UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Escola Superior d'Enginyeries Industrial, Aeroespacial i Audiovisual de Terrassa

INGENIERÍA ELECTRÓNICA INDUSTRIAL Y AUTOMÁTICA

TRABAJO DE FIN DE GRADO

Torrecilla Matencio, Marc

ESTUDIO DE LAS ETAPAS DE AUTOMATIZACIÓN DE UN PROCESO INDUSTRIAL

ANEXO I

Director: Delgado Prieto, Miguel

Co-director: Fernández Sobrino, Ángel

Convocatoria: Julio, 2020

SUMARIO TABLAS

SUMARIO FIGURAS

ANEXO I:

DOCUMENTACIÓN COMPLEMENTARIA

1. COMUNICACIONES INDUSTRIALES

1.1. OBJETOS DE DATO DE PROCESO (PDO)

Los mensajes PDO de los nodos islas se pueden dividirse en dos categorías:

1. **TPDO[1]:** mensajes con información del proceso que el nodo transmite. En este caso la lectura de los sensores.

2. **RPDO[2]**: mensajes con información del proceso que el nodo escucha. En este caso las islas Advantys que escucharán del bus en busca de órdenes para el control de los motores y actuadores.

El programa "Unity Pro XL" tiene configurado el mapeado de los mensajes en las %MW, que permiten la lectura y escritura de las PDO.

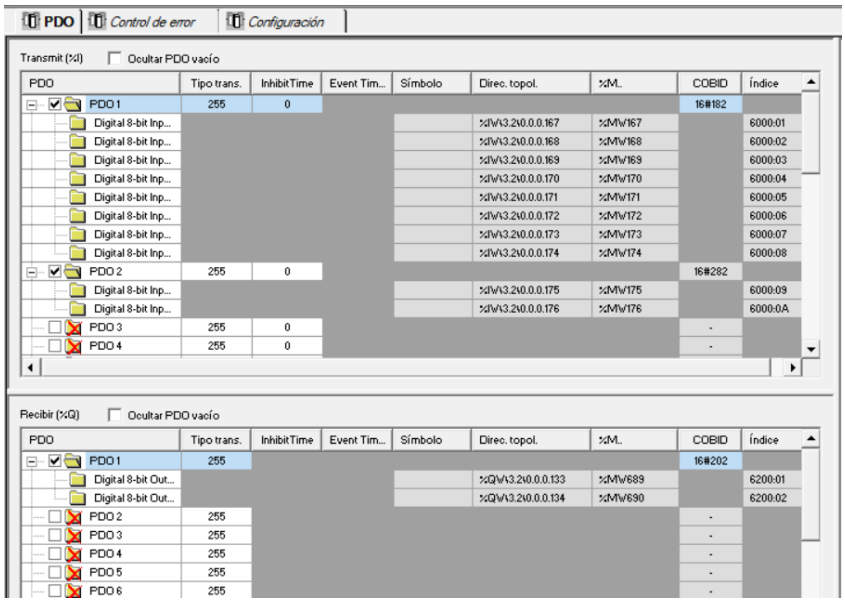


Figura 1. Panel configuración PDO STB_NCO_2212.

[1] **Figura 1**. Transmit (%I) mensajes de transmisión. El paquete PDO1 y PDO2 contienen las entradas digitales que transmite el nodo Advantys.

[2] **Figura 1**. Recibir(%Q) mensajes de recepción. El paquete PDO1 y PDO2 contienen las entradas digitales que recibe el nodo Advantys.

Tal y como se puede observar en la tabla, los objetos de entrada y salida de un esclavo CANopen, siguen la siguiente estructura: **%I,Q W\b.e\r.m.c.d.**

Donde:

Familia	Elemento	Significado
Símbolo	%	Indica un objeto IEC
Tipo de Objeto	I	Objeto de entrada
	Q	Objeto de salida
Formato	W	Palabra WORD 16 bits
Dirección de módulo/canal y punto de conexión	b	Número de bytes
	e	Número de punto de conexión, el esclavo CANopen
Número de bastidor	r	Número de bastidor virtual, siempre 0
Número de módulo	m	Número de módulo virtual, siempre 0
Número de canal	c	Número de canal
Rango de datos del canal	d	Número de datos de esclavo

Tabla 1. Leyenda de direcciones Unity Pro XL.

1.2. IDENTIFICADOR DE OBJETO DE COMUNICACIÓN (COB-ID)

CiA (Can in Automation) establece direcciones estándar de los COB-ID para los cuatro primeros PDO de transmisión y recepción:

Objeto de comunicación (COB)	Identificador del COB (COB-ID)
TxPDO1	384 (180h) + id.nodo
RxPDO1	512 (200h) + id.nodo
TxPDO2	640 (280h) + id.nodo
T_PDO2	768 (300h) + id.nodo
R_PDO3	896 (380h) + id.nodo
T_PDO3	1024 (400h) + id.nodo
R_PDO4	1152 (480h) + id.nodo
T_PDO4	1280 (500h) + id.nodo

Tabla 2. Direcciones estándar para COB-ID.

EL COB-ID está formado por dos partes, el código de función de 4 bits y el identificador del nodo de 7 bits. Cada PDO contiene un COB-ID único.

1.3. RESUMEN DE LOS PROTOCOLOS CONSIDERADOS

	CANopen	Modbus TCP/IP	ETHERNET	HTTP
Características Generales	Sistema de protocolos a nivel de aplicaciones. Topología: en serie y/o con derivaciones. Modelo productor/consumidor y maestro/esclavo. Velocidades 1 Mbps. 64 nodos máximo por segmento y 127 nodos máximos en el bus.	Sistema de protocolos abierto en Ethernet a nivel de aplicaciones. Cable Ethernet con conector RJ-45. Tipología Ethernet en serie o estrella. Modelo cliente / servidor. Velocidad de Ethernet: 100 Mbps.	Estándar de comunicación. Define el nivel físico y el nivel de enlace. Tipología en serie o estrella. CSMA/CD técnica para evitar la colisión entre tramas. Velocidad típica de100 Mbps. Trama de Datos de 42 hasta 15000 Bytes.	Protocolo a nivel de aplicación. Transmisión de documentos hipermedia. Estructura cliente / servidor. Multiplexación, el envío de varios mensajes a través del mismo medio manteniendo la integridad de estos.
Ventajas y beneficios	Insensitivo a interferencias electromagnéticas. Transmisión fiable, COB-ID evita la pérdida de las tramas. Transmisión rápida y en tiempo real. Servicio de 256 puntos de E/S digitales a 1 Mbps. Transparencia entre CANopen y Modbus TCP/IP.	Servicios de I/O Scanning para la comunicación entre los PLC Modicon M340 de Schneider. Sencillez y conexión fiable al encapsular la trama Modbus en un segmento TCP. Cantidad de nodos en unos 1024 gracias a Ethernet. Permite la conexión entre otros dispositivos Modbus como RTU.	Fácil ampliación y conectividad a través de un conmutador (switch). Ideal para crear redes LAN. Transmisión a largas distancias de hasta 10 km.	Transferencia de información a la World Wide Web (red informática mundial). Envío de datos o contenidos a bases de datos a través de servidores web. Funciones de autentificación. Registro del acceso a usuarios autorizados.
Inconvenientes y Limitaciones	Dependencia de la longitud del bus con la tasa de bits de transmisión. Niveles eléctricos de bus restrictivos. Número de nodos limitado. Un maestro en el bus CANopen y los dispositivos esclavos.	Protocolo no determinista. Reserva de espacios de memoria para los datos de comunicación.	Velocidad de transmisión limitada cuando todos los nodos están en el sistema. CSMA / CD, control de acceso al medio no determinista. Paquete Data grande , disminuye la eficiencia de transmisión de datos.	Mensajes de texto plano: mensajes largos, entre 16 kb y 16 MB. Comunicación menos segura al no integrar la encriptación de datos a través de SSL/TLS como si lo hace HTTPS.
Escenarios de uso frecuente	Entornos distribuidos. Dedicada a la automatización industrial. Nivel de máquina. Conexión entre PLC's e islas remotas de E/S.	Nivel de célula, para la supervisión y control. Intercambio de información entre dispositivos, PLC's, administración de E/S distribuidas.	Redes administrativas. Redes de control y supervisión. Conexión entre el nivel de célula y el de gestión a través de una pasarela.	Nivel de planta para comunicación entre los navegadores y servidores web.

Tabla 3. Tabla resumen de los protocolos de comunicación considerados.

2. FASE DE AUTOMATIZACIÓN

2.1. ESTRUCTURAS LÍNEA CAN

2.1.1. ESTRUCTURA PRODUCTO

- **ID:** número único que identifica cada producto producido por la línea.
- **TIPO**: número que determina el tipo de producto, se usa para determinar si en una bandeja o palé, hay un producto o no, y el tipo de producto.
- **BANDEJA:** número único que identifica la bandeja.

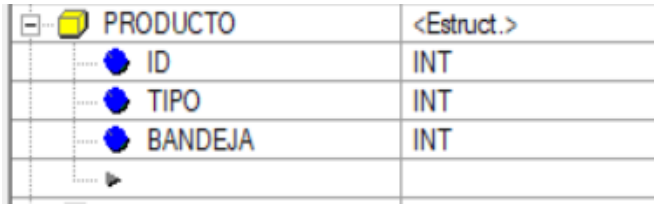


Figura 2. Estructura producto.

2.1.2. ESTRUCTURA RETENEDOR

- **REST:** define el estado en reposo del retenedor o plataforma, indica que no hay bandeja.
- **READY**: define el estado de petición del retenedor o plataforma, indica que hay una bandeja.
- **MOVE:** define el estado de movimiento del retenedor o plataforma, indica que la bandeja se ha puesto o está en movimiento.
- **BLOQ:** define el estado de bloqueo, se usa para bloquear el retenedor o plataforma impidiendo el cambio de un estado en caso de que se necesite.
- **PRODACTUAL:** producto actual en el retenedor o plataforma.
- **PRODARRIVES:** producto que llegará en el retenedor o plataforma.

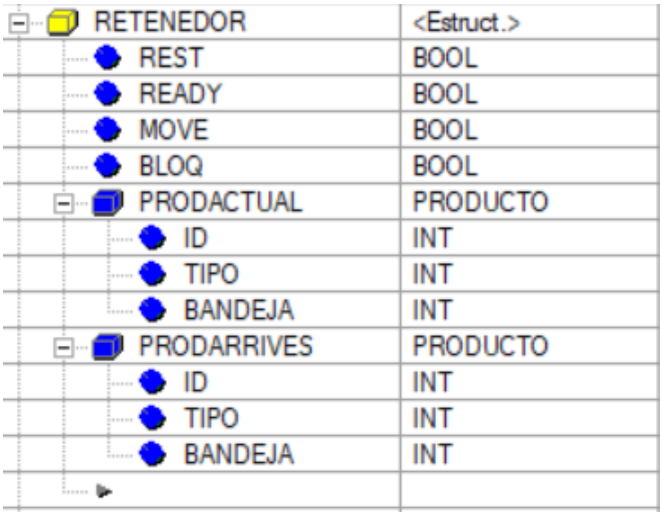


Figura 3. Estructura retenedor.

2.1.3. ESTRUCTURA BIFURCACIÓN

- **RECTO:** define el estado de decisión, dirección hacia el retenedor o plataforma de delante.
- **DESVIO:** define estado de decisión, dirección hacia el retenedor o plataforma lateral.
- **BLOQ_RECTO:** define el estado de bloqueo del retenedor o plataforma de delante.
- **BLOQ_DESVIO:** define el estado de bloqueo del retenedor o plataforma lateral.
- **MOVE_RECTO:** define el estado de movimiento del retenedor o plataforma de delante.
- **MOVE_DESVIO:** define el estado de movimiento del retenedor o plataforma lateral.

BIFURCACION	<Estruct.>
REST	BOOL
READY	BOOL
BLOQ	BOOL
RECTO	BOOL
DESVIO	BOOL
BLOQ_RECTO	BOOL
BLOQ_DESVIO	BOOL
MOVE_RECTO	BOOL
MOVE_DESVIO	BOOL
PRODACTUAL	PRODUCTO
ID	INT
TIPO	INT
BANDEJA	INT
PRODARRIVES	PRODUCTO
ID	INT
TIPO	INT
BANDEJA	INT

Figura 4. Estructura bifurcación

2.1.4. ESTRUCTURA ESTACIÓN

- **ESTACION**: variable que informa de que la estación ha acabado su proceso (0 en el primer flanco de bajada), el tiempo que tarda (1 durante el estado alto), y el bloqueo del retenedor donde se encuentra durante el proceso.

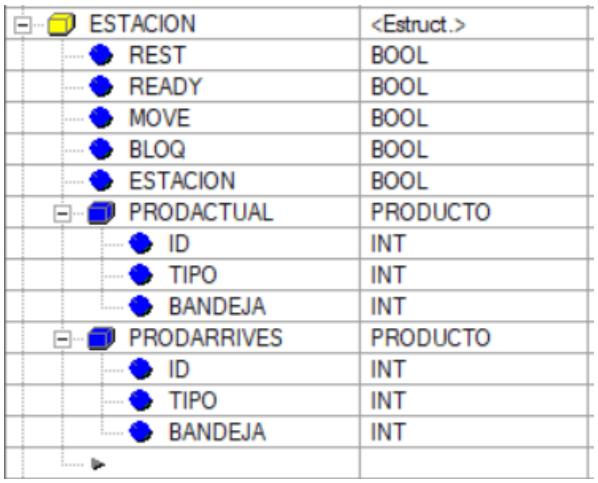


Figura 5. Estructura estación.

2.1.5. ESTRUCTURA FLUJO

La variable de tipo INT “PUNTERO”, permite almacenar la posición actual de la tabla en la que se quiere traspasar el tipo de producto en la bandeja de PT06.

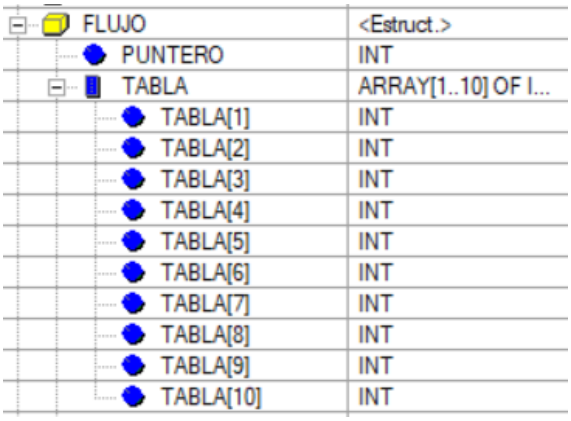


Figura 6. Estructura flujo.

Se ha creado la siguiente variable de este tipo:

- **FLUJO_MATERIAS_PRIMAS:** variable referente a la tabla de flujo de carga de tipos de productos en la carga de materias primas.

2.1.6. ESTRUCTURA LOTE

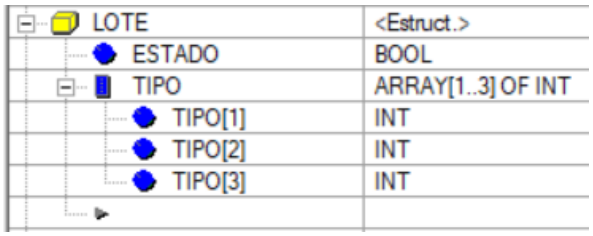


Figura 7. Estructura lote.

Se han creado las siguientes variables de este tipo:

- **LOTE_CANTIDAD:** variable que contiene la cantidad para cada tipo de producto que se tiene que producir en un ciclo de lote.
- **LOTE_CARGADO:** variable que contiene la cantidad de tipos de productos que han sido cargados con materias primas. Cuando el tipo de producto del lote cargado se igual del de "LOTE_CANTIDAD" no se producirá más este tipo de producto.
- **LOTE_FINALIZADO:** variable que contiene la cantidad de productos que ya han sido extraídos. El ciclo de lote se finalizará cuando este lote sea igual de "LOTE_CANTIDAD".

2.2. JUSTIFICACIÓN DE LAS ESTRUCTURAS

La programación documentada en la memoria **"4.1.2 MODIFICACIÓN DE LA TRAZABILIDAD",** permite saber en cada momento el flujo de productos por los que van pasando a través de los retenedores y plataformas en tiempo real[3]. De esta forma podemos saber en cada momento donde se encuentran los productos y que partes de las cintas transportadoras están ocupadas. Esto servirá de base para las implementaciones de gestión como la incorporación del Pulmón.

En el programa anterior, en cambio, el traspaso de información se hacía mediante estructuras de tipo array, y unos contadores que actuaban como punteros para el intercambio de información entre los retenedores y plataformas. Esto hacía difícil el seguimiento del programa, ya que resultaba más compleja su entendibilidad al traducirse en una escritura más extensa.

Tampoco se podía determinar en qué punto de la línea de cada instante se encontraban los productos, pues las tablas contaban con 10 estructuras de tipo "PRODUCTO" en el que se iba reescribiendo de forma cíclica cada posición del array. El intercambio de información solo traspasaba los datos entre las plataformas, sin tener en cuenta los retenedores, utilizando solo el estado "MOVE".

El uso de las arrays hacía aumentar considerablemente la cantidad de variables.

Véase el siguiente comparativa donde se refleja la disminución de las variables almacenadas:

Si solo tenemos en cuenta las matrices, considerando solo el "ID", "TIPO_PRODUCTO" y "BANDEJA" tendremos una cantidad de:

[3] A nivel de lógica programable. En la realidad hay que tener en cuenta que habrá un retraso temporal de los datos debido al traspaso de información entre las comunicaciones.

*10 (arrays) * 10 (posiciones) * 3(variables INT)=* **300 variables almacenadas.**

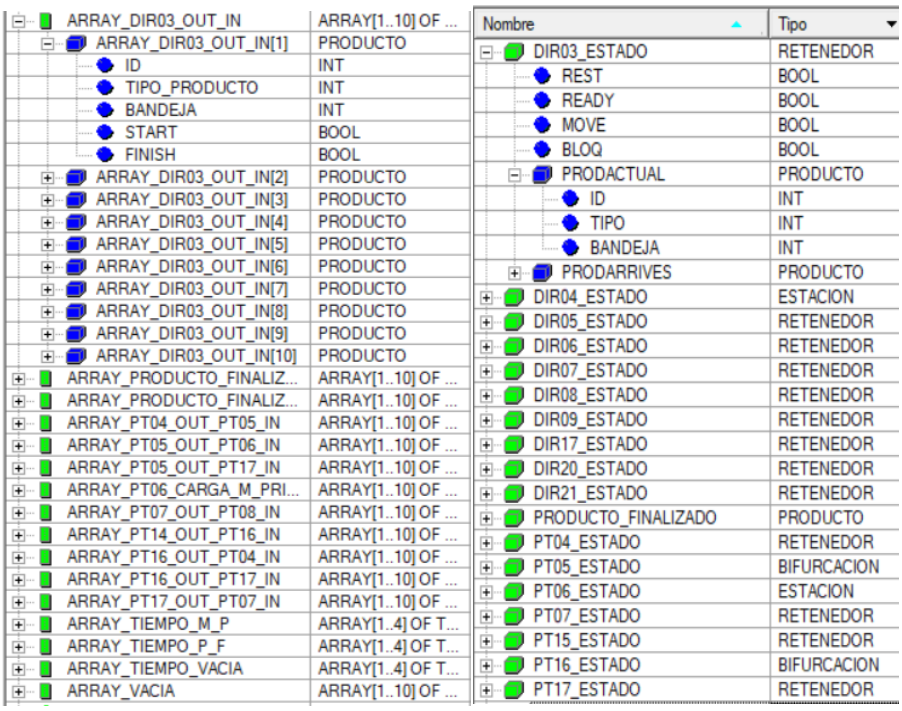


Figura 8. Comparativa matriz con estructuras.

En cambio, ahora tenemos:

18(estructuras) * (3(PRODACTUAL) + 3(PRODANTERIOR))= **108 variables almacenadas**

2.3. BLOQUE DE FUNCIOENS DERIVADOS LÍNEA CAN (DFB)

2.3.1. DOCUMENTACIÓN DFB

Las DFB se utilizan para secuencias de programa que se repiten varias veces en la aplicación. Estas se pueden exportar y la posterior importación de las DFB permite que las utilice otro equipo de programadores para la misma aplicación o en diferentes al estandarizar un programa dentro del bloque.

Estas nos presentan las siguientes ventajas:

- Permiten estructurar y mejorar la aplicación.
- Simplifica el diseño y el aprovechamiento del programa.
- Facilita la depuración del programa, pues todas las variables introducidas por la DFB se identifican en su interfaz.

- Disminuir el volumen de código generado. El código correspondiente a las DFB se carga una vez, sin importar el número de llamadas al DFB en el programa introducido. Sólo se generan los datos correspondientes a las instancias.

por otra parte, también tiene diferentes ventajas respecto el uso de una subrutina:

- Permite parametrizar más fácil el procesamiento.
- Las DFB utilizan variables internas independientes de la aplicación.
- Los programas de las secciones de las DFB pueden probarse independientemente de la aplicación en que se utiliza.

Se podría hacer la analogía de los bloques diseñados, sólo conceptualmente, con una función de transferencia en el uso que le damos en nuestra aplicación. Donde a través de las señales de entrada en el bloque, nos permite tener una respuesta de las señales de salida, es decir, el cambio de los estados de los retenedor o plataformas.

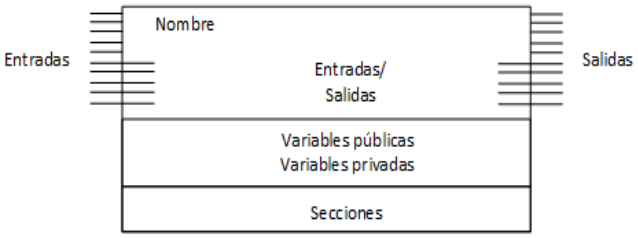


Figura 9. Ilustración DFB.

Se han utilizado los siguientes parámetros:

- **Entradas:** parámetros que permiten pasar los valores del programa aplicación al programa interno del DFB. Estas serán los contactos referentes a los estados que se necesitan para el programa interno.

- **Salidas:** parámetros que permiten pasar los valores del programa interno del DFB al programa de aplicación. Estas serán bobinas para activar o desactivar cada estado con las bobinas de set y reset del programa de aplicación.

- **Variables privadas**: variables internas del DFB que no se pueden acceder desde fuera de ellas. Esta se ha utilizado para el filtrado de los sensores con la FB TON.

- **Secciones:** Programa interno. Se ha utilizado el lenguaje de contactos Ladder.

2.3.2. PARÁMETROS PULMÓN CARGA

- **Entradas:** Los parámetros de entrada definen las condiciones necesarias para que se pueda producir la activación del ciclo de carga.

    1. **CARGA_IN**: parámetro de entrada indicador del ciclo de carga. Impide volver a iniciar la carga del Pulmón mientras ya esté en ciclo de carga.
    2. **DESCARGA_IN:** parámetro de entrada indicador del ciclo de descarga. Impide el inicio de la carga en el Pulmón en caso de que esté activo.
    3. **SALIDA_READY:** parámetro referente al estado de petición de la plataforma PT06. Permite el inicio de carga en caso de que esté activo.
    4. **SALIDA_MOVE:** parámetro de entrada referente al estado de movimiento de la plataforma PT06. Permite el inicio de carga en caso de que esté activo.
    5. **INTERIOR_READY:** parámetro de entrada del estado del retenedor interior del Pulmón DIR06. Impide la activación de la petición de carga en caso de que esté activo.
    6. **ENTRADA_READY:** parámetro de entrada del estado del retenedor de DIR05. Permite activar.
    7. **MAX:** parámetro referente al estado máximo del pulmón. Impide la activación de la petición de descarga en caso de que el pulmón llegue a su almacenamiento máximo.

- **Salidas:** el parámetro de salida habilita el estado de la petición de carga bandeja dependiente de los parámetros de entrada.

    1. **CARGA_OUT:** parámetro de salida indicador del cumplimiento de las condiciones para la carga. Se usa como flanco ascendente para la activación de la petición de carga.

PULMON_CARGA		<DFB>
<entradas>		
CARGA_IN	1	BOOL
DESCARGA_IN	2	BOOL
SALIDA_READY	3	BOOL
SALIDA_MOVE	4	BOOL
INTERIOR_READY	5	BOOL
ENTRADA_READY	6	BOOL
MAX	7	BOOL
<salidas>		
CARGA_OUT	1	EBOOL
<entradas/salidas>		
<público>		
<privado>		
<secciones>		
CARGA		<ST>

Figura 10. Tipo DFB pulmón carga.

2.3.3. PARÁMETROS PULMÓN DESCARGA

- **Entradas:** Los parámetros de entrada definen las condiciones necesarias para que se pueda producir la activación del ciclo de descarga.

    1. **CARGA_IN:** parámetro de entrada indicador del ciclo de carga. Impide volver a iniciar la carga del Pulmón mientras ya esté en ciclo de carga.
    2. **DESCARGA_IN:** parámetro de entrada indicador del ciclo de descarga. Impide el inicio de la carga en el Pulmón en caso de que esté activo.
    3. **INTERIOR_REST:** parámetro de entrada indicador del estado de reposo de DIR06. Debe estar activo para poder habilitar la descarga, ya que indica que no hay bandeja en la plataforma del actuador lineal.
    4. **INTERIOR_MOVE:** parámetro de entrada referente al estado de movimiento del retenedor DIR06. En caso de que esté activo, impedirá el ciclo de descarga, puesto que habrá una bandeja moviéndose de DIR06 a PT06.
    5. **ENTRADA_MOVE:** parámetro de entrada indicador del estado de movimiento del retenedor DIR05. Si está activo impedirá la descarga, ya que habrá una bandeja dirigiéndose a DIR06.
    6. **ENTRADA_READY:** parámetro de entrada del estado de petición del retenedor DIR05.
    7. **MIN:** parámetro referente al estado mínimo del Pulmón. Impide la descarga al no haber palés en el Pulmón.
    8. **DIR06:** parámetro de entrada del sensor inductivo del retenedor del Pulmón. Impide la petición de descarga en caso de que detecte una bandeja.

- **Salidas:** el parámetro de salida habilita el estado de la petición de descarga bandeja dependiente de los parámetros de entrada.

    1. **DESCARGA_OUT:** parámetro de salida indicador del cumplimiento de las condiciones para la descarga. Se usa como flanco ascendente para la activación de la petición de descarga.

PULMON_DESCARGA		<DFB>
<entradas>		
CARGA_IN	1	BOOL
DESCARGA_IN	2	BOOL
INTERIOR_REST	3	BOOL
INTERIOR_MOVE	4	BOOL
ENTRADA_MOVE	5	BOOL
ENTRADA_READY	6	BOOL
MIN	7	BOOL
DIR06	8	BOOL
<salidas>		
DESCARGA_OUT	1	EBOOL
<entradas/salidas>		
<público>		
<privado>		
<secciones>		
DESCARGA		<ST>

Figura 11. Tipo DFB pulmón descarga.

2.3.4. PARÁMETROS RETENEDOR PULMÓN

El siguiente programa permite el mismo algoritmo que el que se hace en el bloque de funciones derivado “RETENEDOR_BASICO”, excepto con unos parámetros de entrada y salida específicos para la carga y descarga.

RETENEDOR_PULMON		<DFB>	
<entradas>			
REST	1	BOOL	
READY	2	BOOL	
MOVE	3	BOOL	
BLOQ	4	BOOL	
NEXT_REST	6	BOOL	
DI	7	BOOL	
CARGA	8	BOOL	
DESCARGA	9	BOOL	
FIN_CARGA	10	BOOL	
FIN_DESCARGA	11	BOOL	
<salidas>			
SET_REST	1	BOOL	
SET_READY	2	BOOL	
SET_MOVE	3	BOOL	
RESET_REST	4	BOOL	
RESET_READY	5	BOOL	
RESET_MOVE	6	BOOL	
CICLO_DESCARGA	7	BOOL	
CICLO_CARGA	8	BOOL	
<entradas/salidas>			
<público>			
<privado>			
FILTRO_SENSOR		TON	
<secciones>			
RETENEDOR_PULM...		<LD>	

Figura 12. Tipo DFB retenedor pulmón.

Parámetros de entrada específicos:

- **CARGA:** parámetro referente a la petición de carga. En el programa de aplicación se conectará a la salida de la DFB "PETICIÓN_PULMON_CARGA". Detecta el flanco ascendente.
- **DESCARGA:** parámetro referente a la petición de descarga. En el programa de aplicación se conectará a la salida de la DFB "PETICIÓN_PULMON_DESCARGA". Detecta el flanco ascendente.
- **FIN_CARGA:** parámetro que finaliza el ciclo de carga. En el programa de aplicación se conectará a la variable "CICLO_CARGA_PULMON" escrita por el PLC PULMON. Detecta el flanco descendente.
- **FIN_DESCARGA:** parámetro que finaliza el ciclo de descarga. En el programa de aplicación se conectará a la variable "DIR06" escrita por el PLC PULMON del sensor inductivo DIR06. Detecta el flanco descendente.

Parámetros de salida específicos:

- **CICLO_CARGA:** parámetro del ciclo de carga. Permite bloquear DIR05 durante el ciclo de carga.
- **CICLO_DESCARGA:** parámetro del ciclo de descarga. Permite bloquear DIR05 durante el ciclo de descarga.

2.4. ESTRUCTURAS PULMÓN

2.4.1. ESTRUCTURAS PASO CADENA

Parámetros de entrada:

- **PETICION:** Petición para la activación del ciclo que permite el movimiento de la cadena.
- **S_LENTA:** parámetro referente al sensor fotoeléctrico que permite activar la velocidad lenta.
- **BLOQ:** parámetro de bloqueo del ciclo de la cadena. Se usa para bloquear un paso tras otro.
- **PASOS:** Cantidad de pasos a realizar en una sola petición.

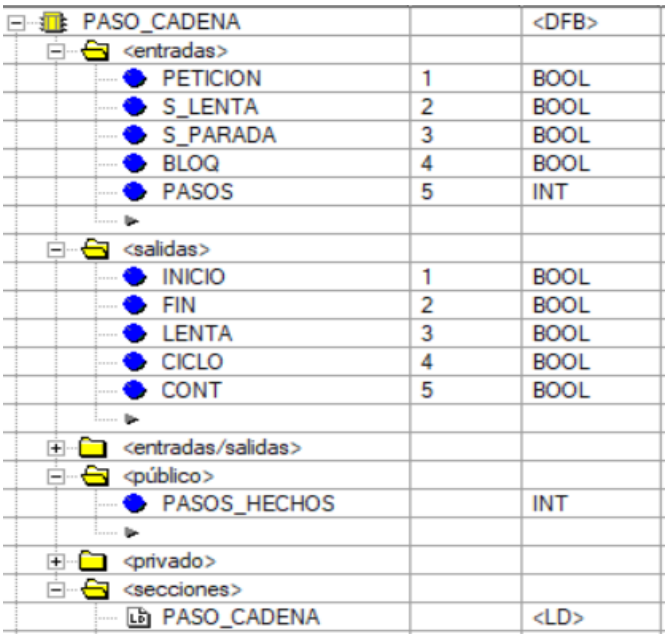
Parámetros de salida:

- **INICIO:** activa el movimiento de la cadena.
- **FIN:** indica el final de movimiento de la cadena. El flanco ascendente indica el fin de un paso.
- **LENTA:** activa la velocidad lenta de la cadena.
- **CICLO:** parámetro del ciclo de movimiento de la cadena**.** Permanece activo hasta que se hayan realizado todos los pasos indicados en el parámetro de entrada "PASOS".

- **CONT:** emite un pulso por cada paso realizado. Permite incrementar o decrementar el contador de pasos de la cadena "CONTADOR_PASOS_CADENA" y el puntero de la matriz de almacenamiento de id de bandejas "PILA.POSICION".

Parámetros públicos:

- **PASOS_HECHOS:** contador de los pasos que se van realizando hasta llegar al establecido por el parámetro de entrada "PASOS".



PASO_CADENA		<DFB>
<entradas>		
PETICION	1	BOOL
S_LENTA	2	BOOL
S_PARADA	3	BOOL
BLOQ	4	BOOL
PASOS	5	INT
<salidas>		
INICIO	1	BOOL
FIN	2	BOOL
LENTA	3	BOOL
CICLO	4	BOOL
CONT	5	BOOL
<entradas/salidas>		
<público>		
PASOS_HECHOS		INT
<privado>		
<secciones>		
PASO_CADENA		<LD>

Figura 13.Tipo DFB paso cadena.

2.4.2. ESTRUCTURAS PROCESO PULMON

Parámetros de entrada:

- **INICIO:** activación el ciclo del proceso (carga o descarga).
- **FINAL:** permite finalizar el ciclo una vez llegado a la tercera etapa "ETAPA3". En el caso de la descarga se usa para que no se finalice el ciclo hasta que "DIR06_MOVE" pase de "1" a "0".
- **CADENA_ON:** al estar activa impide la activación del ciclo del proceso.
- **CADENA_OFF:** Permite resetear la variable de salida "PROCESO" en la segunda etapa "ETAPA2".
- **S_PALE:** impide el inicio de ciclo cuando está activa. Se usa para impedir el proceso de descarga en caso de que no haya una bandeja a descargar "PULMON_VACIO".
- **S_ARRIBA:** activa la segunda etapa "ETAPA2". Se conecta el sensor superior del actuador lineal.
- **S_ABAJO:** activa la primera etapa "ETAPA1". Se conecta al sensor inferior del actuador lineal.

Parámetros de salida:

- **BAJAR:** Activa la bajada del actuador lineal.
- **SUBIR:** Activa la subida del actuador lineal.
- **CICLO:** ciclo del proceso. Al estar activa habilita las etapas y bloquea la petición de un nuevo proceso hasta que no se haya finalizado este ciclo.
- **CONT:** emite un pulso para contar cada proceso realizado. Se ha utilizado para incrementar o decrementar la variable "CONTADOR_BANDEJAS".
- **PROCESO:** activa e indica el inicio del proceso para dejar el proceso en espera hasta que se finalice. En este caso el proceso viene dado por la activación de la cadena retenedora.

Parámetros públicos[4]:

- **ETAPA1:** primera etapa definida por la subida del actuador lineal arriba.
- **ETAPA2:** segunda etapa definida por la realización de un proceso. En este caso el movimiento de un paso de la cadena.
- **ETAPA3:** última etapa definida por la bajada del actuador lineal hasta que se finalice a través del parámetro "FIN".

PROCESO_PULMON		<DFB>
<entradas>		
INICIO	1	BOOL
FINAL	2	BOOL
CADENA_ON	3	BOOL
CADENA_OFF	4	BOOL
S_PALE	5	BOOL
S_ARRIBA	6	BOOL
S_ABAJO	7	BOOL
<salidas>		
BAJAR	1	BOOL
SUBIR	2	BOOL
CICLO	3	BOOL
CONTADOR	4	BOOL
PROCESO	5	BOOL
<entradas/salidas>		
<público>		
ETAPA3		BOOL
ETAPA1		BOOL
ETAPA2		BOOL
<privado>		
<secciones>		
PROCESO		<LD>

Figura 14.Tipo DFB proceso pulmón.

[4] *Las etapas del proceso se han puesto en parámetros públicos para depurar el programa. En una aplicación real industrial, podrían poner-se cómo privadas para que no se tuviera acceso a ellas.*

2.4.3. ESTRUCTURAS PROCESO PULMON

Parámetros de entrada:

- **INCREMENTA:** en cada pulso de entrada incrementa la "PILA" con el id del parámetro R_PALE.
- **DECREMENTA:** en cada pulso de entrada decrementa la variable de la posición "1" de la tabla y almacena el ID en la variable de salida W_PALE.
- **R_PALE**: parámetro que se conecta a la variable a almacenar.

Parámetros de salida:

- **W_PALE:** parámetro que se conecta a la variable a descargar.

Parámetros de entrada/salida:

- **PILA:** parámetro que permite modificar la estructura "PILA".

Parámetro privado:

- **i:** parámetro iterativo para desplazar el contenido de la tabla de la estructura "PILA" en caso de que se incremente o decremente.

PILA_PALE		<DFB>
<entradas>		
INCREMENTA	1	BOOL
DECREMENTA	2	BOOL
R_PALE	4	INT
<salidas>		
W_PALE	4	INT
<entradas/salidas>		
PILA	3	PILA
<público>		
<privado>		
i		INT
<secciones>		
PILA		<ST>

Figura 15. DFB pila palé.

2.5. SECCIÓN MANUAL

Todas las actuaciones manuales del Pulmón quedan inhabilitadas en caso de que se haya iniciado la descarga de palés en serie “INICIO_SERIE”.

La activación de la subida de la cadena, en ciclo manual, se produce si los dos sensores superiores no detectan palé, “PULMON_LLENO” y “PULMON_LLENO2”, por tanto, el Pulmón no está lleno. Si no está bajando “BAJAR_CADENA_MANUAL” y se está manteniendo pulsado el botón “SUBIR_CADENA_bo”.

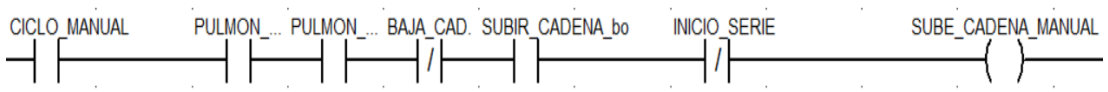


Figura 16. Ladder sección manual pulmón [1].

La activación de la bajada de la cadena se realiza mediante el botón” BAJA_CADENA_bo”, siempre que no se esté subiendo la cadena.

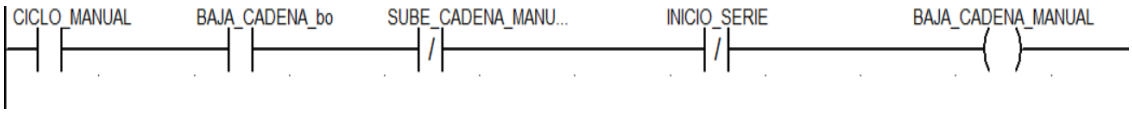


Figura 17. Ladder sección manual pulmón [2].

La subida del actuador lineal se hace mediante el botón “SUBE_ACTUADOR_bo”, en el caso de que no se esté bajando “BAJA_BASE_MANUAL” o ya se haya llegado arriba “SENSOR_ ACTUADOR_ARRIBA”.

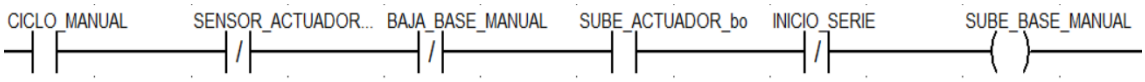


Figura 18. Ladder sección manual pulmón [3].

La bajada del actuador lineal se activa a través del botón “BAJA_ACTUADOR_bo”, y la cadena no se ha puesto a subir “SUBE_BASE_MANUAL” o el actuador lineal ya ha llegado abajo “SENSOR_ACTUADOR_ABAJO”.

La variable “HABILITA_DIR06_MANUAL” se activa siempre que se mantenga el botón de bajada del actuador lineal y este ya se encuentre abajo. Esta variable nos permite desbloquear DIR06 en el programa de la línea CAN.

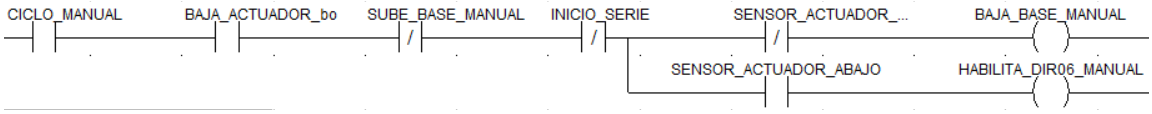


Figura 19. Ladder sección manual pulmón [4].

El cambio de velocidades a través del variador se realiza tanto para la bajada como para la subida.

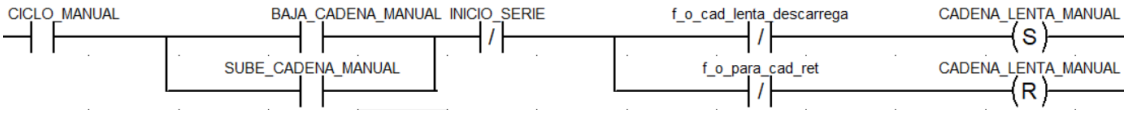


Figura 20. Ladder sección manual pulmón [5].

2.6. CONFIGURACIÓN Y COMUNICACIÓN PROGRAMA PULMON CON PROGRAMA CAN

2.6.1. CONFIGURACIÓN MEMORIAS DE LECTURA Y ESCRITURA PROGRAMA PULMÓN

Se han configurado las variables de lectura y escritura desde el PLC del Pulmón, puesto que este actúa como maestro y el PLC de la línea CAN como esclavo dentro de la red Ethernet Modbus TCP/IP.

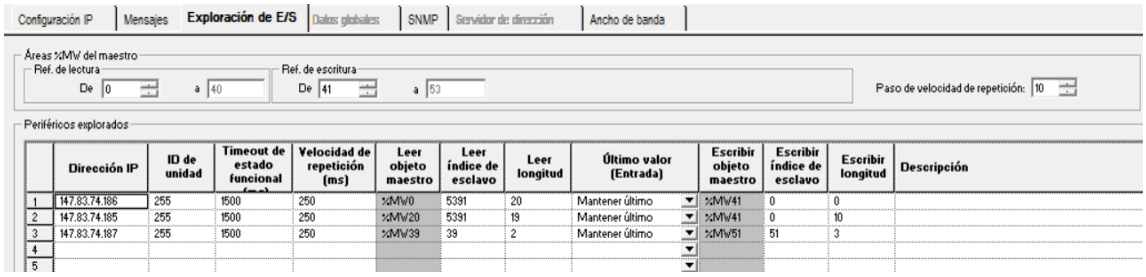


Figura 21. Exploración E/S pulmón.

En la pantalla de Exploración de entradas y salidas se ha puesto la dirección IP correspondiente al del PLC de la línea CAN “147.83.74.187”, y se ha puesto una velocidad de 250 ms para no sobrecargar demasiado la red.

De la misma forma se ha direccionado los espacios de memoria para la escritura y lectura de los datos de las islas Advantys con las direcciones IP de “147.83.74.185” y “147.83.74.186”. Para la lectura, se reservará un espacio de memoria comprendido entre %MW39 y %MW40 donde se almacenarán los bits leídos en la línea CAN de las variables con el mismo índice que estas.

Los espacios de memoria se han direccionados con las siguientes variables: las variables de petición de carga y descarga de bandeja, los estados de reposo, movimiento y bajada del retenedor DIR06, el estado en movimiento de DIR05 y la variable que permite almacenar el ID de una bandeja a cargar.

CARGA_BANDEJA	BOOL	%MW39.0
DESCARGA_BANDEJA	BOOL	%MW39.1
DIR06_REST	BOOL	%MW39.2
DIR06_MOVE	BOOL	%MW39.3
DIR06_DOWN	BOOL	%MW39.4
DIR05_MOVE	BOOL	%MW39.5
ID_BANDEJA_LECTURA	INT	%MW40

Figura 22. Variable elementales lectura.

Para la escritura, se reservará un espació de memoria desde %MW50 hasta %MW51 incluido, los cuales enviarán el contenido de forma respectiva a %MW50 y %MW51 de la línea CAN.

En la sección de salida se han conectado las variables de ciclo de las estructuras de carga y descarga con las palabras de escritura, ya que no se pueden direccionar directamente bit a bit en la pantalla de variables derivadas. “DT_retenedor” y “SENSOR_ACTUADOR_ABAJO” también se han conectado cada uno a un bit de la palabra de escritura, puesto que “DT_retenedor” y “SENSOR_ACTUADOR_ABAJO” ya están direccionados con las variables de entrada de la isla Advantys. Finalmente, los sensores indicadores del almacenamiento máximo y mínimo del Pulmón.

Figura 23. Ladder sección salidas pulmón [3].

Y las variables que se han direccionado en la pantalla de “Variables Elementales”.

CICLO_MANUAL	BOOL	%MW51.3
HABILITA_DIR06_MANUAL	BOOL	%MW51.4
INICIO_SERIE	BOOL	%MW51.8
CICLO_AUTOMATICO	BOOL	%MW51.9
EMERGENCIA	BOOL	%MW51.10
RESET_PULMON	BOOL	%MW51.11
ID_BANDEJA_ESCRITURA	INT	%MW52
CONTADOR_BANDEJAS	INT	%MW53

Figura 24. Variable elementales escritura.

2.6.2. CONFIGURACIÓN PULMÓN CICLO MANUAL

Para la configuración del ciclo manual, la siguiente tabla resume la correspondencia entre las variables:

Variables Pulmón Escritura CAN			
Línea Ethernet (Pulmón)		**Línea CAN**	
Variable	**Dirección**	**variable**	**Dirección**
CICLO_MANUAL	%MW51.3	PULMON_MANUAL	%MW51.3
HABILITA_DIR06_MANUAL	%MW51.4	HABILITA_DIR06_MANUAL	%MW51.4
SENSOR_ACTUADOR_ABAJO	%MW51.5	PULMON_PREPARADO	%MW51.5
INICIO_SERIE	%MW51.8	MANUAL_SERIE_DESCARGA	%MW51.8
CICLO_AUTOMATICO	%MW51.9	PULMON_AUTOMATICO	%MW51.9

Tabla 4. Variables pulmón escritura en CAN.

El envío de la variable "SENSOR_ACTUADOR_ABAJO" permite habilitar el paso de DIR05 a DIR06 si el actuador lineal se encuentra abajo.

Durante el ciclo de descarga en series "INICIO_SERIE" se bloqueará el retenedor DIR05[5] *a través de la variable "MANUAL_SERIE_DESCARGA" de la línea CAN.*

En la sección de "COMUNICACIONES" del programa de la línea CAN se ha programado el siguiente código:

[5] Véase en la memoria Figura 26. *Sección Ladder Línea DFB DIR05 Retenedor – 4.2.1.2 Bloque de Funciones Derivados Retenedor Pulmón y Condicionamiento de los Ciclos.*

```
(*PULMON MANUAL*)
IF PULMON_MANUAL = TRUE THEN
        DIR06_ESTADO.BLOQ:=TRUE;
ELSE DIR06_ESTADO.BLOQ:=FALSE;
END_IF;

IF HABILITA_DIR06_MANUAL = TRUE THEN
        DIR06_ESTADO.BLOQ:=FALSE;
END_IF;
```

Figura 25. Sección comunicaciones [1].

Cuando el Pulmón esté en ciclo manual, el retenedor DIR06 quedará bloqueado. Se desbloqueará cuando se mantenga pulsado el botón "BAJA_ACTUADOR_bo" y el actuador lineal esté abajo, o el ciclo de descarga serie "INICIO_SERIE" esté a "1". Entonces se activará "HABILITA_ DIR06_MANUAL".

2.6.3. CONFIGURACIÓN PULMÓN

La siguiente tabla [6]muestra la correspondencia entre las variables de lectura de la línea CAN y el Pulmón ,para la comunicación e integración del Pulmón en Los ciclos Automático y Lote de la línea CAN.

Variables Pulmón Lectura CAN			
Línea Ethernet (Pulmón)		**Línea CAN**	
Variable	**Dirección**	**variable**	**Dirección**
CARGA_BANDEJA	%MW39.0	CARGA_PULMON	%MW39.0
DESCARGA_BANDEJA	%MW39.1	DESCARGA_PULMON	%MW39.1
DIR06_REST	%MW39.2	DIR06_ESTADO.REST (activadora)	%MW39.2
DIR06_MOVE	%MW39.3	DIR06_ESTADO.MOVE (activadora)	%MW39.3
DIR06_DOWN	%MW39.4	DIR06_DOWN	%MW39.4
DIR05_MOVE	%MW39.5	DIR05_ESTADO.MOVE (activadora)	%MW39.5
EMERGENCIA_CAN	%MW39.6	EMERGENCY_LOCK_PULMON	%MW39.6
ID_BANDEJA_LECTURA	%MW40	ID_BANDEJA_PULMON	%MW40

Tabla 5. Variables pulmón escritura en CAN.

Durante el ciclo Automático o Lote, la activación de descarga o carga del pulmón se envía a través de la variable de petición "CARGA_PULMON" y "DESCARGA_PULMON". Y las variables de estado de reposo y movimiento de DIR06, así como la variable activadora de la bajada del

[6] *Las variables marcadas con (activadora) son variables que transfieren su estado a la memoria determinada.*

retenedor del Pulmón. También la variable de movimiento de DIR05 para que no se suba el actuador lineal y el id de la bandeja.

Tabla[7] de correspondencia entre las variables escritas por el Pulmón en la línea CAN:

Variables Pulmón Escritura CAN			
Línea Ethernet (Pulmón)		**Línea CAN**	
Variable	**Dirección**	**variable**	**Dirección**
CARGA.CICLO (activadora)	%MW51.0	FIN_CARGA (activada)	%MW51.1
DESCARGA.CICLO (activadora)	%MW51.1	FIN_DESCARGA (activada)	%MW51.1
DT_RETENEDOR (activadora)	%MW51.2	DIR06	%MW51.2
SENSOR_ACTUADOR_ABAJO (activadora)	%MW51.5	PULMON_PREPARADO	%MW51.5
PULMON_VACIO (activadora)	%MW51.6	PULMON_MIN	%MW51.6
PULMON_LLENO, PULMON_LLENO2 (activadoras)	%MW51.7	PULMON_MAX	%MW51.7
EMERGENCIA	%MW51.10	EMERGENCIA_PULMON	%MW51.10
ID_BANDEJA_ESCRITURA	%MW52	DIR06_ESTADO.PRODARRIVES.BANDEJA	%MW52
CONTADOR_BANDEJA	%MW53	BANDEJAS_PULMON	%MW53

Tabla 6. Variables pulmón escritura en CAN.

Se necesita escribir el estado del retenedor del Pulmón para el cambio de estados en la sección "LINEA" del PLC de la línea CAN. El estado del sensor actuador lineal inferior para habilitar el paso de DIR05 a DIR06. Los estados de almacenamiento del Pulmón para impedir la carga en caso de que el pulmón esté a su máximo y la descarga en caso de que esté sin bandejas. Finalmente el id de la bandeja para la descarga y la cantidad de bandejas almacenadas.

[7] *Las variables marcadas como (activada) son activadas por el contenido de la dirección de memoria*

2.7. PANTALLAS DE OPERADOR

2.7.1. PANTALLA OPERADOR LÍNEA CAN

La siguiente leyenda describe diferentes indicadores y elementos de la pantalla de operación:

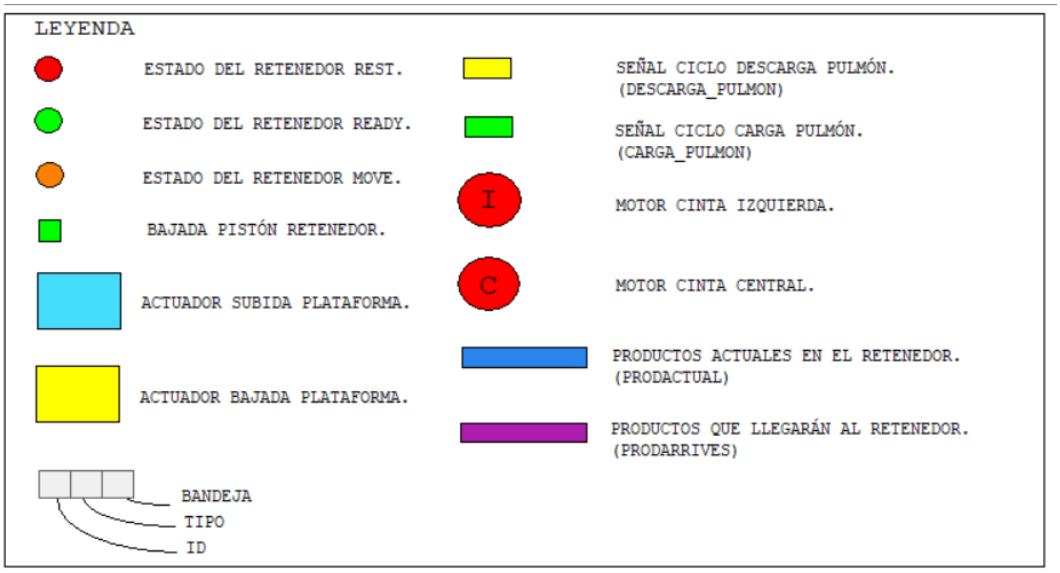


Figura 26. Pantalla operador línea CAN [1].

- **N.A:** los botones N.A simulan los sensores inductivos normalmente abiertos.
- **N.C:** los botones N.C simulan a los sensores inductivos normalmente cerrados.

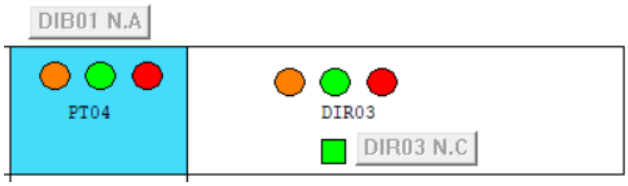


Figura 27. Pantalla operador línea CAN [2].

- **DESCARGA y CARGA:** son las señales referentes al “CICLO_CARGA_PULMON”, color verde cuando está a “1” y “CICLO_DESCARGA_PULMON”, color amarillo cuando está a “1”.
- **CICLO DESCARGA (C.D.P) y CICLO CARGA (C.C.P):** estos botones simulan la escritura en las variables “CICLO_CARGA_PULMON” y “CICLO_DESCARGA_PULMON”.
- **PRODUCTO A ALMACENAR:** visualiza la variable que será leída por el PLC del Pulmón.
- **BANDEJAS PULMÓN:** simula la variable escrita por el PLC Pulmón de la cantidad de bandejas que tiene almacenada.

- **SEÑAL PULMÓN LLENO y SEÑAL PULMÓN VACÍO:** simulan las variables escritas por el PLC Pulmón cuando este está lleno y cuando está vacío
- **ID BANDEJA DESCARGADA PULMÓN:** simula la variable escrita por el PLC Pulmón referente a la última bandeja descargada.
- **R.A.L:** simula la escritura de la variable del PLC Pulmón referente al sensor inferior del actuador lineal.

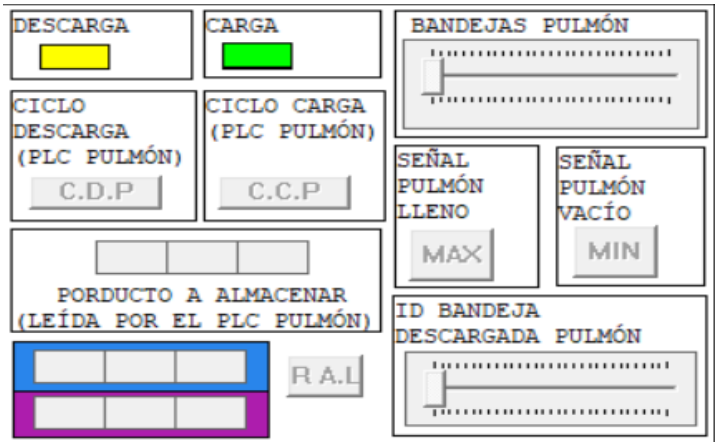


Figura 28. Pantalla operador línea CAN [3].

- **OCUPABILIDAD PROFIBUS:** Simulan la variable escrita por el PLC Profibus
- **BANDEJAS PROFIBUS:** Simulan la variable escrita por el PLC Profibus de la cantidad de bandejas ocupadas por cada retenedor o bandeja.
- **SETA EMERGENCIA:** simula el botón con enclavamiento de la seta de emergencia.
- **PULSADOR REARME:** simula el botón del pulsador rearme.

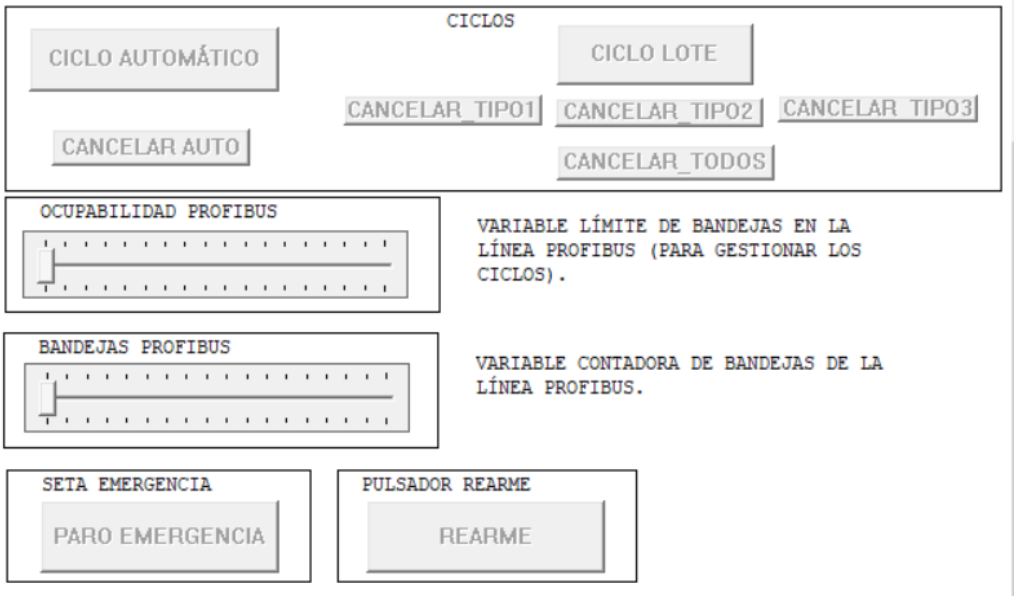


Figura 29. Pantalla operador línea CAN [4].

2.7.2. PANTALLA OPERADOR PULMÓN

La siguiente leyenda describe diferentes indicadores y elementos de la pantalla de operación:

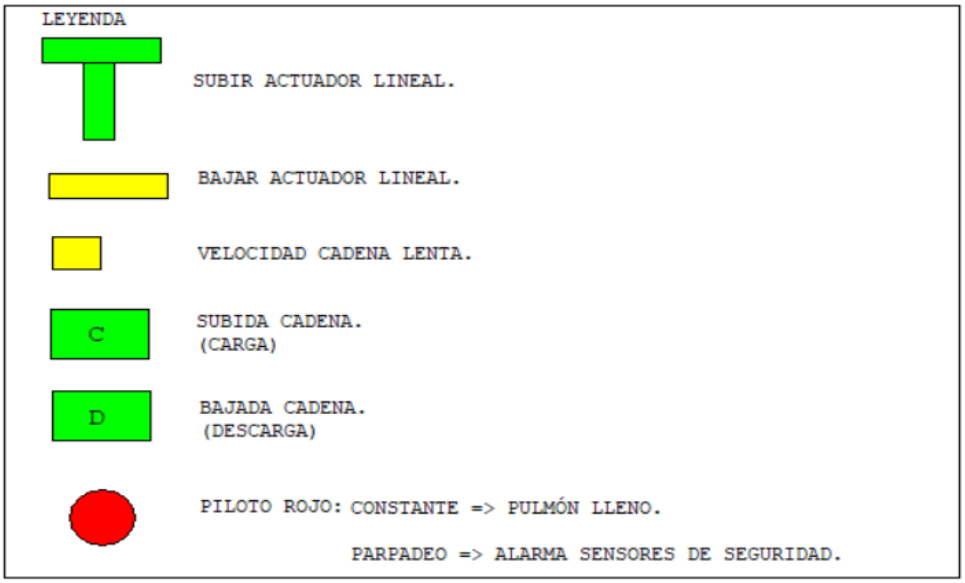


Figura 30. Pantalla operador pulmón [1].

- **EMERGENCIA:** simula el botón con enclavamiento de la seta de emergencia normalmente cerrado.
- **REARME:** simula el botón del pulsador rearme.
- **PUERTA 1, PURETS 1, TÉRMICO CADENA, BLOQUEO VERIADOR, TÉRMICO ACTUADOR LINEAL:** simulan los botones de los sensores de emergencia.
- **AUTOMÁTICO Y MANUAL:** simulan el switch de activación del ciclo automático o el manual.
- **DESCARGA SERIES:** simula el botón de la Dashboard de Node-red para descarga una tanda de bandejas en ciclo manual.
- **PASOS A DESCARGAR:** simula las bandejas que se descargarán en el ciclo de descarga de serie.
- **ID BANDEJA EN DIR06:** simula la variable leída en el PLC CAN para el almacenamiento del ID de la bandeja cuando se carga.

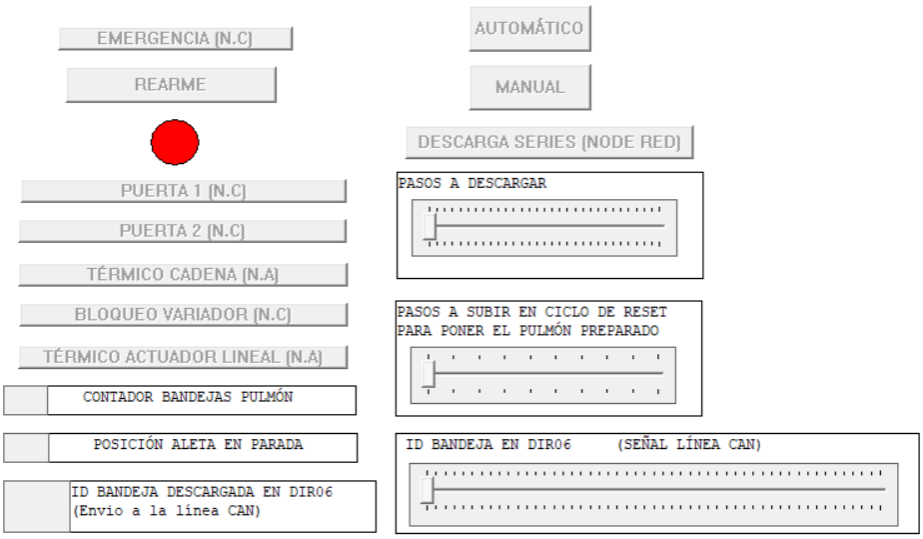


Figura 31. Pantalla operador pulmón [2].

- **CARGA LENTA, PARADA y DESCARGA LENTA:** simulan los sensores inductivos para la velocidad lenta en la carga, la parada de la cadena, y el cambio de velocidad lenta en la descarga.
- **SEÑAL DESCARGA y SEÑAL CARGA:** simulan las señales de petición leídas en el Pulmón CAN.
- **BOTON RESTE PULMÓN:** simula el botón de reset del Pulmón.
- **DIR06 MOVE, DIR06 REST y DIR05 MOVE:** simulan las variables de estado leídas en el PLC CAN.
- **ARRIBA y ABAJO:** simulan los sensores de final de carrera del actuador lineal.
- **DETECTOR BANDEJA:** simula en detector normalmente cerrado de la parte inferior del Pulmón que determina cuando el Pulmón está vacío o no.
- **DIR06:** simula el sensor inductivo normalmente cerrado del retenedor DIR06.

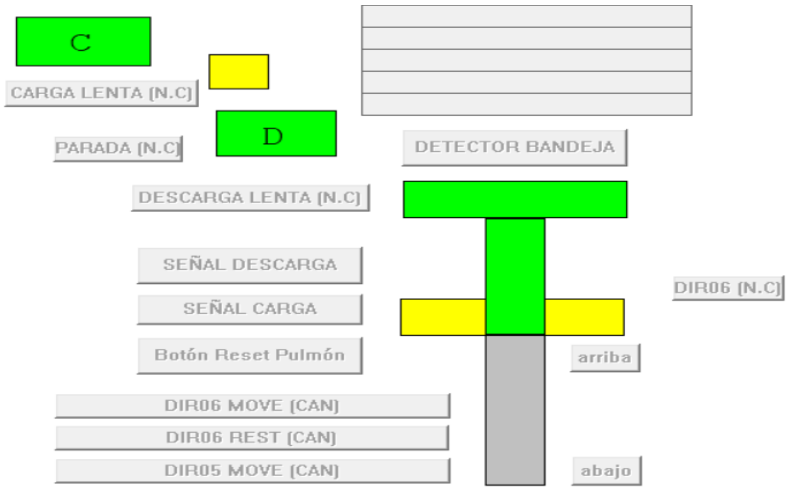


Figura 32. Pantalla operador pulmón [3].

3. FASE DE NODE-RED

3.1. ELEMENTOS DE NODE-RED

NODO MODBUS TCP/IP

Nodo que permite comunicar-se con Modbus TCP, para poder realizar el traspaso de datos entre la Dashboard de Node-Red y el PLC CAN y PLC PULMÓN.

El nodo se configura tal y como se ilustra en la siguiente imagen:

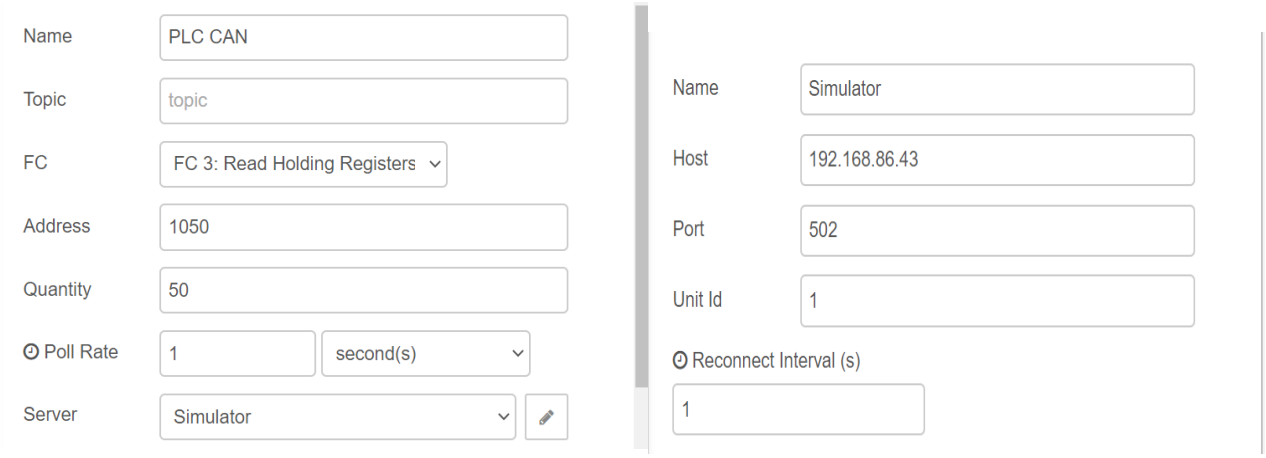


Figura 33.Propiedades modbustcp (izquierda) y modbustcp-server (derecha).

En el caso del nodo “modbustcp” tenemos:

- **Name:** nombre que le asignamos a nuestro nodo.
- **FC (function code):** tipo de función de lectura, en este caso FC3 permite leer Registros de palabras de memoria interna.
- **Adress**: dirección del espacio de memoria de origen a leer, en este caso %MW1050.
- **Quantity:** cantidad de espacios a memoria a leer, en este caso 50 palabras a partir de %MW1050.
- **Poll Rate:** frecuencia de lectura por parte del nodo, en este caso cada segundo.
- **Server:** servidor del nodo cliente, este sería el PLC CAN y el PLC PULMÓN. En este caso se ha realizado mediante el servidor local, para la simulación.

En el caso del nodo “modbustcp-server”:

- **Name:** nombre asignado al servidor.
- **Host:** termino que se refiere a los dispositivos conectados a una red que proveen y utilizan servicios sobre ella. En este caso se ha puesto la dirección IP de host utilizada

por el simulador de Unity Pro. Tanto para el Pulmón como para la línea CAN se configurarían con sus IP respectivas “147.83.74.184” y “147.83.74.187”.

- **Port:** puerto por donde el nodo escucha i el puerto 502 está reservado para las aplicaciones Modbus.
- **Unit Id:** permite identificar un servidor local remoto que no está en la misma red. En este caso se obvia al poner 1 o 0.

Se pueden realizar múltiples lecturas de diferentes registros poniendo varios nodos Modbus TCP o inyectando un código JSON con dos objetos[8] que definen las características del nodo.

VARIABLE DE CONTEXTO FUNCIONES DE NODO

Se han usado las variables de contexto que permiten almacenar datos:

- **Flow**: el contexto que alcance a nivel de flujo. En una pestaña de flujo, donde se programan los nodos.
- **Global**: el contexto que alcanza a nivel global. Accesible por los otros flujos.

Para almacenar los datos se usan las funciones de tipo “setter”**:**

flow.set(“dato_guardado”, dato_a_aguardar);

Y para extraer los datos guardados usando las funciones de tipo “getter”:

flow.get(“dato_guardado”);

[8] Véase en la memoria 5.4.4 FLUJO PULMON, donde se configuran dos objetos JSON para dos ciclos de lectura de Modbus TCP.

3.2. VARIABLES DE ANÁLISIS

Se han identificado las variables de interés analítico, y se han direccionado los espacios de memoria:

General

- **Variables estados de los retenedores (CAN):** permiten poder supervisar los estados de los retenedores y plataformas de las líneas, y los productos que hay actualmente. **%MW1050-%MW1193.**
- **PRODUCTO_FINALIZADO (CAN):** último producto finalizado. La variable "ESTACION" de DIR04 permite saber el tiempo que dura la extracción de este producto. **%MW1194-%MW1196.**
- **CONTADOR_CANTIDAD_M_PRIMAS (CAN):** cantidad actual de materias primas que quedan por abastecer. **%MW1197.**

Ciclo Automático

- **CONTADOR_AUT_PROD_CARGA_1 (CAN):** cantidad de productos del tipo 1 cargados en la línea en el ciclo automático. **%MW1198.**
- **CONTADOR_AUT_PROD_CARGA_2 (CAN):** cantidad de productos del tipo 2 cargados en la línea en el ciclo automático. **%MW1199.**
- **CONTADOR_AUT_PROD_CARGA_3 (CAN):** cantidad de productos del tipo 3 cargados en la línea en el ciclo automático. **%MW1200.**

Ciclo Lote

- **LOTE_CARGADO (CAN):** contiene las variables contadoras del tipo de producto cargado en la línea. Permite informar los tipos de productos en los que ya se han cargado las materias primas para fabricarse. **%MW1201-%MW1204.**
- **LOTE_FINALIZADO (CAN):** contiene las variables contadoras del tipo de producto finalizado en la estación DIR04. Permite informar de los tipos de productos que ya se han realizado. **%MW1205-%MW1208.**

Contadores de las estaciones

- **CONTADOR_E_1 (CAN):** piezas realizadas por la estación 1. **%MW1209.**
- **CONTADOR_E_2 (CAN):** piezas realizadas por la estación 2. **%MW1210.**
- **CONTADOR_E_3 (CAN):** piezas realizadas por la estación 3. **%MW1211.**

3.3. IDENTIFICACIÓN VARIABLES DE CONTROL

General

- **FLUJO_MATERIAS_PRIMAS (CAN):** variable que contiene la matriz con el orden de producción de tipo de productos. Permite gestionar la frecuencia de producción de los productos. **%MW1212- %MW1222.**
- **CAMBIO_FLUJO_PRODUCCION:** variable que permite bloquear la plataforma PT06 hasta que se haya cambiado el flujo de producción. **%MW1223.0.**
- **FINALIZADO_CAMBIO_FLUJO_P:** variable que permite el cambio del flujo de producción actual. **%MW1223.1.**
- **ABASTECIMIENTO_M_PRIMAS:** variable de confirmación del abastecimiento de la cantidad de materias primas. **%MW1223.2.**
- **CANTIDAD_M_PRIMAS_ABASTECIDAS:** variable que tiene el número de materias primas a cargar en la línea. **%MW1224.**
- **OCUPABILIDAD_PROFIBUS:** variable de control que determina a partir de cuantas bandejas se considera la línea Profibus, lo suficientemente ocupada para tomar decisiones de gestión. **%MW1225.**

Ciclo Automático

- **CICLO_AUTOMATICO:** variable de inicio del ciclo automático. **%MW1223.3.**
- **CANCELAR_AUTO:** variable de cancelación del ciclo automático. **%MW1223.4.**

Ciclo Lote

- **CICLO_LOTE:** variable de inicio del ciclo de lote. **%MW1223.5.**
- **LOTE_CANTIDAD:** variable que contiene la cantidad de cada tipo de producto que se tiene que realizar. **%MW1226-%MW1229.**
- **CANCELAR_TIPO1:** variable de cancelación del producto de tipo 1. **%MW1223.6.**
- **CANCELAR_TIPO2:** variable de cancelación del producto de tipo 2. **%MW1223.7.**
- **CANCELAR_TIPO3:** variable de cancelación del producto de tipo 3. **%MW1223.8.**
- **CANCELAR_TODOS:** variable de cancelación de todos los productos. **%MW1223.9.**

3.4. VARIABLES INFLUXDB

Se ha usado el nodo "InfluxDB" para almacenar los datos en la base de datos "DATABASE1" para tener un registro de los diferentes datos de interés:

- **Materias cargadas:** permitirán saber con qué frecuencia y cantidad de materias primas se abastece la línea. Para gestionar las previsiones y comandas.
- **Producto final extraído:** permite identificar el último producto finalizado.
- **Lote cargado y finalizado:** para tener un registro de la cantidad de tipos de productos cargados y finalizados por Lote.
- **Variable "ESTACION"[9] de DIR04 y PT06:** al almacenarse en series temporales se podrá determinar cuánto tiempo están activas estas estaciones.
- **Automático cargado y finalizado:** para tener un registro de la cantidad de tipos de productos cargados y finalizados en ciclo automático.
- **Estados "MOVE" retenedores y plataformas:** estos datos pueden determinar durante cuánto tiempo están en movimiento. Al almacenarse los datos en series temporales se podría determinar la velocidad, y poder prevenir fallos en los variadores frecuencias.
- **ID última bandeja descargada del Pulmón:** para tener identificada la última bandeja descargada del Pulmón

[9] Se ha considerado usar el valor de la variable "ESTADO" de DIR04 y PT06 para determinar los tiempos de introducción de materias primas y extracción del producto, porque son estaciones de entrada y salida. Los tiempos de interés, por tanto, son aproximados puesto que la lectura de las variables del PLC (Poll Rate) son a 1 segundo. El tiempo de producción en las estaciones de la línea Profibus, sí que podrían ser más críticos dependiendo del tipo de producción. En dicho caso se almacenaría el tiempo del contador del programa del PLC.