** Es una señal que envía el robot en la que nos dice que el robot tiene los selectores en el modo automático, y que el modo remoto ha sido activado.
- **SYSRDY.** Indica que los motores tienen tensión.
- **PROGRUN.** Se activa cuando un programa está siendo ejecutado por parte del robot.
- **PAUSED.** Un programa ha sido parado y está a la espera de ser reiniciado.
- **HELD.** Esta señal la envía el robot cuando la tecla “hold” es pulsada, deteniendo el programa, en el momento que deja de ser pulsada continua el programa.
- **FAULT.** El sistema presenta un error, el programa se para y para anular el error se necesita una UI para anular ese defecto. Si hay una alarma no se emite esta señal.
- **ATPERCH.** Esta señal indica que el robot se encuentra en una posición de referencia, normalmente la posición de inicio y final de trayectoria.
- **TPENBL.** Recibimos un 1 cuando el selector del teachpendant está en ON.
- **BATALM.** Esta señal la recibimos cuando las baterías internas del robot están demasiado bajas, estas baterías valen para que los encoders no pierdan su referencia cuando se les corta la tensión.

- **BUSY.** Recibimos esta señal cuando la CPU está ocupada o se está ejecutando un programa.

Del mismo modo como entradas [UI] tenemos:

- **IMSTP.** Es una señal de paro que se da mediante software, lo que hace es forzar un estado de error, por lo que es una especie de señal de seguridad (no es de seguridad como tal ya que nuestra seguridad la controla el autómata), y siempre tiene que estar a uno. Si pasa a cero a lo largo de la ejecución del programa, el robot de detendrá, se desconectarán los servos.
- **Hold.** Es una señal normalmente a 1, y que servirá para detener el programa en ejecución cuando pasa a cero. Similar a la señal anterior, pero si generar un error en el sistema.
- **SFSPD.** Es una señal que permite dar al robot la velocidad máxima de ejecución siempre y cuando tenga las condiciones de automático. Es decir, en manual, la velocidad máxima de movimiento es inferior a la de en automático.
- **Fault-Reset.** Es una señal externa que nosotros asociaremos al botón de reseteo de la instalación. Se resetean los defectos y el programa pausado vuelve a ejecutarse desde el punto en el que estaba.
- **Start.** Esta señal, será la que permita ejecutar la rutina de trabajo del robot, en nuestro caso será cuando exista una pieza cargada correctamente en el montaje. Si la señal está a cero, el programa se reanuda y puede continuar, en caso de que pase a uno, el programa se detendrá por lo que es una señal que debe de estar a uno hasta que tenga condiciones de trabajo.
- **CSTOPI.** Es una señal que indica la parada de ciclo
- **HOME.** Esta señal nos dice que el robot se debe mover a la posición registrada como HOME.
- **ENBL.** Esta señal habilita que el movimiento del robot sea posible, siempre debe estar a uno, en el caso de que pasa a cero, trabajando en modo remoto aparecerá un defecto.
- **PNSTROBE.** Es una señal de validación del código enviado, dicha validación se produce con un flanco ascendente.
- **PROD_START.** Esta señal inicia la ejecución del programa, siempre que sea sistema PNS, en caso de que no lo sea significa la ejecución del programa desde el cursor.

Para que un robot por arranque remoto funcione, debemos de seguir una secuencia de eventos:

1. Debemos en primer lugar habilitar las señales UI dentro de nuestra controladora. Debemos activar la opción de “ENABLE UI SIGNALS”.
2. El robot debe darnos la señal de salida UO CMD ENABLE, y para ello se deben cumplir las siguientes condiciones:

a. Necesitamos en primer lugar la UI:1 IMSTP en ON, es decir que no se recibe ningún paro por software.
b. UI:2 HOLD a ON, no se recibe ningún paro de programa externo.
c. UI:3 SFSPD a ON, no se recibe paro debido a una velocidad determinada del robot.
d. UI:8 ENABLE para poder permitir el movimiento del robot.
e. Tener el selector en AUTO, con lo que las seguridades externas de hardware quedan habilitadas.
f. Tener el controlador en modo remoto de modo que se pueda ejecutar el arranque del robot por medio de un motor macha que estará asociado a la entrada UI:6 START. La cual generará un impulso descendente.
g. Quitar las condiciones paso a paso de ejecución del programa.
h. UO: 2 SYS READY en ON de modo que el robot no tiene fallos o han sido reseteados.

Con estas condiciones y un flanco descendente, se consigue el robot arranque de forma remota.

Ciclo Automático.

El operario espera la autorización a trabajar por parte de la instalación de formación. Para tener dicha autorización, se deben cumplir las siguiente33333 condiciones:

- Montaje sin memoria y sin pieza.
- Montaje con las condiciones mecánicas de carga correctas, es decir, ventosa retrocedida.
- Robot fuera de la zona de trabajo del operario.
- Operario fuera de las barreras inmateriales de seguridad.

Cuando éstas condiciones se cumplan, la baliza de color verde se iluminará. Dando pie al operador a cargar la pieza.

Ilustración 68: Baliza indicadora de la autorización de trabajo.

El operador toma la pieza, con los guantes apropiados para ello, cruza la barrera inmaterial. En el momento en el que cruza la barrera, el robot queda bloqueado por su FB de seguridad. De éste modo el robot jamás se moverá en automático, es una medida de seguridad que nos permite en caso de que alguien cruce la barrera mientras el robot está trabajando el robot se pare de modo inmediato. El robot, aunque no haya nada bloqueando la barrera, no retomará su trabajo, ya que es un defecto grave y es necesario rearmar en el panel del operador.

El operador una vez coloca la pieza y abandona la zona de carga, deberá realizar una validación. De este modo garantizamos que el operador se encuentra fuera de la zona de trabajo, y comprobamos que la pieza ha sido bien cargada. En caso de que no sea así la baliza naranja quedará fija pidiendo intervención de nuevo por parte del operario. Cuando la pieza es correcta y ha sido validada, la ventosa de nuestro montaje actuará, sujetando la pieza firmemente. En caso de que la pieza sea la correcta, la baliza pasará a color rojo indicando prohibición de trabajo.

Ilustración 69: Baliza Operario Indiciando Prohibición de trabajo.

El robot ya ha recibido el ciclo en el momento que el operario valida correctamente la pieza y está a la espera de recibir el evento de poder coger. Para poder coger, se contempla que el montaje tenga la pieza correcta, las condiciones mecánicas sean buenas y que el robot se encuentre en la posición de espera de evento, de este modo garantizamos que el robot está en un punto conocido. El robot es el que controla sus elementos, por lo que controlará mediante detecciones el estado de la misma.

En el punto de fin de cogida y con la información del detector embarcado en la garra de que hay pieza, el robot avanzará el piloto, y cuando garantice que está avanzado, es decir que tenemos la detección de avance, el imán se avanzará. De este modo garantizamos que el robot tiene la pieza cogida de forma correcta.

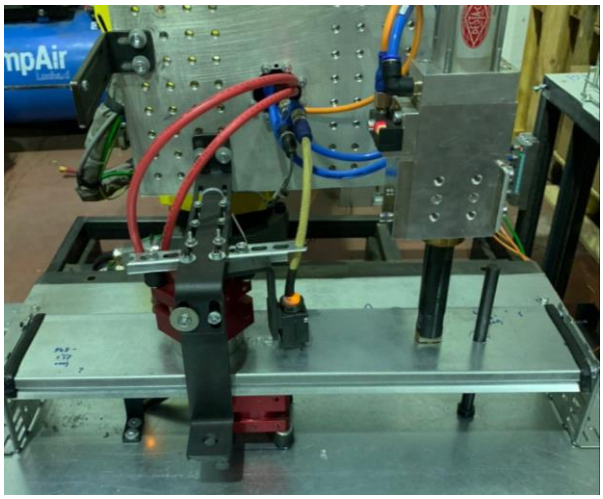


Ilustración 70: Fin de cogida del Robot.

Con esas condiciones, el robot se puede retirar con la pieza, en el momento en el que llegue a una posición fuera de peligro, dará la seguridad al montaje, de modo que si se cumplen las condiciones será posible cargar otra pieza.

Nuestro robot, se irá con la pieza a otro montaje para dejarla. En el punto de fin de dejada, el robot retrocederá imanes y pilotos transfiriendo la información de la pieza al montaje siguiente.

Ilustración 71: Fin de dejada Robot.

Una vez el robot ha dejado la pieza en nuestro puesto de salidas, volverá a su posición de “Home” que hayamos programado. Es muy importante que el robot siempre acabe la trayectoria en ese punto, ya que envía una señal al autómata que le permitirá realizar la adquisición de un nuevo ciclo de programa. En el caso de que el robot no se encuentre en dicho punto, pese a que el operador cargue la pieza de

manera correcta, y le demos ciclo y evento al robot, éste no se moverá, será necesario pasar el robot a manual y llevarle a ese punto.

Conclusiones.

Dentro de mis tareas en la empresa, la realización de éste aula de formación no era la más prioritaria, debido en parte a la falta de espacio en el taller el comienzo de la realización del aula se alargó. Con el añadido de la falta de stock por parte de nuestros proveedores y las negociaciones con Fanuc hizo que se alargara aún más el inicio del aula de formación.

La línea temporal con la que he realizado este proyecto ha sido muy similar a la que he narrado en los objetivos. Primero hice el pedido de todos los materiales para mis armarios eléctricos y posteriormente los cableé, aproximadamente tardé una semana en el cableado y testeo del armario grande ZB, que mide cerca de 1,8 metros, y alrededor de 2 días en el armario de operador OPB, que es un tercio del anterior. Una vez cableados necesité la ayuda para colocarlos dentro de su encajonamiento metálico y posterior colocación en el aula de formación.

Con los armarios ya cableados y probados, comencé a realizar la interconexión de 24V y profinet así como la configuración de los distintos elementos de mi instalación. Sin embargo, por aquel entonces aún no disponía de actuadores por lo que tenía una instalación sin ningún movimiento.

Antes de la parada de verano, que es el pico de trabajo de mi empresa por el cual estaría cerca de 3 meses sin poder realizar ningún avance llegó el robot. Realicé la interconexión de los elementos de la garra con sus actuadores y señales tal y como comento en Programación Fanuc. También realicé el diseño e implantación mecánica de la garra con los elementos disponibles.

Aún no tenía realizado el reconocimiento por parte del autómata de los distintos elementos de la instalación, debido en gran manera a que la memoria de mi PLC estaba corrupta. Una memoria nueva cuesta alrededor de 150€, por lo que invertí cierto tiempo en intentar arreglar la memoria de tipo NAND, finalmente conseguí arreglarla perdiendo aproximadamente la mitad de la capacidad, pero permitiéndome cargar mi programa y poder comenzar el reconocimiento del equipo.

En la vuelta del verano, con una carga de trabajo muy inferior, comencé a realizar la mecánica del puesto de carga del operario, y del puesto en el que el robot dejaría la pieza. La mecánica y el cableado de los detectores y los actuadores me llevó aproximadamente dos semanas, debido al inicio de un nuevo proyecto del que sería responsable.

El siguiente gran problema al que debí hacer frente, fue la configuración de las cartas de comunicaciones de mi robot Fanuc. Debido a la falta de información existente, y a que dicha configuración es específica necesité la ayuda del jefe de robótica para poder dar con el problema. Finalmente, una vez establecidas las comunicaciones, el aula de formación estaba funcional, a la espera de un buen programa de autómata y la creación de las trayectorias del robot.

A modo conclusión, es un proyecto factible con el que he aprendido, pese a mis años de experiencia a elaborar un programa desde cero con un modelo de autómata distinto, debido a la reutilización de elementos de otras instalaciones he tenido que pensar mucho el modo de funcionamiento y diseño de la instalación. Actualmente, la instalación está lista para la incorporación de los nuevos miembros de Icare Automation y para su formación en los estándares del grupo Renault.

Tabla de Ilustraciones

Bibliografía

[1]«https://www.slideshare.net/automatizacionplc/resea-historica-de-la-automatizacion,» [En línea].

[2]«https://www.mcr.es/automatizacion-industrial-como-funciona/,» [En línea].

[3]«http://www.ctinmx.com/que-es-un-plc/,» [En línea].

[4]«https://automatizacionindustrial.wordpress.com/2011/02/09/queeslaautomatizacionindustrial/,» [En línea].

[5] «http://automatica.mex.tl/imagesnew/5/0/1/4/2/Presentaci%C3%B3n%20P.L.C..pdf,» [En línea].

[6]«http://recursostic.educacion.es/observatorio/web/gl/component/content/article/502-monografico-lenguajes-de-programacion?start=2,» [En línea].

[7]«http://www.sc.ehu.es/sbweb/webcentro/automatica/WebCQMH1/PAGINA%20PRINCIPAL/PLC/ESTRUCTURAS/ESTRUCTURA%20INTERNA/CPU/cpu.htm,» [En línea].

[8] «https://revistadigital.inesem.es/gestion-integrada/buses-de-campo/,» [En línea].

[9] «https://vinssa.com/robot-scara,» [En línea].

[10] «http://www.mekkam.com/robotica-industrial/robot-cartesiano/,» [En línea].

[11]«https://www.researchgate.net/publication/327010808_Aplicaciones_de_los_Robots_Paralelos,» [En línea].

[12]«http://www.unicauca.edu.co/ai/publicaciones/ISAShow_Vivas.pdf,» [En línea].

[13] «https://arhatarahant.files.wordpress.com/2014/07/robots_antropomorficos.pdf,» [En línea].

[14] «https://www.elfinanciero.com.mx/tech/que-son-los-robots-colaborativos-y-por-que-son-ideales-para-las-pymes,» [En línea].

[15]«https://cache.industry.siemens.com/dl/files/834/12403834/att_347/v1/et200eco_operating_instructions_es-ES_es-ES.pdf,» [En línea].

[16]«http://www.cnomo.com/fr/index.php?PHPSESSID=qm6kooesin2oar82032gjknfg4,» [En línea].

[17]«https://www.academia.edu/27814336/ROB%C3%93TICA_PROGRAMACI%C3%93N_FANUC,» [En línea].

[18]«http://galia.fc.uaslp.mx/~cantocar/automatas/PRESENTACIONES_PLC_PDF_S/4_EL_PLC.PDF,» [En línea].

Anexos.

Anexo I. Programa del Autómata.

Programa de Seguridad Instalación.

SIMATIC

FB1110 – <offline>

"TSZ1_AU_ES_FZ" ZONA 1: FB Gestion SAFETY AU, ES, FZ
Nom : **Famille :**
Auteur : **Version :** 0.3
Version de bloc : 2
Horodatage Code : 24/01/2020 12:06:47
Interface : 20/12/2019 09:22:28
Longueur (bloc/code /données locales) : 01402 00894 00010

Propriétés de l'objet :
S7_language 12(1) Français (France) 04/03/2008 15:32:00

Nom	Type de données	Adresse	Valeur initiale	Commentaire
IN		0.0		
OUT		0.0		
IN_OUT		0.0		
STAT		0.0		
Z1AU	BFS_BSEC1	0.0		Z1: GESTION BUCLE PARADA DE EMERGENCIA
Z1ES	BFS_BSEC1	30.0		Z1: GESTION BUCLE PUESTA EN SERVICIO
Z1FZ	BFS_BSEC1	60.0		Z1: GESTION BUCLE ZONA CERRADA
Z1PORTE1	BFS_PORTE1	90.0		Z1: GESTION PUERTA 1
B0RB	ROB_PROFISAFE	96.0		B1_RB: GESTIÓN SEGURIDAD ROBOT
TEMP		0.0		
X1_IBFF	Bool	0.0		Aux. 1 Info Buen Funcionamiento Filerie
X2_IBFF	Bool	0.1		Aux. 2 Info Buen Funcionamiento Filerie
X3_IBFF	Bool	0.2		Aux. 3 Info Buen Funcionamiento Filerie
X4_IBFF	Bool	0.3		Aux. 4 Info Buen Funcionamiento Filerie
X5_IBFF	Bool	0.4		Aux. 5 Info Buen Funcionamiento Filerie

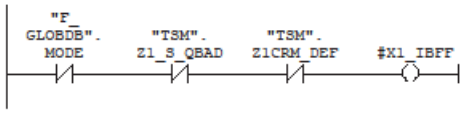
Bloc : FB1110 Bloque funcional de seguridad

Réseau : 1 //////////////////IBFF//////////////////////////

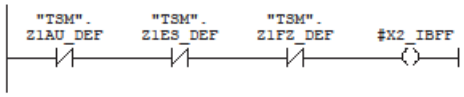
Página de

SIMATIC

Réseau : 2 X1_IBFF



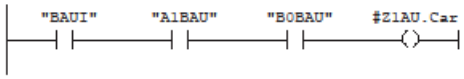
Réseau : 3 X2_IBFF



Réseau : 4 IBFF

Réseau : 5 ////////////////// PARADA DE EMERGENCIA ////////////////////

Réseau : 6 AU: Condiciones con rearme



Réseau : 7 AU: Condiciones sin rearme

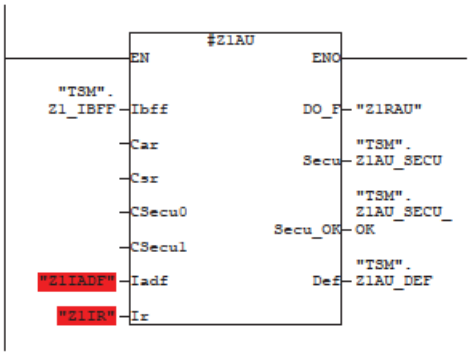
Página de

SIMATIC

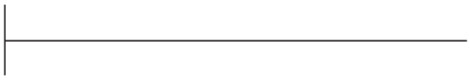
Réseau : 8 Control Bucle Seguridad AU a 0

Réseau : 9 Control Bucle Seguridad AU a 1

Réseau : 10 GESTION BUCLE SEGURIDAD PARADA DE EMERGENCIA



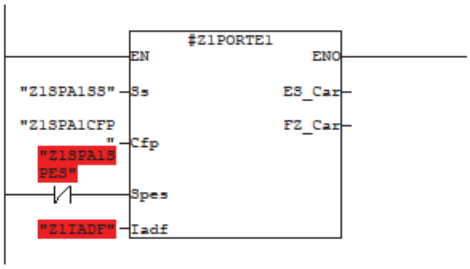
Réseau : 11 ////////////////////Gestion puertas ////////////////////



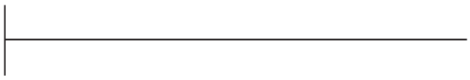
Página de

SIMATIC

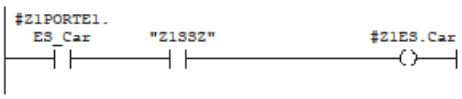
Réseau : 12 Z1: GESTION PUERTA 1



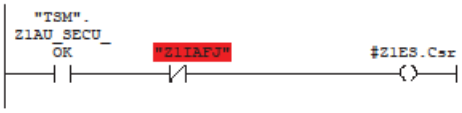
Réseau : 13 //////////////////// PUESTA EN SERVICIO ////////////////////////



Réseau : 14 ES: CONDICIONES CON REARME



Réseau : 15 ES: CONDICIONES SIN REARME

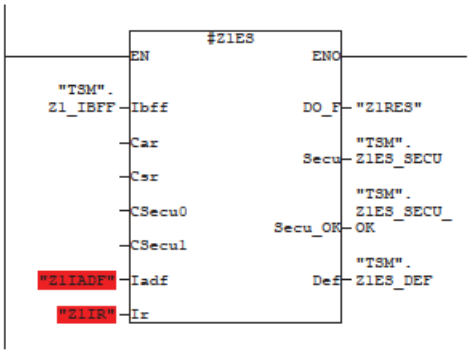


Réseau : 16 ES: CONTROL RETORNO SIN PILOTAGE

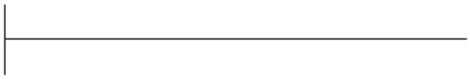
SIMATIC

Réseau : 17 ES: CONTROL RETORNO CON PILOTAGE

Réseau : 18 Z1 : GESTION BUCLE DE SEGURIDAD PUESTA EN SERVICIO



Réseau : 19 ////////////////////BUCLE ZONA CERRADA ////////////////////



Réseau : 20 FZ: CONDICIONES CON REARME

Página de

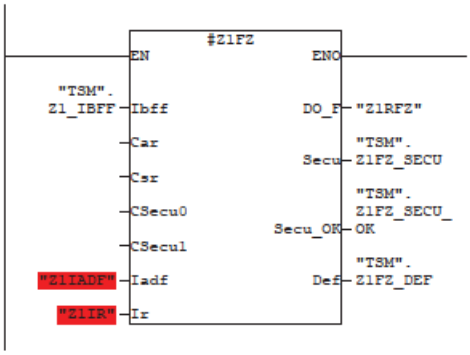
SIMATIC

Réseau : 21 FZ: CONDICIONES SIN REARME

Réseau : 22 Z1 : CONTROL RETORNO SIN PILOTAGE

Réseau : 23 Z1 : CONTROL RETORNO CON PILOTAGE

Réseau : 24 Z1 : GESTION BUCLE DE SEGURIDAD ZONA CERRADA

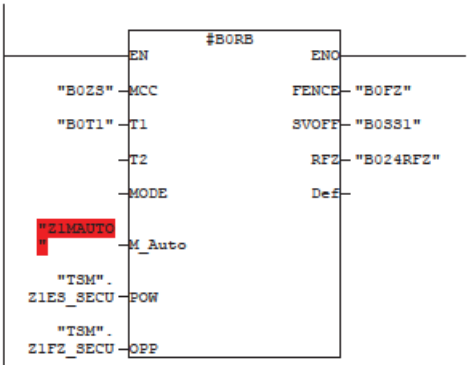


Página de

SIMATIC

Réseau : 25 //////////////////GESTION SEGURIDAD ROBOT//////////////////

Réseau : 26 B0RB : GESTION SEGURIDAD ROBOT FANUC R30ib Profisafe



Programa de Seguridad Operario.

SIMATIC

FB1111 - <offline>

"TSZ1_SOP" ZONA 1: FB Gestion SAFETY SOP
Nom : **Famille :**
Auteur : **Version :** 0.3
Version de bloc : 2
Horodatage Code : 20/12/2019 11:56:11
Interface : 20/12/2019 11:56:11
Longueur (bloc/code /données locales) : 00748 00390 00008

Propriétés de l'objet :
S7_language 12(1) Français (France) 04/03/2008 15:32:00

Nom	Type de données	Adresse	Valeur initiale	Commentaire
IN		0.0		
OUT		0.0		
IN_OUT		0.0		
STAT		0.0		
A01SOP	BFS_BSEC1	0.0		
A01EVEMP	BFS_BSEC1	30.0		Z1: GESTION RELE PUESTA EN SERVICIO
A01CS1	BFS_BSEC1	60.0		Z1: GESTION BUCLE DE SEGURIDAD AU
TEMP		0.0		

Bloc : FB1111 Bloque funcional de seguridad

Réseau : 1 ////// UTIL A01 //////

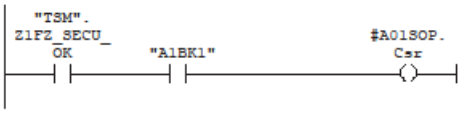
Réseau : 2 Gestion bucle Seguridad Operador A1

Réseau : 3 Condicion con rearme bucle de seguridad operador A1SOP

Página de

SIMATIC

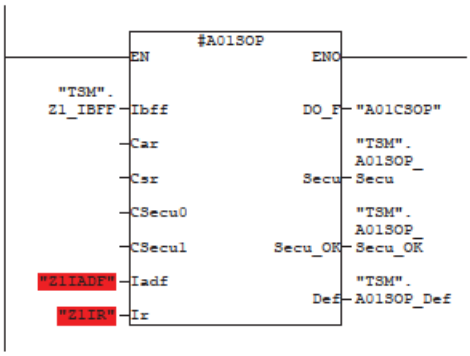
Réseau : 4 Condicion sin rearme bucle de seguridad operador A1SOP



Réseau : 5 A01: Control bucle Seguridad SOP a 0

Réseau : 6 A01: Control bucle seguridad SOP a 1

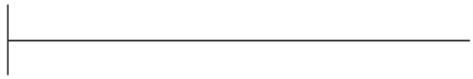
Réseau : 7 GESTION BUCLE SEGURIDAD OPERADOR MONTAGE A01



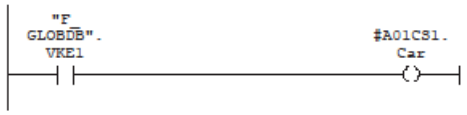
SIMATIC

Réseau : 8 A1: COMUN ELEC SEGURIDAD OPERADOR OK

Réseau : 9 ///////////////MODULO SALIDA A01CS1 ///////////////



Réseau : 10 CONDICIONES CON REARME



Réseau : 11 CONDICIONES SIN REARME

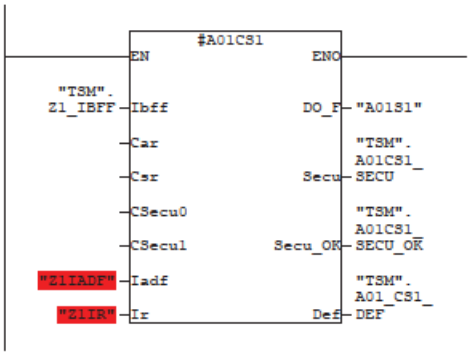
Réseau : 12 Control Bucle Seguridad a 0

Réseau : 13 Control Bucle Seguridad a 1

Página de

SIMATIC

Réseau : 14 BF A01CS1



Programa del Operario.

SIMATIC

FB103 - <offline>

"1_A1_OPERARIO"

Nom : **Famille :**
Auteur : **Version :** 0.1
Version de bloc : 2
Horodatage Code : 26/02/2020 19:40:46
Interface : 26/02/2020 19:40:46
Longueur (bloc/code /données locales) : 00586 00280 00008

Nom	Type de données	Adresse	Valeur initiale	Commentaire
IN		0.0		
OUT		0.0		
IN_OUT		0.0		
STAT		0.0		
A1TATOP	TON	0.0		A1_OP: TEMPO ESPERA OPERARIO
A1FTOPDV	TON	22.0		H8_OP: FIN DE TRABAJO TEMPORIZADO MARCHA VACIO
A1LIEN_PUP	LIEN_PUP	44.0		ENLACE PUPITRE PARA GESTION MANU
A1TDEF	TON	46.0		A1_OP: FIN DE TRABAJO TEMPORIZADO MARCHA VACIO
A1TATOP_PT	Time	68.0	T#0MS	A1_OP: UMBRAL TIEMPO ESPERA OPERARIO
S1A1DRMESBI	Bool	72.0	FALSE	A1 - DEFECTO BI PARA REARMAR PUESTA EN SERVICIO
TAO	Int	74.0	0	A1_OP: UMBRAL TIEMPO ESPERA OPERARIO
A1MTOP	Bool	76.0	FALSE	A1_OP: MEMORIA CRUCE OPERARIO
A1MRBV	Bool	76.1	FALSE	A1_OP: MEMORIA LIBERACION PULSADOR OPERARIO
S1A1DINTRID	Bool	76.2	FALSE	A1_OP: DEF INTERNO CORTINA ALBANY
S1A1DBVOP	Bool	76.3	FALSE	A1_OP: DEF LIBERACION BP VALIDACION
TEMP		0.0		
RetVal	Int	0.0		
DEF_BUS	Bool	2.0		DEFECTO BUS / DEFAUT BUS
tpsop	Time	4.0		

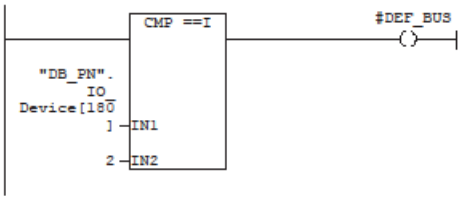
Bloc : FB103 OPERADOR

Página de

SIMATIC

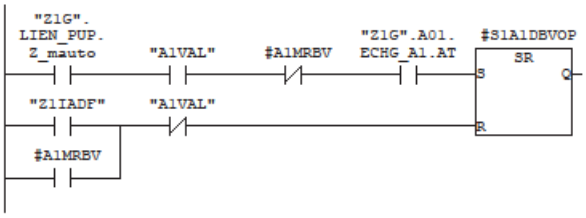
Réseau : 1 ////////////////////// PROGRAMA SEGURIDAD //////////////////////

Réseau : 2 DEFECTO BUS



Réseau : 3 ////////////////////// GESTION OPERADOR //////////////////////

Réseau : 4 DEFECTO CAIDA PULSADOR DE VALIDACION OPERARIO

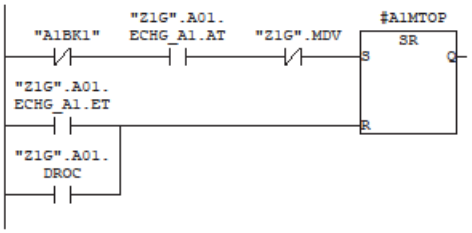


Réseau : 5 ////////////////////// MEMORIAS OPERADOR //////////////////////

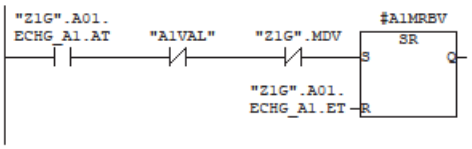
Página de

SIMATIC

Réseau : 6 MEMORIA CRUCE OPERARIO

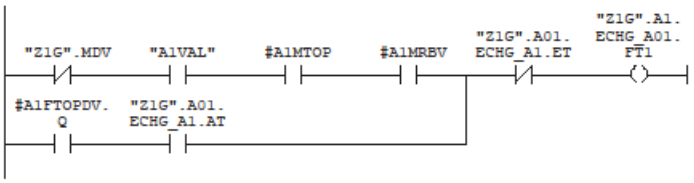


Réseau : 7 MEMORIA CAIDA DEL PULSADOR DE VALIDACION



Réseau : 8 ////////////////////// INTERCAMBIOS CON A01 //////////////////////

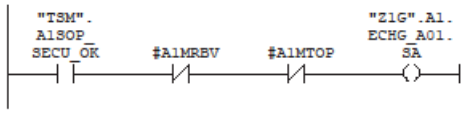
Réseau : 9 FIN DE TRABAJO: FT



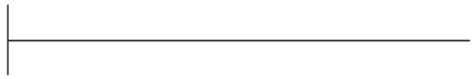
Página de

SIMATIC

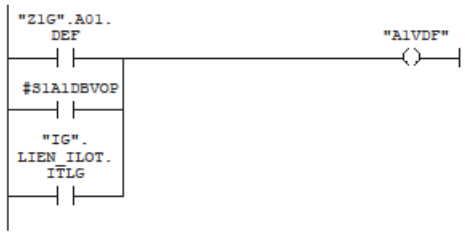
Réseau : 10 SEGURIDAD TRASERA OP: SA



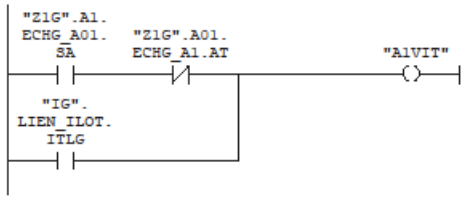
Réseau : 11 ////////////////////// OPERADOR BALIZA //////////////////////////



Réseau : 12 LAMPARA DEFECTO PUPITRE OPERARIO



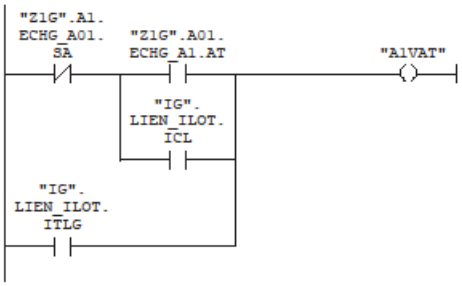
Réseau : 13 LAMPARA PROHIBICION DE TRABAJO OPERARIO



Página de

SIMATIC

Réseau : 14 LAMPARA AUTORIZACION DE TRABAJO OPERARIO



Página de

Programa del Útil.

SIMATIC

FB106 - <offline>

"1_A01_UTIL"

Nom : **Famille :**
Auteur : **Version :** 0.1
Version de bloc : 2
Horodatage Code : 26/02/2020 19:46:21
Interface : 26/02/2020 19:46:21
Longueur (bloc/code /données locales) : 01792 01106 00032

Nom	Type de données	Adresse	Valeur initiale	Commentaire
IN		0.0		
OUT		0.0		
IN_OUT		0.0		
STAT		0.0		
A01VMA	BF_MOUV2P1	0.0		TIPO_ELEMENTO VMA
A01TDPIEC	TON	82.0		A01 - TEMPORIZACION DEFECTO PIEZA CARGA OP
S1A01DVMA	Bool	104.0	FALSE	DEFECTO MOVIMIENTOS SRA
S1A01DPIEC	Bool	104.1	FALSE	A01 - DEFECTO PIEZA CARGA OPERARIO
IHMDPIEC	Int	106.0	0	DIVERSIDAD PARA DEFECTO PIEZA
TEMP		0.0		
RetVal	Int	0.0		

Bloc : FB106 UTIL A01

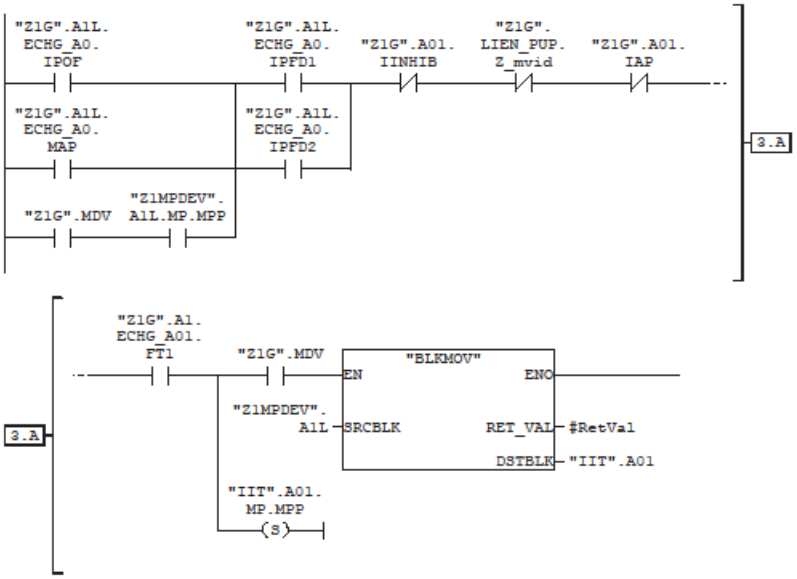
Réseau : 1 //////////// INTERCAMBIOS OPERADOR A1 ////////////

Réseau : 2 //////////////////// TABLA PIEZA ////////////////////////////

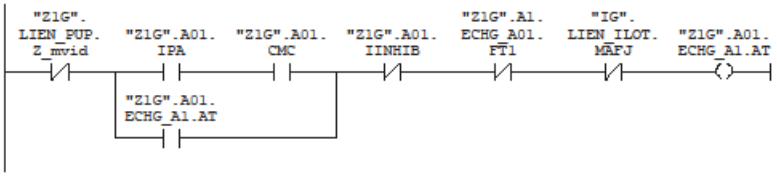
```
*******************************************************************************
**                          PART TABLE MANAGEMENT                            **
*******************************************************************************
```

SIMATIC

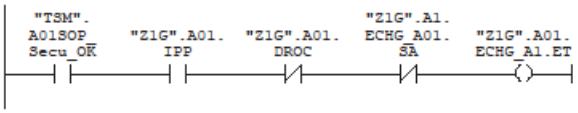
Réseau : 3 SET MPP



Réseau : 4 OPERADOR A1 AT AUTORIZACION DE TRABAJO



Réseau : 5 OPERADOR A1 ET REGISTRO DE TRABAJO

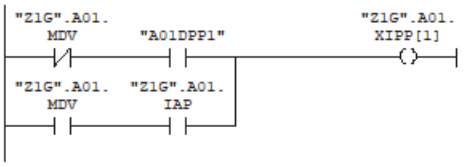


Página de

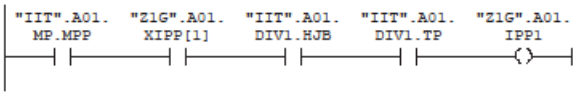
SIMATIC

Réseau : 6 ////////////////// IPP INFORMACION / IPP EN EL PUESTO //////////

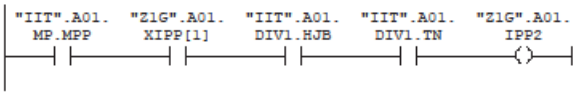
Réseau : 7 XIPP



Réseau : 8 IPP1



Réseau : 9 IPP2



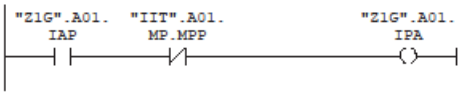
Réseau : 10 INFORMACION PIEZA PRESENTE

Página de

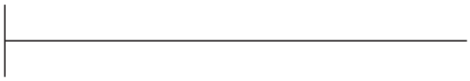
SIMATIC

Réseau : 11 IAP

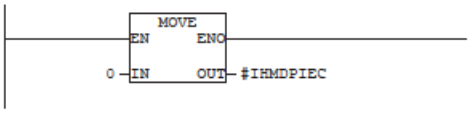
Réseau : 12 IPA



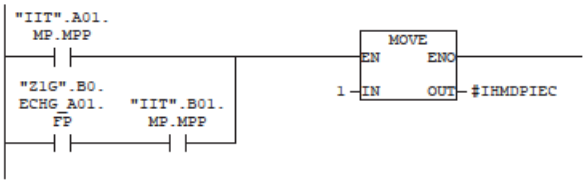
Réseau : 13 ////////////// GESTION DEFECTO PIEZA ////////////////////



Réseau : 14 RAZ DIVERSIDAD PARA DEFECTO PIEZA



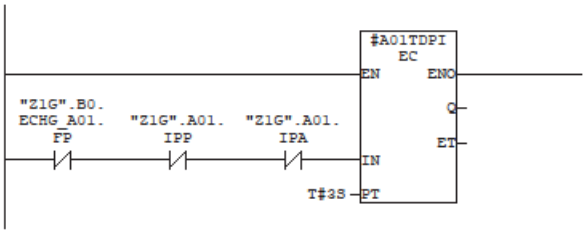
Réseau : 15 DEFECTO PIEZA



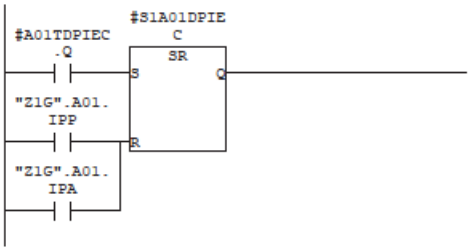
Página de

SIMATIC

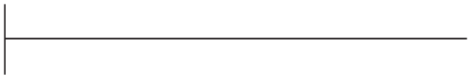
Réseau : 16 TEMPORIZACION DEFECTO PRESENCIA PIEZA



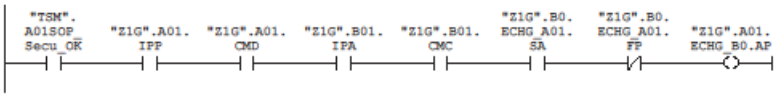
Réseau : 17 DEFECTO PRESENCIA PIEZA



Réseau : 18 ////////////////////// INTERCAMBIO ROBOT B0 ////////////////////



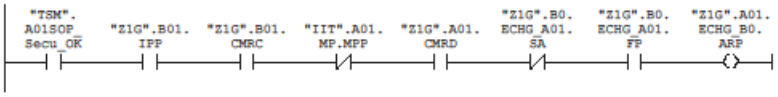
Réseau : 19 AUTORIZACION COGIDA B0



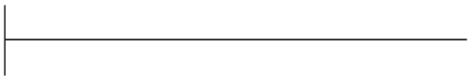
Página de

SIMATIC

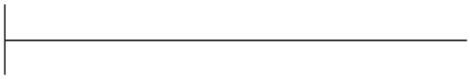
Réseau : 20 AUTORIZACION RETROCESO COGIDA B0



Réseau : 21 /////////////// MOVIMIENTOS //////////////

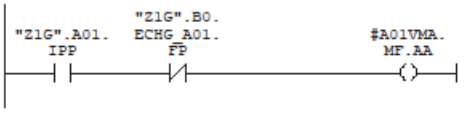


Réseau : 22 ********** VMA VENTOSA 1 **********



Réseau : 23 SEGURIDAD DE AVANCE VMA

Réseau : 24 AUTORIZACION DE AVANCE VMA



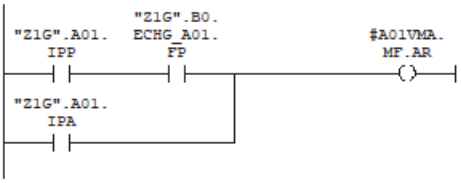
Réseau : 25 INFORMACION DE AVANCE VMA

Página de

SIMATIC

Réseau : 26 SEGURIDAD DE RETROCESO VMA

Réseau : 27 AUTORIZACION DE RETROCESO VMA

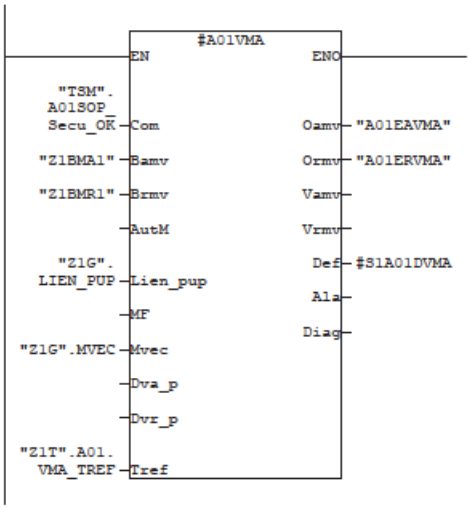


Réseau : 28 INFORMACION DE RETROCESO VMA

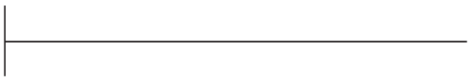
Página de

SIMATIC

Réseau : 29 BF VMA



Réseau : 30 ********** CONDICIONES MECANICAS **********



Réseau : 31 CONDICION MAQUINA AVANZADA = CMA

Réseau : 32 CONDICION MAQUINA RETROCEDIDA = CMR

Página de

SIMATIC

Réseau : 33 CONDICION MAQUINA CARGA = CMC

Réseau : 34 CONDICION MAQUINA RETROCESO CARGA = CMRC

Réseau : 35 CONDICION MAQUINA DESCARGA = CMD

Réseau : 36 CONDICION MAQUINA RETROCESO DESCARGA = CMRD

Réseau : 37 DEFECTO

Página de

Programa del Robot.

SIMATIC

FB110 - <offline>

"1_B0_ROBOT"

Nom : **Famille :**
Auteur : **Version :** 0.1
Version de bloc : 2
Horodatage Code : 18/02/2020 15:28:04
Interface : 17/01/2020 10:41:11
Longueur (bloc/code /données locales) : 02780 00844 00032

Nom	Type de données	Adresse	Valeur initiale	Commentaire
IN		0.0		
OUT		0.0		
IN_OUT		0.0		
STAT		0.0		
B0ROBOT	BF_RB_NISSAN32	0.0		B0 - ROBOT MANAGEMENT FONCTION BLOC / FB DE GESTION DU ROBOT
B0SUIVI	BF_TCY_SUIV	362.0		B0 - CYCLE TIME MONITORING / SUIVI DU TEMPS DE CYCLE ROBOT
B0xCODE	Bool	406.0	FALSE	B0 - ROBOT CYCLE CODE / CODE CYCLE DU ROBOT
B0CODE	Byte	407.0	B#16#0	B0 - ROBOT CYCLE CODE / CODE CYCLE DU ROBOT
S1B0NDR	Word	408.0	W#16#0	B0 - ROBOT FAULT NUMBER / NUMERO DE DEFAUT ROBOT
ST1	Int	410.0	0	B0 - T1 THRESHOLD (1/10S) / SEUIL DE T1 (BASE 1/10S)
ST2	Int	412.0	0	B0 - T2 THRESHOLD (1/10S) / SEUIL DE T2 (BASE 1/10S)
S1B0DIV	Int	414.0	0	B0 - ROBOT DIVERSITY IN PROGRESS / DIVERSITE EN COURS DE TRAVAIL
S1B0TCP	Int	416.0	0	B0 - OWN CYCLE TIME (1/10S) / TEMPS DE CYCLE PROPRE (BASE 1/10S)
S1B0TCT	Int	418.0	0	B0 - TOTAL CYCLE TIME (1/10S) / TEMP DE CYCLE TOTAL (BASE 1/10S)

Página de

SIMATIC

Nom	Type de données	Adresse	Valeur initiale	Commentaire
S1B0NCY	Int	420.0	0	B0 - ROBOT CYCLE NUMBER / NUMERO DE CYCLE ROBOT
S1B0APP	Bool	422.0	FALSE	B0 - ROBOT GENERAL FAULT / DEFAUT GENERAL ROBOT
S1B0APE	Bool	422.1	FALSE	B0 - EXPLOITATION OWN STOPPAGE / ARRET PROPRE EXPLOITATION
S1B0APF	Bool	422.2	FALSE	B0 - FUNCTIONNAL OWN STOPPAGE / ARRET PROPRE FONCTIONNEL
S1B0ALR	Bool	422.3	FALSE	B0 - OPERATOR CALL / APPEL OPERATEUR
S1B0AR	Bool	422.4	FALSE	B0 - STOP ROBOT / ARRET ROBOT
S1B0ATT	Bool	422.5	FALSE	B0 - ROBOT WAITING / ROBOT EN ATTENTE
S1B0TOPCY	Bool	422.6	FALSE	B0 - ROBOT TOP CYCLE / TOP CYCLE ROBOT
S1B0AIA	Bool	422.7	FALSE	B0 - WAIT FOR EVENT / ATTENTE CODE EVENEMENT
S1B0HPR	Bool	423.0	FALSE	B0 - NOT IN SERVICE / HORS PRODUCTION
S1B0DER	Bool	423.1	FALSE	B0 - ROBOT DERIVE / DERIVE ROBOT
S1B0DTC	Bool	423.2	FALSE	B0 - CYCLE TIME OVERRUN / DEPASSEMENT TEMPS T1
S1B0DT2	Bool	423.3	FALSE	B0 - CYCLE TIME OUT / DEPASSEMENT TEMPS T2
S1B0RML	Bool	423.4	FALSE	B0 - LOCAL MODE ROBOT / ROBOT MODE LOCAL
B0RAZROD	Bool	423.5	FALSE	B0 - RESET TIP DRESS COUNTER / RAZ COMPTEUR RODAGE
B0NBROD	Word	424.0	W#16#0	B0 - TIP DRESS COUNTER / COMPTEUR DE RODAGE ELECTRODE

SIMATIC

Nom	Type de données	Adresse	Valeur initiale	Commentaire
B0ALRCF	Word	426.0	W#16#0	B0 - TIP DRESSER CHANGE ALARM THRESHOLD VALUE / VALEUR SEUIL ALARM CHGT FRAISE
B0DEMCF	Word	428.0	W#16#0	B0 - TIP DRESSER CHANGE FAULT THRESHOLD VALUE/VALEUR SEUIL DEFAUT CHGT FRAISE
B0DROD	Bool	430.0	FALSE	B0 - TIP DRESS REQUEST / DEMANDE RODAGE
B0ALROD	Bool	430.1	FALSE	B0 - TIP DRESS ALARM / ALARME RODAGE
S1B01ACFR	Bool	430.2	FALSE	B0 - TIP DRESSER CHANGE ALARM / ALARME CHANGEMENT DE FRAISE
S1B01DCFR	Bool	430.3	FALSE	B0 - TIP DRESSER CHANGE REQUEST / DEMANDE CHANGEMENT DE FRAISE
S1B01AACE	Bool	430.4	FALSE	B0 - TIP CHANGE WARNING BEFORE ALARM / ALARME AVANT CHANGEMENT ELECTRODE
S1B01ACE	Bool	430.5	FALSE	B0 - TIP CHANGE ALARM / ALARME CHANGEMENT ELECTRODE
S1B01DCE	Bool	430.6	FALSE	B0 - TIP CHANGE REQUEST / DEMANDE CHANGEMENT ELECTRODE
S1B0PNO	Bool	430.7	FALSE	B0 - Non-working program
S1B0DIB	Bool	431.0	FALSE	B0 - Robot Internal error
B0TDCHELM	Word	432.0	W#16#0	B0 - WORD TIP DRESS CHELMAN
STATE_1	Struct	434.0		B0 - STATE
X0	Bool	434.0	FALSE	
X1	Bool	434.1	FALSE	
X2	Bool	434.2	FALSE	
X3	Bool	434.3	FALSE	
X4	Bool	434.4	FALSE	
X5	Bool	434.5	FALSE	
X6	Bool	434.6	FALSE	
X7	Bool	434.7	FALSE	

SIMATIC

Nom	Type de données	Adresse	Valeur initiale	Commentaire
X8	Bool	435.0	FALSE	
X9	Bool	435.1	FALSE	
X10	Bool	435.2	FALSE	
X11	Bool	435.3	FALSE	
X12	Bool	435.4	FALSE	
X13	Bool	435.5	FALSE	
X14	Bool	435.6	FALSE	
X15	Bool	435.7	FALSE	
X16	Bool	436.0	FALSE	
X17	Bool	436.1	FALSE	
X18	Bool	436.2	FALSE	
X19	Bool	436.3	FALSE	
X20	Bool	436.4	FALSE	
X21	Bool	436.5	FALSE	
X22	Bool	436.6	FALSE	
X23	Bool	436.7	FALSE	
X24	Bool	437.0	FALSE	
X25	Bool	437.1	FALSE	
X26	Bool	437.2	FALSE	
X27	Bool	437.3	FALSE	
X28	Bool	437.4	FALSE	
X29	Bool	437.5	FALSE	
X30	Bool	437.6	FALSE	
X31	Bool	437.7	FALSE	
S1B0DWC	Bool	438.0	FALSE	B0 - WELD CABLE FAULT
S1B0DW	Bool	438.1	FALSE	B0 - WELD FAULT
TB0DWF	TON	440.0		B0 - TIMER WATER FLOW FAULT
S1B0DWF	Bool	462.0	FALSE	B0 - WATER FLOW FAULT
S1B0DFDJRR	Bool	462.1	FALSE	B0 - ATD BREAKER FAULT
S1B0RBDCSEC	Bool	462.2	FALSE	B0 - SAFETY PLATE ERROR
S1B0SPEED	Bool	462.3	FALSE	B0 - VELOCIDAD 100 / SPEED 100
DEF_BUS	Bool	462.4	FALSE	
PULSE_TIPDRESS	Bool	462.5	FALSE	D01 - PULSE TIP DRESS
PULSE_TIPCHANGE	Bool	462.6	FALSE	D01 - PULSE TIP CHANGE
D01NBRODALR	Word	464.0	W#16#0	D01 - TIP DRESS COUNTER / COMPTEUR DE RODAGE ELECTRODE

SIMATIC

Nom	Type de données	Adresse	Valeur initiale	Commentaire
D01ALRELEC	Word	466.0	W#16#0	D01 - TIP DRESSER CHANGE ALARM THRESHOLD VALUE / VALEUR SEUIL ALARM CHGT ELEC
D01FPCPLC	Bool	468.0	FALSE	D01 - FRONT PREMIER CYCLE PLC
D01CODAPLISOLD	Int	470.0	0	D01 - CODIGO DEFECTO APLICACION SOLDADURA
S1D01SINSOLD	Bool	472.0	FALSE	D01 - SOLDADURA DESACTIVADA EN ROBOT
S1D01ALDELEC	Bool	472.1	FALSE	D01 - ALARMA DESGASTE ELECTRODOS
S1D01DFDDELEC	Bool	472.2	FALSE	D01 - DEFECTO DESGASTE ELECTRODOS
S1D01DFELECSC	Bool	472.3	FALSE	D01 - DEFECTO ELECTRODOS SIN CAMBIAR
S1D01ALDFRES	Bool	472.4	FALSE	D01 - ALARMA DESGASTE FRESADO
S1D01DFDGINS	Bool	472.5	FALSE	D01 - DEFECTO DESGASTE INSUFICIENTE
S1D01DFROTFRES	Bool	472.6	FALSE	D01 - DEFECTO ROTACION MOTOR FRESA
S1D01DFCDELEC	Bool	472.7	FALSE	D01 - DEFECTO COMPENSCION DESGASTE ELECTRODOS
S1D01DFVNO100	Bool	473.0	FALSE	D01 - DEFECTO VELOCIDAD DE TRABAJO NO AL 100%
FLANCOR	Bool	473.1	FALSE	E01 - FLANCO RODAGE
FLANCOE	Bool	473.2	FALSE	E01 - FLANCO ELEC
D01RCHELMSELEC	Bool	473.3	FALSE	D01: Flanco Descendente RAZ Seleccion CHELM
S1D01DWAPP	Bool	473.4	FALSE	D01 - WELD GENERAL FAULT / DEFECTO GENERAL SOLDADURA
D01CHELMSELEC	Bool	473.5	FALSE	D01: Flanco Seleccion CHELM
D01MRAZROD	Bool	473.6	FALSE	D01 - MEMORIA PARA RAZROD EN ROBOT
TD01DEFROD	TON	474.0		D01 - CONTROL TIEMPO FRRESADO

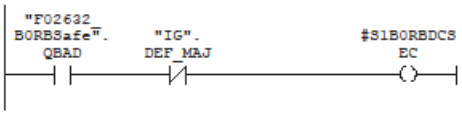
SIMATIC

Nom	Type de données	Adresse	Valeur initiale	Commentaire
S1D01DEFTFRE	Bool	496.0	FALSE	D01 - DEFECTO TIEMPO FRRESADO EXCEDIDO
S1D01CAMBFRES	Bool	496.1	FALSE	D01 - REVISION O CAMBIO DE FRESA
TEMP		0.0		
RETVAL	Int	0.0		

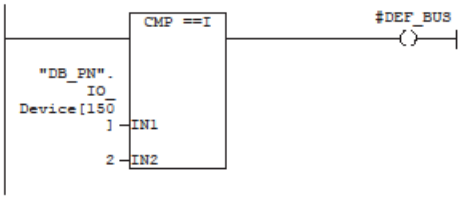
Bloc : FB110 ROBOT

GESTION DE ROBOT B1

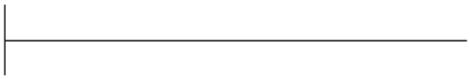
Réseau : 1 DEFECTO CARTA SEGURIDAD



Réseau : 2 DEFECTO BUS ROBOT



Réseau : 3 //////////////ROBOT ORDENES/////////



SIMATIC

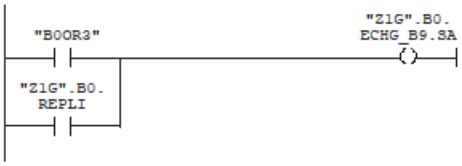
Réseau : 4 SEGURIDAD TRASERA A01

****
**
****

Réseau : 5 FIN DE COGIDA A01

Réseau : 6 SEGURIDAD TRASERA B9

****
**
****



Réseau : 7 FIN DE DEJADA B9

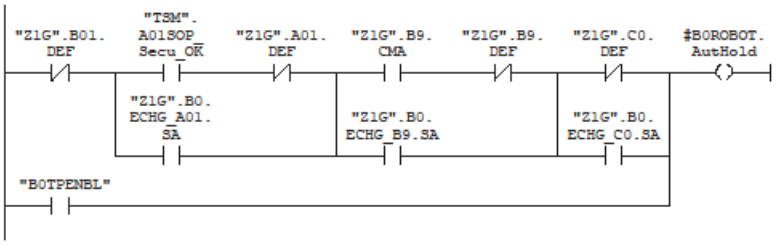
Página de

SIMATIC

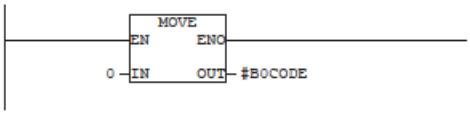
Réseau : 8 SEGURIDAD TRASERA C0

Réseau : 9 FIN DE DEJADA C0

Réseau : 10 AUTORIZACION DE EVOLUCION



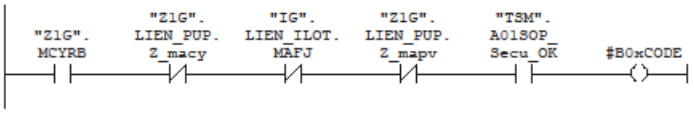
Réseau : 11 RESET CODIGO CICLO



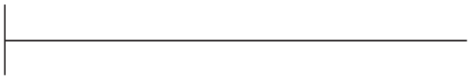
Página de

SIMATIC

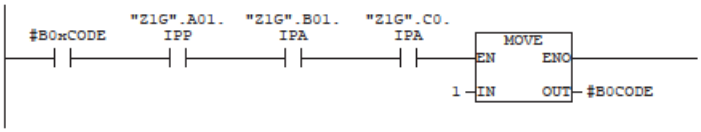
Réseau : 12 AUXILIAR CICLO PRINCIPAL



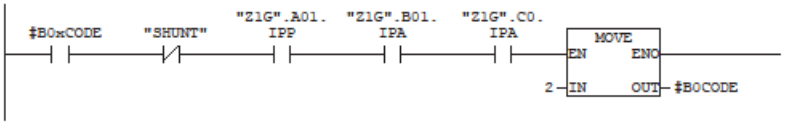
Réseau : 13 /////////CICLOS ROBOT//////////



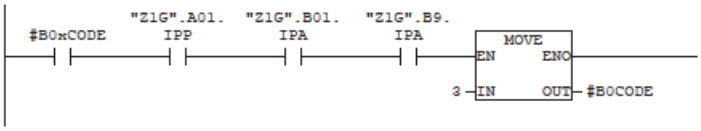
Réseau : 14 CICLO 1 - AGUILLAJE



Réseau : 15 CICLO 2 - DEJADA EN C0



Réseau : 16 CICLO 3 - DEJADA B9

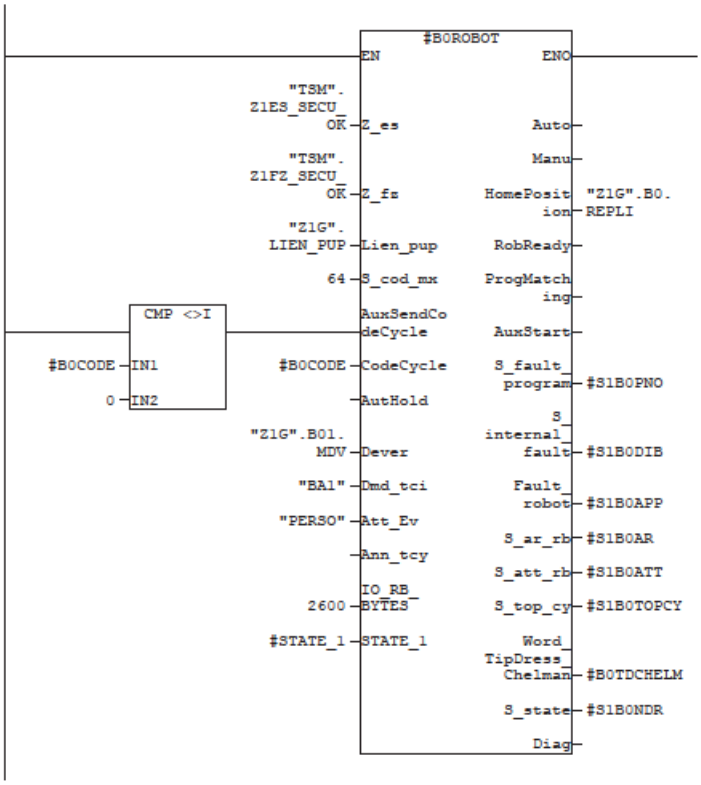


Página de

SIMATIC

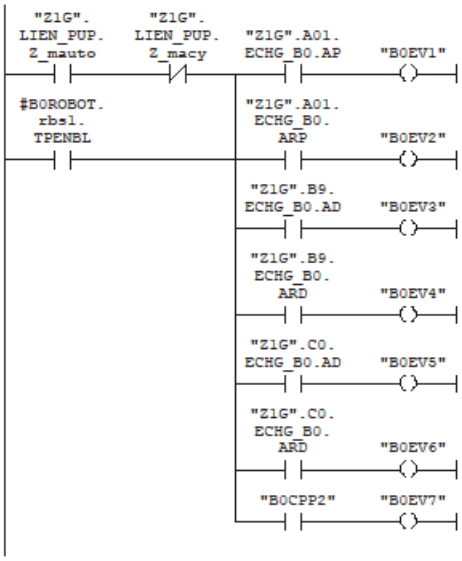
Réseau : 17 FB CONTROL ROBOT / ROBOT CONTROL BF

GESTION BF ROBOT / BF GESTION ROBOT



SIMATIC

Réseau : 18 ////////////////ROBOT EVENTOS////////////////////



Programa de la Garra.

SIMATIC

FB111 - <offline>

"1_B01_GARRA"

Nom : **Famille :**
Auteur : **Version :** 0.1
Version de bloc : 2
Horodatage Code : 26/02/2020 19:47:48
Interface : 26/02/2020 19:47:48
Longueur (bloc/code /données locales) : 02476 01500 00034

Nom	Type de données	Adresse	Valeur initiale	Commentaire
IN		0.0		
OUT		0.0		
IN_OUT		0.0		
STAT		0.0		
A01_TF_TP_B01	BF_TF_TP	0.0		B01: TRANSFERENCIA TABLA PIEZA A01 -> B01
B01PLB	BF_MOUV2P1	128.0		B01: TIPO_ELEMENTO PLB
B01VMA	BF_MOUV2P1	210.0		B01: TIPO_ELEMENTO VMA
B01TDPIEC	TON	292.0		B01 - TEMPORIZACION DEFECTO PIEZA
S1B01DPIEC	Bool	314.0	FALSE	B01 - DEFECTO PIEZA
IHMDPIEC	Int	316.0	0	B01 - DIVERSIDAD PARA DEFECTO PIEZA
S1B01DVMA	Bool	318.0	FALSE	DEFECTO MOVIMIENTOS SRA
S1B01DPLB	Bool	318.1	FALSE	DEFECTO MOVIMIENTOS PLB
TEMP		0.0		
RETVAL	Int	0.0		RETURN VALUE FOR SFC20 / MOT POUBELLE POUR LA FONCTION SFC20
BAMV	Bool	2.0		SEQUENCE MOUVEMENT ADVANCE / AVANCE MOUVEMENT DE LA SEQUENCE
BRMV	Bool	2.1		SEQUENCE MOUVEMENT RETURN / RECUL MOUVEMENT DE LA SEQUENCE

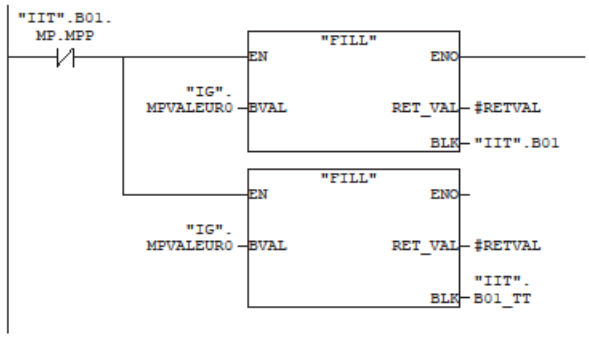
Bloc : FB111 GARRA B01

Página de

SIMATIC

Réseau : 1

Réseau : 2 RAZ IIT



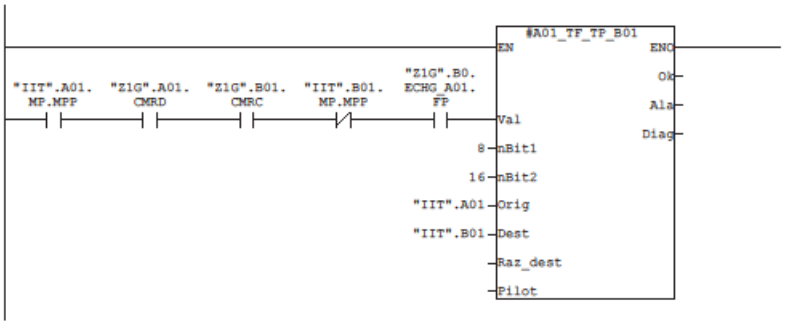
Réseau : 3 ////////////////////// TRANSFERENCIA TABLA PIEZA ////////////////

```
*******************************************************************************
**                          PART TABLE MANAGEMENT                            **
*******************************************************************************
```

Página de

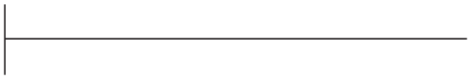
SIMATIC

Réseau : 4 TRANSFERENCIA DE LA TABLA PIEZA: A01 -> B01

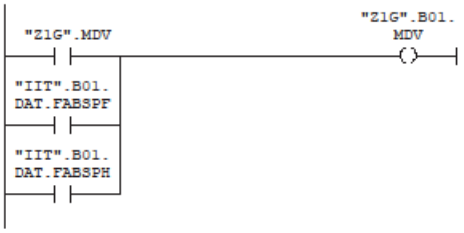


Réseau : 5 ////////////////////// PALABRA PIEZA ////////////////////////////

****
** PART MEMORY **
****



Réseau : 6 MEMORIA MARCHA VACIO MONTAJE



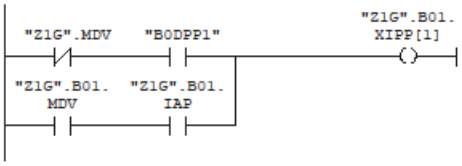
Página de

SIMATIC

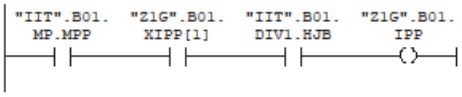
Réseau : 7 ********** IPP INFORMACION / IPP EN EL PUESTO **********

****
** IPP INFORMACION / IPP EN EL PUESTO **
****

Réseau : 8 XIPP HJB



Réseau : 9 IPP

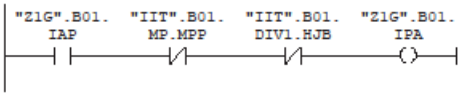


Réseau : 10 IAP

INFORMACION AUSENCIA PIEZA (NINGUN DETECTOR PIEZA ACTIVO)

Réseau : 11 IPA

INFORMACION PIEZA AUSENTE



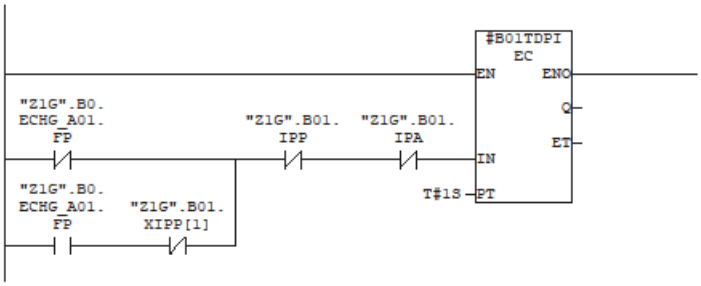
Página de

SIMATIC

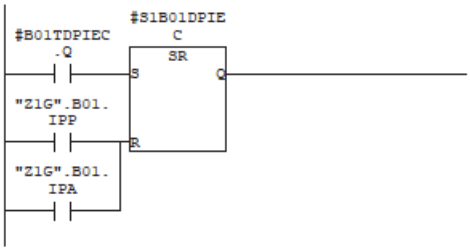
Réseau : 12 //////////////////// GESTION DEFECTO PIEZA ////////////////////

//////////////////// GESTION DEFECTO PIEZA ////////////////////

Réseau : 13 TEMPORIZACION DEFECTO PRESENCIA PIEZA



Réseau : 14 SET RESET - DEFECTO PRESENCIA PIEZA



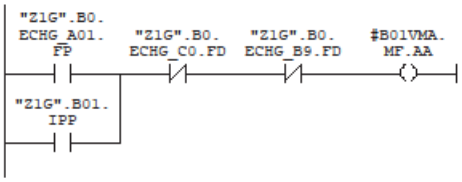
Réseau : 15 ********** VMA VENTOSA 1 **********

Página de

SIMATIC

Réseau : 16 SEGURIDAD DE AVANCE VMA

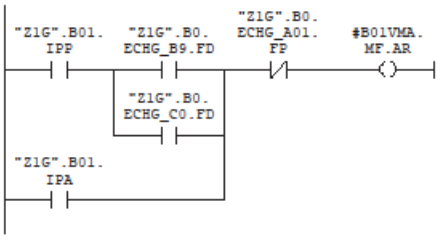
Réseau : 17 AUTORIZACION DE AVANCE VMA



Réseau : 18 INFORMACION DE AVANCE VMA

Réseau : 19 SEGURIDAD DE RETROCESO VMA

Réseau : 20 AUTORIZACION DE RETROCESO VMA

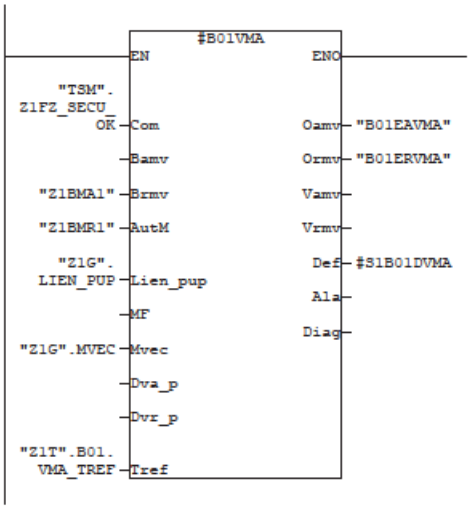


Página de

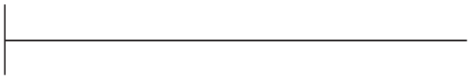
SIMATIC

Réseau : 21 INFORMACION DE RETROCESO VMA

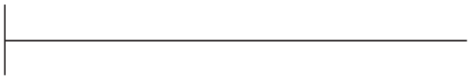
Réseau : 22 BF VMA



Réseau : 23



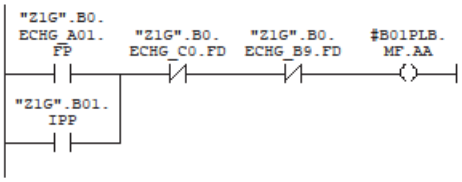
Réseau : 24 ********** plb **********



SIMATIC

Réseau : 25 SEGURIDAD DE AVANCE plb

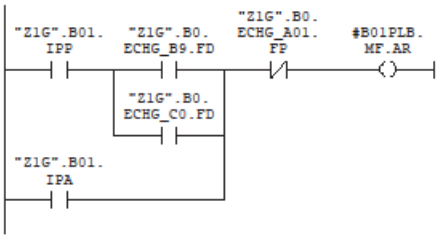
Réseau : 26 AUTORIZACION DE AVANCE plb



Réseau : 27 INFORMACION DE AVANCE plb

Réseau : 28 SEGURIDAD DE RETROCESO plb

Réseau : 29 AUTORIZACION DE RETROCESO plb

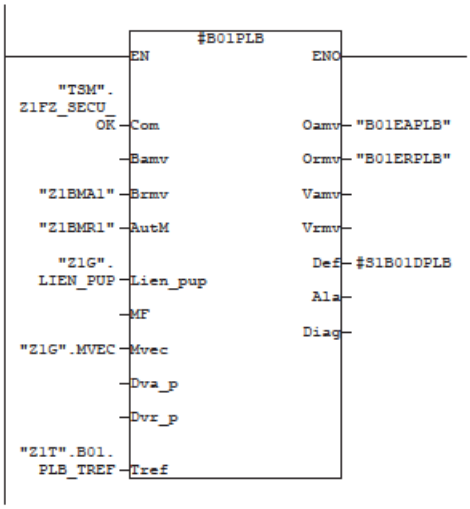


Página de

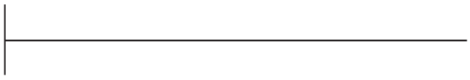
SIMATIC

Réseau : 30 INFORMACION DE RETROCESO plb

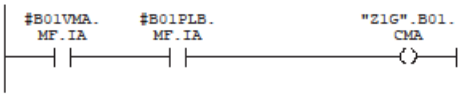
Réseau : 31 BF plb



Réseau : 32 ********** CONDICIONES MECANICAS **********

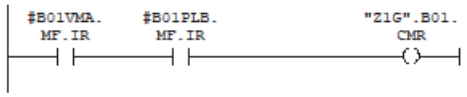


Réseau : 33 CONDICION MAQUINA AVANZADA = CMA

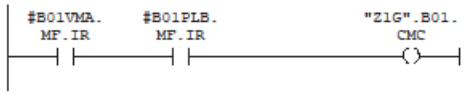


SIMATIC

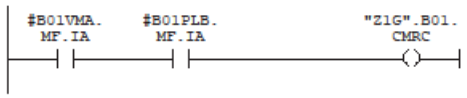
Réseau : 34 CONDICION MAQUINA RETROCEDIDA = CMR



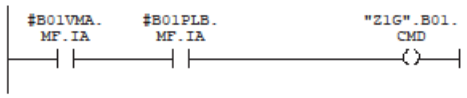
Réseau : 35 CONDICION MAQUINA CARGA = CMC



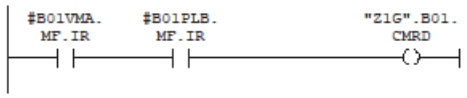
Réseau : 36 CONDICION MAQUINA RETROCESO CARGA = CMRC



Réseau : 37 CONDICION MAQUINA DESCARGA = CMD



Réseau : 38 CONDICION MAQUINA RETROCESO DESCARGA = CMRD



SIMATIC

Réseau : 39 DEFECTO

Página de

Programa del Puesto de Salidas.

SIMATIC

FB112 - <offline>

"1_B9_PES"

Nom : **Famille :**
Auteur : **Version :** 0.1
Version de bloc : 2
Horodatage Code : 26/02/2020 19:48:46
Interface : 26/02/2020 19:48:46
Longueur (bloc/code /données locales) : 01170 00698 00028

Nom	Type de données	Adresse	Valeur initiale	Commentaire
IN		0.0		
OUT		0.0		
IN_OUT		0.0		
STAT		0.0		
B01_TF_TP_B9	BF_TF_TP	0.0		B9 : TRANSFERENCIA TABLA PIEZA B01 -> B9
B9TDFPC	TON	128.0		B9 : TEMPORIZACION DEFECTO PIEZA
S1B9DPIEC	Bool	150.0	FALSE	B9 : DEFECTO PIEZA
S1B9DEF	Bool	150.1	FALSE	B9 : DEFECTO CARRO
IHMDPIEC	Int	152.0	0	B9 : PALABRA GESTION DEFECTO PIEZA
TEMP		0.0		
RETVAL	Int	0.0		RETURN VALUE FOR SFC20 / PALABRA BASURA PARA LA FUNCION SFC20

Bloc : FB112

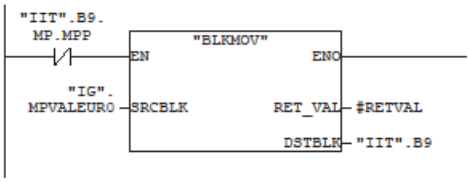
Réseau : 1 ////////////////////// SMP //

```
*******************************************************************************
**                          MONITORING / SEGUIMIENTO                         **
*******************************************************************************
```

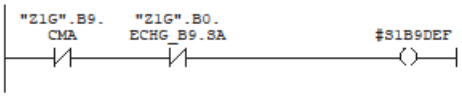
Página de

SIMATIC

Réseau : 2 RAZ IIT

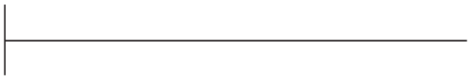


Réseau : 3 DEF

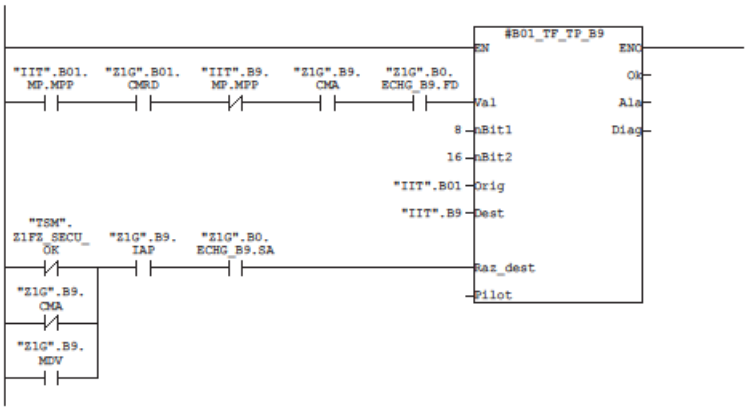


Réseau : 4 //////////////////// MEMORIA PIEZA ////////////////////////////

```
*******************************************************************************
**                                MEMORIA PIEZA                                **
*******************************************************************************
```



Réseau : 5 TRANSFERENCIA PALABRA PIEZA: B01 -> B9

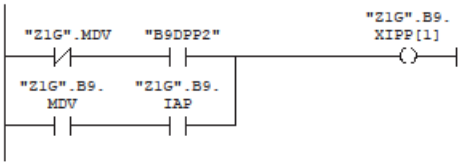


SIMATIC

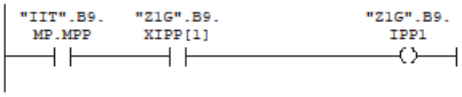
Réseau : 6 //////////////////// IPP INFORMACION / IPP EN EL PUESTO ///////

****
** IPP INFORMACION / IPP EN EL PUESTO **
****

Réseau : 7 XIPP1



Réseau : 8 IPP1



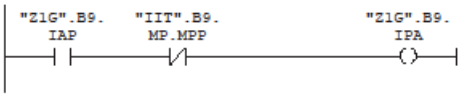
Réseau : 9 IPP

Réseau : 10 IAP

Página de

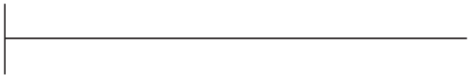
SIMATIC

Réseau : 11 IPA

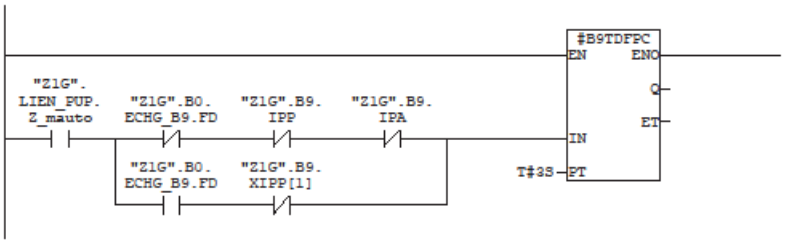


Réseau : 12 //////////////////// DEFECTO PIEZA ////////////////////////////

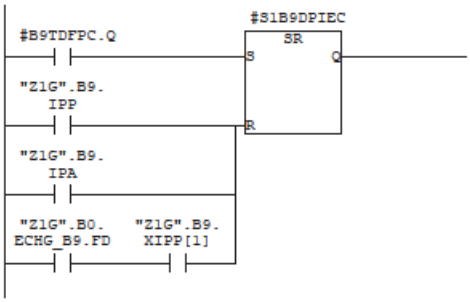
****
** DEFECTO PIEZA **
****



Réseau : 13 TEMPORIZACION DEFECTO PRESENCIA PIEZA



Réseau : 14 DEFECTO PRESENCIA PIEZA



Página de

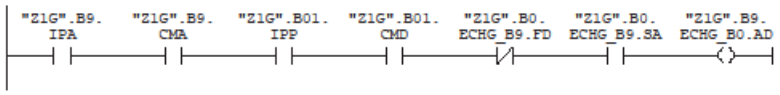
SIMATIC

Réseau : 15 //////////////////// INTERCAMBIOS ////////////////////////////

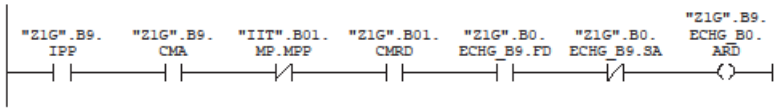
```
*******************************************************************************
**                              INTERCAMBIOS                                 **
*******************************************************************************
```

Réseau : 16 ********** INTERCAMBIOS CON B0 ****************************

Réseau : 17 AUTORIZACION DE DEJADA B0



Réseau : 18 AUTORIZACION DE RETROCESO DEJADA B0



Réseau : 19 //////////////////// CONDICIONES MECANICAS ////////////////////

```
*******************************************************************************
**                            CONDICIONES MECANICAS                          **
*******************************************************************************
```

SIMATIC

Réseau : 20 CONDICION MAQUINA AVANZADA

Réseau : 21 DEFECTO PUESTO E/S

```
**********************************************************************************
**                                  E/S FAULT                                   **
**********************************************************************************
```

Anexo II. Programa del Robot.

```
/PROG  PNS0001
/ATTR
OWNER       = MNEDITOR;
COMMENT     = "";
PROG_SIZE   = 3304;
CREATE      = DATE 20-02-06  TIME 16:16:56;
MODIFIED    = DATE 20-02-18  TIME 15:42:52;
FILE_NAME   = ;
VERSION     = 0;
LINE_COUNT  = 102;
MEMORY_SIZE = 3668;
PROTECT     = READ_WRITE;
TCD:  STACK_SIZE    = 0,
      TASK_PRIORITY = 50,
      TIME_SLICE    = 0,
      BUSY_LAMP_OFF = 0,
      ABORT_REQUEST = 0,
      PAUSE_REQUEST = 0;
DEFAULT_GROUP   = 1,*,*,*,*;
CONTROL_CODE    = 00000000 00000000;
/MN
   1:  !******************************** ;
   2:  !* Trayectoria de:               * ;
   3:  !* COGIDA EN A01 ;
   4:  !******************************** ;
   5:   ;
   6:  UFRAME_NUM=1 ;
   7:  UTOOL_NUM=1 ;
   8:  !Carga payload: PREH_VIDE ;
   9:  PAYLOAD[1] ;
  10:   ;
  11:  !Position de rebouclage ;
  12:J PR[2:P REB] 100% FINE    ;
  13:   ;
  14:  !OR1: SEG. TRASERA A01 ;
  15:  !OR3: SEG. TRASERA B9 ;
  16:  CALL ORD_PLC('1010000000000000') ;
  17:   ;
  18:  !EV1: AUT.COGER A01 ;
  19:  CALL EVT_PLC('1000000000000000') ;
  20:   ;
  21:  !OR1: RAZ SEG. TRASERA A01 ;
  22:  !OR3: SEG. TRASERA B9 ;
  23:  CALL ORD_PLC('0010000000000000') ;
  24:   ;
  25:J P[1] 100% CNT100    ;
  26:L P[2] 300mm/sec CNT20    ;
  27:L P[3:PT COGIDA] 100mm/sec FINE    ;
  28:   ;
  29:  !OR1: RAZ SEG. TRASERA A01 ;
  30:  !OR2: FIN COGIDA A01 ;
  31:  !OR3: SEG. TRASERA B9 ;
  32:  CALL ORD_PLC('0110000000000000') ;
  33:   ;
  34:  !EV2: AUT.RETOCESO COGIDA A01 ;
  35:  CALL EVT_PLC('0100000000000000') ;
  36:   ;
  37:  !OR1: RAZ SEG. TRASERA A01 ;
  38:  !OR2: FIN COGIDA A01 ;
  39:  !OR3: SEG. TRASERA B9 ;
  40:  CALL ORD_PLC('0010000000000000') ;
  41:   ;
  42:  !Carga payload: PREH PLEIN ;
  43:  PAYLOAD[2] ;
  44:   ;
  45:L P[4] 100mm/sec FINE    ;
  46:J P[5:P_FIN_PR] 100% FINE    ;
  47:   ;
  48:  !OR1: SEG. TRASERA A01 ;
  49:  !OR3: SEG. TRASERA B9 ;
  50:  !OR5: SEG. TRASERA C0 ;
```

```
   51:  CALL ORD_PLC('1010100000000000') ;
   52:   ;
   53:  !******************************** ;
   54:  !* Trayectoria de:              * ;
   55:  !* DEJADA EN B9 ;
   56:  !******************************** ;
   57:   ;
   58:  UFRAME_NUM=2 ;
   59:  UTOOL_NUM=1 ;
   60:  !Carga payload: PREH_VIDE ;
   61:  PAYLOAD[1] ;
   62:   ;
   63:J P[6:P DB DP B9] 10% CNT100    ;
   64:   ;
   65:  !OR1: SEG. TRASERA A01 ;
   66:  !OR3: SEG. TRASERA B9 ;
   67:  CALL ORD_PLC('1010000000000000') ;
   68:   ;
   69:  !EV3: AUT. DEJADA B9 ;
   70:  CALL EVT_PLC('0010000000000000') ;
   71:   ;
   72:  !OR1: SEG. TRASERA A01 ;
   73:  !OR3: RAZ SEG. TRASERA B9 ;
   74:  CALL ORD_PLC('1000000000000000') ;
   75:   ;
   76:J P[7] 50% CNT100    ;
   77:L P[8] 300mm/sec CNT20    ;
   78:L P[9:PT DEJADA] 100mm/sec FINE    ;
   79:   ;
   80:  !Carga payload: PREH VIDE ;
   81:  PAYLOAD[1] ;
   82:   ;
   83:  !OR1: SEG. TRASERA A01 ;
   84:  !OR3: RAZ SEG. TRASERA B9 ;
   85:  !OR4: FIN DE DEJADA  B9 ;
   86:  CALL ORD_PLC('1001000000000000') ;
   87:   ;
   88:  !EV4: AUT. RETROCESO DEJADA B9 ;
   89:  CALL EVT_PLC('0001000000000000') ;
   90:   ;
   91:  !OR1: SEG. TRASERA A01 ;
   92:  !OR3: RAZ SEG. TRASERA B9 ;
   93:  !OR4: RAZ FIN DE DEJADA  B9 ;
   94:  CALL ORD_PLC('1000000000000000') ;
   95:   ;
   96:L P[10] 100mm/sec FINE    ;
   97:J PR[2:P_REB] 60% FINE    ;
   98:   ;
   99:  !OR1: SEG. TRASERA A01 ;
  100:  !OR3: SEG. TRASERA B9 ;
  101:  !OR5: SEG. TRASERA C0 ;
  102:  CALL ORD_PLC('1010100000000000') ;
/POS
P[1]{
   GP1:
     UF : 1, UT : 1,
     J1=   -75.664 deg,  J2=   -33.757 deg,  J3=    -6.776 deg,
     J4=   -45.042 deg,  J5=    14.194 deg,  J6=   223.807 deg
};
P[2]{
   GP1:
     UF : 1, UT : 1,     CONFIG : 'F U T, 0, 0, 1',
     X =    -8.277  mm,  Y =    -4.089  mm,  Z =  -195.529  mm,
     W =     -.834 deg,  P =     2.803 deg,  R =      .598 deg
};
P[3:"PT_COGIDA"]{
   GP1:
     UF : 1, UT : 1,     CONFIG : 'F U T, 0, 0, 1',
     X =     1.021  mm,  Y =    -1.207  mm,  Z =    -4.969  mm,
     W =     -.836 deg,  P =     4.000 deg,  R =      .580 deg
};
```

```
P[4]{
   GP1:
    UF : 1, UT : 1,
    J1=   -75.525 deg,  J2=   -32.291 deg,  J3=   -31.284 deg,
    J4=   -18.087 deg,  J5=    32.077 deg,  J6=   194.364 deg
};
P[5:"P_FIN_PR"]{
   GP1:
    UF : 1, UT : 1,     CONFIG : 'F U T, 0, 0, 1',
    X =  -675.861  mm,  Y =  -353.972  mm,  Z =  -635.034  mm,
    W =    -2.859 deg,  P =    22.570 deg,  R =   -61.510 deg
};
P[6:"P DB DP B9"]{
   GP1:
    UF : 2, UT : 1,     CONFIG : 'F U T, 0, 0, 1',
    X =  -235.417  mm,  Y =   128.366  mm,  Z =  -636.271  mm,
    W =     2.029 deg,  P =    23.528 deg,  R =    32.167 deg
};
P[7]{
   GP1:
    UF : 2, UT : 1,     CONFIG : 'F U T, 0, 0, 0',
    X =     -.960  mm,  Y =    -7.388  mm,  Z =  -250.549  mm,
    W =     3.259 deg,  P =      .154 deg,  R =      .189 deg
};
P[8]{
   GP1:
    UF : 2, UT : 1,     CONFIG : 'F U T, 0, 0, -1',
    X =     -.510  mm,  Y =    -5.051  mm,  Z =   -84.386  mm,
    W =     1.227 deg,  P =      .157 deg,  R =      .187 deg
};
P[9:"PT_DEJADA"]{
   GP1:
    UF : 2, UT : 1,     CONFIG : 'F U T, 0, 0, -1',
    X =     -.286  mm,  Y =    -6.718  mm,  Z =    -6.283  mm,
    W =     1.226 deg,  P =      .157 deg,  R =      .188 deg
};
P[10]{
   GP1:
    UF : 2, UT : 1,     CONFIG : 'F U T, 0, 0, 0',
    X =     -.868  mm,  Y =    -2.303  mm,  Z =  -212.635  mm,
    W =     1.226 deg,  P =      .157 deg,  R =      .188 deg
};
/END
```

Anexo III. Presupuesto.

Apdo.	Concepto	Ud.	€	Total
1	**Montajes Elementos Electronicos**			
1.1	Armario de Zona	1	2.400,00 €	2.400,00 €
1.2	Armario de Operador	1	1.500,00 €	1.500,00 €
1.3	Montaje Festo Operador	1	1.200,00 €	1.200,00 €
1.4	Montaje Festo Garra	1	850,00 €	850,00 €
1.5	Puertas de Seguridad Euchner	1	1.450,00 €	1.450,00 €
1.6	Barreras de Seguridad SICK	1	850,50 €	850,50 €
1.7	Robot Fanuc : Fanuc M10iA/12	1	16.000,00 €	16.000,00 €
1.8	Horas de Electricista.	50	20,00 €	1.000,00 €
2	**Montajes Elementos Electronicos**			
2.1	Materiales Montaje Operador	1	150,00 €	150,00 €
2.2	Materiales Montaje Puesto de Salida	1	80,00 €	80,00 €
2.3	Horas de Mecánico.	40	30,00 €	1.200,00 €
	Total: Intalación y Montaje de Elecmentos Elec y Mec.			26.680,50 €
3	**Programación de la Instalación**			
3.1	Horas de Ingeniero Automatista.	60	50,00 €	3.000,00 €
3.2	Horas de Ingeniero Robótico.	40	50,00 €	2.000,00 €
3.3	Horas Documentación de la Instalación.	100	50,00 €	5.000,00 €
4	**Equipos y Programas**			
4.1	Ordenador Dell Inspiron 15 500	1	1.500,00 €	1.500,00 €
4.2	SIMATIC STEP 7 Basic V15 SP 5.5	1	2.534,00 €	2.534,00 €
	Total: Programación Autómata y Robótico y Programas Informaticos.			14.034,00 €
	Total Proyecto:			40.714,50 €