CURSO SYSMAC STUDIO

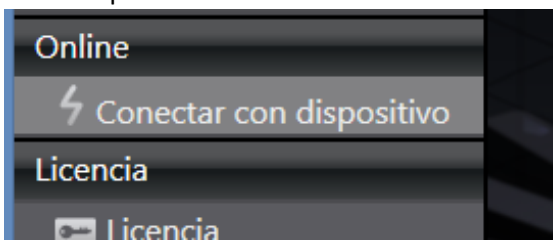
I. Introducción. Generalidades
II. Primeros pasos
III. Empezando a programar
IV. Funciones
V. Servomotores
VI. Actualización de firmware y Backup

I. Introducción. Generalidades:

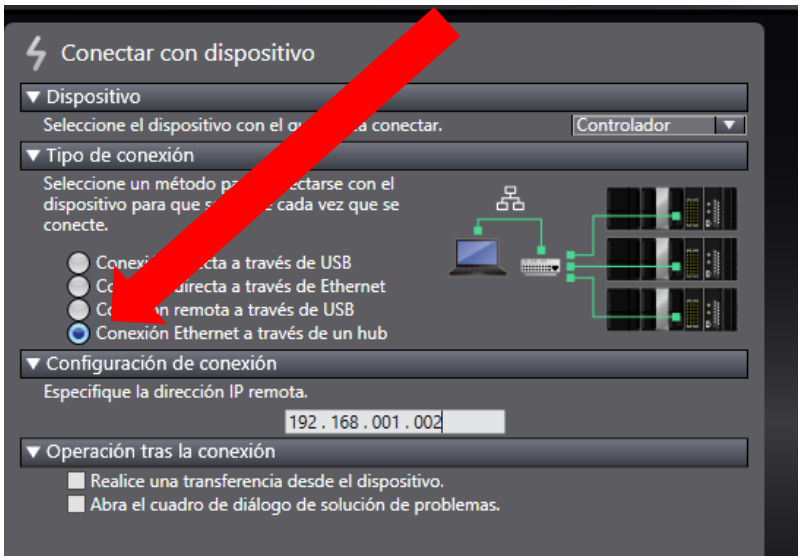
- NJ es compatible con CJ. Lleva el bus compatible, el conector lateral típico. Le sirven sus módulos de E/S. La CPU NX no es compatible.
- El número de servos que vamos a controlar como si fueran un eje es el patrón diferenciador de escoger una CPU u otra. La diferencia no es la memoria de programa. En memoria de programa todas las CPU NJ y NX van sobradas.
- Las CPU NJ y NX no son un simple PLC, tienen instrucciones de motion (ejes), de base de datos (DB connection), de robótica (cinemática inversa para robots, la CPU coordina todos los ejes)…
- El refresco de datos de la cabecera Ethercat en NX/NJ es más rápido que el protocolo que usa Omron en las CPU CJ2 (PCI 21). Esto permite más precisión en el movimiento de servomotores. Al tener una rápida actualización en EtherCat, tenemos buena resolución en el movimiento.
- Por defecto el Sysmac guarda los archivos en una ruta propia. No tenemos “Abrir” proyecto/”Guardar como”, tenemos “Importar”/”Exportar”.
- Se pueden arrancar programas desde la SD sin necesitad de cargarlos en la CPU. Para ello el Switch 2 debe estar a ON. Y debemos cargar el programa en el directorio /autoload/ en donde se encuentra el archivo “NJBackup.dat” .

II. Primeros pasos:

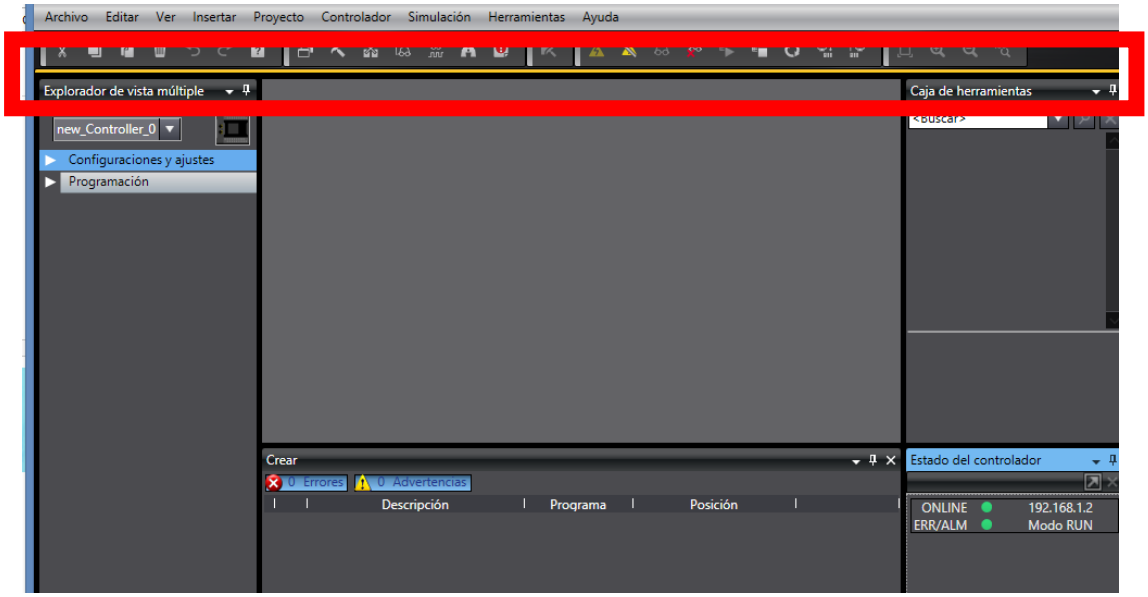
1. Seleccionar “conectar con dispositivo”.



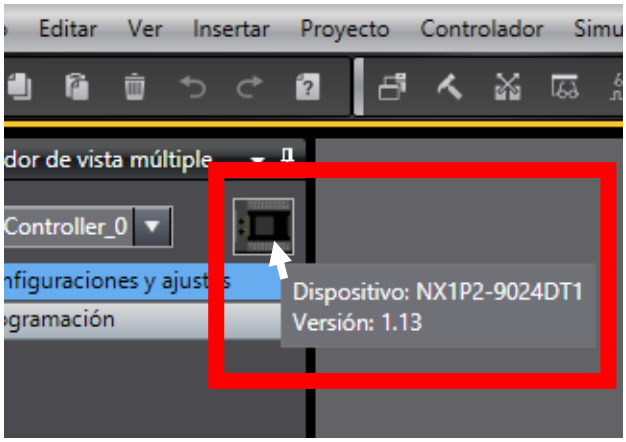
Debemos elegir si conectarnos por USB o por ETHERNET. Si nos conectamos por Ethernet recomendamos usar el tipo de conexión “HUB” y especificar la dirección IP. Si escogemos conexión directa por Ethernet nos ahorramos poner la IP pero a veces falla.



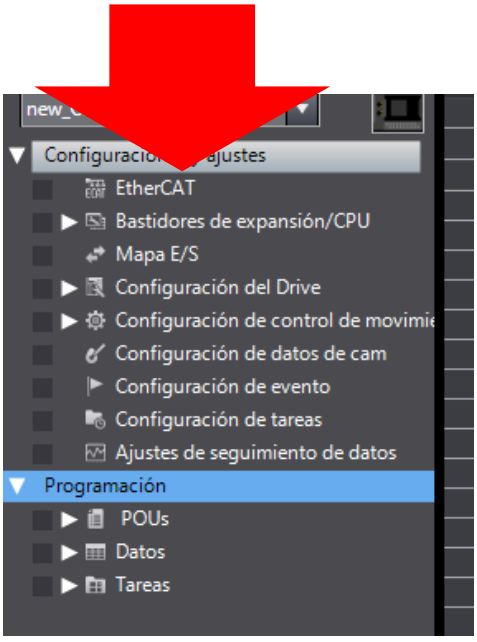
Detectará el modelo de CPU y versión y se conectará automáticamente.
Una vez conectado se quedará en amarillo una línea horizontal indicando que estamos online.



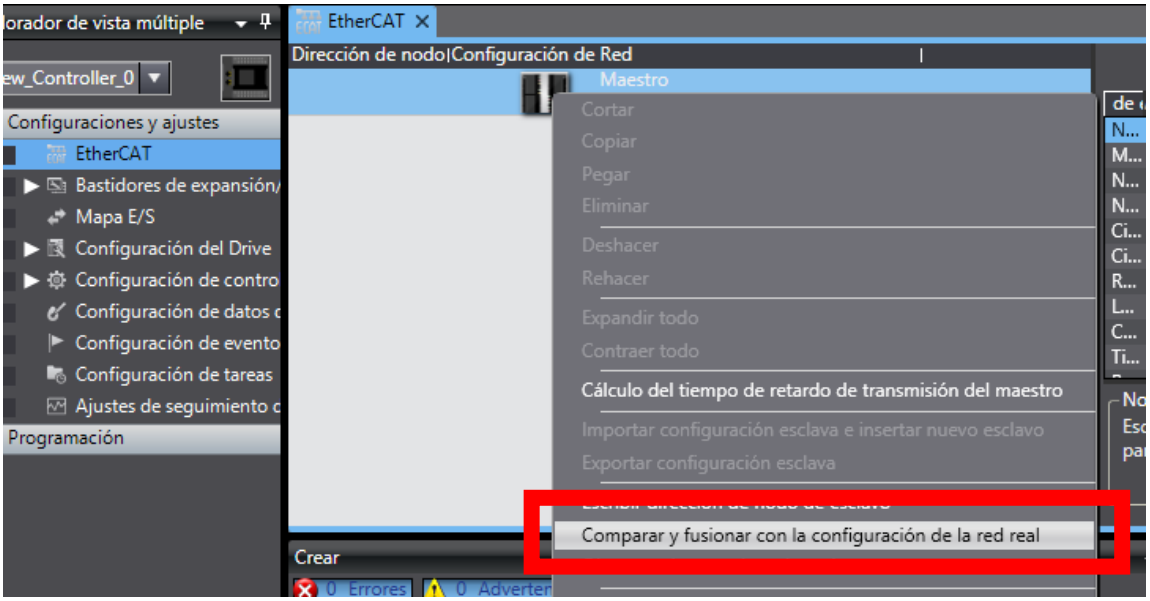
Ahora si ponemos el cursor encima de la CPU, tal y como se ve en la imagen siguiente, podremos ver el dispositivo exacto y versión.



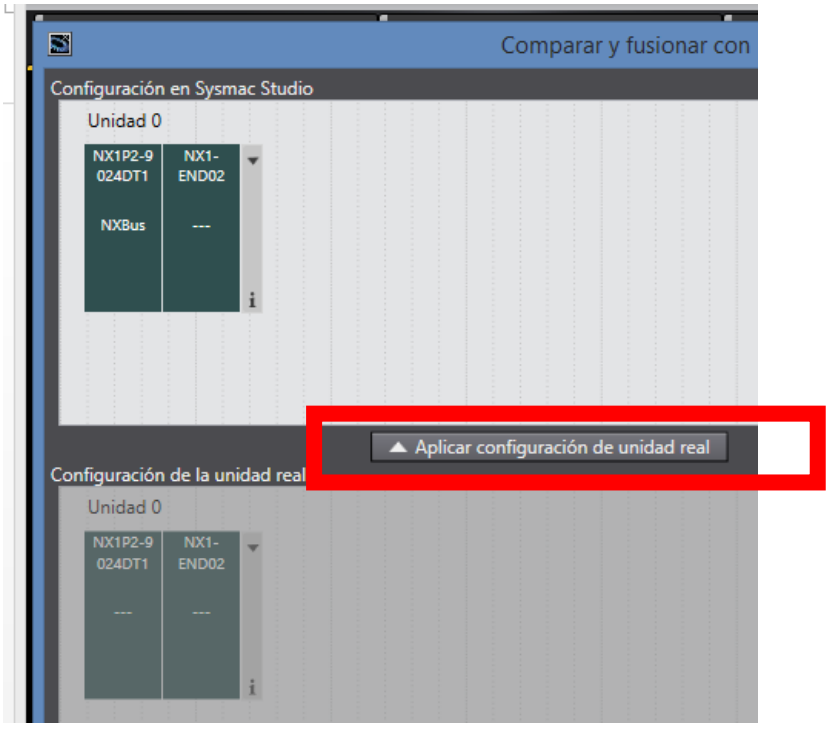
2. Los siguientes pasos los dedicaremos a configurar la CPU. Sysmac studio está pensado para hacer la configuración siguiendo el orden de arriba hacia abajo. Así que primero comenzamos a configurar las comunicaciones EtherCAT.



3. Configuramos Ethercat. Primero tendremos que poner físicamente el número de nodo en cada dispositivo que tengamos conectado. Ya sea una cabecera de E/S distribuida, un servo o un variador, tendremos que asignarle un número de nodo con unos interruptores rotatorios.
   El árbol con dispositivos y nodos se puede hacer manual o detectarlo automáticamente. Para que detecte automáticamente la red EtherCAT, hay que darle doble click a “EtherCAT”. Y luego botón derecho en la CPU “Comparar y fusionar con la configuración de la red real” anatorregrosa@inemur.com

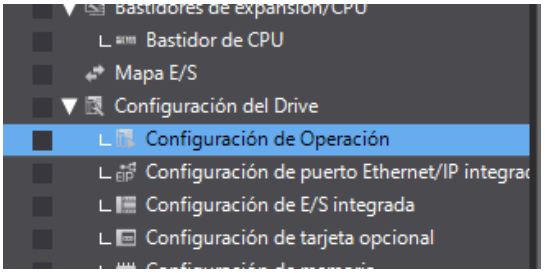


4. Ahora vamos con los "Bastidores de expansión CJ". Aquí hacemos lo mismo. Clicamos dos veces. Y una vez dentro, en el dibujo de la CPU le damos botón derecho "Comparar y fusionar con la configuración de la red real". Y una vez dentro le damos a "Aplicar configuración de unidad real"

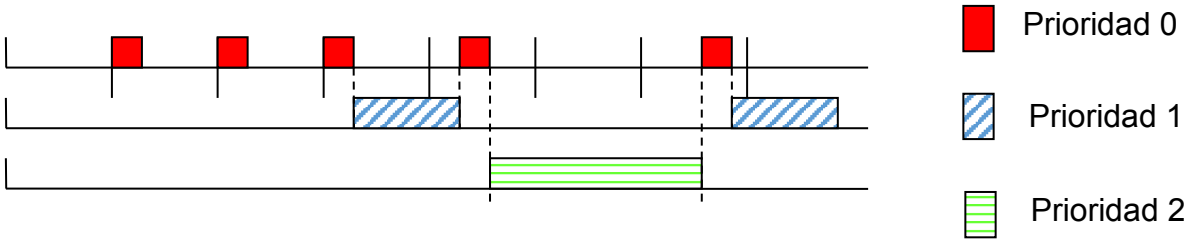


5. Clicamos en "Mapa de E/S". Veremos los nodos que tenemos en la red, servos, cabeceras, tarjetas de cada cabecera, etc. En las CPU de Sysmac la zona de memoria es fija, no hay que asignar una dirección. Sólo tenemos que ir ligando etiquetas a las entradas/salidas físicas.
   Podemos ver todas las variables que hemos asignado en "Programación" → "Datos" → "Variables globales".

6. Ahora abrimos el árbol de “Configuración del drive” → “Configuración de operación”.

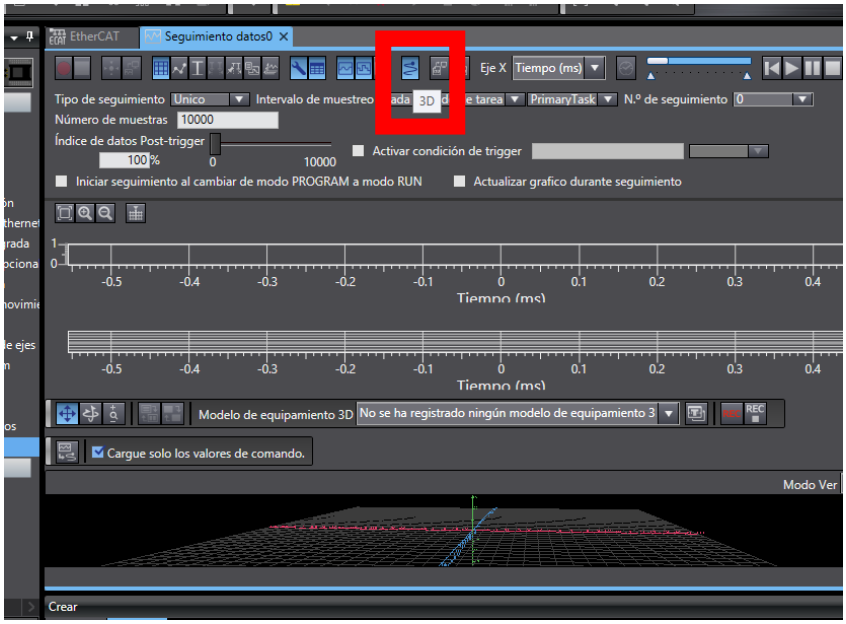


   Aquí podemos configurar el modo de arranque de la CPU. Por defecto lo dejaremos en modo RUN, para que al arrancar se ponga en marcha.
7. En “Configuración del puerto Ethernet” ponemos la IP que va a tener la CPU.
8. Ahora vendría“Configuración de control de movimiento” pero esto lo dejaremos para más adelante.
9. Tenemos después “Configuración de evento”. Esto se utiliza si queremos meter el log de fallos internos de la CPU en el histórico de alarmas del HMI. Esto nunca se suele utilizar así que lo vamos a obviar.
10. Entramos en “Configuración de tareas”. Aquí tenemos que crear tareas, asociar prioridades de tiempo/eventos y luego indicar qué programas se van a ejecutar en cada tarea. A menor número mayor prioridad.
    Al poder gestionar los hilos de ejecución permitimos el tiempo real.
    Una tarea con mayor prioridad puede llegar a interrumpir las de menor prioridad. Y estas sólo se ejecutarán en los intervalos en los que la prioritaria no actúa.



   Este concepto de alternar tareas permite ejecuciones en tiempo real y es necesario para aplicaciones de Motion, donde es fundamental la rapidez para tener precisión. Esto permite asignar tareas de 1ms prioritarias en donde en ese intervalo leemos la posición del encoder y se ejecuta el envío de la información por EtherCAT.

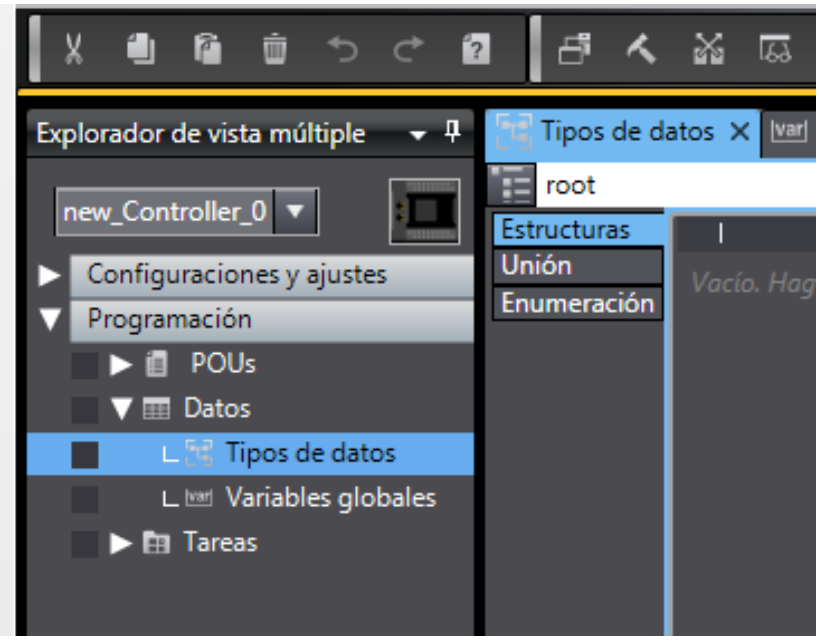
11. En “Ajuste de seguimiento de datos” si le damos al botón derecho pordremos añadir un seguimiento de datos. En esta pantalla podemos monitorizar datos de los servos (torque, intensidad…) para poder hacer ajustes durante la puesta en marcha.
    Podemos representar una vista en 3D y ver el movimiento real de los ejes.



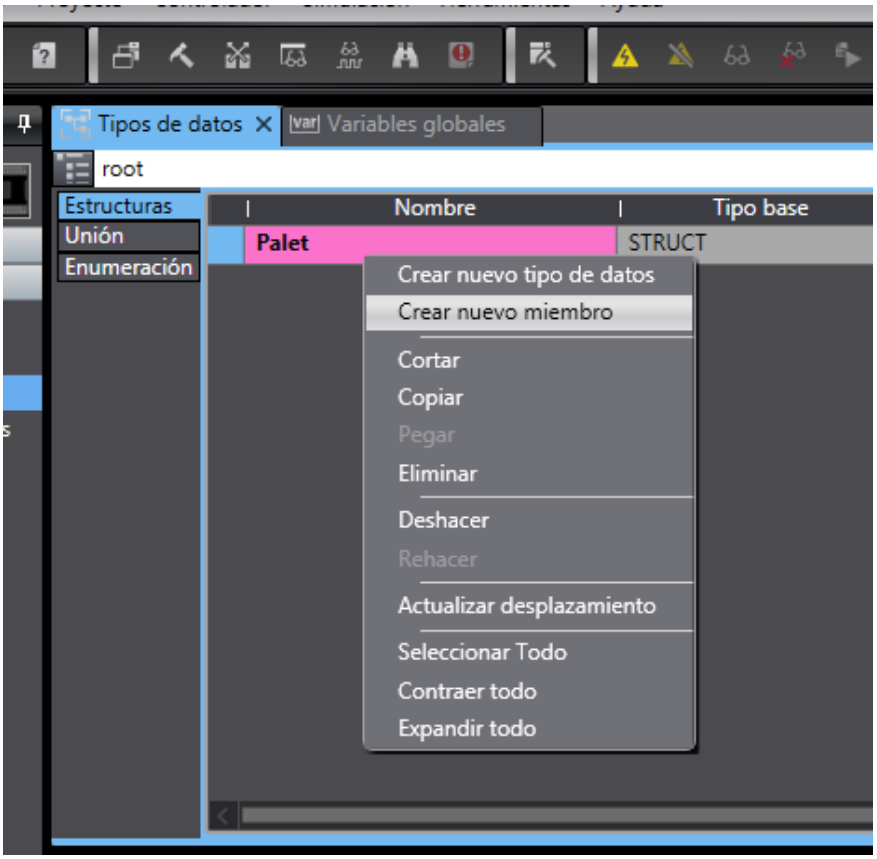
III. Empezando a programar:

1. Para comenzar a programar vamos a "Programación"→"POU's"→"Programas"→"Programa 0"→"Sección 0"
2. Si le damos a "Ayuda"→"Referencia de asignaciones de teclado" podremos ver cómo están mapeadas las teclas. A diferencia de CX-Programmer el mapeado no se puede modificar/personalizar.
   Las teclas más importantes son:
   1. C para crear un contacto NA
   2. / para crear un contacto NC
   3. O para poner una bobina
   4. R para crear una nueva línea
   5. F para introducir una función
   6. I para introducir una instrucción

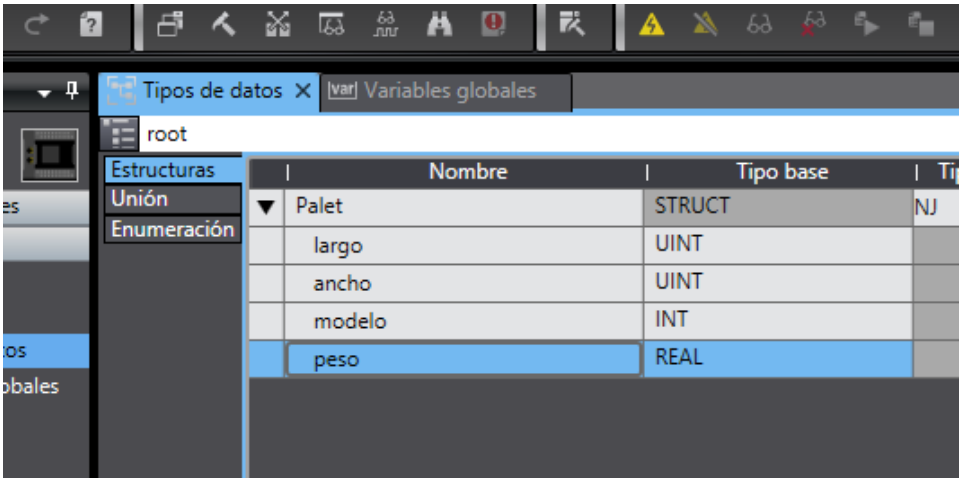
3. En el programa podremos añadir variables globales (se representan en rojo) y variables locales (se representan en negro). Para crear una nueva variable local basta con introducir una etiqueta que no exista y la tomará como variable local.
4. Cuando tengamos que declarar muchas variables del mismo tipo de golpe podemos usar vectores. Por ejemplo, si declaramos una variable con nombre PILOTO y tipo de dato BOOL[1000], habremos generado de una tacada 1000 variables de tipo booleano. Tendremos que hacer referencia a ellas de la siguiente manera: PILOTO[numero]. Esto tiene la ventaja que podemos recorrer todo el vector con una variable que actúe de puntero. Según valga "numero" activamos un PILOTO u otro. Es decir podemos hacer direccionamiento indirecto con arrays.
5. Podemos crear una estructura de datos de la siguiente manera. Nos vamos a Datos → Tipos de datos → Estructuras.



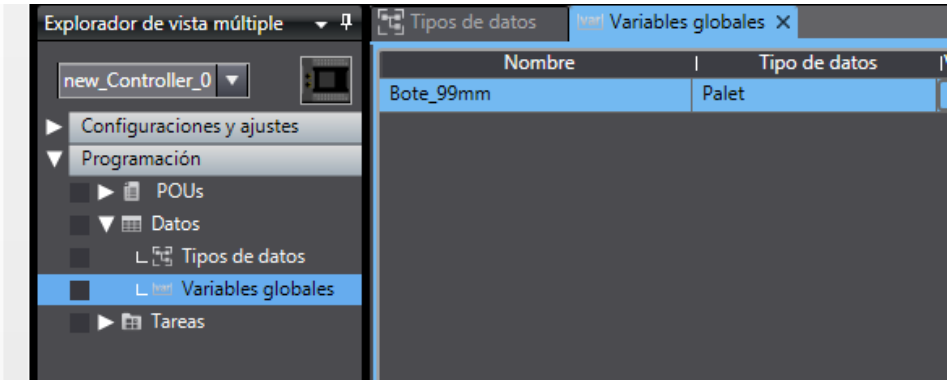
Clicamos dos veces e introducimos el nombre de la estructura. En este caso se llamará "Palet". A continuación le damos botón derecho → Crear nuevo miembro.



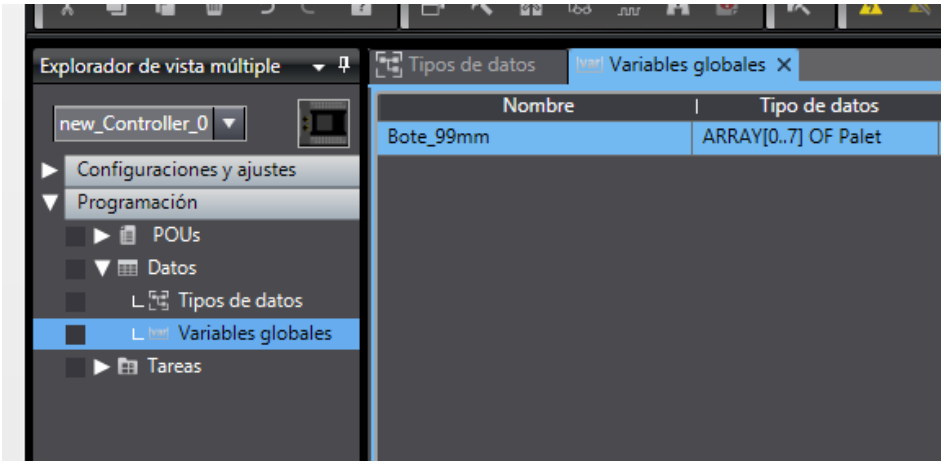
Y vamos añadiendo todos los tipos de datos que contendrá nuestra estructura. Para este ejemplo el tipo de datos "Palet" contendrá los datos: "largo", "ancho", "modelo" y "peso".



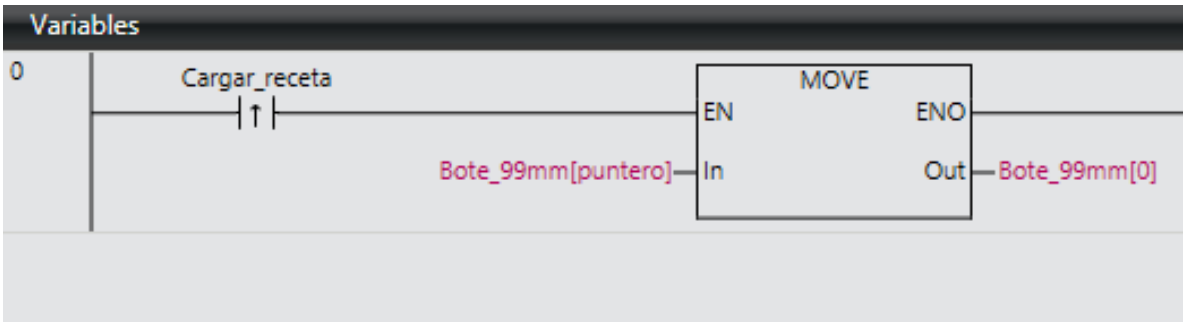
Para hacer uso de esta estructura podemos crear una variable que se llame "Bote_99mm", en donde el tipo de dato es "Palet". Para ello nos vamos "variables globales" y lo creamos de la siguiente manera:



Pero… también podemos crear un array de tamaño 8 para crear de una tacada 8 variables, y así poder acceder a cada una a través de un puntero. Una aplicación que tiene esto es para las recetas y formatos. Con esta técnica organizamos el cambio de receta rápidamente.

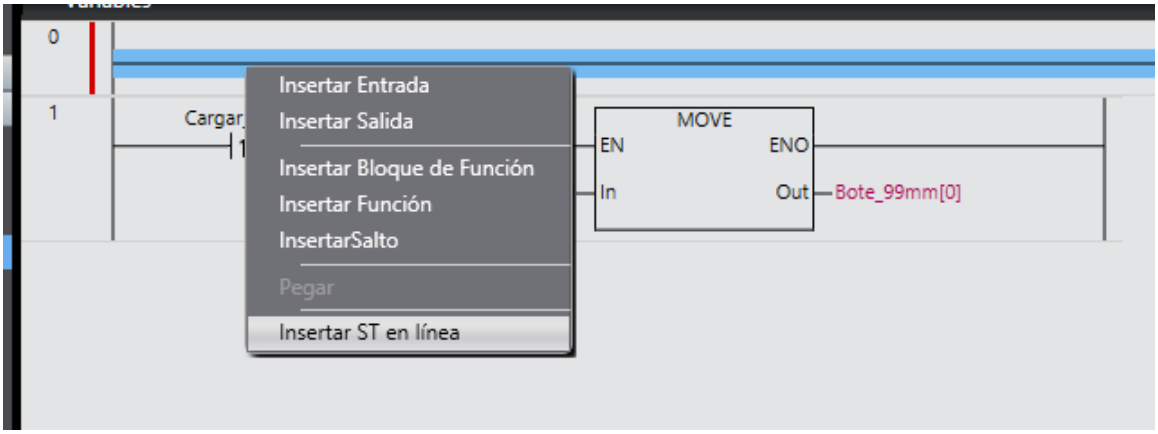


Una forma de trabajar es guardar la posición 0 del vector para usarla como variable de trabajo. Como receta activa. Y en ella cargamos la receta que queramos cambiar.
En el programa queda de la siguiente manera:

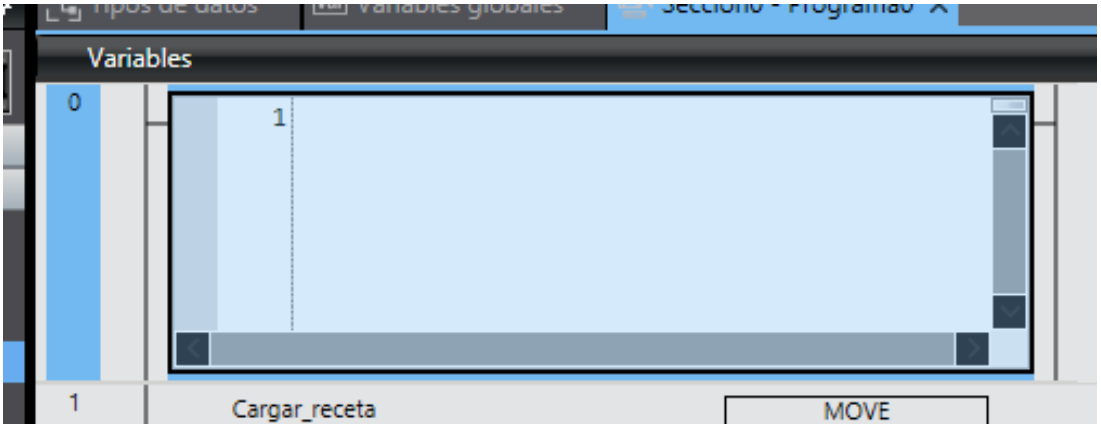


Así cuando queramos cargar una nueva receta nueva, sólo tendremos que cambiar el valor de “puntero” en el HMI. Y después activamos el bit “Cargar_receta”. De esta manera tan simple haríamos el cambio de receta/formato.

6. Ahora vamos a explicar el uso de la programación en texto estructurado ST. En un programa de diagrama de contactos podemos introducir bloques de programación estructurada de la siguiente manera: Nos situamos encima de una línea de programa, le damos botón derecho → "insertar ST en línea".

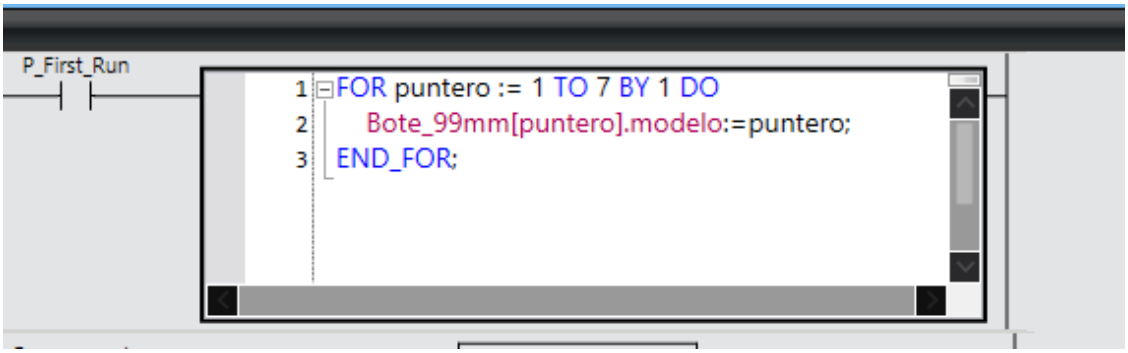


Se nos crea un bloque donde podemos escribir instrucciones en texto estructurado.

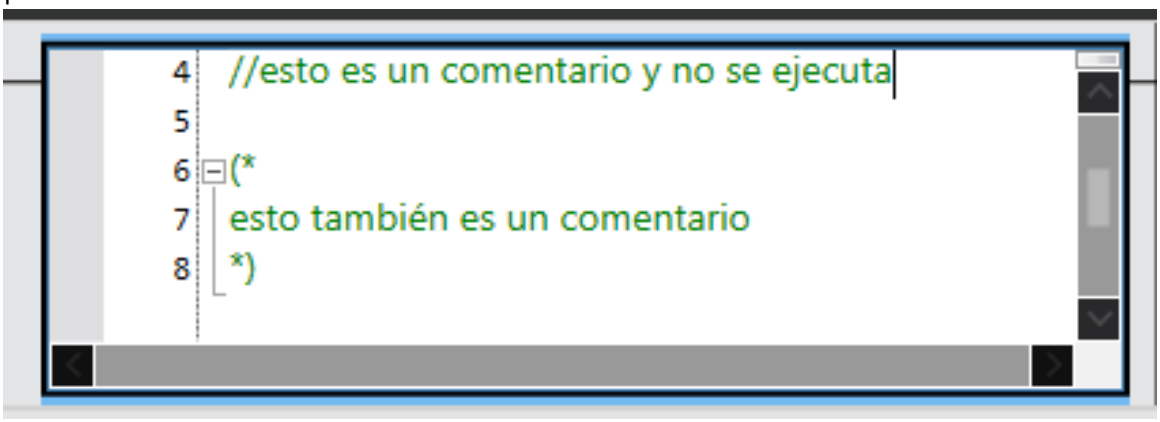


El texto estructurado es muy útil para inicializar variables, hacer operaciones matemáticas… que de otra manera sería muy tedioso hacer el diagramas de contactos.

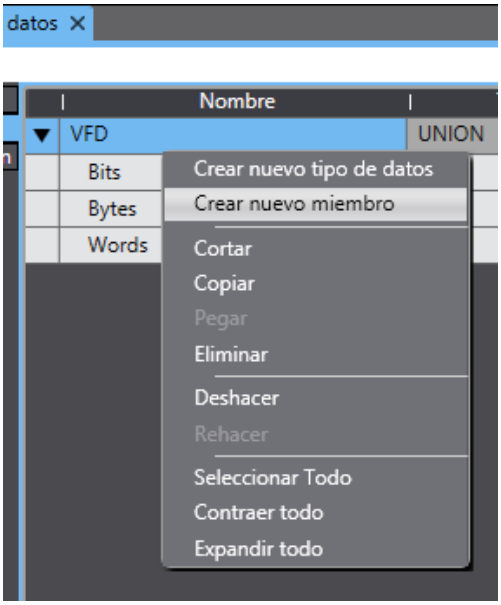
Por ejemplo, podemos inicializar arrays en el primer ciclo de scan con un bucle FOR de la siguiente manera.



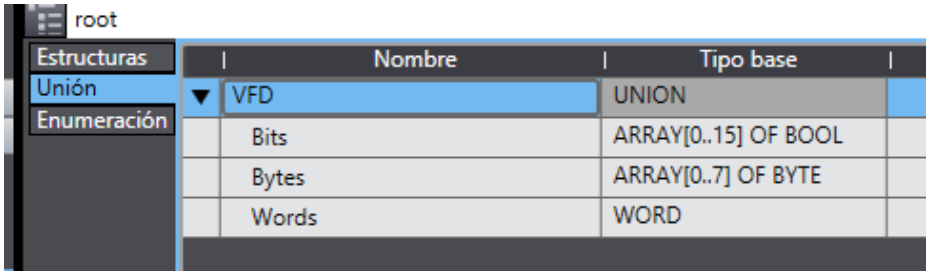
Siguiendo con el texto estructurado, podemos comentar líneas para que no se ejecuten haciendo uso de // ó (* *). Esto es muy útil cuando estamos haciendo pruebas.



7. Tipos de dato UNION. Permiten ver una zona de memoria con diferentes formatos. En CX-Programmer si teníamos una variable tipo WORD en la zona de memoria W100 y queríamos acceder a cada uno de sus bits individualmente podíamos hacerlo accediendo a W100.XX . Esto en Sysmac no es posible porque no tenemos acceso a las direcciones de memoria. Trabajamos con etiquetas y a las direcciones no podemos acceder. Para poder acceder a diferentes bits de una variable Word, la manera de hacerlo es creando un tipo de dato que Omron llama UNION.
   Este tipo de dato Union se usa mucho para leer el estado de los variadores de frecuencia, veamos un ejemplo:
   1. Primero nos creamos el tipo de datos Union. Para ello nos vamos a “Tipos de datos” y debajo de “Estructuras” entramos en “Unión”. En este ejemplo vamos a crear un tipo de datos unión que llamará VFD.
   2. Cremos nuevos miembros. Botón derecho→”Crear nuevo miembro”

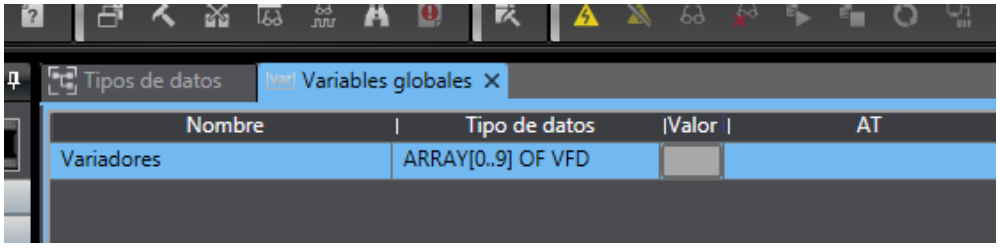


Aquí insertamos miembros de tipo Booleano, Byte y Word tal y como vemos en la siguiente imagen. La variable principal queremos que sea tipo Word, por eso hacemos un array de 16 bits y un array de 8 bytes.

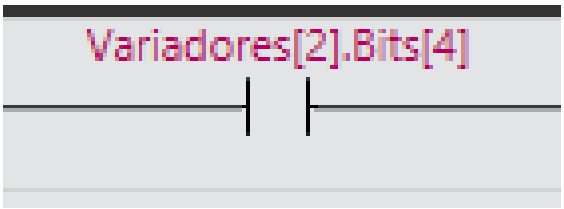


Como vemos, la forma se asemeja mucho al tipo de datos Estructura visto anteriormente. Realmente, una Union es una Estructura con la peculiaridad que cada miembro apunta a la misma dirección de memoria.

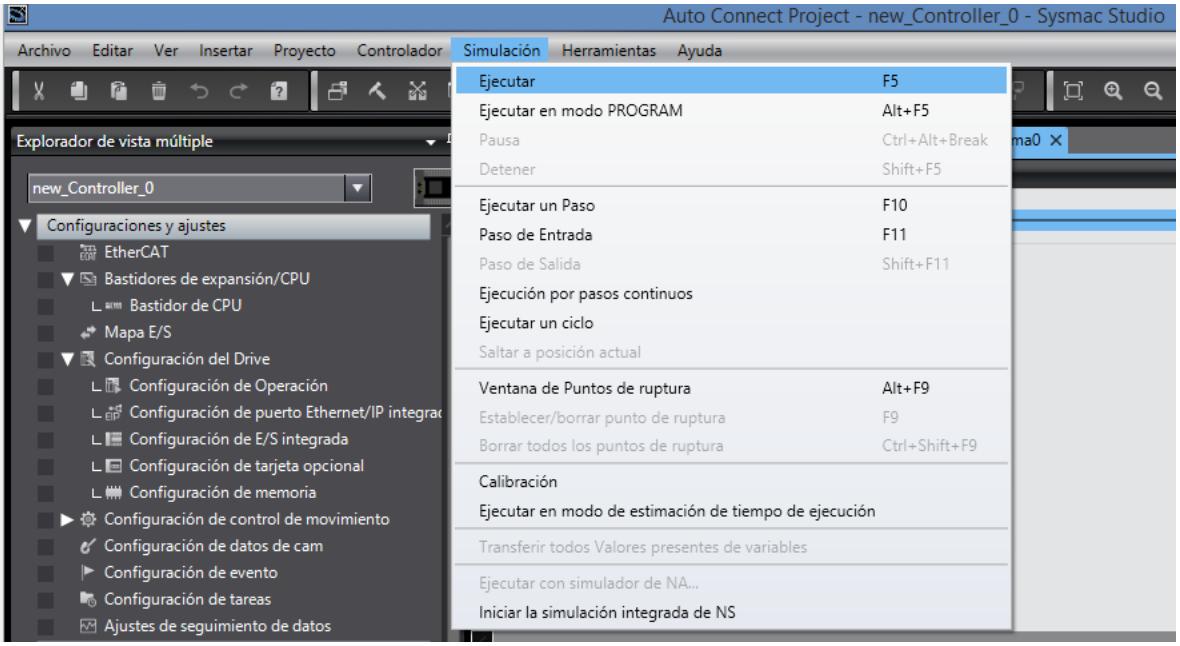
3. Ahora creamos una variable global que se llame “Variadores” y en tipo de datos ponemos “VFD[10]”. Se nos quedará algo de la siguiente manera. Lo que hemos hecho ha sifo crear 10 variables de una tacada para leer el estado de los variadores de frecuencia.

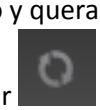


4. Ahora podremos acceder a cada variable de cada variador y leerlo en formato Word, Byte o Bit. Por ejemplo, si queremos acceder al bit 4 del variador 2 se hará de la siguiente manera.

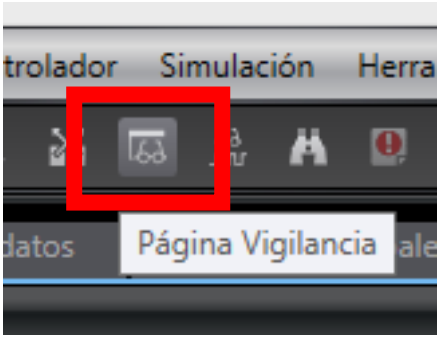


8. Para simular un programa simplemente le damos a “Simulación” → “Ejecutar”

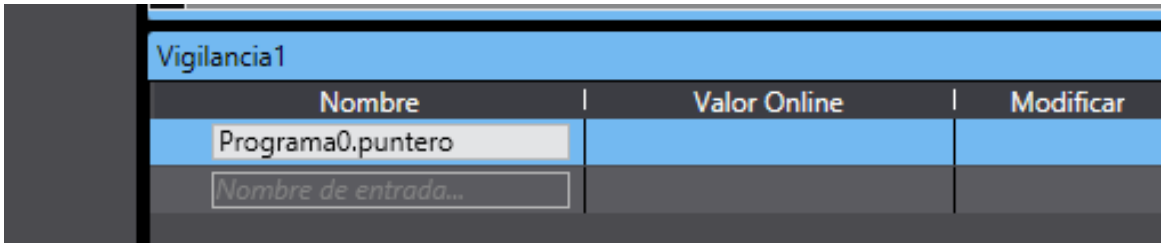


9. Una vez tengamos un programa hecho y queramos transferirlo, para compararlo con el que tiene la CPU le damos a sincronizar  .

10. Una vez hemos cargado el programa en la CPU y estamos en modo PROGRAM o estamos simulando, podemos visualizar los valores que van tomando las variables de la siguiente manera. Clicamos en “Página de vigilancia”.



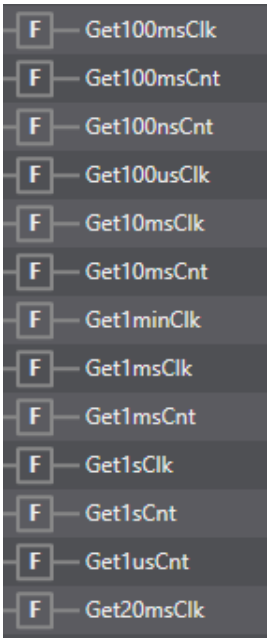
Esta herramienta es muy útil para diagnosticar y probar. Porque podemos visualizar y modificar el valor de las variables (globales y locales). Para las variables locales hay que indicar el programa al que pertenecen. Siguiendo con el ejemplo anterior, si queremos ver el valor de la variable local “puntero” tendríamos que poner “Programa0.puntero” tal y como vemos en la siguiente imagen:



Si queremos modificar el valor de una variable tenemos que introducirlo en la casilla “Modificar” y pulsar la tecla Enter.

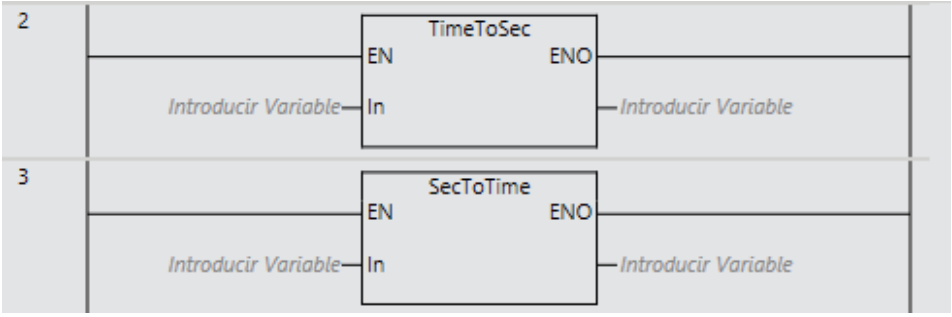
IV. Funciones:

1. Funciones interesantes:
    a. Get10msClk. En sysmac studio los generadores de pulsos vienen en formato función.

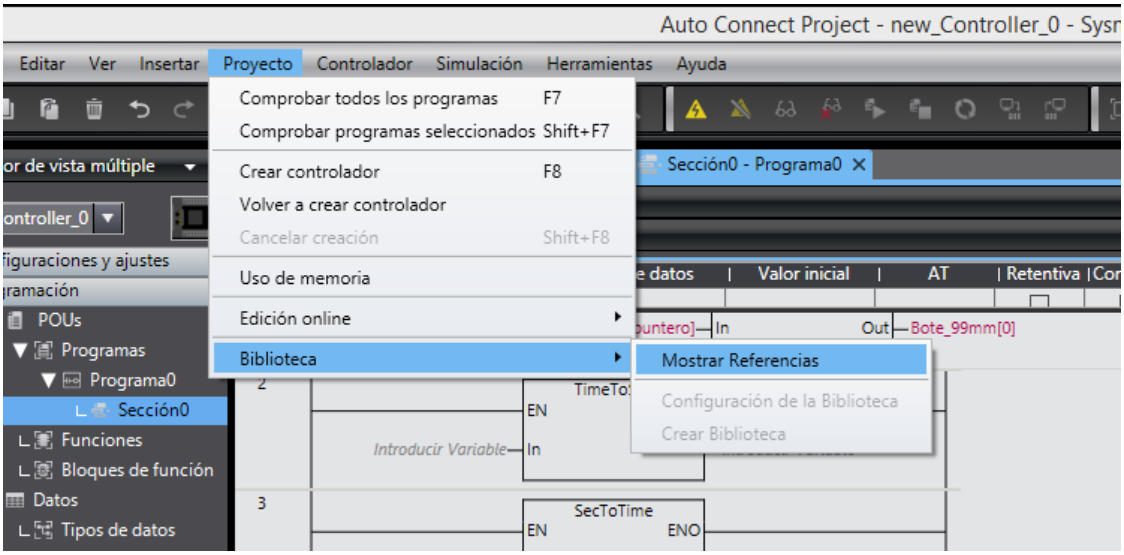


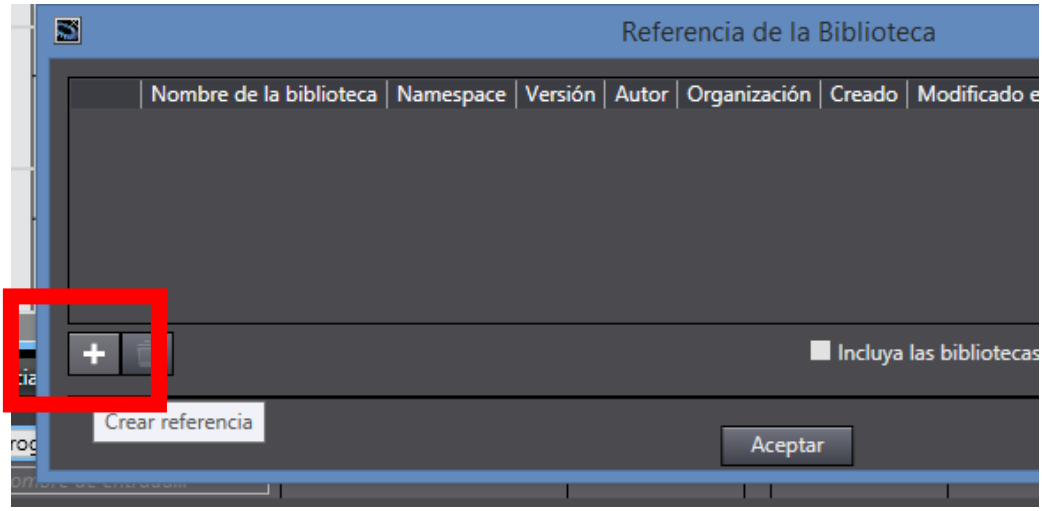
    b. PWLApprox. Hace una aproximación lineal. Útil para sensores que no tengan un escalado lineal, por ejemplo si queremos atacar a un termistor NTC sin tarjeta de linealización. Si sabemos la ecuación la metemos en texto estructurado y no hay problema. Pero si no tenemos una ecuación, y tenemos por ejemplo una serie de puntos que hemos obtenido empíricamente, esta función hace un multiescalado, permite linealizar entre punto y punto por lo que podemos calcular todos los puntos intermedios.
    Antes esto se tenía que hacer con Excel, se metían los puntos en Excel y se le pedía que nos sacase una ecuación que aproximase lo máximo posible a esa nube de puntos. Pero con esta función es más cómodo.
    c. TimeStamp. Para las E/S EtherCAT podemos leer y establecer cuando queremos desactivar o activar una cabecera de E/S.

2. Para cambiar y visualizar el valor de un temporizador desde un HMI tenemos que hacer uso de las funciones siguientes: “TimeToSec” y “SecToTime”. Estas funciones cambian el tipo de dato de la forma “T#1m4s” a tipo entero y viceversa.

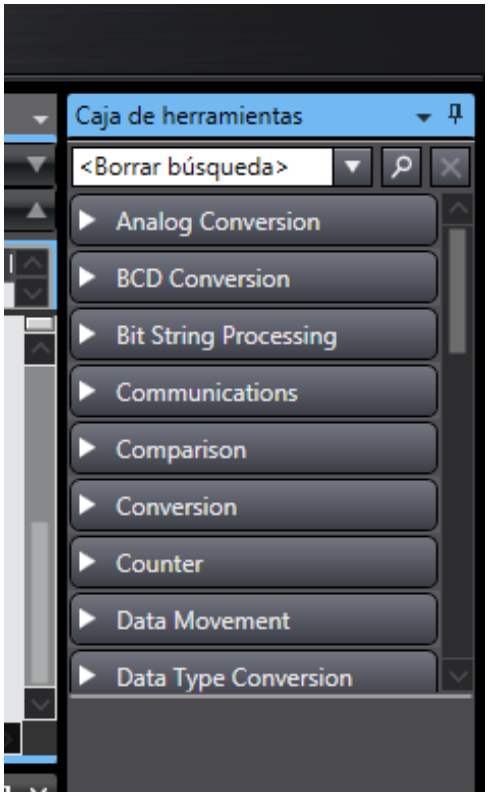


3. Las funciones aquí trabajan dentro de librerías. Porque llevan funciones y datos. Para importar funciones tenemos que importar librerías. La forma de cargarlas es la siguiente:





Una vez cargada, nos aparecerán las funciones nuevas en la caja de herramientas.



V. Servomotores:

Primero vamos a explicar la diferencia entre ejes controlados, ejes reales y ejes virtuales.

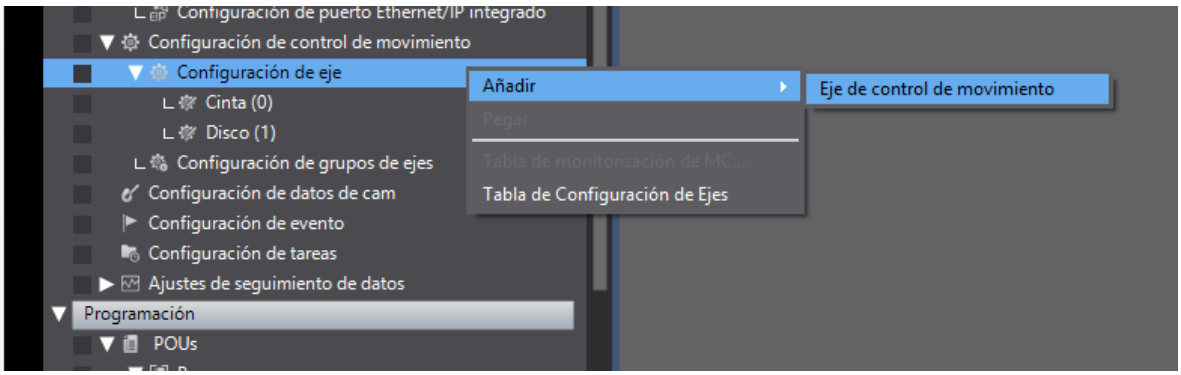
- Número de ejes controlados = número de ejes que la CPU puede controlar. Incluye reales y virtuales.
- Número de ejes reales = número de ejes físicos que se pueden conectar. En definitiva, número de drivers que podemos pinchar.
- Número de ejes virtuales = Numero ejes controlados – Numero ejes reales.

Un eje virtual es un eje que creamos por software.

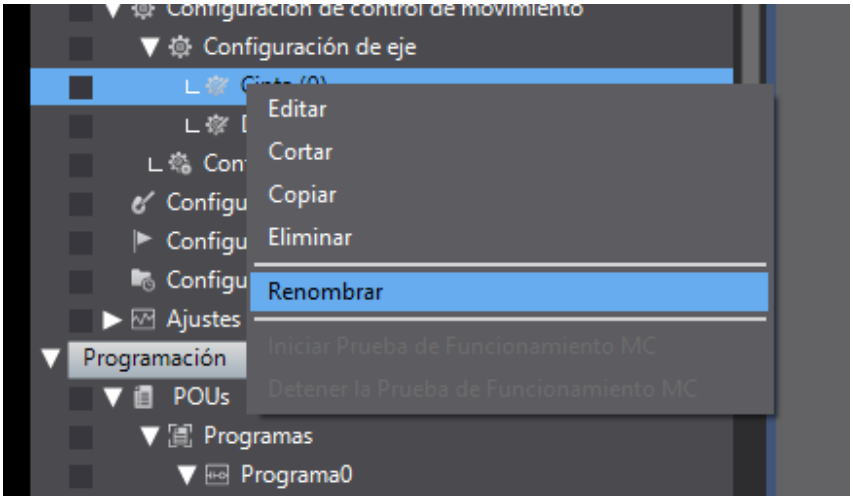
Un eje virtual puede ser un enconder. Si tenemos un cinta transportadora, que no está movida por servo sino por VFD, al ponerle un encoder, al ser un sincronismo, puede ser un eje. Pero no es un eje real porque no tiene entrada directa por driver de servo. Entonces es eje virtual.

Un eje virtual también puede ser una unión de dos ejes reales. Por ejemplo, si dos ejes reales queremos unirlos para que se muevan en conjunción, estamos creando un eje nuevo realmente, un eje virtual.

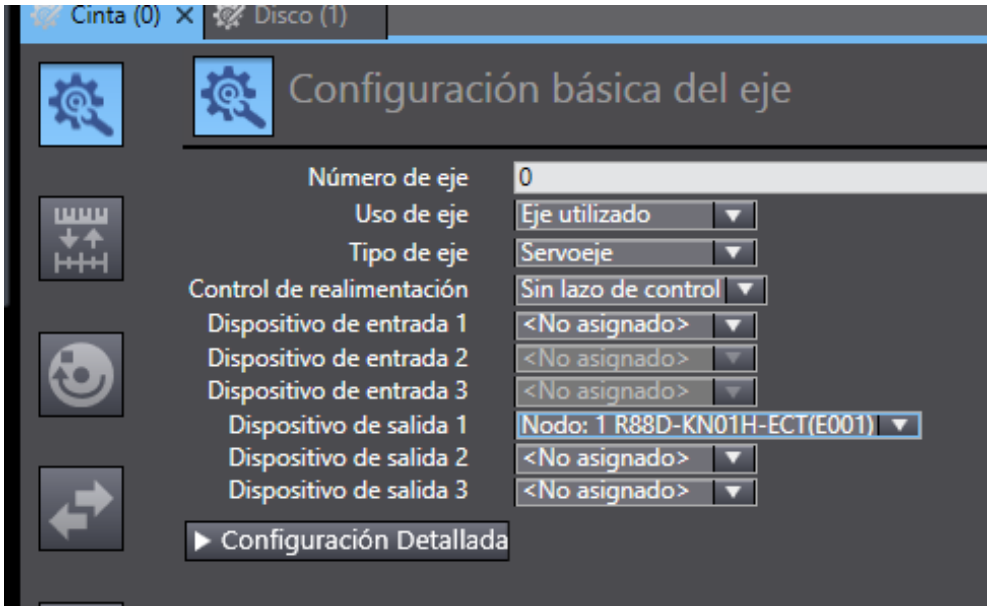
1. Añadir un nuevo servomotor. Nos vamos a “Configuración de eje”→”Añadir”→”Eje de control de movimiento”.



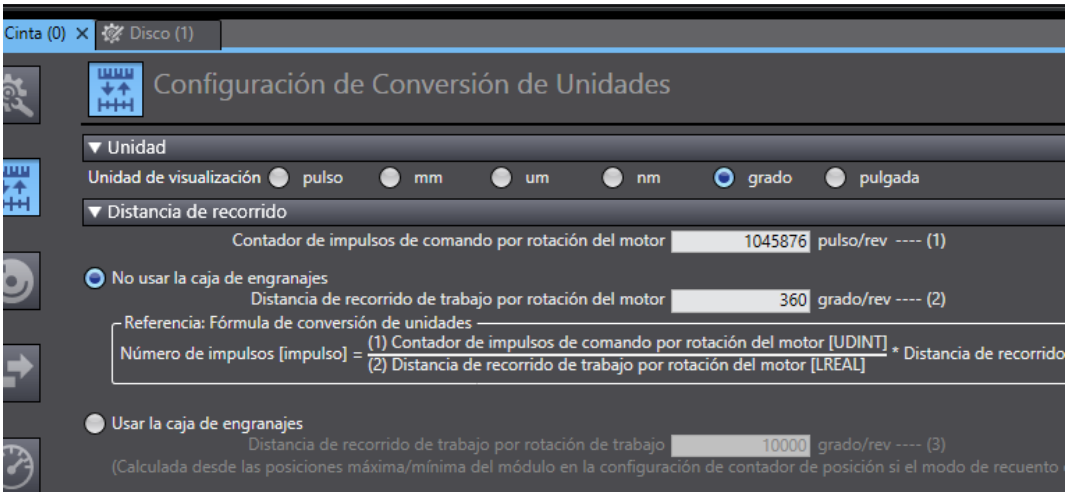
2. Cambiar el nombre del eje. Nos ponemos encima del servomotor y le damos botón derecho “Renombrar”.



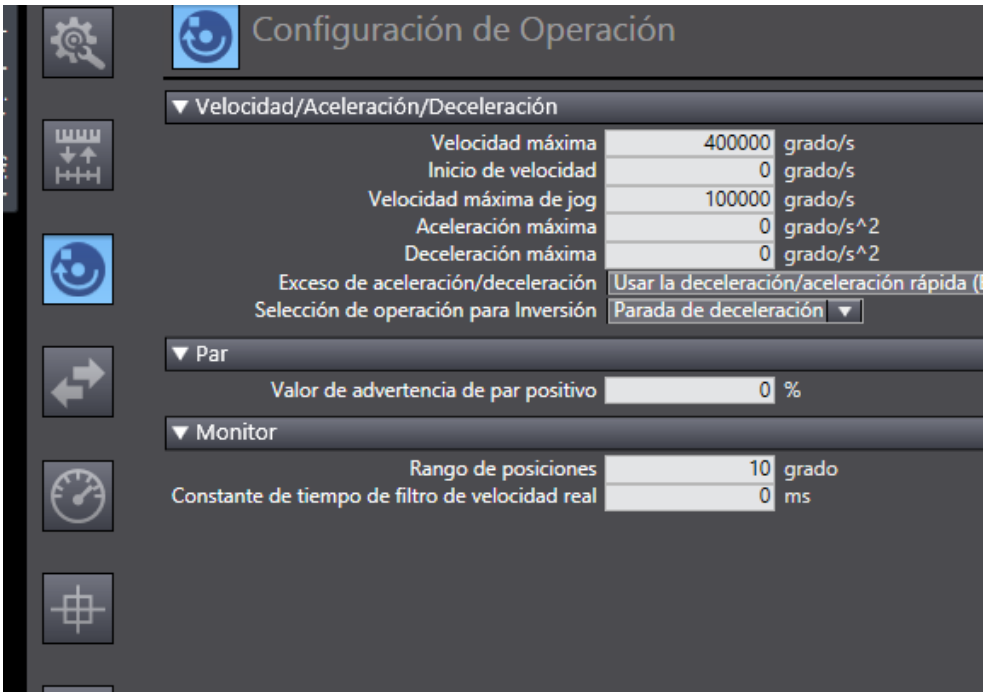
3. Configuración. Pulsamos dos veces sobre el eje.
    1. Configuración básica del eje.



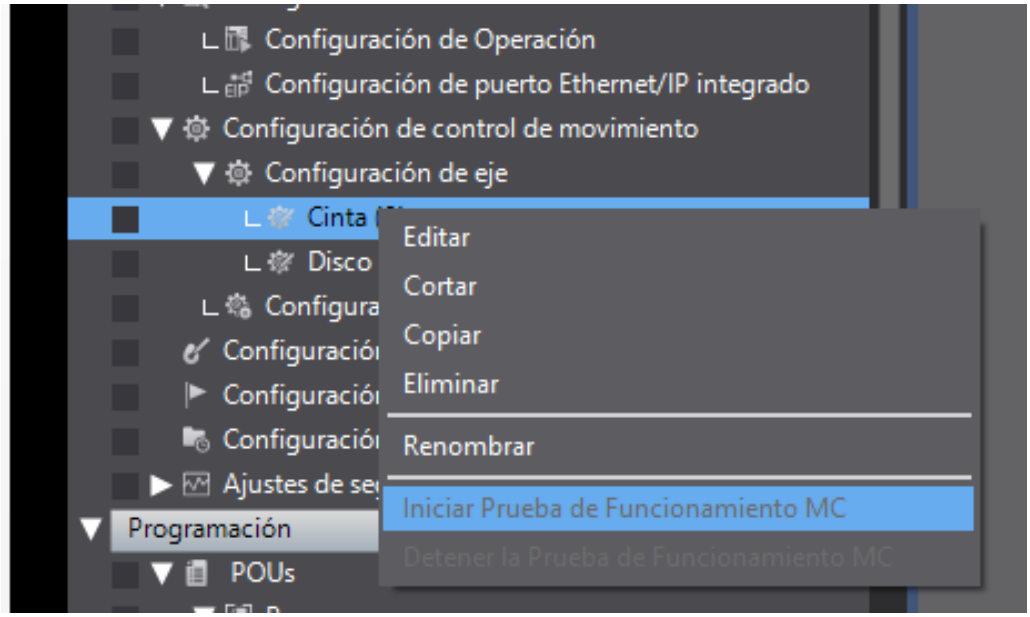
    2. Configuración de Conversión de Unidades. Aquí configuramos los pulsos por vuelta y escalamos el servo.
    En nuestro caso tenemos un servo con un encoder que tiene una resolución de 20 bits. 2^20=1045876. Este es el valor que ponemos en el dato de pulsos/rev. Para este ejemplo vamos a escalar el servo en grados y para simplificar pondremos 360 grados/rev.



3. Configuración de operación. Aquí configuramos las velocidades según los límites que tenga nuestro accionamiento real.



4. Comprobación de funcionamiento. Nos situamos encima del servo, botón derecho → "Iniciar prueba de funcionamiento MC".



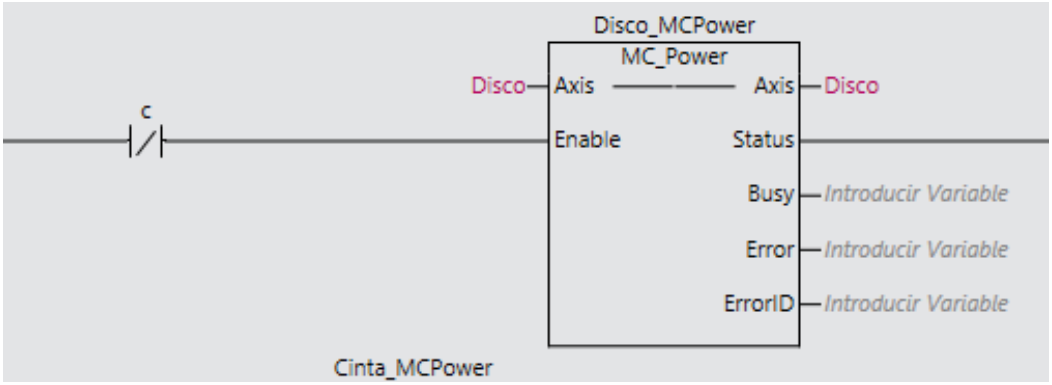
Se nos abrirá un asistente en el que podemos mover manualmente el servo para comprobar que la configuración es correcta. Para ello tenemos que activar el servo primero donde pone “Servo OFF” y le damos después al botón de desplazar.

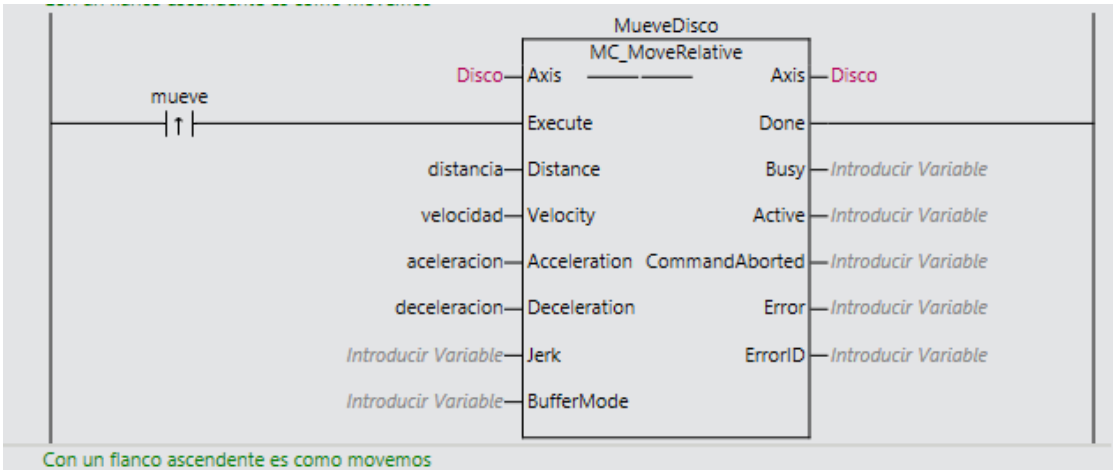
5. Movimiento relativo.

Si queremos hacer movimiento relativo tenemos que hacer uso primero de la función “MC_Power” y después “MC_MoveRelative”.

La función “MC_Power” pone el servo a ON, es decir el servo empieza a aplicar un par a velocidad cero, para quedarse bloqueado. Esta señal se mantiene por nivel, mientras que la señal de las siguientes funciones que vamos a usar se mantiene por pulso. La función “MC_MoveRelative” es la que ejecuta el movimiento, y es donde le indicamos la consigna de distancia y los valores de velocidad, aceleración, deceleración…

A continuación se muestra un programa para movimiento relativo:

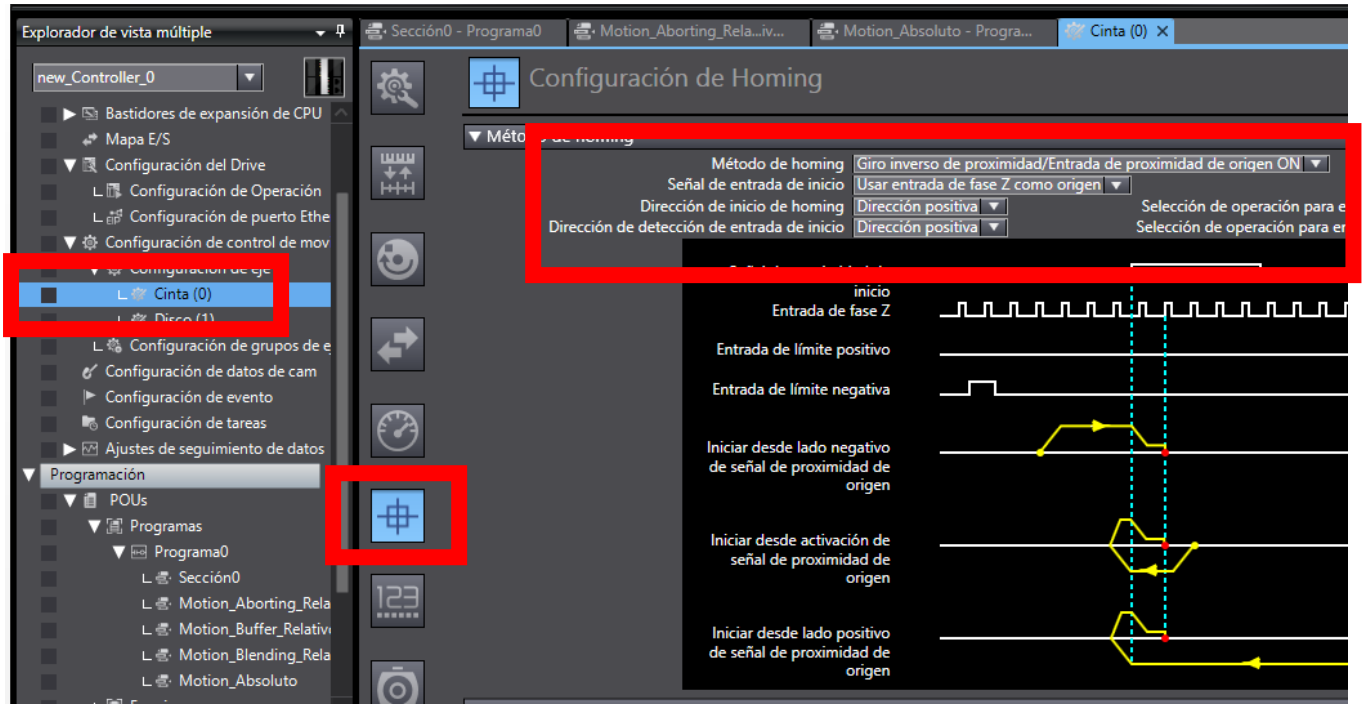




6. Movimiento absoluto.

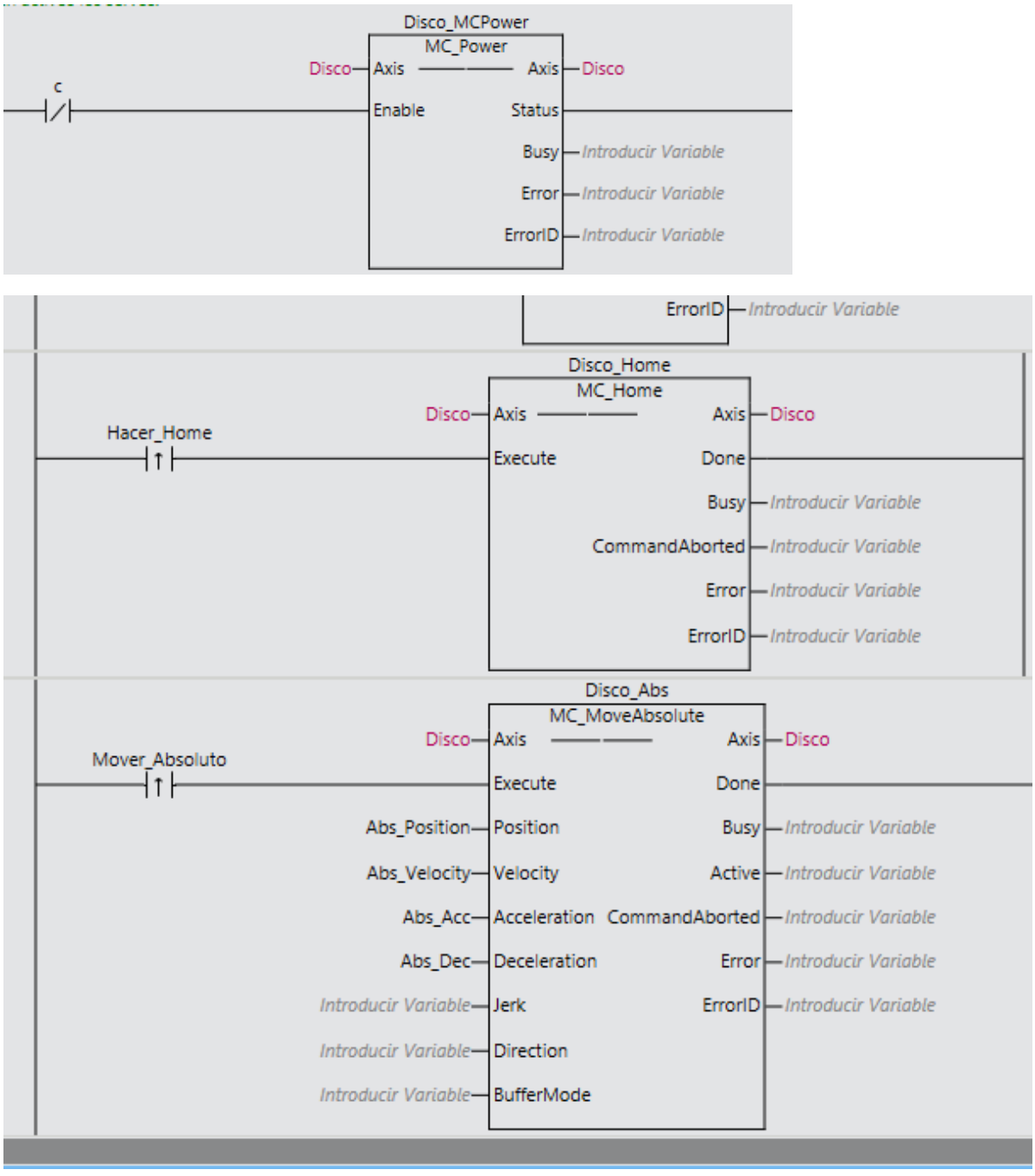
Para movimiento absoluto usamos los bloques “MC_Power”, “MC_Home” y “MC_MoveAbsolute”. El bloque “MC_Home” se usa para referenciar el origen. Si queremos hacer movimiento absoluto es necesario tener una referencia, un origen. Mientras que en movimiento relativo el origen del movimiento siguiente es el punto actual.

Hay que ajustar en “Configuración del Homing” cómo queremos que sea el proceso de Homing. A continuación se ve cómo se accede a dicha configuración.



Para configurar el homing se toma como referencia la fase Z del encoder (1 pulso por vuelta), para tener más precisión.

Veamos un ejemplo del programa con movimiento absoluto:



7. Señales de salida de los bloques.

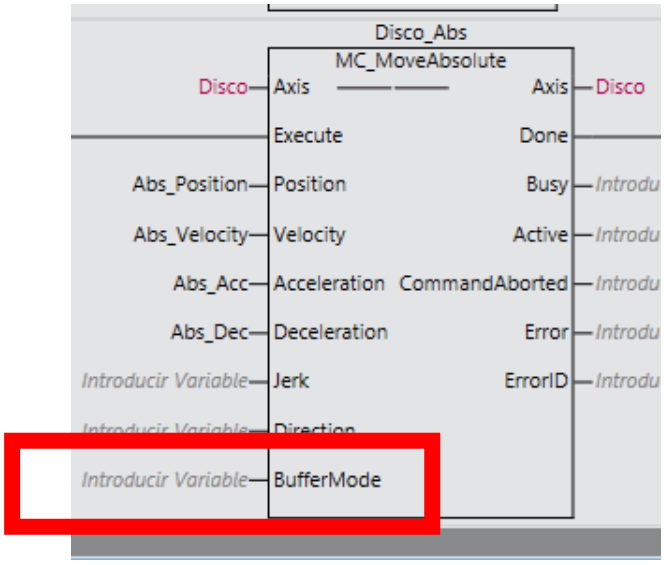
En todos los bloques de movimiento vemos que tienen las señales de salida:

- Busy: Dará un 1 si le hemos lanzado la petición de movimiento. Pero no significa que todavía se mueva.
- Active: Dará un 1 si el eje se está moviendo.
- Error: Muestra un código de error.

- Done: Se pone a 1 cuando ha ejecutado el movimiento. Cuando enlazamos movimientos es útil.

8. Buffer de movimientos.

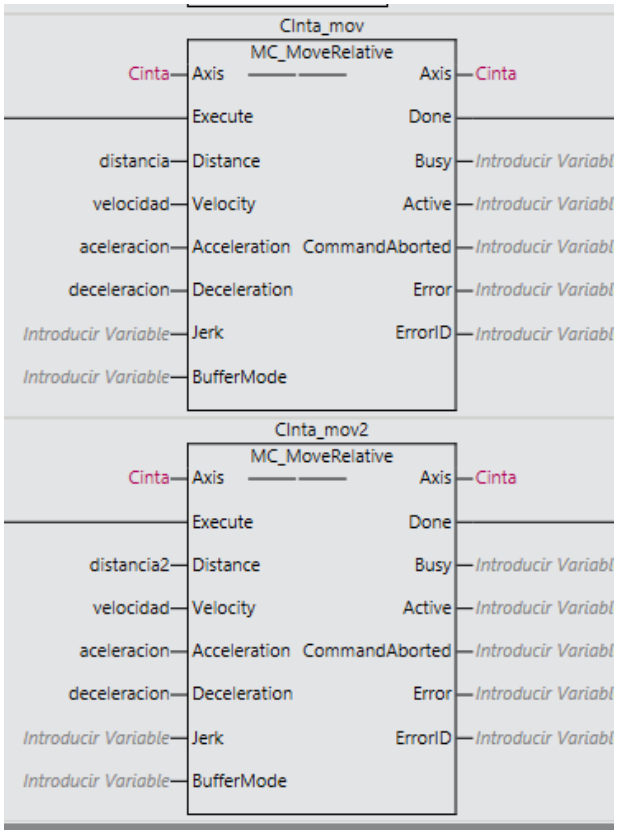
En las funciones “MC_Move” podemos configurar el Buffer.



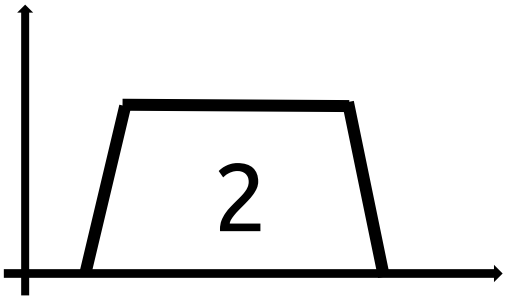
Pasemos a explicar las diferentes funciones que podemos configurar en el buffer. Imaginemos que tenemos dos instrucciones de movimiento en el programa, una tras otra.

Vamos a ver el comportamiento que hace el servo en función del buffer configurado:

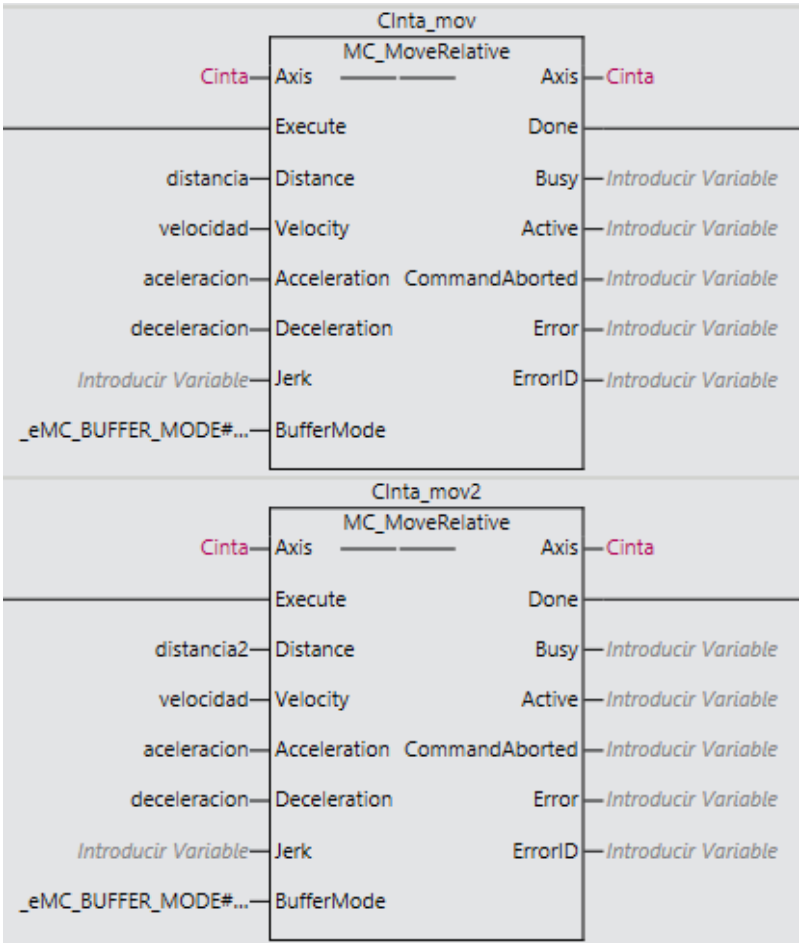
- ABORTING. Por defecto si no se rellena el campo “BufferMode” se queda en abortig. En el programa tendríamos lo siguiente.



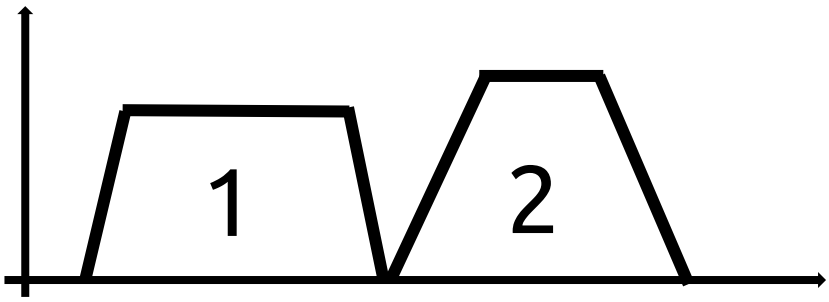
Lo que ocurre aquí es que la CPU lee el movimiento 2. Como si se tratase de una bobina, se queda con el último movimiento que lee. Así que sólo se ejecuta el movimiento 2 tal y como vemos en la siguiente imagen:



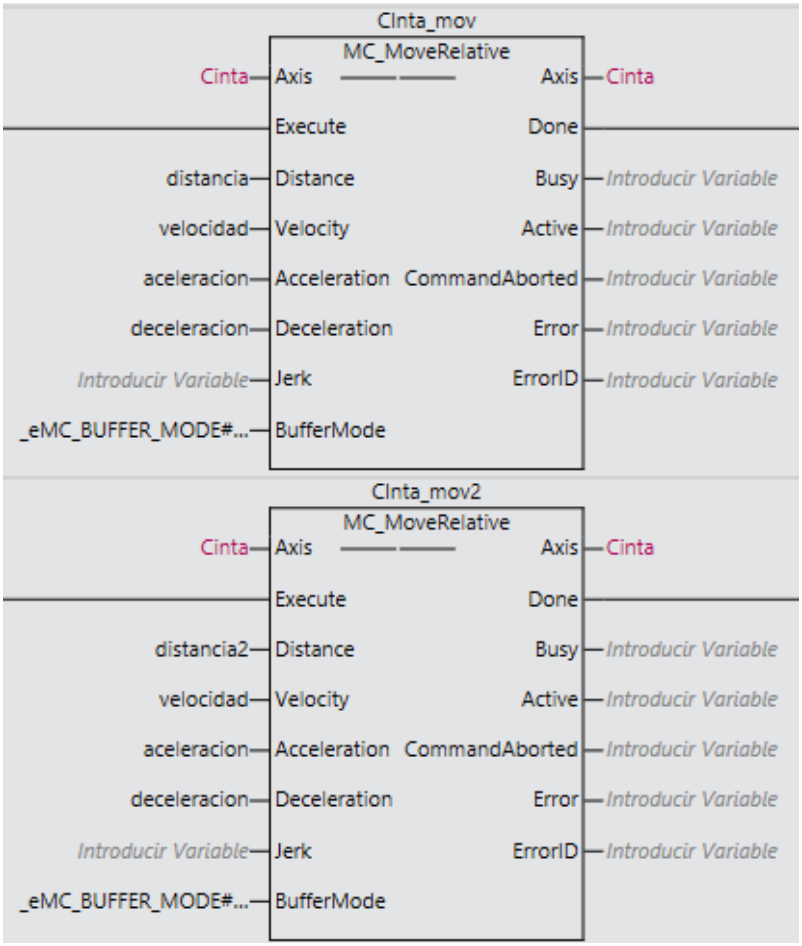
- BUFFER. Para configurarlo en este modo tenemos que rellenar el campo "BufferMode" con lo siguiente: "_eMC_BUFFER_MODE#_mcBuffered".



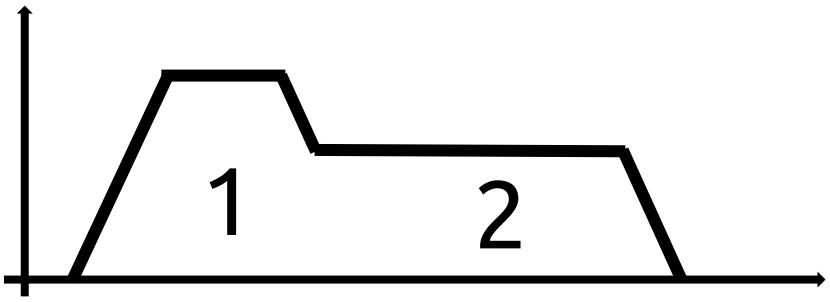
En este modo lo que ocurre es que si activamos los dos movimientos con el mismo trigger, se ejecuta primero el movimiento 1, y el movimiento 2 se guarda en buffer para ejecutarlo en cuanto termine el 1. Tal y como vemos en la siguiente imagen se ejecuta un movimiento y después otro:



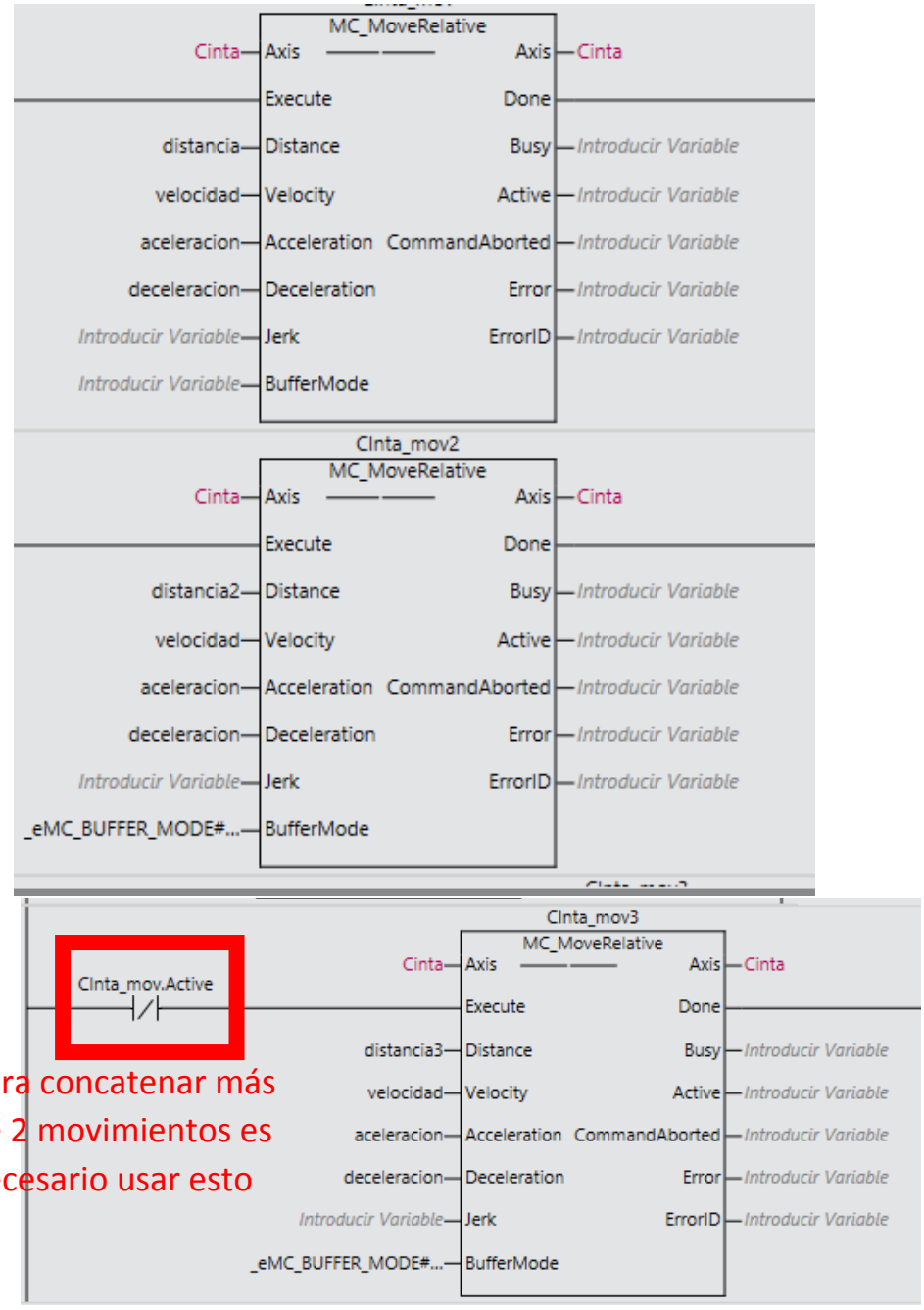
- BLENDING. Para configurarlo en este modo tenemos que rellenar el campo "BufferMode" con lo siguiente: "_eMC_BUFFER_MODE#_mcBlendingLow".



Lo que ocurre en este modo es que se mezclan las dos instrucciones tal y como vemos en la siguiente imagen:



Cabe destacar que en movimiento eje a eje, el buffer es de 1 solamente. Por lo que si queremos concatenar más de dos movimientos a través de buffer, tenemos que usar las señales “Active”. Vamos a mostrar un ejemplo de cómo se haría para concatenar 3 movimientos:



Para concatenar más de 2 movimientos es necesario usar esto

9. Cambiar la velocidad del servo en orden de marcha: se puede usando la instrucción “MC_combine_axis”.

10. Enlazar dos ejes. Es decir sincronización servo-servo. Se hace usando la instrucción “MC_GearIn”. Habrá momentos en los que interese tener “conectados” los dos servos y otras veces no. Para desacoplarlos usamos la instrucción “MC_GearOut”.

11. Hacer una leva. Es decir sincronización servo-objeto. Se hace usando “MC_DigitalCamSwitch”.

12. Creación de grupos de ejes. Cuando hacemos interpolaciones tenemos que asociar los ejes a un grupo. Una interpolación es cuando dos ejes arrancan y paran a la vez haciendo distancias diferentes. Las instrucciones que tenemos que usar son “MC_GroupEnable” y “MC_MoveLinear”. Para hacer una interpolación circular usamos “MC_MoveCircular2D”.

13. Control de par. Usamos “MC_TorqueControl”. Le ponemos el torque y la velocidad límite. En control de par siempre es necesario poner una velocidad máxima, porque si en algún momento no hay par resistente interesa que la velocidad aumente hasta cierto límite buscando el par.

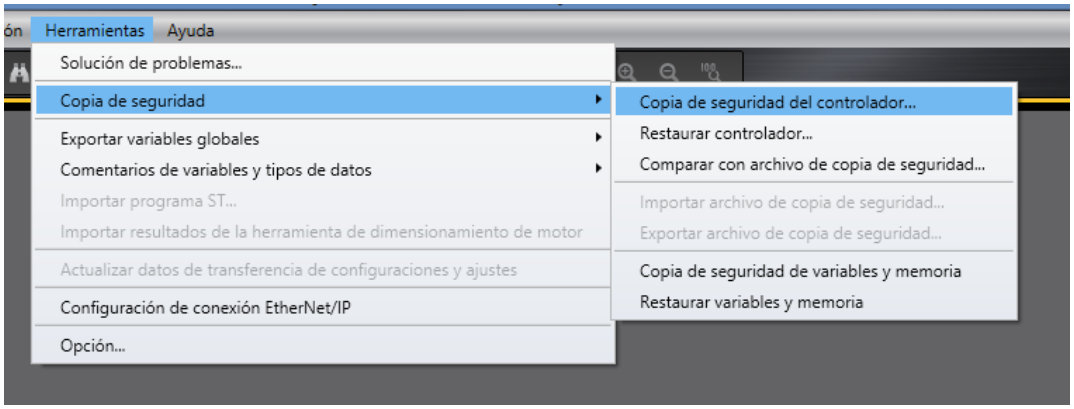
14. Observaciones.

Algunos servos controlan el encoder desde el driver. En otros, es el PLC el que lee el valor del encoder del servo. Como el ciclo de scan es de 1ms, a cada ms lee el valor del encoder y envía el valor de corrección al servo según la consigna. Realmente nosotros usamos una instrucción Move. Pero por debajo, de manera transparente, cuando lanzamos una instrucción MOVE, la CPU hace muchos procesos de lectura del encoder y cálculos según consigna.

VI. Actualización de firmware y backup

1. Actualización de firmware de la CPU.
    1. Solicitamos el firmware a Omron (no se puede descargar de su página ni por Sysmac Studio).
    2. Lo cargamos en una SD.
    3. Con la CPU apagada introducimos la SD y ponemos los 4 dip-switch a ON.
    4. Encendemos la CPU.

2. Generación de Backups. Varias formas de hacerlo:
    1. Desde Sysmac Studio. Si le damos a "Herramientas" → "Copia de seguridad" → "Copia de seguridad del controlador" hacemos una copia de seguridad de todo (programas, configuración, memoria).



Si le damos a Restaurar controlador podremos cargarle un Backup previamente realizado.

2. Por Switches. Poniendo los dip-switch en el modo Backup se hace una copia de todo: programa, configuraciones, memoria. Se guarda en la SD.
3. Por HMI. Si está conectado a un HMI podemos usar la instrucción "BackupToMemoryCard". Se guarda en la SD. Para poder hacer el Backup hay que activar el bit de programa desde el HMI. Ver sección 9 del manual de NJ, NX.