# PROYECTO FINAL DE CARRERA

# AUTOMATIZACIÓN DE UNA LÍNEA PARA PROCESADO DE PESCADO

**Alumno: Abel Ortega Carrasco**
**NIU: 2109421**
**Tutor: Juan José Ramos**

# ÍNDICE

## *1.* PLANTEAMIENTO DEL PROYECTO A DESARROLLAR

Este proyecto surge de una necesidad industrial por parte de una empresa dedicada al procesado y envasado de pescado fresco y congelado (Elaborados Freiremar S.A). Para satisfacer esta necesidad, se decidió colocar una línea de proceso totalmente automatizada.

Dicha automatización, consta de 3 partes totalmente diferenciales pero relacionadas entre ellas, puesto que sin alguna de estas partes la línea no podría funcionar correctamente:

- **Software de control (PLC)**: Se encarga de controlar y actuar sobre la máquina con los diferentes sensores, células de carga, motores…
- **SCADA en Visual Basic (Pantalla táctil)**: Se encarga de mantener una comunicación continua con el software de control, y va almacenando todos los datos referentes a la línea y sus rendimientos.
- **Software de Consultas**: Se instala en un PC de oficinas y se utiliza para sacar informes con todos los datos obtenidos del SCADA en Visual Basic.

Esta línea consta de 2 partes, una primera para el fileteado manual del fletan y otra para el envase para la directa comercialización del mismo. Además de la maquinaria implantada por Palinox, esta línea está alimentada por una Badeer encargada de filetear el fletan o el rodaballo para su posterior repaso manual.

En la primera parte de la línea, como se puede observar en la figura 1, se sitúan 5 operarios de la empresa para filetear y en la segunda parte de la línea (figura 2) se colocan 3 operarios más que se dedican a envasar. De esta forma, se consigue que el proceso sea lo mas ágil posible y a su vez que la empresa Freiremar se ahorre costes con lo que los beneficios son mayores, o bien el precio del mercado es menor.

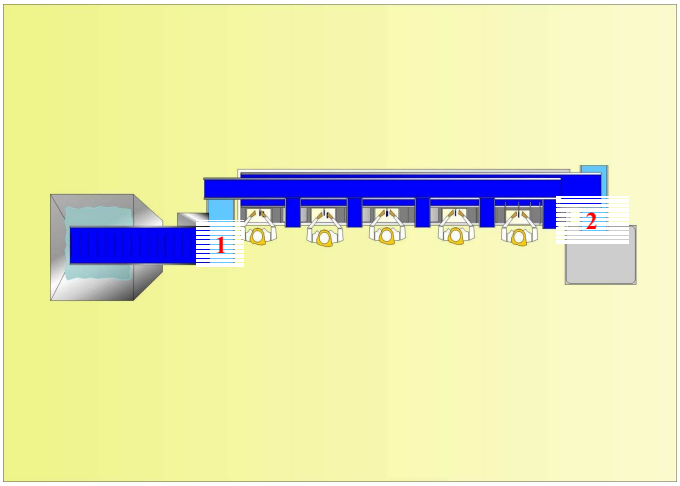



Figura 1: 1ª parte de la línea (fileteado con control de rendimientos)

En la figura 1, se puede observar como es la primera parte de la línea anteriormente comentada, de la cual se procede a explicar su funcionamiento. La primera observación que hay que hacer, es que existen 2 cintas (números 1 y 2) que están situadas al principio y al final de la línea, que son básculas provistas de un moto tambor inversor que permite descargar el género hacia un lado o hacia el otro dependiendo de las necesidades del momento y del espacio disponible en la sala.

Como se puede observar, la línea comienza con una cinta que descarga género desde una tolva de alimentación hacia la primera de las básculas comentadas anteriormente (número 1). Esta cinta tiene que estar en marcha, siempre y cuando no se supere el valor de 10 Kg. en la báscula de entrada (más adelante se aclarará el porque se hacen pesadas de 10 Kg.). Una vez se ha llegado a este peso, este género (que se ha almacenado en la báscula) se tiene que servir a uno de los operarios que están fileteando, siempre habiendo guardado previamente el valor de la pesada.

Si esto se realiza por primera vez desde que se ha iniciado el trabajo, se realiza un ciclo donde se le otorga género a todos y cada uno de los operarios, por orden, para que de esta manera se pueda asegurar que al arrancar la línea, todos los operarios tienen género para poder comenzar con su trabajo. Si no se encuentra en el primer ciclo, cada operario tiene un pulsador de Seta, que sirve para demandar género cada vez que se estén quedando sin nada para filetear. Cabe destacar que esta báscula está colocada principalmente para saber en cada momento el peso que ha recibido cada operario, y así poder realizar un control de rendimientos individual.

Una vez finalizada la primera descarga de género a cada operario, existe una cinta inferior y 5 cintas de tamaño reducido (se encargan de transportar el género que ha fileteado cada operario hacia la báscula de final de línea) para así poder controlar también el peso de filete limpio que ha acabado cada operario.

Además, también se utiliza la segunda báscula para poder llenar bañeras con un número de kilos específico, así siempre saldrán con los kilos necesarios y controlando el número de bañeras que se han realizado cada día.

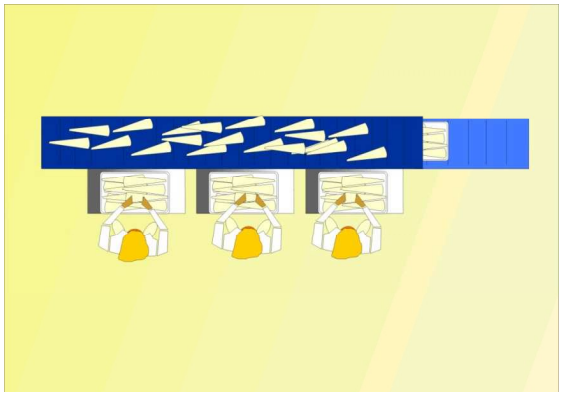


Figura 2: 2ª parte de la línea (envasado manual con control de cajas y peso)

En la segunda parte de la línea, que se puede observar en la figura 2, trabajan 3 operarios que se dedican, básicamente, a envasar el producto final en cajas de unos determinados kilos. Para esto, cada uno de ellos dispone de una báscula a la cual se le introducen dos umbrales de peso, un peso inferior y un peso superior, para hacer las cajas con un pequeño margen con respecto a un peso prefijado.

El encargado de los operarios ha de saber en todo momento el número de kilos de cada uno de los operarios, así como el número de cajas que ha realizado durante todo el día, para así poder pagar dependiendo únicamente de la producción. Todo esto se explica con más detalle en los siguientes apartados, profundizando mas en las funcionalidades de cada uno de los programas realizados para el correcto y deseado funcionamiento de la línea.

Por tanto, una vez descrita la necesidad que hay que satisfacer y el funcionamiento de la línea, apuntamos los principales objetivos de dicho proyecto:

- Realización del programa de control de un PLC Siemens para el correcto funcionamiento de los actuadotes y elementos de la línea.
- Realización de un programa en Visual Basic para el control de los operarios y para mantener una información actualizada del estado de la máquina.
- Realización de un programa de consultas para poder obtener los rendimientos de los operarios desde las oficinas de la empresa.
- Correcto funcionamiento e implantación de la línea en el cliente.
- Total automatización de una línea que, en un principio, se había planteado totalmente manual.

## *2.* COMPONENTES DEL SISTEMA

Este apartado explica con detalle los diversos elementos utilizados en la línea para el correcto funcionamiento de la manera más fiable posible. Dentro de estos elementos se diferencian los sistemas de pesaje, los autómatas programables con sus módulos y finalmente la pantalla táctil.

Las básculas utilizadas en la primera parte de la línea son unas básculas de la casa UTILCELL (SMART), que se pueden observar en la figura 4, la misma que las células de carga (mayor fiabilidad), ya que este modelo se caracteriza por su gran estabilidad y precisión que garantiza su salida analógica utilizada en el autómata, en este caso 0-10V. También hay que destacar la gran facilidad que tiene para poder realizar una auto-tara, después de cada pesada, desde el autómata.

Para poder convertir los 0-10V de la salida analógica en peso, lo que se ha hecho ha sido utilizar un módulo de entradas analógicas en el autómata para que transforme dicha señal a una cantidad de puntos entre 0 y 32768. Una vez ya está transformado, lo único que queda es utilizar dichos los puntos correspondientes a la tara y a un peso conocido para poder hacer una simple división y saber a cuanto equivale cada punto. En la figura 3 se puede observar un diagrama donde se explica gráficamente:

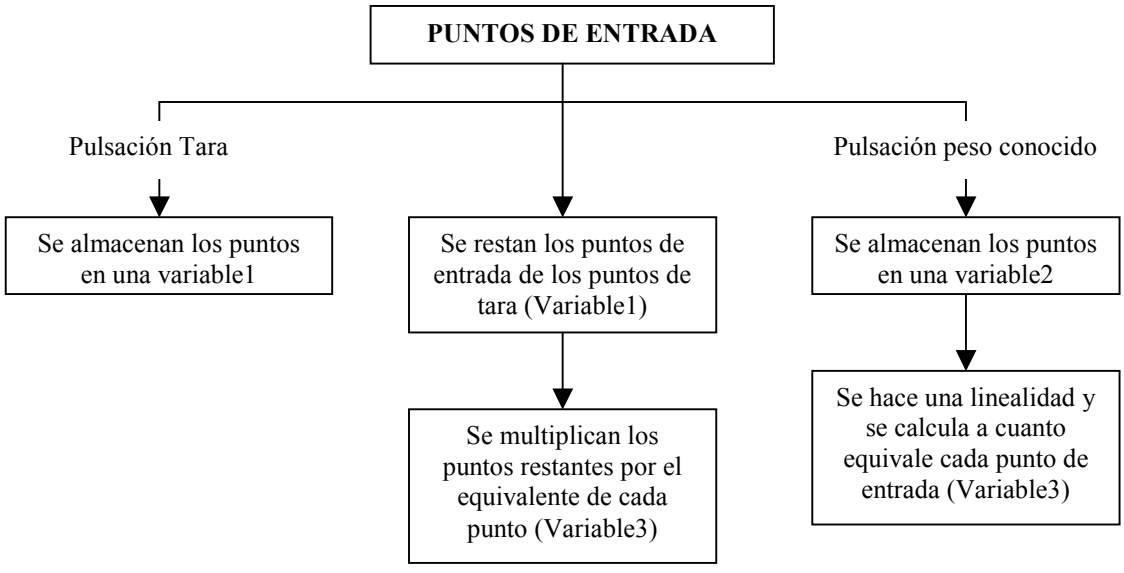



Figura 3: Diagrama de tratamiento a la señal analógica para su transformación a peso.

A continuación se muestran sus características detalladas:

- Indicador digital de pesaje de **última tecnología** electrónica.
- **Homologado** según **OIML R76** y **EN 45501**, monorrango o multirrango, 6000 divisiones.
- **Configuración y ajuste por teclado.**
- **Alta resolución**: A/D 24 bits, 16 millones de divisiones internas; 100.000 divisiones externas.
- **Puerto RS-232 bidireccional** para comunicaciones a PC o impresora.
- **Protección** aumentada contra **interferencias electromagnéticas**, mediante filtros, ferritas y blindajes adecuados.
- **Corrección de linealidad.**


Figura 4: Visor digital de peso SMART (Utilcell)

En cuanto a las básculas de la segunda parte de la línea, hay que decir que son de la marca EPELSA (figura 6) y que fueron las elegidas debido a que en esta parte lo que se pretende hacer es un llenado de cajas, es decir, un "Over-Under".

En este modo "Over-Under", se le asigna a la báscula un nivel bajo y un nivel alto y entre estos valores (a expensas de otros factores externos) el peso de la caja es el correcto. Además de tener esta funcionalidad, la báscula elegida también dispone de una salida 0-10V (tratada igual que las de la primera parte de la línea) y de una placa de relés (contiene 4 salidas) que dan una señal libre de potencial dependiendo de si el peso se encuentra en cero, por debajo del nivel bajo, dentro de los límites o por encima del nivel alto.

En la figura 5, se muestra un diagrama explicando como se utilizan las salidas digitales que contiene la báscula para encender un pequeño semáforo.

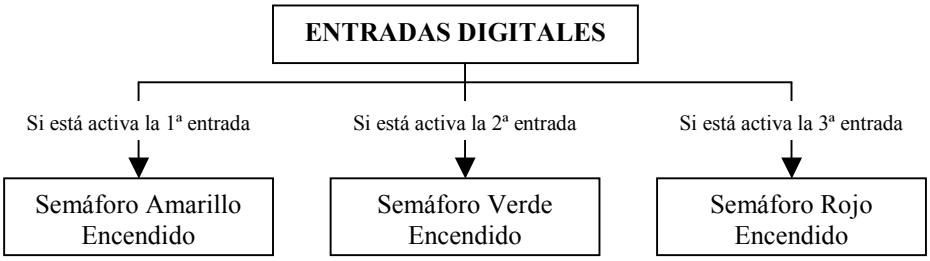



Figura 5: Diagrama de tratamiento de las salidas digitales de las básculas.

A continuación se muestran las características técnicas de estas básculas:

- **Visor multifunción**: Tara-Peso, cuentapiezas, pesacamiones, controlador +/- y pesa-ruedas.
- Posibilidad de conectar **hasta 4 células** de carga.
- **Visor de seguridad intrínseca.**
- **1.000.000 de puntos** de resolución interna.
- Hasta **100.000 divisiones de ajuste** a nivel externo.


Figura 6: Visor multifunción de peso CN-10 (EPELSA)

Finalmente, se describe el modelo de autómata utilizado para realizar dicho proyecto y los diferentes módulos que han hecho falta.

El autómata utilizado para realizar la automatización de la línea ha sido el modelo CPU 226 REL 02.00 de la marca Siemens (figura 7). Se decidió utilizar este debido a que es uno de los más fiables del mercado y por su gran facilidad de incorporación de módulos, ya sean de entradas y salidas digitales o bien de entradas y salidas analógicas. Otro factor que hizo que al final se colocase este modelo fue que habitualmente se trabaja con autómatas Siemens, por lo que la programación es más familiar que no con Omron o Allan-Bradley.

Además de la propia CPU 226, que incluye 16 Kbytes de memoria programable y 2 puertos de comunicación (uno de ellos utilizado para la comunicación con el PC), se ha necesitado la utilización de diferentes módulos:

- Módulo 4 EA: Módulo de 4 entradas analógicas. (figura 8)
- Módulo 16E / 16S: Módulo de 16 entradas y 16 salidas digitales. (figura 9)

A continuación se muestran las características técnicas del autómata y módulos:

### *2.1. Autómata CPU 226 Siemens*

- Alimentación a **24 VDC**.
- **10 KBytes** de memoria de datos y **24 KBytes** de memoria de programa.
- Tiempo de ciclo de scan: **0,22µs**.
- **256 temporizadores** programables.
- 2 puertos de comunicación **RS-485 y PPI**.

Figura 7: CPU 226 programable (SIEMENS)

### *2.2. Módulo 4 Entradas analógicas EM223*

- **Consumo de 60mA** del bus de la CPU.
- Resolución de la señal analógica de **12 bits**.
- Tiempo de conversión de analógica a puntos: **250µs**.
- Rango de señales convertidas de **-32.000 a 32.000 puntos**.

Figura 8: Módulo 4 Entradas Analógicas EM223 (SIEMENS)

### *2.3. Módulo de 16 Entradas digtales / 16 Salidas digitales EM231*

- Alimentación a **24 VDC**.
- **Consumo de 160 mA** del bus de CPU.
- **16 entradas** digitales y **16 salidas** digitales **aisladas galvánicamente**.
- Retardo de entradas de **4,5ms**.

Figura 9: Módulo 16 Salidas/Entradas Digitales EM231 (SIEMENS)

## *3.* RED DE PETRI BINARIA

REPOSO MÁQUINA

Inicio programa pantalla táctil

ESPERA CONFIRMACIÓN MARCHA

Pulsación botón de marcha colocado en máquina

**REPARTO GÉNERO**

**FUNCIONALIDAD SW**

ARRANQUE CINTA ENTRADA

Llegada a 8Kg. en báscula por señal analógica

DETENER CINTA Y ENVIO DE TRAMA PC

***INTERACCIÓN ENTRE PC Y PLC***

Recibe trama de confirmación

***INTERACCIÓN ENTRE PC Y PLC***

ARRANQUE CINTA PESO Y CINTA DISTRIBUCIÓN

Confirmación cinta marcha

APERTURA TRAMPILLA OPERARIO (i)

Reparto a todos los operarios

ESTADO ESPERA PETICIÓN

Petición por pulsador

**RECOGIDA GÉNERO FILETEADO**

Espera pulsación confirmación (bañera colocada)

ARRANQUE CINTA INFERIOR (Recogida)

Contador usuario a descargar (1-5)

ARRANQUE CINTA INDIVIDUAL

Tiempo para llevar género

DETENER CINTA Y ENVIAR TRAMA PC

***INTERACCION ENTRE PC Y PLC***

***INTERACCION ENTRE PC Y PLC***

Recepción confirmación

ARRANQUE CINTA PESO

ESPERA RECIBIMIENTO SEÑAL AUTÓMATA

Recibe trama comunicación para peso

ENVIO CONFIRMACIÓN A PLC

Tratamiento de datos

ALMACENAMIENTO DATOS Y CÁLCULOS

Recepción trama

ENVIO CONFIRMACIÓN

Tratamiento de datos

ALMACENAMIENTO DATOS Y CÁLCULOS

**REPOSO MÁQUINA ENVASADO**

Inicio programa pantalla táctil

ESPERA CONFIRMACIÓN MARCHA

Pulsación botón de marcha de la zona de envasado

ESPERA SEÑAL LECTURA PESO AUTÓMATA

ARRANQUE CINTA REPARTO GÉNERO

*INTERACCIÓN PC-PLC*

Recepción de trama del PLC

SE ALMACENAN LOS DATOS DE PESO RECIBIDOS

Pulsador Operario2

Pulsador Operario1

Pulsador Operario3

COMPROVACIÓN SI ESTÁ ENTRE LÍMITES

Está OK

No está en límites

ENVIO DE TRAMA HACIA PC

SEÑAL ACÚSTICA DURANTE 3 SEG.

Datos correctos

ENVIO DE TRAMA DE VALIDACIÓN

INTERACCIÓN PC-PLC

Recepción trama validación

## *4.* SOFTWARE DEL PLC SIEMENS

La función principal del programa realizado para el autómata Siemens, es la de actuar sobre los diversos componentes de la máquina (motores, pilotos,…), dependiendo del estado en el que se encuentren cada uno de ellos, y recibir cualquier orden desde la misma (pulsadores, peticiones, paro emergencia,…) lanzada por el operario. También se encarga de ir controlando el estado de todos los componentes del sistema, como la lectura constante del peso de las diferentes básculas y la señal de los disyuntores, así como la comunicación con la pantalla táctil para controlar en todo momento en que situación se encuentra la línea.

La parte más destacada de este software, reside en la correcta configuración y utilización de la comunicación con una pantalla táctil, ya que esta es la encargada de gestionar todo lo que tiene que hacer el PLC, dependiendo de los diferentes datos que vaya recibiendo del mismo. Así mismo, el programa del autómata se puede subdividir en 3 partes importantes (comunicaciones, fileteado y envasado)

### *4.1. Comunicaciones*

El autómata consta de 2 puertos de comunicación (puerto 0 y 1), de los cuales se ha configurado el 0 como freeport[1] para la comunicación entre la pantalla táctil y el PLC. La configuración que se le aplicó fue paridad "even", 8 bits de datos y velocidad de transferencia de 9600 bps (figura 10).

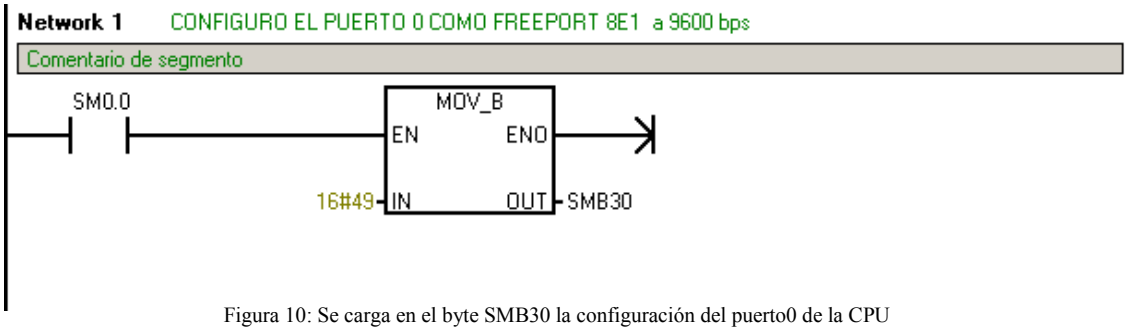



Figura 10: Se carga en el byte SMB30 la configuración del puerto0 de la CPU

Además de estas configuraciones básicas, también hay que configurar otros parámetros para el correcto control de la recepción de mensajes a través del puerto 0. Dicha configuración se realiza insertando un valor hexadecimal en el símbolo SMB87.

[1] **Freeport:** Comunicación programable por el usuario. El programa define la velocidad de transferencia, los bits por carácter, la paridad y el protocolo.

**SMB87 Control de recepción de mensajes**

**SM87.1 0 = ignorar condición BREAK; 1 = utilizar condición BREAK al comienzo de mensajes.**

**SM87.2 0 = ignorar SMW92, 1 = finalizar recepción si se excede el tiempo en SMW92.**

**SM87.3 0 = temporizador entre caracteres, 1 = temporizador de mensajes.**

**SM87.4 0 = ignorar SMW90, 1 = usar SMW90 para detectar condición de inactividad.**

**SM87.5 0 = ignorar SMB89, 1 = usar SMB89 para detectar fin del mensaje.**

**SM87.6 0 = ignorar SMB88, 1 = usar SMB88 para detectar comienzo del mensaje.**

**SM87.7 0 = recepción de mensajes inhibida, 1 = recepción de mensajes habilitada.**

En la configuración necesaria para este caso, hay activos los bits SM87.7, SM87.6 y SM87.4. Esto da un número en binario (1101 0000) que pasado a valor hexadecimal equivale al valor insertado: ***D0*** (figura 11).

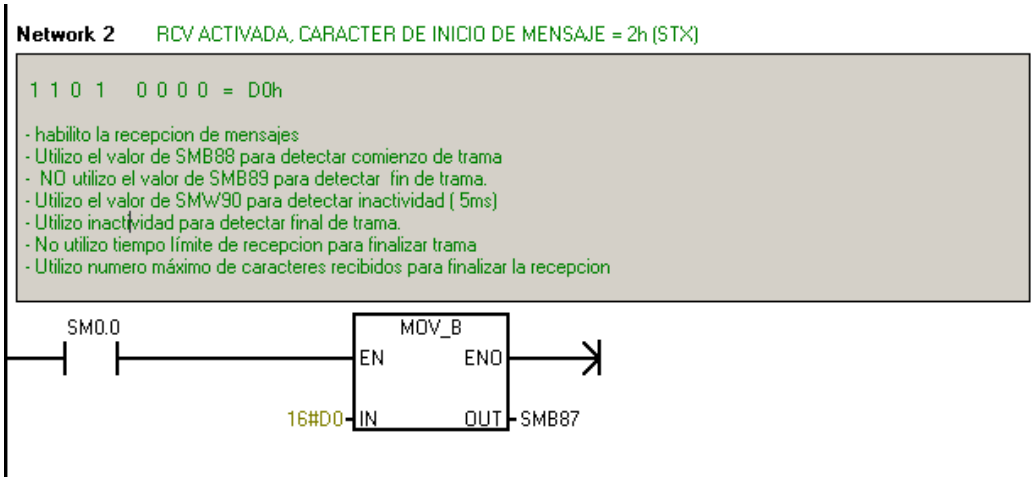



Figura 11: Se configura el valor D0 dentro del byte de configuración SMB87

Además del símbolo SMB87, también se deben configurar como habilitados los bytes SMB88, SMW90 y SMB94, que corresponden a la detección del comienzo de una trama, al tiempo de inactividad del puerto y al número máximo de caracteres recibidos respectivamente. En cuanto al byte SMB94, se ha configurado a 19 bytes, valor suficiente para todos los datos que se necesitan enviar en cada una de las diferentes tramas utilizadas para la comunicación entre PLC y PC.

Cada vez que dicho software recibe una trama de comunicación, tiene asignada una trama de respuesta correspondiente que rellena con los datos necesarios por el PC y se encarga de enviarla lo más rápidamente posible.

Se pueden consultar todas las tramas de comunicación, así como sus respectivas respuestas en el **Anexo1**.

Cada una de las diferentes tramas de comunicación, contiene 1 byte dedicado exclusivamente al control de errores en la transmisión, utilizando un CRC basado en una XOR de todos los bytes recibidos en la trama, esto quiere decir una XOR de los 18 bytes recibidos sin contar el CRC.

En la figura 12 se muestra como se calcula dicho CRC comentado anteriormente, en el software del autómata.

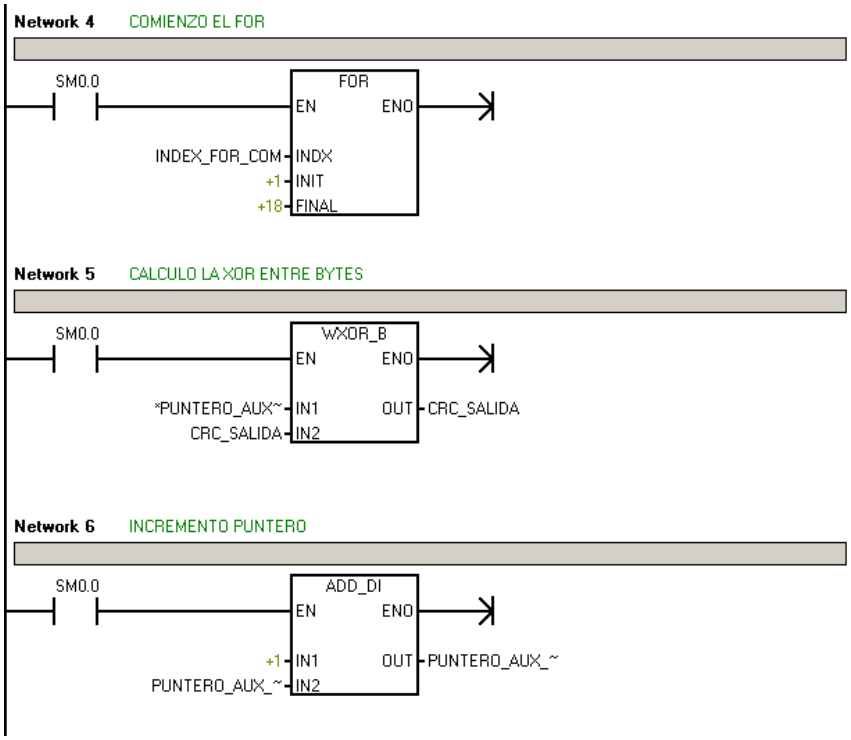



Figura 12: Se calcula el CRC con una XOR de todos los bytes recibidos en la trama

### *4.2. Funcionamiento 1ª parte (fileteado)*

En lo referente al funcionamiento de la primera parte de la línea ***(reparto género)***, destacar que consta de una cinta de alimentación de producto, que desemboca directamente en una báscula donde se van haciendo pesadas de 5 Kg (figura 13), controladas directamente por el autómata que está recibiendo constantemente el peso a través de la señal analógica. Si existe alguna petición de género por parte de algún operario (cada uno tiene un pulsador para realizarla), se descarga el género de la báscula en una cinta con trampillas abatibles y se abre la trampilla necesaria dependiendo del operario que ha demandado producto. La cinta de la báscula se queda en marcha solo el tiempo necesario para dar una vuelta entera, que es el tiempo mínimo para garantizar que no se queda nada de género encima. Cuando pasa ese tiempo, se realiza una auto

tara sobre el visor de peso para que, si se ha quedado algún resto, no afecte a la siguiente pesada.

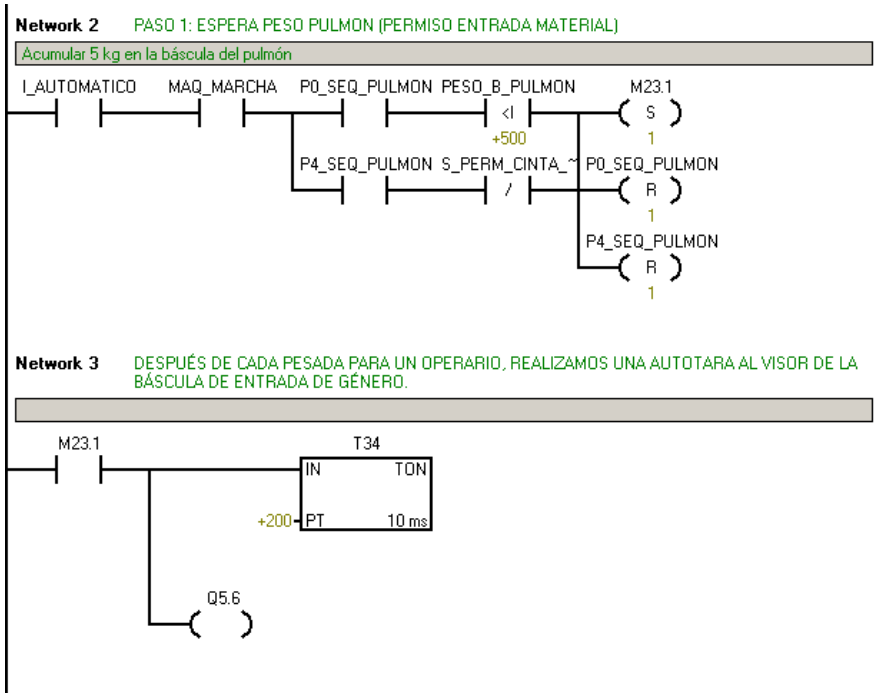



Figura 13: Secuencia que para la cinta una vez se ha llegado a 5Kg. y hace una autotara. La línea tiene que estar en automático (I_AUTOMATICO) y con la máquina en marcha (MAQ_MARCHA).

Cuando se pone en marcha la línea, se ejecuta un primer ciclo de descarga de género hacia cada uno de los operarios, es decir, como si cada uno de ellos hubiese pulsado el botón de ***Petición***. De esta manera, se agiliza mucho el proceso inicial ya que así cuando llegan los operarios, tienen sus cajas correspondientes llenas de producto y no es necesario que esperen. Si esto no fuese así, el último operario en pedir género estaría bastante tiempo sin trabajar hasta que no se cumpliesen las 4 peticiones anteriores (en el caso extremo de que estuviesen trabajando los 5 operarios).

Una vez servido todo el género de este primer ciclo, siempre se queda en espera de recibir más peticiones por parte de los operarios. En caso que haya más de una, se van almacenando en una tabla (realmente es una cola FIFO) para ir sirviendo correlativamente una vez que se tenga el peso preparado (las pesadas de 5 Kg.).

Si no existe ninguna petición, se deja preparado el peso encima de la báscula hasta que algún operario realice alguna, así no se pierde tiempo cada vez que un operario necesita mas género para seguir con su trabajo. Mientras se realiza la pesada o cuando tenemos el peso de 5Kg encima de la báscula, la cinta de alimentación de producto se detiene para no falsear el peso.

Hay que destacar la manera en como se guardan las peticiones en la cola. Cada operario dispone de un código interno (100 para el operario1, 200 para el operario2,…) y cada vez que realizan una petición, se va guardando dicho número para posteriormente servir el producto (figura 14). Cada vez que un operario añade una petición, tienen que pasar 60 segundos para que el mismo pueda realizar otra.

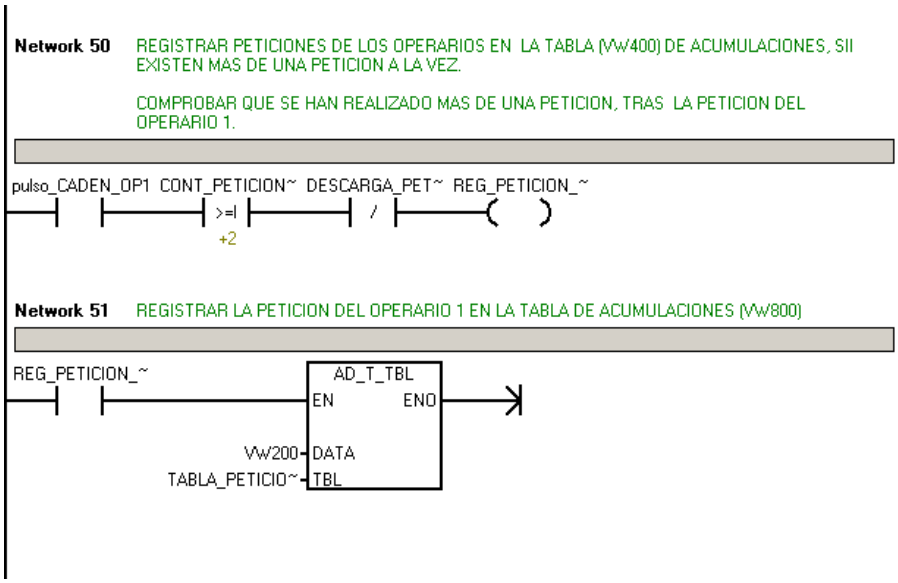



Figura 14: Se guarda el valor 100 (Operario1) en la tabla de peticiones cuando se pulsa el botón (REG_PETICION_OP1).

Sin embargo, hay que tener en cuenta que si existe alguna petición realizada y durante 3 minutos no se alcanzan los 5 Kg. definidos, se sirve el género que haya en ese momento encima de la báscula, ya que existe la posibilidad de que se haya acabado todo el producto y sea el resto que queda.

Una vez finalizado todo el primer ciclo de repartición de género a los operarios, se comienza con la parte de recogida de producto final, que se encuentra fileteado y repasado, y con el llenado de bañeras automático.

En esa parte de la línea ***(recogida género fileteado)***, cada operario dispone de una pequeña cinta donde puede almacenar el producto listo para el llenado de bañeras. Este género se va recogiendo secuencialmente a todos los trabajadores, para así poder controlar en todo momento el peso total realizado por cada uno de ellos independientemente. Cada vez que se tiene que recoger el género, lo único que hay que hacer es poner en marcha la cinta de cada uno de ellos y la cinta transportadora hacia la báscula.

Para poder controlar en cada momento el operario que ha descargado sobre la báscula, se lleva un control interno en el PLC con una variable que se va incrementando

con cada pesada y que, cada vez que se pesa el producto del último operario (Operario 5), se vuelve a quedar con valor 1 para volver a empezar (figura 15).

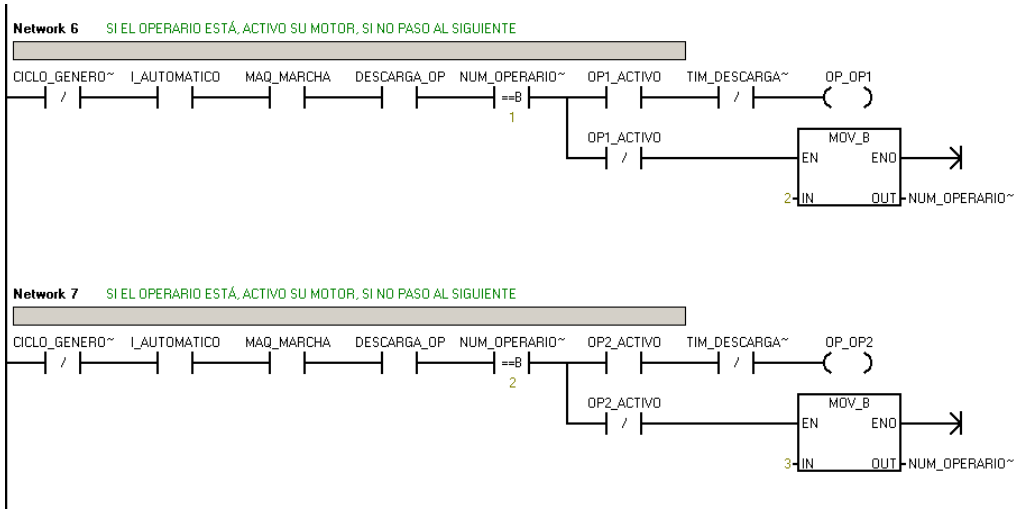



Figura 15: Se comprueba el operario a descargar y, si está trabajando (OP1_ACTIVO), se activa su motor (OP_OP1)

Cuando el producto llega a la báscula, se mantiene unos segundos antes de validar la pesada, para así estabilizarlo. Una vez pasado este tiempo, se manda una señal hacia la pantalla táctil para que pueda almacenar el dato y descargar ese producto, ya pesado, hacia la bañera. En este momento se puede considerar que hay una interacción entre los diferentes elementos utilizados para la automatización (PC – PLC).

Para poder poner en marcha todo el proceso de descarga, se tiene que haber confirmado la bañera previamente para así asegurar que en ningún momento el producto, ya fileteado y repasado, pueda caer al suelo. La confirmación de la bañera simplemente se trata de un pulsador que activa, a su vez, una entrada digital en el autómata.

Este botón, que se encuentra situado encima de donde va colocada la bañera, y hay que pulsarlo una vez colocada en su posición. Cuando se llega al peso máximo de la bañera, configurado a través de la pantalla táctil antes de comenzar a trabajar, se vuelve a parar todo el proceso de recogida de producto hasta que se vuelve a confirmar la siguiente. Antes de retirarla, también se debe confirmar con otro pulsador, ya que se encuentra totalmente finalizada.

Una vez confirmada la colocación de la bañera, el programa tiene la posibilidad de ponerle un número determinado de litros de agua y otros tantos de salmuera (figura 16), todo configurado previamente en la pantalla táctil. Para poder contar el número de litros, lo único que hace falta es la colocación de un cuenta litros de agua y un cuenta litros teflonado para la salmuera, ya que es mucho más corrosiva.

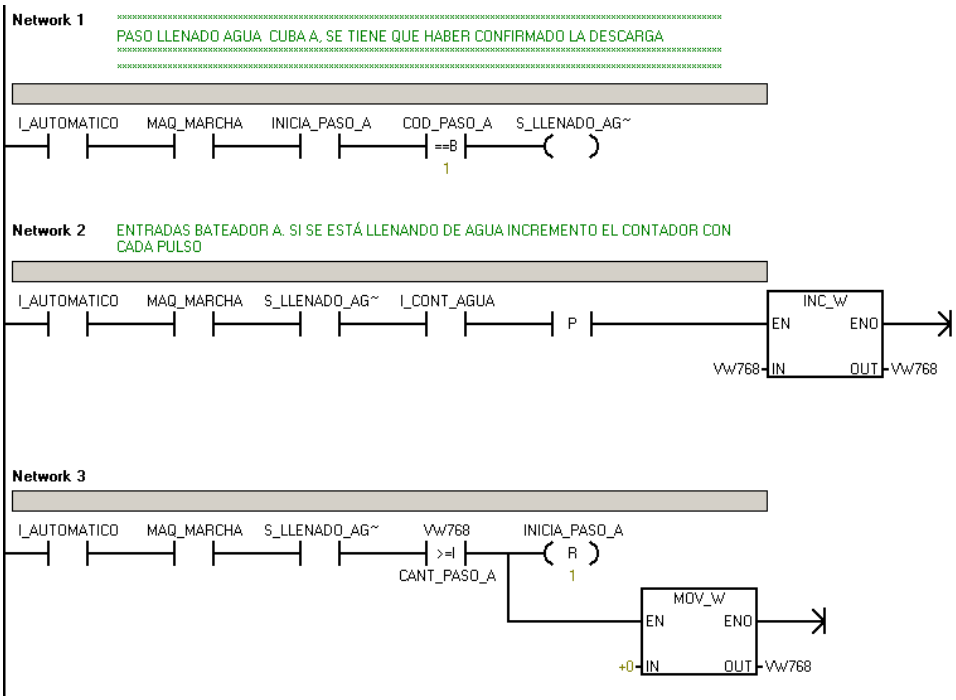



Figura 16: Si la línea se encuentra llenando de agua, se recogen los pulsos que lanza el cuenta litros y se almacenan

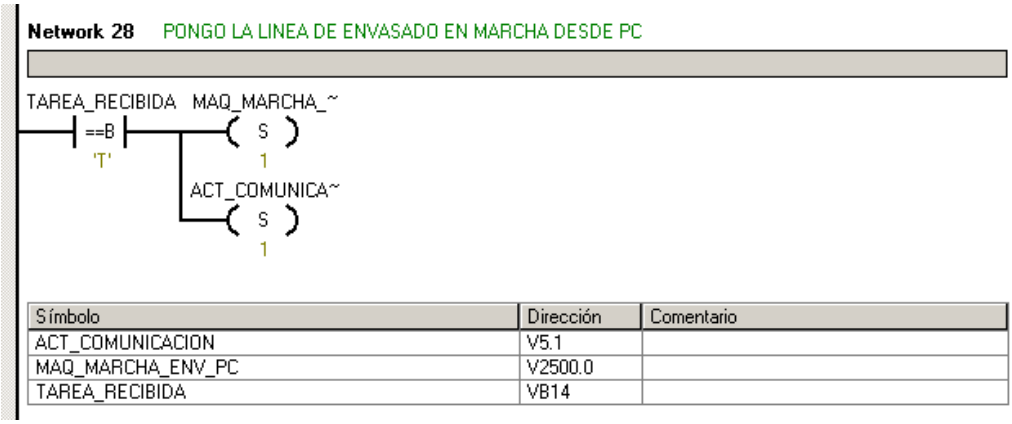
También existe la posibilidad de darle un cierto tiempo de bateo por aire justo cuando se acaba de llenar de producto. Esto se configuraría de la misma manera que se procede con los litros, es decir, guardando un tratamiento con los 3 parámetros (Agua, salmuera y minutos de bateo).

En esta parte es donde mas datos se guardan, puesto que hay que saber toda la producción de la línea, así como también la producción individual de todos y cada uno de los operarios y la merma introducida en el proceso. También hay que tener en cuenta que se guardan todos los datos referentes a los tratamientos que se les tiene que aplicar a los diferentes productos producidos en la línea.

### *4.3. Funcionamiento 2ª parte (envasado)*

En cuanto al envasado, destacar que consiste en una cinta de descarga de género desde una primera cuba hacia una cinta superior, que se encarga de transportar el género a baja velocidad a los operarios para que puedan ir llenando las cajas de producto, y una cinta inferior para ir sacando las cajas ya finalizadas y pesadas correctamente hacia un operario que se encarga de ir montando los palets.

Para poner en marcha la zona de envasado, hay que pulsar el botón de marcha en la pantalla táctil y así, se activa la subrutina del programa del autómata que se encarga de todas las maniobras de esta parte de la línea (figura 17).




| Símbolo | Dirección | Comentario |
|---|---|---|
| ACT_COMUNICACION | V5.1 | |
| MAQ_MARCHA_ENV_PC | V2500.0 | |
| TAREA_RECIBIDA | VB14 | |

Figura 17: Cuando se recibe la trama de marcha de envasado, se activa la marca de MAQ_MARCHA_ENV_PC

Las básculas colocadas en esta parte de la línea (CN-10 EPELSA) se encargan de dar una salida digital opto acoplada dependiendo del peso de la caja, si se encuentra por debajo del peso mínimo, entre los pesos o por encima del peso máximo (estos pesos están guardados como recetas dentro de la misma báscula para no tener que estar introduciendo los valores cada vez que se tiene que trabajar con u producto diferente). Mediante el autómata se van encendiendo unos pilotos, en función del estado de las salidas opto acopladas anteriormente mencionadas, para facilitar la visualización por parte del operario del estado de la caja que está llenando (ámbar, verde y rojo).

Además de estos pilotos, el autómata también controla el número de cajas y el peso de cada una de ellas que ha realizado cada operario. Para controlar dicha acción, cada operario dispone de un pulsador en forma de seta que deben accionar una vez se encuentra el peso dentro de los límites (piloto verde encendido).

Si por algún motivo, un operario pulsa este botón cuando todavía no se encuentra dentro de los límites, el valor de la caja no se le acumula al operario y se activa una alarma sonora durante 3 segundos para que el operario y el encargado de sala se den cuenta. Esta alarma sirve principalmente para que en las siguientes cajas intenten pulsar una vez tengan el peso bien estabilizado dentro del peso correcto. Cada vez que un operario pulsa cuando tiene la caja finalizada, el autómata se encarga de enviar el peso a la pantalla táctil y de notificarle que se ha acabado otra caja más para almacenarlas (figura 18).

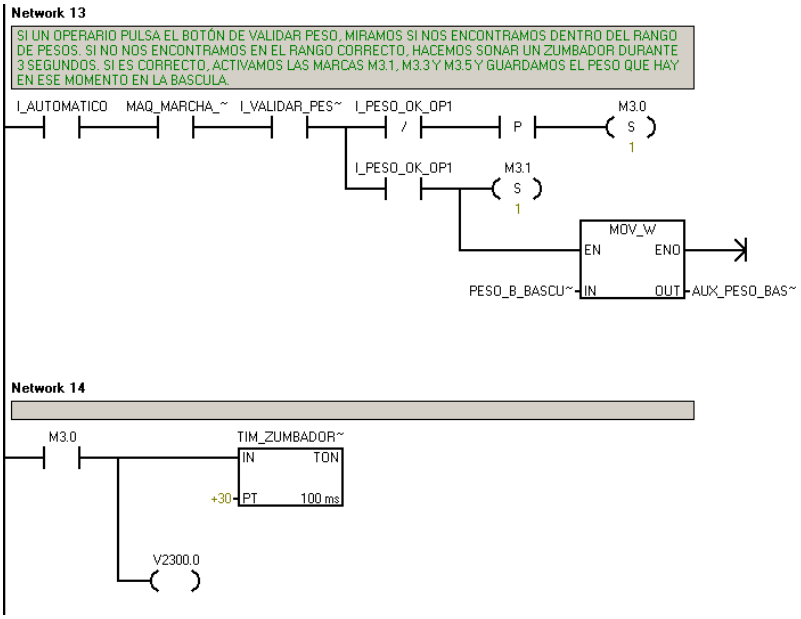



Figura 18: Si cuando pulsa un operario no se encuentra en peso correcto (I_PESO_OK), suena un zumbador 3seg.

En el **Anexo2** se puede consultar el listado con todas las entradas y salidas utilizadas en el autómata y su funcionalidad.

# 5. DESARROLLO DEL PROGRAMA DE CONTROL (PANTALLA TÁCTIL)

En la línea de proceso existe, además del autómata SIEMENS, una pantalla táctil con un ordenador integrado (Panel PC) para poder ir enviando y recibiendo los datos necesarios hacia el PLC y para tener un entorno gráfico para el uso de los operarios.

Con este programa, lo que se pretende conseguir es un control total sobre la línea de proceso a automatizar. Además de enviar y recibir los datos, también se encarga de ir almacenando todos los datos necesarios para tener todos los rendimientos de la línea, tanto de los operarios como del proceso en general.

Para poder conseguir todo esto, se puede subdividir el programa en 2 partes importantes, y sin una de las cuales no funcionaría bien nada del proceso (Comunicaciones, Base de datos).

## *5.1. Comunicaciones*

En este software, existen unas tramas de comunicación para poder tener una constante comunicación con el autómata. El estilo de trama es el mismo que en el plc, pero con la diferencia que el que envía siempre la primera es el PC, es decir, el autómata solo se dedica a satisfacer las peticiones que realiza el software de control. Si en algún momento fallan las comunicaciones del PC, la línea dejaría de funcionar porque el autómata no recibiría la confirmación de lo que tiene que hacer. En el **Anexo1** se pueden observar las tramas que se envían desde el Pc.

Para la configuración de la comunicación, se ha creado una librería OCX[2] con las principales funciones de la comunicación, para solo tener que hacer llamadas a las funciones creadas y así no tener ningún tipo de problema. Las principales funciones que componen esta ocx son las de abrir y cerrar el puerto de comunicación, aunque no son menos importantes las de envío, recepción y escucha de los eventos producidos en el puerto de comunicación.

A continuación se muestra una lista detallada y con una explicación de la funcionalidad de todas y cada una de las funciones que forman parte de la OCX. En el **Anexo3** se puede observar su código fuente.

[2] **Librería OCX:** Módulo independiente del programa principal, que contiene las principales funciones a utilizar en las comunicaciones. Tienen la ventaja que son muy portables entre unos lenguajes y otros.

- ***CommOpen()***: Función donde se puede configurar todos los parámetros necesarios para que la recepción del puerto sea la adecuada. Se tienen que configurar básicamente la longitud de trama, el número de puerto y todos los datos referentes a la velocidad y la paridad. Una vez configurado todo el puerto, también se encarga de abrir el puerto y dejarlo a punto para el envio y recepción de tramas desde el autómata.
- ***CommClose()***: Función que se encarga única y exclusivamente de cerrar el puerto de comunicaciones una vez que se ha finalizado la ejecución del programa. Si por algún error no se cerrase el puerto, ninguna otra aplicación lo podría utilizar aunque el programa estuviese cerrado.
- ***Envia()***: Función que monta la trama antes de ser enviada por el puerto de comunicación. Delante de la trama de datos enviada desde el PC, se le coloca una pequeña cabecera de 4 bytes para saber que los datos que están llegando pertenecen, efectivamente, a la línea. Una vez montada la trama, se deja colocada en el puerto para ser enviada.
- ***OnComm()***: Esta función es la encargada de recibir cualquier evento que se da en el puerto de comunicaciones, es decir, cuando el PC está esperando una respuesta por parte del autómata, se entera a través de esta función puesto que detecta un evento y entonces pasa a leer lo que hay en el puerto de entrada.
- ***Gestion_Recivido()***: Es la encargada de desglosar toda la trama y mirar, principalmente, si el CRC que se envía en el último byte de la misma cuadra con el que teóricamente se calcula. Para calcular este CRC, lo único que hay que hacer es una XOR entre los 19 bytes de la trama. Si por algún motivo ha habido un fallo de comunicación y no son iguales el CRC calculado y enviado, se lanza una excepción y no se mira el resto de la trama.

Además de estas funciones, existe una función principal para las comunicaciones, que es la encargada de recibir las tramas utilizando las funciones del componente OCX anteriormente citadas. También tiene implementado un **WATCH DOG**[3] para saber en que momento se ha recibido la trama y si hace un timeout. Siempre se realizan 3 reintentos por si la comunicación ha fallado, y se lanzan los errores en caso de que siga fallando.

[3] **Watch Dog:** Temporizador que se utiliza para la detección de algún fallo, o como en este caso, que detecta una inactividad en el puerto de comunicaciones transcurrido un cierto tiempo.

En el **Anexo3** también podemos observar el código fuente de la principal función de las comunicaciones. Esta función (***talxXd***) se utiliza constantemente, tanto cuando se tiene que enviar una trama hacia el autómata como cuando se tiene que recibir alguna trama desde el PLC.

Cuando se recibe una trama, se lanza el evento (generado por Gestión_Recivido) para que la función Ctm800Comm_RecEv pueda comprobar la integridad de la trama con el CRC, por si ha llegado erróneamente. Una vez comprobado, se puede continuar con el proceso normal, a través de la función talkXD.

Un claro ejemplo de envío y recepción de tramas se puede observar cuando un operario realiza una pesada, y el PC tiene que recibir dicho peso para poder ir controlando la producción. A continuación se puede observar dicha función:

```
Public Sub Tarea_Peso_Operario(ByVal iNodo As Integer)
Dim sAux As String
Dim iNumLinea As Integer
Dim dPeso As Double
Dim iOperario As Integer
Dim Descripcion As String
Dim i As Integer
On Error GoTo RutinaError
iNumLinea = Numero_Linea_Por_Nodo(iNodo)
sAux = talkXd(iNodo, "O") 'Envio de la trama "O" hacia el autómata
If sAux = "ERRORCOMM" Then
  Error_Comm iNodo, T_PESO_OPERARIO
  Call FinalTarea
  Exit Sub
End If
iOperario = CInt(Asc(Mid(sAux, 5, 1))) 'Recepción de datos desde el autómata
dPeso = CInt(lPasaALong2Bytes(Asc(Mid(sAux, 6, 1)), Asc(Mid(sAux, 7, 1)))) / 100
'Para que NUNCA nos pueda dar un peso negativo en los históricos
If dPeso < 0 Then
    dPeso = 0
End If
'Se opera con los datos para obtener la producción de cada operario
ParamLinea(iNumLinea).dPeso_Operario(iOperario) =
Format(ParamLinea(iNumLinea).dPeso_Operario(iOperario) + dPeso, "###0.00")
ParamLinea(iNumLinea).dProduccion_Operarios =
Format(ParamLinea(iNumLinea).dProduccion_Operarios + dPeso, "###0.00")
'AHORA MIRO EL PESO/HORA DE CADA OPERARIO
If Not ParamLinea(iNumLinea).tTiempo_MarchaOperario(iOperario) = 0 Then
    ParamLinea(iNumLinea).dKgH(iOperario) =
Format(ParamLinea(iNumLinea).dPeso_Operario(iOperario) /
(CDbl(ParamLinea(iNumLinea).tTiempo_MarchaOperario(iOperario)) * 24), "###0.0")End If
```

```
'Guardo el peso/hora operario para la gráfica y el número de pesada
ParamLinea(iNumLinea).iCnt_Samples_Peso(iOperario) =
ParamLinea(iNumLinea).iCnt_Samples_Peso(iOperario) + 1
ParamLinea(iNumLinea).iPeso_Hora_Operario(ParamLinea(iNumLinea).iCnt_Samples_Peso(iOperario),
iOperario) = ParamLinea(iNumLinea).dKgH(iOperario)
'EN FUNCIÓN DEL BATEADOR QUE SE ESTÉ LLENANDO SUMO EL PESO
If ParamLinea(iNumLinea).bLlenandoBateadorA Then
    ParamLinea(iNumLinea).dPesoBateadorA = Format(ParamLinea(iNumLinea).dPesoBateadorA +
dPeso, "###0.00")
End If
If ParamLinea(iNumLinea).bLlenandoBateadorB Then
    ParamLinea(iNumLinea).dPesoBateadorB = Format(ParamLinea(iNumLinea).dPesoBateadorB +
dPeso, "###0.00")
End If
'Meto los datos en el histórico
'Descripcion = "Operario " & ParamLinea(iNumLinea).iCodigo_Operario(iOperario) & ": " & dPeso
& "Kg" & " /  " & ParamLinea(iNumLinea).dPeso_Operario(iOperario) & "Kg" & " /  " &
ParamLinea(iNumLinea).tTiempo_MarchaOperario(iOperario) & " /  " &
ParamLinea(iNumLinea).dKgH(iOperario) & " Kg/h"
Descripcion = Textos(306) & " " & ParamLinea(iNumLinea).iCodigo_Operario(iOperario) & ": " &
dPeso & Textos(307) & " /  " & ParamLinea(iNumLinea).dPeso_Operario(iOperario) & Textos(307)
& " /  " & ParamLinea(iNumLinea).tTiempo_MarchaOperario(iOperario) & " /  " &
ParamLinea(iNumLinea).dKgH(iOperario) & Textos(177)
With Data1
    .Commands("cmdHistoricos").CommandText = "Insert Into Historicos (NumeroPartida, Fecha,
Hora, Descripcion, NumLinea, Orden) values ('" & ParamLinea(iNumLinea).sNumero_Partida & "',
'" & Date & "', '" & Time & "', '" & Descripcion & "', " & iNumLinea & ", " &
ParamLinea(iNumLinea).iOrden & ")"
    .Commands("cmdHistoricos").Execute
End With
'Guardo datos en la tabla PesosOperarios
'Incremento el número de pesada
ParamLinea(iNumLinea).iNumeroPesadaOperario(iOperario) =
ParamLinea(iNumLinea).iNumeroPesadaOperario(iOperario) + 1
With Data1
    .Commands("cmdPesosOperarios").CommandText = "Insert Into PesosOperarios (NumeroPartida,
CodigoOperario, NumeroPesada, Peso, Hora, HoraInicioOperario, Fecha, Numlinea, NumeroPuesto,
Orden, PesoAcc, KgH) values ('" & ParamLinea(iNumLinea).sNumero_Partida & "'," &
ParamLinea(iNumLinea).iCodigo_Operario(iOperario) & ", " &
ParamLinea(iNumLinea).iNumeroPesadaOperario(iOperario) & ", '" & dPeso & "', '" & Time & "',
'" & ParamLinea(iNumLinea).tHoraInicioOperario(iOperario) & "', '" & Date & "', " & iNumLinea
& ", " & iOperario & ", " & ParamLinea(iNumLinea).iOrden & ", '" &
ParamLinea(iNumLinea).dPeso_Operario(iOperario) & "', '" &
ParamLinea(iNumLinea).dKgH(iOperario) & "')"
    .Commands("cmdPesosOperarios").Execute
End With
'Guardo el peso actual de los bateadores en la tabla Estadolinea para el soft de
'consultas
With Data1
```

```
    .Commands("cmdEstadoLinea").CommandText = "Update EstadoLinea set Peso_A='" &
ParamLinea(iNumLinea).dPesoBateadorA & "', Peso_B= '" &
ParamLinea(iNumLinea).dPesoBateadorB & "' where N_Linea='" &
ParamLinea(iNumLinea).iNumLinea & "'"
    .Commands("cmdEstadoLinea").Execute
End With
Call FinalTarea
Exit Sub
RutinaError:
    Call FinalTarea
    Exit Sub
End Sub
```

Todas las tramas que se envían hacia el autómata, se configuran previamente cada una en una función del programa. Para no saturar el canal de comunicación entre el PC y el PLC, lo que se hace es crear una pila donde se van almacenando todas las tramas que se tienen que enviar, y hasta que no llega la respuesta por parte del autómata, no se envía otra. Con esto se puede garantizar que todas las tramas llegan a su destino, puesto que si no se recibe la confirmación por parte del autómata no se envía otra trama.

En el **Anexo3** se pueden observar también las principales funciones relacionadas con las tareas que tiene el programa en cuanto al envío de las tramas de comunicación. A continuación se detalla la funcionalidad de cada una de ellas:

- **AñadirTarea(iNumTarea, iNodo)**: Esta función se encarga de mirar si la tarea que hay que ejecutar ya se encuentra en la pila de tareas pendientes, y si no se encuentra se añade. Además de esta, solo existe otra posibilidad para no poder añadir una tarea, es que la pila esté llena y no se pueda añadir nada, pero esto solo pasa cuando falla la comunicación y no se van eliminando de la pila. El número máximo de tareas que se pueden tener en la pila son 50.
- **BorraTarea(iTarea)**: Una vez que ya se encuentran las tareas en la pila, se van lanzando según el orden de llegada, y cuando ya se han ejecutado correctamente y han recibido también su correspondiente contestación, se hace la llamada a esta función para borrar dicha tarea de la pila. Para eliminarla de la pila, lo único que tiene que hacer es mirar en que posición se encuentra la tarea y colocarle un valor 0.
- **ConfiguraTareas()**: Cuando el Pc se queda sin tareas a ejecutar, se llama a esta función cada cierto tiempo para que añada a la pila la trama de estado “E”, que se encarga de informar continuamente en que estado se encuentra la línea para así poder seguir su evolución y mostrarla por pantalla.

### *5.2. Base de Datos*

En cuanto a la organización del programa, hay que destacar que está centrado en una base de datos relacional, creada en Access, donde se van almacenando todos los datos referentes a la línea y que sirven para, posteriormente, poder sacar informes con los rendimientos. También se encuentran guardados todos los datos relacionados con los procesos que hay que hacer en la línea, así como los diferentes tratamientos a aplicar y los operarios que trabajan en la línea.

Todos los datos se van guardando con un nombre de partida y de orden, sea en la tabla que sea, para así poder saber en todo momento de que lote se están sacando los datos. En la figura 19 se puede observar la tabla de relaciones de la base de datos, y a continuación una explicación de cada una de las tablas que se utilizan.

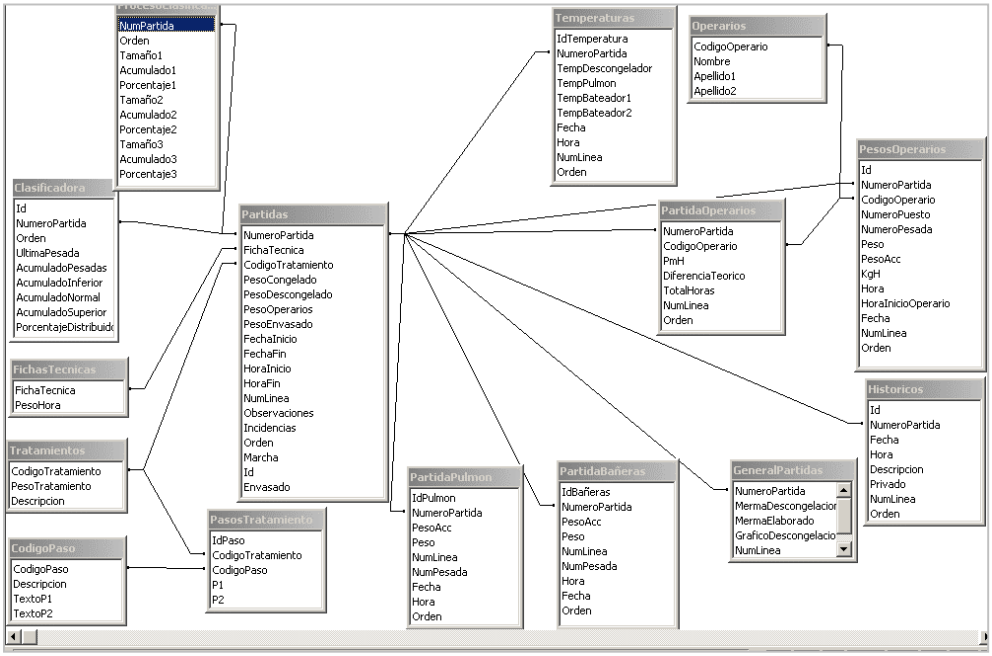



Figura 19: Diagrama de relaciones de la base de datos creada en Microsoft Access.

En este esquema se pueden ver las diferentes tablas que se encuentran relacionadas entre sí. Hay que decir que la más importante es la tabla ***Partidas***, ya que es donde se guardan todos los datos de una partida una vez finalizada. Entre estos datos se encuentran todas las mermas, todos los tratamientos que se han aplicado, los rendimientos esperados por parte de los operarios e incluso si esta partida ha sido ya envasada o no.

Además de esta, otra de las importantes es la tabla ***Tratamientos*** ya que es donde se encuentran almacenados todos los tratamientos que se le tienen que aplicar a las partidas. Siempre antes de empezar con una nueva partida, se indica cual se le

aplicará para poder meter los litros de agua y de sal adecuados, así como los minutos de bateo por aire.

Finalmente, otra tabla también muy importante es la tabla ***PesosOperarios*** ya que es donde se van guardando todas las pesadas de los operarios. Además de las pesadas, también se van guardando los pesos acumulados, la hora a la que se ha realizado la pesada y el tiempo que el operario lleva en la línea. Con estos datos, se puede obtener el rendimiento que está haciendo el operario.

Lo mismo sucede con la tabla ***PesosEntradaOperarios***, ya que es exactamente igual pero en vez de ir almacenando los pesos a la salida de los operarios, se guardan los pesos que han ido demandando los operarios. Entre estas dos tablas se puede sacar la merma de cada uno de los operarios haciendo una sencilla división entre los kilos de salida y los kilos de entrada (Siempre será mayor el peso a la entrada que a la salida).

Todas estas tablas se van actualizando en tiempo real, es decir, dependiendo del estado en que se encuentra la línea, se van guardando datos en una tabla o en otra. Por ejemplo, cada vez que un operario recibe peso se actualiza al momento la tabla PesosEntradaOperarios, mientras que la tabla Partidas, solo se actualiza una vez que se ha finalizado por completo la ejecución.[4]

Una vez que se ha iniciado la partida, y por lo tanto se encuentra toda la línea en marcha, se comienza a enviar continuamente una trama de estado de línea para ir conociendo el estado de la misma. Dentro de esta trama existen diferentes valores en los diferentes bytes para saber que hay que hacer o que valores se tienen que ir leyendo del autómata.

Los principales valores que se devuelven con la trama de estado son un 1 en algún byte avisando que ya está disponible el valor de la pesada de los operarios, ya sea de entrada o de salida para que, de esta forma, se le pueda enviar al autómata la trama pidiendo el peso de los operarios.

```
'Se ha pesado la entrada de un operario
If Asc(Mid(sAux, 10, 1)) = 1 Then
    If ParamLinea(iNumLinea).bReposo_op_entrada Then
        'Pedimos el peso que le ha llegado
        PeticionModificarPila T_PESO_OPERARIO_ENTRADA, iNodo, True, False
        ParamLinea(iNumLinea).bReposo_op_entrada = False
  End If
End If
```

[4] En el anexo5 se puede aprender más sobre el funcionamiento de este programa con el manual adjunto.

```
'Se ha pesado a un operario
If Asc(Mid(sAux, 11, 1)) = 1 Then
    If ParamLinea(iNumLinea).bReposo_op Then
        'Pedimos el peso que ha soltado
        PeticionModificarPila T_PESO_OPERARIO, iNodo, True, False
        ParamLinea(iNumLinea).bReposo_op = False
    End If
End If
```

Además de este caso, hay que decir que también hay diferentes bytes reservados a saber en que estado se encuentran las bañeras o los bateadores, es decir, para saber si se le está metiendo agua, si se le está echando sal o si se está bateando. Todo esto se puede ir sabiendo gracias a la comunicación en tiempo real que existe entre el autómata y la pantalla táctil.

Finalmente, cabe destacar que cada vez que se inicia una nueva partida, se envía al autómata todos los datos necesarios para que pueda realizar correctamente todas sus funciones. Un claro ejemplo de esto es que se envía una trama con los operarios que se encontrarán trabajando en la línea para que, si algún operario no se encuentra trabajando, no ponga su cinta en marcha y así ganar el máximo tiempo posible en el ciclo de descarga.

## *6.* DESARROLLO DEL PROGRAMA DE CONSULTAS (PC OFICINA)

Existe también un software de consultas, diseñado especialmente para la extracción de datos de la línea desde cualquier ordenador de oficina que esté incluido dentro de la misma red de trabajo. Hay que decir que este programa está diseñado a petición del cliente, es decir, en este programa sale única y exclusivamente la información que el cliente necesita ya que es el software sobre el cual más modificaciones se realizan en función de las peticiones.

Cada vez que se arranca el programa, se hace una pequeña comprobación sobre las bases de datos configuradas, ya que el programa no tiene una única configurada sino que el usuario tiene la posibilidad de configurar tantas como quiera. Con esto se gana flexibilidad ya que los usuarios pueden ir creando copias de seguridad sin necesidad de perder los datos existentes.

Para realizar dichas comprobaciones, existe una tabla dentro de la base de datos donde se van guardando las diferentes rutas. Si no existe ninguna ruta en el momento de entrar al programa, se lanza un aviso al usuario para que tenga la posibilidad de configurar tantas como quiera.

La función que se encarga de hacer estas comprobaciones es la IniciaBasesDatos(), donde primero se guardan todas las rutas de las bases de datos en un vector para así poder acceder a todas ellas. Una vez guardadas, se comprueba las que están activas y si no existe ninguna, se lanza un error por pantalla. Si por el contrario existe alguna base de datos activa, se deja para poder extraer datos de ella. En el **Anexo4** se puede observar el código fuente de dicha función.

Una vez realizadas dichas comprobaciones, ya se puede asegurar que nunca se realizarán consultas sobre bases de datos no existentes y que, por tanto, no existirán problemas con la existencia de datos.

En la figura 20 se pueden observar las diferentes tablas que hay creadas en la base de datos de consultas, principalmente para almacenar datos temporalmente y así poder imprimir los informes sin problema y no utilizar recursos de la pantalla táctil de la línea, sino que se trabaja sobre el propio ordenador local.

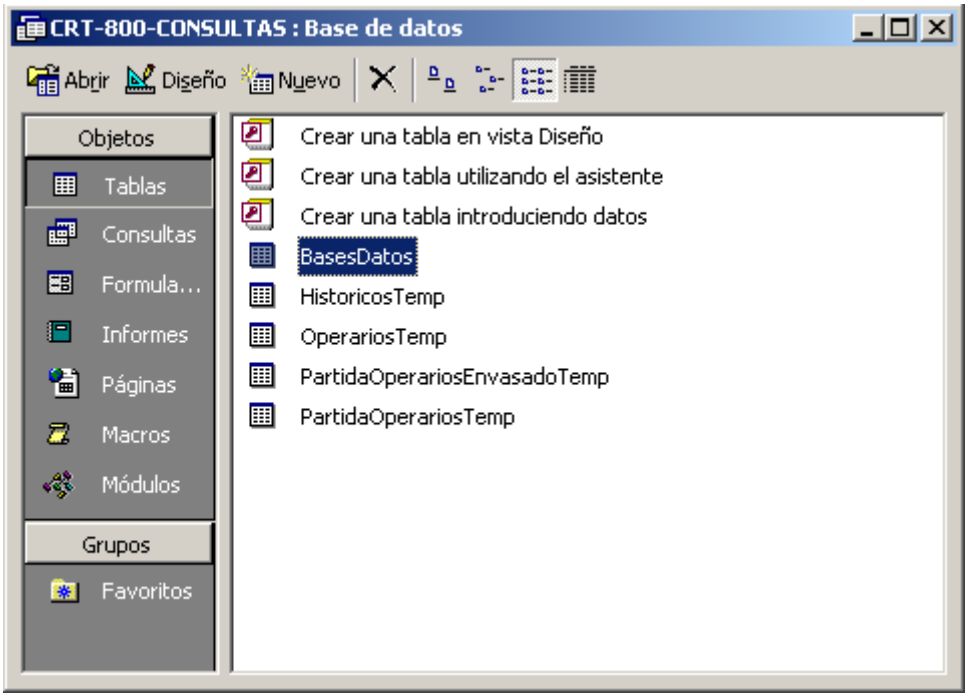



Figura 20: Tablas de la base de datos del software de consultas, realizadas con Microsoft Access.

Como se puede observar en la figura 20, existe una tabla ***BaseDatos*** que es donde se van guardando todas las rutas de las diferentes bases de datos añadidas para hacer consultas. Además de esta tabla, existen 4 tablas temporales que es donde se van guardando todos los datos referentes a las partidas antes de imprimirlas, ya que siempre es más efectivo hacer las consultas de la base de datos en local que no en otro equipo de la red.

Desde este software, además de poder consultar los datos almacenado, también se pueden sacar informes detallados de todos los operarios y de todos los procesos de la línea. Con esto, se pretende pagar más a los operarios que estén trabajando de una manera más eficaz para la empresa.

Para extraer los datos de la base de datos, solo hay que saber el número de partida y el número de lote del cual hay que sacar la información. Una vez se tienen esos datos, solo queda consultar e imprimir el informe. Estos datos únicamente los conoce el responsable de la línea, para así no tener datos relacionados con el rendimiento de los trabajadores a la vista de todo el mundo.

Tampoco hace falta que el responsable se conozca todas las partidas, sino que el programa tiene implementada una función (***BuscarPartida***) que se encarga de acceder a la base de datos correspondiente y mostrar por pantalla todas las partidas, así como los

lotes de dichas partidas. En el **Anexo4** también se puede observar el código fuente de esta función y las siguientes.

La función ***CargaLote*** es la encargada de hacer la consulta a la base de datos sobre las diferentes partidas y lotes que existen para poder mostrarlas todas por pantalla. Para mostrarlas, se hace siempre de 10 en 10, puesto que debido al gran número de partidas que puede haber no se podrían mostrar todas de una vez.

Una vez cargadas todas las partidas, se va refrescando la lista con la función ***Refresca_Lista*** para ir dando al usuario la posibilidad de seleccionar la que desee. Cuando el usuario selecciona el nombre de la partida, el programa realiza unas operaciones para extraer todos los datos y poder generar automáticamente el informe sin tener que pedir ningún dato más al operario. Estas operaciones se realizan en la función ***VisualizarDatosPartida*** y se detallan a continuación:

- Se borran todos los datos existentes en las tablas ***PartidaOperariosTemp*** y ***PartidaOperariosEnvasadoTemp*** para no mezclar datos de diferentes partidas, puesto que el informe se imprime de los datos existentes en dichas tablas.
- Se extraen los datos de la tabla ***Partidas*** de la pantalla táctil. Los que se extraen primeramente son los referentes a los pesos totales de la línea, es decir, el peso bruto insertado en la línea, el peso de operarios y el peso de envasado. También se obtienen la fecha y la hora de inicio y de final de la partida.
- Se extraen los datos de ***FichaTecnica***, que es la producción esperada para cada uno de los operarios, y que depende del tipo de producto que se trabaja en la línea.
- Se extraen los operarios que han intervenido en la línea, de las tablas ***PesosOperarios*** y ***PesoEnvasado*** de la base de datos que está en la pantalla táctil**.**
- Se obtienen los datos de todos los operarios, tanto los pesos de entrada como los pesos de salida y los envasados. Esto se encuentra en las tablas ***PesosOperarios***, ***PesosEntradaOperarios*** y ***PesosEnvasado.*** Los datos más relevantes que se tiene que saber son los Kg. producidos por cada uno de los operarios, su producción y su merma entre el peso de entrada y el peso de salida.
- Finalmente se guardan todos los pesos y todos los datos en las tablas temporales que se encuentran en el ordenador local, y se genera el informe calculando previamente las mermas que ha habido entre los diferentes pesos de la línea.

Una vez obtenidos todos los datos de la base de datos, se genera un informe con todos esos datos para que el jefe de planta pueda observar el rendimiento de todos y cada uno de sus trabajadores.

En la figura 21 se puede observar un informe de ejemplo, pero sin datos reales.

# ELABORADOS FREIREMAR

DELEGACION VALENCIA

VALENCIA

**LOTE: 1120** **102** 10/12/2005

| | | | | |
|---|---|---|---|---|
| Fecha Inicio: | 13/04/2005 | Hora Inicio: | 15:44:46 | Total tiempo marcha: |
| Fecha Fin: | 13/04/2005 | Hora Fin: | 15:49:15 | 0:04:29 |

| Peso Congelado | Peso Descongelado | Peso Operarios | Peso Elaborado |
|---|---|---|---|
| **50000 Kg** | **0 Kg** | **0,0135 Kg** | **0 Kg** |

**100,0 %** **< 0 %** **100,0 %**

**100,0 %**

**100,0 %**

| | | | | |
|---|---|---|---|---|
| Ficha Técnica: | FICHA 6 | Kg/h Teóricos: | 362 | Tanques: |
| Tratamiento: | 147 | Kg/h Obtenidos: | 0,36 | |

Observaciones: -

Incidencias: LLLL

OPERARIOS QUE HAN INTERVENIDO 1120 102 Página: 1/1

| NOMBRE | PESO ENT. | MERMA | PRODUCIDO | KG/H | DIF. TEÓR. | HORAS TRAB. | NIVEL |
|---|---|---|---|---|---|---|---|
| | | | | | | | |
| OPERARIO 1 | 12,3 | 100 | 0 | 0 | -362 | 00:00:00 | – |
| OPERARIO 2 | 10,25 | 99,98 | 0,0021 | 0,4 | -361,6 | 0:00:19 | – |
| OPERARIO 3 | 10,25 | 99,97 | 0,003 | 0,5 | -361,5 | 0:00:22 | – |
| OPERARIO 5 | 14,35 | 99,99 | 0,0021 | 0,4 | -361,6 | 0:00:19 | – |
| OPERARIO 6 | 10,25 | 99,98 | 0,0021 | 0,4 | -361,6 | 0:00:19 | – |
| OPERARIO 7 | 8,2 | 99,97 | 0,0021 | 0,4 | -361,6 | 0:00:19 | – |
| OPERARIO 8 | 10,25 | 99,98 | 0,0021 | 0,4 | -361,6 | 0:00:19 | – |

Figura 21: Informe que se extrae con todos los datos referentes a la línea, generado con el propio Visual Basic.

# *7.* CONCLUSIONES

Lo primero que hay que decir en cuanto al proyecto realizado, es que los objetivos se han alcanzado sobradamente, ya que primeramente se planteaba la línea de una manera semiautomática, y finalmente ha quedado una línea de proceso totalmente automatizada, cumpliendo todos los requisitos del cliente y sus necesidades.

En cuanto a los objetivos planteados inicialmente, también se han podido cumplir todos ellos, pero siempre atacándolos de una manera ordenada y organizada, ya que plantaban problemas bastante complejos de solucionar y que han requerido una solución individualmente, para posteriormente ser unidos en el mismo programa de control.

Todos los puntos, en cuanto a diseño de aplicaciones, que se han tenido que plantear desde un principio han sufrido bastantes modificaciones, debido a que las soluciones aportadas inicialmente podían no ser las más adecuadas para resolver el tipo de problema al que se hacía frente.

A continuación se da una breve explicación de los problemas más destacados que se han encontrado a la hora del desarrollo y la puesta en marcha de la línea de proceso, tanto en el taller del fabricante como en el cliente final.

## *7.1.* PRINCIPALES PROBLEMAS ENCONTRADOS

**- Problemas en la fase de desarrollo**

Los principales problemas que se encontraron en la fase de desarrollo fueron debidos a la gran cantidad de líneas de código que se han tenido que hacer tanto en el programa del autómata como en el programa realizado en Visual Basic.

Uno de los problemas más graves que se encontraron en el momento del diseño fue el tema de la comunicación. Se tenía que diseñar un protocolo totalmente fiable para que no hubiese errores en las tramas y que no se pudiese descontrolar la máquina debido a este error. Al principio costó bastante encontrar la manera de realizarla, pero a medida que se iba pensando el funcionamiento de la máquina y el contenido de las tramas todo fue un poco más fácil.

Otro de los problemas que surgió en la fase de desarrollo fue debido al tiempo de retardo que había en la comunicación entre el autómata y la pantalla táctil, ya que cuando todavía no se había recibido la contestación ya se estaba enviando la siguiente trama con el consiguiente error.

Otro frecuente error al que se tuvo que hacer frente fue al desconocimiento de lo que quería el cliente desde un principio. Esto llevó a un desconcierto en cuanto al verdadero funcionamiento de la máquina y por tanto a un retraso en la realización del programa ya que se empezó a hacer sin tener claro totalmente lo que tenía que contabilizar o mostrar.

Los principales problemas surgidos fueron estos, aunque siempre iban saliendo detalles que se tenían que ir puliendo con el paso de los días y con la estrecha colaboración del cliente.

**- Problemas en la fase de puesta en marcha**

Los principales problemas de un programa de autómata o de un programa que controla toda la línea vienen en el momento de la puesta en marcha.

Una vez se colocó el programa, se probó lo máximo posible en la empresa Palinox antes de salir la máquina hacia el cliente. Se tuvieron que modificar algunas cosas pero nada en exceso hasta que se llegó al cliente.

Cuando la máquina ya estaba colocada en la sala de producción, empezaron a llegar algunos problemas. El primero de ellos surgió nada más colocar la máquina puesto que los paros de emergencia y los paros normales de máquina se habían colocado al revés (lógica inversa) en el programa del autómata.

Para cumplir normativa eléctrica, tienen que ser normalmente cerrados y dejar de dar señal al autómata cuando se pulsan o bien si se corta un cable. Esto hizo que la línea no diese señales de vida nada más arrancar. Esto no tuvo mayor problema una vez se detectó la situación que lo provocaba.

Otro de los problemas llegó a la hora de pesar los kilos tanto de entrada como de salida de los operarios. En ningún momento llegaban los valores de peso correctamente a la pantalla táctil. En un principio tenía toda la pinta de ser problema del autómata, que es el encargado de hacer una regla de tres para sacar los pesos a través de los puntos. Finalmente, resultó ser problema del autómata porque recibe hasta 32000 puntos en la línea analógica mientras que el visor de peso estaba configurado para 64000 puntos y sin una salida totalmente lineal, por lo que el valor 0 no podía corresponder a 0 puntos.

Una vez solucionado el problema anterior, la línea comenzó a funcionar, incluso teniendo todavía algunos detalles que solucionar. El siguiente problema que se encontró fue que las pesadas de entrada a los operarios se realizaban de 10 kilos, pero como la cinta de entrada llevaba una inercia en el momento de la parada, caían casi 20 kilos en la báscula y no cabía en la caja de los operarios. La solución a este problema fue cambiar este valor en el autómata por 5Kg.

Finalmente, el último problema que surgió fue a las 3 semanas de estar trabajando la línea. El problema fue que no se había contemplado en el programa la posibilidad que al finalizar el proceso quedasen peticiones sin satisfacer, con lo cual, en el momento en el que paraban con alguna petición realizada el programa dejaba de funcionar, realizaba el primer ciclo de reparto de género a los operarios pero luego no recibía ninguna petición.

Este problema se solucionó reseteando toda la tabla de peticiones al iniciar un nuevo proceso, o mejor dicho, al parar la máquina.

En cuanto al programa de consultas, también hubo algún pequeño problema que se tuvo que solucionar el día de la puesta en marcha. El principal problema que surgió fue que en el momento de acceder a la base de datos de la pantalla táctil, los datos no guardaban el mismo formato, es decir, en unos había dos decimales colocados mientras que en los otros no se habían colocado.

Esto hacía que todo el tema relacionado con las mermas, con las producciones, etc... no funcionase correctamente ya que no cuadraba ningún dato.

Actualmente, la línea está funcionando sin ningún tipo de problema en el cliente, es más, se les van añadiendo pequeñas modificaciones que el cliente va pidiendo por necesidad de mercado. La mayor parte de las modificaciones que se están realizando son de rendimientos.

Lo que se ha hecho para poder solucionar cualquier tipo de problema, ha sido conseguir un acceso a la pantalla táctil mediante el programa de acceso remoto VNC (Virtual Network Connection) e instalar el programa para acceder al autómata (MicroWin). Con esto instalado, desde la oficina técnica de Palinox se puede acceder sin ningún tipo de problema y sin la necesidad de desplazarse a Valencia.

## ANEXO 1: Tramas de comunicación

| | PLC>>PC | | | |
|---|---|---|---|---|
| | TRAMA DE ESTADO 'E' | TRAMA PULMON 'L' | TRAMA PESO OPERARIO 'O' | TRAMA ARRANCAR 'A' |
| BYTE 1 | 2 | 2 | 2 | 2 |
| BYTE 2 | Z | Z | Z | Z |
| BYTE 3 | NODO | NODO | NODO | NODO |
| BYTE 4 | **E** | **Q** | **O** | **A** |
| BYTE 5 | AUTO/MAN | NUMERO DE OPERARIO | NUMERO DE OPERARIO | O |
| BYTE 6 | ALARMA | PESO ENTRADA OPERARIO | PESO SALIDA OPERARIO | K |
| BYTE 7 | ESTADO TRATAMIENTO 1 | | | |
| BYTE 8 | ESTADO TRATAMIENTO 2 | | | |
| BYTE 9 | TERMICOS | | | |
| BYTE 10 | LEER PESO PULMON | | | |
| BYTE 11 | LEER PESO OPERARIO | | | |
| BYTE 12 | | | | |
| BYTE 13 | TARA PULMON | | | |
| BYTE 14 | TARA BAÑERA | | | |
| BYTE 15 | ESTADO CUBA 1 | | | |
| BYTE 16 | ESTADO CUBA 2 | | | |
| BYTE 17 | REPOSO CUBA 1 | | | |
| BYTE 18 | REPOSO CUBA 2 | | | |
| BYTE 19 | CRC | CRC | CRC | CRC |

| | PC>>PLC | | | |
|---|---|---|---|---|
| | TRAMA DE ESTADO 'E' | TRAMA PESO PULMON 'L' | TRAMA PESO OPERARIO 'O' | TRAMA ARRANCAR 'A' |
| BYTE 1 | 2 | 2 | 2 | 2 |
| BYTE 2 | N | N | N | N |
| BYTE 3 | NODO | NODO | NODO | NODO |
| BYTE 4 | **E** | **Q** | **O** | **A** |
| BYTE 5 | | | | |
| BYTE 6 | | | | |
| BYTE 7 | | | | |
| BYTE 8 | | | | |
| BYTE 9 | | | | |
| BYTE 10 | | | | |
| BYTE 11 | | | | |
| BYTE 12 | | | | |
| BYTE 13 | | | | |
| BYTE 14 | | | | |
| BYTE 15 | | | | |
| BYTE 16 | | | | |
| BYTE 17 | | | | |
| BYTE 18 | | | | |
| BYTE 19 | CRC | CRC | CRC | CRC |

| | PLC>>PC | | | |
|---|---|---|---|---|
| | TRAMA PASO TRAT. A | TRAMA PASO TRAT. B | TRAMA DE PARO 'R' | TRAMA AJ. BÁSCULAS 'Z' |
| BYTE 1 | 2 | 2 | 2 | 2 |
| BYTE 2 | Z | Z | Z | Z |
| BYTE 3 | NODO | NODO | NODO | NODO |
| BYTE 4 | **F** | **G** | **R** | **U** |
| BYTE 5 | O | O | O | O |
| BYTE 6 | K | K | K | K |
| BYTE 7 | | | | |
| BYTE 8 | | | | |
| BYTE 9 | | | | |
| BYTE 10 | | | | |
| BYTE 11 | | | | |
| BYTE 12 | | | | |
| BYTE 13 | | | | |
| BYTE 14 | | | | |
| BYTE 15 | | | | |
| BYTE 16 | | | | |
| BYTE 17 | | | | |
| BYTE 18 | | | | |
| BYTE 19 | CRC | CRC | CRC | CRC |

| | PC>>PLC | | | |
|---|---|---|---|---|
| | TRAMA PASO TRAT. A | TRAMA PASO TRAT. B | TRAMA DE PARO 'R' | TRAMA AJ. BASCULAS 'Z' |
| BYTE 1 | 2 | 2 | 2 | 2 |
| BYTE 2 | N | N | N | Z |
| BYTE 3 | NODO | NODO | NODO | NODO |
| BYTE 4 | **F** | **G** | **R** | **U** |
| BYTE 5 | COD. PASO | COD. PASO | | NUMERO DE BÁSCULA |
| BYTE 6 | CANTIDAD | CANTIDAD | | 1-TARA, 2-PESO |
| BYTE 7 | | | | PESO BÁSCULA |
| BYTE 8 | | | | |
| BYTE 9 | | | | |
| BYTE 10 | | | | |
| BYTE 11 | | | | |
| BYTE 12 | | | | |
| BYTE 13 | | | | |
| BYTE 14 | | | | |
| BYTE 15 | | | | |
| BYTE 16 | | | | |
| BYTE 17 | | | | |
| BYTE 18 | | | | |
| BYTE 19 | CRC | CRC | CRC | CRC |

| | PLC>>PC | | | |
|---|---|---|---|---|
| | TRAMA PESOS 'K' | TRAMA OPERARIOS1 'D' | TRAMA OPERARIOS2 'J' | TRAMA TIEMPOS 'L' |
| BYTE 1 | 2 | 2 | 2 | 2 |
| BYTE 2 | Z | Z | Z | Z |
| BYTE 3 | NODO | NODO | NODO | NODO |
| BYTE 4 | **K** | **D** | **J** | **L** |
| BYTE 5 | PESO ENTRADA | O | O | O |
| BYTE 6 | | K | K | K |
| BYTE 7 | | | | |
| BYTE 8 | | | | |
| BYTE 9 | PESO SALIDA OPERARIO | | | |
| BYTE 10 | | | | |
| BYTE 11 | PESO BASCULA 1 | | | |
| BYTE 12 | | | | |
| BYTE 13 | PESO BASCULA 2 | | | |
| BYTE 14 | | | | |
| BYTE 15 | PESO BASCULA 3 | | | |
| BYTE 16 | | | | |
| BYTE 17 | | | | |
| BYTE 18 | | | | |
| BYTE 19 | CRC | CRC | CRC | CRC |

| | PC>>PLC | | | |
|---|---|---|---|---|
| | TRAMA PESOS 'K' | TRAMA OPERARIOS1 'D' | TRAMA OPERARIOS2 'J' | TRAMA TIEMPOS 'L' |
| BYTE 1 | 2 | 2 | 2 | 2 |
| BYTE 2 | Z | Z | Z | Z |
| BYTE 3 | NODO | NODO | NODO | NODO |
| BYTE 4 | **K** | **D** | **J** | **L** |
| BYTE 5 | | OPERARIOS ACTIVOS | TIM_OP7 | TIM_CICLO |
| BYTE 6 | | | | |
| BYTE 7 | | TIM_OP1 | TIM_OP8 | TIM_DESCARGA |
| BYTE 8 | | | | |
| BYTE 9 | | TIM_OP2 | TIM_OP9 | TIM_PESADA |
| BYTE 10 | | | | |
| BYTE 11 | | TIM_OP3 | TIM_OP10 | TIM_CINTA_A |
| BYTE 12 | | | | |
| BYTE 13 | | TIM_OP4 | TIM_OP11 | TIM_CINTA_B |
| BYTE 14 | | | | |
| BYTE 15 | | TIM_OP5 | TIM_OP12 | |
| BYTE 16 | | | | |
| BYTE 17 | | TIM_OP6 | TIM_OP13 | |
| BYTE 18 | | | | |
| BYTE 19 | CRC | CRC | CRC | CRC |

| | PLC>>PC | | | |
|---|---|---|---|---|
| | TRAMA TEMP1 'M' | TRAMA TIEMPOS2 | TRAMA CINTA BATEADOR | TRAMA PAUSA |
| BYTE 1 | 2 | 2 | 2 | 2 |
| BYTE 2 | Z | Z | Z | Z |
| BYTE 3 | NODO | NODO | NODO | NODO |
| BYTE 4 | **M** | **H** | **P** | **I** |
| BYTE 5 | LEER PESO BASCULA 1 | O | O | O |
| BYTE 6 | LEER PESO BASCULA 2 | K | K | K |
| BYTE 7 | LEER PESO BASCULA 3 | | | |
| BYTE 8 | | | | |
| BYTE 9 | | | | |
| BYTE 10 | | | | |
| BYTE 11 | | | | |
| BYTE 12 | | | | |
| BYTE 13 | | | | |
| BYTE 14 | | | | |
| BYTE 15 | | | | |
| BYTE 16 | | | | |
| BYTE 17 | | | | |
| BYTE 18 | | | | |
| BYTE 19 | CRC | CRC | CRC | CRC |

| | PC>>PLC | | | |
|---|---|---|---|---|
| | TRAMA TEMP1 'M' | TRAMA TIEMPOS2 | TRAMA CINTA BATEADOR | TRAMA PAUSA |
| BYTE 1 | 2 | 2 | 2 | 2 |
| BYTE 2 | Z | Z | Z | Z |
| BYTE 3 | NODO | NODO | NODO | NODO |
| BYTE 4 | **M** | **H** | **P** | **I** |
| BYTE 5 | | TIM_EXTRA_BATEO | 0-->A, 1-->B | 1-->SI, 0-->NO |
| BYTE 6 | | | | |
| BYTE 7 | | TIM_VACIADO | | |
| BYTE 8 | | | | |
| BYTE 9 | | TIM_DESFASE | | |
| BYTE 10 | | | | |
| BYTE 11 | | TIM_SALIDA | | |
| BYTE 12 | | | | |
| BYTE 13 | | | | |
| BYTE 14 | | | | |
| BYTE 15 | | | | |
| BYTE 16 | | | | |
| BYTE 17 | | | | |
| BYTE 18 | | | | |
| BYTE 19 | CRC | CRC | CRC | CRC |

| | PLC>>PC | | | |
|---|---|---|---|---|
| | TRAMA SOBREPESO | TRAMA FIN CICLO | TRAMA MARCHA ENV. | TRAMA PARO ENV. |
| BYTE 1 | 2 | 2 | 2 | 2 |
| BYTE 2 | Z | Z | Z | Z |
| BYTE 3 | NODO | NODO | NODO | NODO |
| BYTE 4 | **S** | **W** | **T** | **t** |
| BYTE 5 | O | O | O | O |
| BYTE 6 | K | K | K | K |
| BYTE 7 | | | | |
| BYTE 8 | | | | |
| BYTE 9 | | | | |
| BYTE 10 | | | | |
| BYTE 11 | | | | |
| BYTE 12 | | | | |
| BYTE 13 | | | | |
| BYTE 14 | | | | |
| BYTE 15 | | | | |
| BYTE 16 | | | | |
| BYTE 17 | | | | |
| BYTE 18 | | | | |
| BYTE 19 | CRC | CRC | CRC | CRC |

| | PC>>PLC | | | |
|---|---|---|---|---|
| | TRAMA SOBREPESO | TRAMA FIN CICLO | TRAMA MARCHA ENV. | TRAMA PARO ENV. |
| BYTE 1 | 2 | 2 | 2 | 2 |
| BYTE 2 | Z | Z | Z | Z |
| BYTE 3 | NODO | NODO | NODO | NODO |
| BYTE 4 | **S** | **W** | **T** | **t** |
| BYTE 5 | BAT A (1-->SI, 0-->NO) | BAT A (1-->SI, 0-->NO) | | |
| BYTE 6 | BAT B (1-->SI, 0-->NO) | BAT B (1-->SI, 0-->NO) | | |
| BYTE 7 | | | | |
| BYTE 8 | | | | |
| BYTE 9 | | | | |
| BYTE 10 | | | | |
| BYTE 11 | | | | |
| BYTE 12 | | | | |
| BYTE 13 | | | | |
| BYTE 14 | | | | |
| BYTE 15 | | | | |
| BYTE 16 | | | | |
| BYTE 17 | | | | |
| BYTE 18 | | | | |
| BYTE 19 | CRC | CRC | CRC | CRC |

| | PLC>>PC |
|---|---|
| | TRAMA PESOS ENV. |
| BYTE 1 | 2 |
| BYTE 2 | Z |
| BYTE 3 | NODO |
| BYTE 4 | **N** |
| BYTE 5 | PESO BASCULA 1 |
| BYTE 6 | |
| BYTE 7 | PESO BASCULA 2 |
| BYTE 8 | |
| BYTE 9 | PESO BASCULA 3 |
| BYTE 10 | |
| BYTE 11 | |
| BYTE 12 | |
| BYTE 13 | |
| BYTE 14 | |
| BYTE 15 | |
| BYTE 16 | |
| BYTE 17 | |
| BYTE 18 | |
| BYTE 19 | CRC |

| | PC>>PLC |
|---|---|
| | TRAMA PESOS ENV |
| BYTE 1 | 2 |
| BYTE 2 | Z |
| BYTE 3 | NODO |
| BYTE 4 | **N** |
| BYTE 5 | |
| BYTE 6 | |
| BYTE 7 | |
| BYTE 8 | |
| BYTE 9 | |
| BYTE 10 | |
| BYTE 11 | |
| BYTE 12 | |
| BYTE 13 | |
| BYTE 14 | |
| BYTE 15 | |
| BYTE 16 | |
| BYTE 17 | |
| BYTE 18 | |
| BYTE 19 | CRC |

## ANEXO 2: Listado de entradas y salidas del autómata

# ENTRADAS

| *Módulo Entradas Discretas I0* | *Módulo Entradas Discretas I3* |
|---|---|
| **I0.0** => Pulsador Paro Emergencia (NC) | **I3.0** => LIBRE |
| **I0.1** => Pulsador Paro Línea Fileteado (NC) | **I3.1** => LIBRE |
| **I0.2** => Pulsador Marcha Línea Fileteado | **I3.2** => LIBRE |
| **I0.3** => Pulsador Petición Operario 1 | **I3.3** => Falta Peso Báscula 1 |
| **I0.4** => Pulsador Petición Operario 2 | **I3.4** => Peso Ok Báscula 1 |
| **I0.5** => Pulsador Petición Operario 3 | **I3.5** => Exceso Peso Báscula 1 |
| **I0.6** => Pulsador Petición Operario 4 | **I3.6** => Falta Peso Báscula 2 |
| **I0.7** => Pulsador Petición Operario 5 | **I3.7** => Peso Ok Báscula 2 |

| *Módulo Entradas Discretas I1* | *Módulo Entradas Discretas I4* |
|---|---|
| **I1.0** => Bañera Colocada (Confirmación Descarga) | **I4.0** => Exceso Peso Báscula 2 |
| **I1.1** => Térmico Operario 1 (NC) | **I4.1** => Falta Peso Báscula 3 |
| **I1.2** => Térmico Operario 2 (NC) | **I4.2** => Peso Ok Báscula 3 |
| **I1.3** => Térmico Operario 3 (NC) | **I4.3** => Exceso Peso Báscula 3 |
| **I1.4** => Térmico Operario 4 (NC) | **I4.4** => Validar Peso Báscula 1 |
| **I1.5** => Térmico Operario 5 (NC) | **I4.5** => Validar Peso Báscula 2 |
| **I1.6** => Térmico Cinta Báscula Entrada (NC) | **I4.6** => Validar Peso Báscula 3 |
| **I1.7** => Térmico Cinta Superior | **I4.7** => Validar Peso Báscula 4 (PREVISIÓN) |

| *Módulo Entradas Discretas I2* | *Módulo Entradas Discretas I5* |
|---|---|
| **I2.0** => Térmico Cinta Inferior | **I5.0** => Falta Peso Báscula 4 (PREVISIÓN) |
| **I2.1** => Bañera Finalizada (Fin Ciclo) | **I5.1** => Peso Ok Báscula 4 (PREVISIÓN) |
| **I2.2** => Tara Báscula Entrada | **I5.2** => Exceso Peso Báscula 4 (PREVISIÓN) |
| **I2.3** => Tara Báscula Producción | **I5.3** => LIBRE |
| **I2.4** => Selector Limpieza | **I5.4** => LIBRE |
| **I2.5** => Contador Agua | **I5.5** => LIBRE |
| **I2.6** => Contador Sal | **I5.6** => Selector Manual / Automático |
| **I2.7** => LIBRE | **I5.7** => LIBRE |

# SALIDAS

| *Módulo Salidas Discretas Q0* | *Módulo Salidas Discretas Q3* |
|---|---|
| **Q0.0** => Marcha Cinta Tolva Alimentación Báscula | **Q3.0** => Piloto Falta Peso Bascula 1 |
| **Q0.1** => Marcha Cinta Báscula | **Q3.1** => Piloto Peso OK Bascula 1 |
| **Q0.2** => Marcha Cinta Superior | **Q3.2** => Piloto Exceso Peso Bascula 1 |
| **Q0.3** => Marcha Cinta Inferior | **Q3.3** => LIBRE |
| **Q0.4** => Marcha Cinta Báscula Producción Derecha | **Q3.4** => Piloto Falta Peso Bascula 2 |
| **Q0.5** => Marcha Cinta Operario 1 | **Q3.5** => Piloto Peso OK Bascula 2 |
| **Q0.6** => Marcha Cinta Operario 2 | **Q3.6** => Piloto Exceso Peso Bascula 2 |
| **Q0.7** => Marcha Cinta Operario 3 | **Q3.7** => LIBRE |

| *Módulo Salidas Discretas Q1* | *Módulo Salidas Discretas Q4* |
|---|---|
| **Q1.0** => Marcha Cinta Operario 4 | **Q4.0** => Piloto Falta Peso Bascula 3 |
| **Q1.1** => Marcha Cinta Operario 5 | **Q4.1** => Piloto Peso OK Bascula 3 |
| **Q1.2** => Trampilla Operario 1 | **Q4.2** => Piloto Exceso Peso Bascula 3 |
| **Q1.3** => Trampilla Operario 2 | **Q4.3** => LIBRE |
| **Q1.4** => Trampilla Operario 3 | **Q4.4** => Piloto Falta Peso Bascula 4 |
| **Q1.5** => Trampilla Operario 4 | **Q4.5** => Piloto Peso OK Bascula 4 |
| **Q1.6** => Trampilla Operario 5 | **Q4.6** => Piloto Exceso Peso Bascula 4 |
| **Q1.7** => LIBRE | **Q4.7** => LIBRE |

| *Módulo Salidas Discretas Q2* | *Módulo Salidas Discretas Q5* |
|---|---|
| **Q2.0** => LIBRE | **Q5.0** => LIBRE |
| **Q2.1** => Piloto Línea Marcha | **Q5.1** => LIBRE |
| **Q2.2** => Luz Fin Ciclo | **Q5.2** => LIBRE |
| **Q2.3** => Piloto Térmicos Operarios | **Q5.3** => LIBRE |
| **Q2.4** => Piloto Térmicos Basculas | **Q5.4** => LIBRE |
| **Q2.5** => Piloto Línea Envasado Marcha | **Q5.5** => LIBRE |
| **Q2.6** => Piloto Térmicos Cintas Superior / Inferior | **Q5.6** => Auto Tara Báscula Entrada |
| **Q2.7** => LIBRE | **Q5.7** => Auto Tara Báscula Producción |

## ANEXO 3: Funciones de librería OCX y de comunicaciones

- ### Función CommOpen()

```
Public Function CommOpen() As Boolean
Timer1.Enabled = False
Timer1.Interval = Me.TimeOut
MSComm1.RThreshold = Me.iLongTrama + 1
MSComm1.SThreshold = Me.iLongTrama + 1
bConfirma = True
With MSComm1
  If .PortOpen Then
    CommOpen = False
    Exit Function
  Else
    .CommPort = Me.NumPuerto
    .Settings = Me.ConfigPuerto
    .PortOpen = True
  End If
End With
CommOpen = True
End Function
```

- ### Función CommClose()

```
Public Function CommClose() As Boolean
With MSComm1
  If .PortOpen Then
    .PortOpen = False
  End If
End With
CommClose = True
End Function
```

- ### Función Envia()

```
Public Sub Envia()
Dim sMsg As String
If bConfirma Then
  bConfirma = False
  sAux = ""
  sMsg = Me.Mensaje
  Timer1.Enabled = True
  sMsg = Chr$(2) & "N" & Chr$(Nodo) & sMsg
  Me.sTramaSalida = sCheck(sMsg, Me.iLongTrama)
  bEnvio = True
  MSComm1.Output = Me.sTramaSalida
  delay (Me.lDelay)
End If
End Sub
```

- ### Función MSComm1_OnComm()

```
Private Sub MSComm1_OnComm()
Select Case MSComm1.CommEvent
  Case comEvReceive
    If bEnvio Then
      Timer1.Enabled = False
      sAux = sAux & MSComm1.Input
      Gestion_Recivido
    End If
End Select
End Sub
```

- ### Función Gestion_Recivido()

```
Private Sub Gestion_Recivido()
Dim sRecivido As String
Dim sChkRec As String
Dim iAux As Integer
If InStr(1, sAux, Chr$(2)) <> 0 Then
  sRecivido = Mid(sAux, 1, Len(sAux) - 1)
  sChkRec = Mid(sAux, Len(sAux), 1
  If bCheckOk(sRecivido, sChkRec) Then
```

```
    bEnvio = False
    Me.Recivido = sRecivido
    Label1 = ""
    For iAux = 1 To Len(sRecivido)
      Label1 = Label1 & " ; " & Asc(Mid(sRecivido, iAux, 1))
    Next iAux
    RaiseEvent RecEv(0)
  Else
    Me.Recivido = sAux & MSComm1.Input
    RaiseEvent RecEv(1)
  End If
Else
  bEnvio = False
  Me.Recivido = sAux & MSComm1.Input
  RaiseEvent RecEv(2)
End If
End Sub
```

- **Función talkXd(iNodo, sMsg)**

```
Public Function talkXd(ByVal iNodo As Integer, ByVal sMsg As String) As String
Dim iAux As Integer
Dim iNumReintentos As Integer
Dim iPos As Integer
Dim sTime_Inicio_Watch_Dog As String
' Variables de fin de recepción
bRecepcion = False
' Variable para los reintentos
iNumReintentos = 1
reintentos:
' Form donde se encuentre el ocx de comunicación
With frmPrincipal
  sTime_Inicio_Watch_Dog = Now
  lCNT_WATCH_DOG_PILA = 0
  .Ctm800Comm.bConfirma = True
  .Ctm800Comm.VaciaBuffer
  .Ctm800Comm.Nodo = iNodo
  .Ctm800Comm.Mensaje = sMsg
  bRecepcion = False
  .Ctm800Comm.Envia
  While Not bRecepcion
    DoEvents
    ' --> Perro guardián
    If lCNT_WATCH_DOG_PILA < TIME_WATCH_DOG_PILA Then
      lCNT_WATCH_DOG_PILA = lCNT_WATCH_DOG_PILA + 1
    Else
      bRecepcion = True
      frmDepuracion.lstErrores.AddItem Time & " >> Salta Watch Dog. Espera: " &
SecondsToStrHorMinSec(segundos(sTime_Inicio_Watch_Dog, Now))
      iFalloComm = 5
      sTrama = ""
    End If
  Wend
End With
' Fallo de comunicación
If iFalloComm <> 0 Or sTrama = "" Then
  If iNumReintentos < MAX_REINTENTOS And tPila(0).iNumTarea <> T_ESTADO Then
    iNumReintentos = iNumReintentos + 1
    GoTo reintentos
  End If
  talkXd = "ERRORCOMM"
  frmPrincipal.stBar.Panels("N" & iNodo) = "Err N" & iNodo
  Exit Function
Else
  ' >> Escribimos comOk en status bar
  With frmPrincipal
    If InStr(1, .stBar.Panels("N" & iNodo), "Err") <> 0 Or Len(.stBar.Panels("N" &
iNodo)) = 0 Then
      .stBar.Panels("N" & iNodo) = "OK N" & iNodo
    End If
  End With
  talkXd = sTrama
End If
End Function
```

- **Función Ctm800Comm_RecEv()**

```
Private Sub Ctm800Comm_RecEv(iError As Integer)
Select Case iError
  Case 0
    sTrama = Me.Ctm800Comm.Recivido
    If InStr(1, sTrama, "KO") <> 0 Then
      iError = 15
      GoTo errorcrc
    ElseIf Len(sTrama) < 18 Then
      iError = 16
      GoTo errorcrc
    End If
  Case Else
errorcrc:
    sTrama = "Error"
End Select
Me.Ctm800Comm.bConfirma = True
bRecepcion = True
iFalloComm = iError
End Sub
```

- **Función AñadirTarea(iNumTarea, iNodo, bASaco)**

```
Public Sub AñadirTarea(ByVal iNumTarea As Integer, ByVal iNodo As Integer, ByVal bASaco
As Boolean)
Dim iCntTareas As Integer
If bEstaEnLaPila(iNodo, iNumTarea) And Not bASaco Then
  Exit Sub
End If
For iCntTareas = 0 To MAX_PILA
  If tPila(iCntTareas).iNodo = 0 And tPila(iCntTareas).iNumTarea = 0 Then Exit For
Next iCntTareas
If iCntTareas >= MAX_PILA Then
  MsgBox "Waiting: La pila está llena", vbInformation + vbOKOnly
  Exit Sub
Else
  tPila(iCntTareas).iNodo = iNodo
  tPila(iCntTareas).iNumTarea = iNumTarea
  Exit Sub
End If
End Sub
```

- **Función BorraTarea (iTarea)**

```
Public Sub BorraTarea(ByVal iTarea As Integer)
Dim iCntTareas As Integer
Dim iPos As Integer
iPos = DamePosicionDeTarea(iTarea)
While iPos <> -1
  For iCntTareas = iPos To MAX_PILA - 1
    tPila(iCntTareas) = tPila(iCntTareas + 1)
  Next iCntTareas
  tPila(MAX_PILA).iNodo = 0
  tPila(MAX_PILA).iNumTarea = 0
  iPos = DamePosicionDeTarea(iTarea)
Wend
End Sub
```

- **Función ConfiguraTareas()**

```
' Si nos hemos quedado sin tareas, configuramos las que estan por defecto
' Mirar el estado de los PLC's
Public Sub ConfiguraTareas()
Dim iCnt As Integer
Dim iAux As Integer
ReDim tPila(MAX_PILA)
For iCnt = 1 To NumeroLineas
      AñadirTarea T_ESTADO, 1, False
      If ParamLinea(iCnt).iTipo_Pantalla = PANTALLA_CONFIG_BASCULAS Then
        AñadirTarea T_LEER_PESOS_CONFIGURACION, ParamLinea(iCnt).iNodo, False
      End If
Next iCnt
Call EjecutaTarea
End Sub
```

## Anexo 4: Funciones del software de consultas

- **Función IniciaBasesDatos()**

```
Public Sub IniciaBasesDatos()
Dim i As Integer
'Inicio la variable de conexión activa
iConexionActiva = 0
'Data2.Connection1.ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=" &
App.Path & "\CRT-800.mdb;Mode=Read;Persist Security Info=False"
'Data3.Connection1.ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=" &
App.Path & "\CRT-800.mdb;Mode=Read;Persist Security Info=False"
'Reseteo los vectores donde guardo la información de las bases de datos
For i = 1 To 10
    BasesDatos(i) = ""
    BaseDatosActiva(i) = 0
    NBasesDatos = 0
Next i
'En primer lugar guardo las bases de datos que estén dadas de alta
With Data1
    .Commands("cmdBasesDatos").CommandText = "Select * from BasesDatos order by Id"
    .Commands("cmdBasesDatos").Execute
    .rscmdBasesDatos.Open
    If .rscmdBasesDatos.RecordCount = 0 Then
        .rscmdBasesDatos.Close
    Else
         i = 1
        While Not .rscmdBasesDatos.EOF
            BasesDatos(i) = .rscmdBasesDatos.Fields("Nombre")
            DescripcionBd(i) = .rscmdBasesDatos.Fields("Descripcion")
            NBasesDatos = i
            i = i + 1
            .rscmdBasesDatos.MoveNext
        Wend
        .rscmdBasesDatos.Close
    End If
End With
'De las bases de datos que haya dadas de alta compruebo la conexión y le doy
'el nuevo ConnectionString
For i = 1 To NBasesDatos
    Select Case i
        Case 1
           CargaBd1
        Case 2
            CargaBd2
        Case 3
            CargaBd3
        Case 4
            CargaBd4
        Case 5
            CargaBd5
```

```
        Case 6
            CargaBd6
        Case 7
            CargaBd7
        Case 8
            CargaBd8
        Case 9
            CargaBd9
        Case 10
            CargaBd10
    End Select
Next i
'Ahora guardo en la base de datos qué rutas son correctas y cuales no para saber
'si la conexión está activa
For i = 1 To NBasesDatos
    With Data1
        .Commands("cmdBasesDatos").CommandText = "Update BasesDatos set Activa='" &
BaseDatosActiva(i) & "' where Nombre='" & BasesDatos(i) & "'"
        .Commands("cmdBasesDatos").Execute
    End With
Next i
'Miro la primera de las bases de datos que esté activa y es la que dejo conectada
'por defecto
For i = 1 To NBasesDatos
    If BaseDatosActiva(i) = 1 Then
        iConexionActiva = i
        Exit For
    End If
Next i
'Si no hay ninguna activa lo indico con un mensaje
If iConexionActiva = 0 Then
    'frmInfo.lblInfo.Caption = "ERROR GRAVE. NO HAY CONEXIONES ACTIVAS"
    frmInfo.lblInfo.Caption = Textos(315)
    frmInfo.Show vbModal
End If
End Sub
```

- **Función BuscarPartida()**

```
Private Sub cmdBuscarPartida_Click()
        iBusqueda = 1
        'BorraLabels
        BorraLabelsLista
        'lblDesc1.Caption = "LOTE"
        'lblDesc2.Caption = "ORDEN"
        lblDesc1.Caption = Textos(5)
        lblDesc2.Caption = Textos(6)
        CargaLote
        Call Refresca_Lista(iLote, sLote)
        frBuscar.Visible = True
End Sub
```

- **Función CargaLote()**

```
Private Sub CargaLote()
Dim i As Integer
iPantallas = 1
'With Data1
With ConexionBd(iConexionActiva)
  .Commands("cmdPartidas").CommandText = "Select * from Partidas order by Id Desc"
  .Commands("cmdPartidas").Execute
  .rscmdPartidas.Open
  If .rscmdPartidas.RecordCount = 0 Then
    .rscmdPartidas.Close
    Exit Sub
  Else
    iLote = .rscmdPartidas.RecordCount
    ReDim sLote(iLote, 2)
    i = 1
    While Not .rscmdPartidas.EOF
      sLote(i, 1) = .rscmdPartidas.Fields("NumeroPartida")
      sLote(i, 2) = .rscmdPartidas.Fields("Orden")
      .rscmdPartidas.MoveNext
      i = i + 1
    Wend
    .rscmdPartidas.Close
  End If
End With
iNumPantallas = Int(iLote / Me.lblDescripcion1.Count)
If (iLote Mod Me.lblDescripcion1.Count) <> 0 Then iNumPantallas = iNumPantallas + 1
End Sub
```

- **Función Refresca_Lista(iNumRegistros, sArray())**

```
Private Sub Refresca_Lista(iNumRegistros As Integer, sArray() As String)
Dim iCnt As Integer
For iCnt = 1 To Me.lblDescripcion1.Count
  If ((iPantallas - 1) * Me.lblDescripcion1.Count) + iCnt <= iNumRegistros Then
    Me.lblDescripcion1(iCnt) = sArray(((iPantallas - 1) * Me.lblDescripcion1.Count) +
iCnt, 1)
    Me.lblDescripcion2(iCnt) = sArray(((iPantallas - 1) * Me.lblDescripcion1.Count) +
iCnt, 2)
  Else
    Me.lblDescripcion1(iCnt) = ""
    Me.lblDescripcion2(iCnt) = ""
  End If
Next iCnt
End Sub
```

- **Función VisualizarDatosPartida()**

```
Private Sub VisualizarDatosPartida()
Dim i, ii As Integer
Dim dPesoOperarios As Double
Dim dPesoOperariosEnvasado As Double
Dim dKghObtenidos As Double
Dim dMerma As Double
Dim HorasTrabajadas As String
Dim DiferenciaTeorico As Double
Dim Nivel As String

dKghObtenidos = 0
dPesoOperarios = 0
dHorasTrabajadas = 0
DiferenciaTeorico = 0
Nivel = ""

BorraTodoPartidas

If Me.txtPartida.Text = "" Or Me.txtOrden.Text = "" Then Exit Sub
'Borro la tabla PartidaOperariosTemp
With Data1
    .Commands("cmdPartidaOperariosTemp").CommandText = "Delete * from
PartidaOperariosTemp"
    .Commands("cmdPartidaOperariosTemp").Execute
End With
With Data1
    .Commands("cmdPartidaOperariosEnvasadoTemp").CommandText = "Delete * from
PartidaOperariosEnvasadoTemp"
    .Commands("cmdPartidaOperariosEnvasadoTemp").Execute
End With
'Dependiendo si queremos todas las órdenes o solamente una seleccionamos todas
'las partidas o solo la partida con su orden
'Primero obtengo los datos referentes a la partida
With ConexionBd(iConexionActiva)
    .Commands("cmdPartidas").CommandText = "Select * from Partidas where
NumeroPartida='" & Me.txtPartida.Text & "' and Orden=" & Me.txtOrden.Text & ""
    .Commands("cmdPartidas").Execute
    .rscmdPartidas.Open
    If .rscmdPartidas.RecordCount = 0 Then
        .rscmdPartidas.Close
        'frmInfo.lblInfo.Caption = "NO SE HA ENCONTRADO EL LOTE"
        frmInfo.lblInfo.Caption = Textos(7)
        frmInfo.Show
        Exit Sub
    Else
        If Not .rscmdPartidas.Fields("PesoCongelado") = Empty Then
            Me.txtPesoCongelado.Text = .rscmdPartidas.Fields("PesoCongelado")
        Else
            Me.txtPesoCongelado.Text = 0
```

```
End If
If Not .rscmdPartidas.Fields("PesoDescongelado") = Empty Then
    Me.txtPesoDescongelado.Text = .rscmdPartidas.Fields("PesoDescongelado")
Else
    Me.txtPesoDescongelado.Text = 0
End If
If Not .rscmdPartidas.Fields("PesoEnvasado") = Empty Then
    Me.txtPesoElaborado.Text = .rscmdPartidas.Fields("PesoEnvasado")
Else
    Me.txtPesoElaborado.Text = 0
End If
If Not .rscmdPartidas.Fields("PesoOperarios") = Empty Then
    Me.txtPesoOperarios.Text = .rscmdPartidas.Fields("PesoOperarios") / 100
Else
    Me.txtPesoOperarios.Text = 0
End If
If Not .rscmdPartidas.Fields("CodigoTratamiento") = Empty Then
    Me.txtCodTratamiento.Text = .rscmdPartidas.Fields("CodigoTratamiento")
Else
    Me.txtCodTratamiento.Text = ""
End If
If Not .rscmdPartidas.Fields("FichaTecnica") = Empty Then
    Me.txtFichaTecnica.Text = .rscmdPartidas.Fields("FichaTecnica")
Else
    Me.txtFichaTecnica.Text = ""
End If
If Not .rscmdPartidas.Fields("Observaciones") = Empty Then
    Me.txtObservaciones.Text = .rscmdPartidas.Fields("Observaciones")
Else
    Me.txtObservaciones.Text = ""
End If
If Not .rscmdPartidas.Fields("Incidencias") = Empty Then
    Me.txtIncidencias.Text = .rscmdPartidas.Fields("Incidencias")
Else
    Me.txtIncidencias.Text = ""
End If
If Not .rscmdPartidas.Fields("FechaInicio") = Empty Then
    Me.txtFechaInicial.Text = .rscmdPartidas.Fields("FechaInicio")
Else
    Me.txtFechaInicial.Text = ""
End If
If Not .rscmdPartidas.Fields("FechaFin") = Empty Then
    Me.txtFechaFinal.Text = .rscmdPartidas.Fields("FechaFin")
Else
    Me.txtFechaFinal.Text = ""
End If
If Not .rscmdPartidas.Fields("HoraInicio") = Empty Then
    Me.txtHoraInicio.Text = .rscmdPartidas.Fields("HoraInicio")
Else
    Me.txtHoraInicio.Text = ""
End If
```

```
        If Not .rscmdPartidas.Fields("HoraFin") = Empty Then
            Me.txtHoraFin.Text = .rscmdPartidas.Fields("HoraFin")
        Else
            Me.txtHoraFin.Text = ""
        End If
        .rscmdPartidas.Close
    End If
End With
'Ahora cojo los datos referentes a la ficha técnica
With ConexionBd(iConexionActiva)
    .Commands("cmdFichasTecnicas").CommandText = "Select * from FichasTecnicas Where
FichaTecnica='" & Me.txtFichaTecnica.Text & "'"
    .Commands("cmdFichasTecnicas").Execute
    .rscmdFichasTecnicas.Open
    If .rscmdFichasTecnicas.RecordCount = 0 Then
        .rscmdFichasTecnicas.Close
    Else
        If Not .rscmdFichasTecnicas.Fields("PesoHora") = Empty Then
            Me.txtKghTeoricos.Text = .rscmdFichasTecnicas.Fields("PesoHora")
        Else
            Me.txtKghTeoricos.Text = ""
        End If
        .rscmdFichasTecnicas.Close
    End If
End With

'Ahora obtengo todos los operarios que han participado en esta partida
With ConexionBd(iConexionActiva)
    .Commands("cmdPesosOperarios").CommandText = "Select distinct NumeroPartida,
CodigoOperario, Orden from PesosOperarios Where NumeroPartida='" & Me.txtPartida.Text &
"' and Orden=" & Me.txtOrden.Text & ""
    .Commands("cmdPesosOperarios").Execute
    .rscmdPesosOperarios.Open
    If .rscmdPesosOperarios.RecordCount = 0 Then
        .rscmdPesosOperarios.Close
    Else
        i = 1
        NOperarios = .rscmdPesosOperarios.RecordCount
        ReDim Operarios(NOperarios)
        ReDim NPesos(NOperarios)
        ReDim NPesosEntrada(NOperarios)
        While Not .rscmdPesosOperarios.EOF
            Operarios(i).CodigoOperario = .rscmdPesosOperarios.Fields("CodigoOperario")
            i = i + 1
            .rscmdPesosOperarios.MoveNext
        Wend
        .rscmdPesosOperarios.Close
    End If
End With
'Ahora obtengo todos los operarios que han participado en el envasado
With ConexionBd(iConexionActiva)
```

```
    .Commands("cmdPesosOperarios").CommandText = "Select distinct NumeroPartida,
CodigoOperario, Orden from PesosEnvasado Where NumeroPartida='" & Me.txtPartida.Text &
"' and Orden=" & Me.txtOrden.Text & ""
    .Commands("cmdPesosOperarios").Execute
    .rscmdPesosOperarios.Open
    If .rscmdPesosOperarios.RecordCount = 0 Then
        .rscmdPesosOperarios.Close
    Else
        i = 1
        NOperariosEnvasado = .rscmdPesosOperarios.RecordCount
        ReDim OperariosEnvasado(NOperariosEnvasado)
        ReDim NPesosEnvasado(NOperariosEnvasado)
        While Not .rscmdPesosOperarios.EOF
            OperariosEnvasado(i).CodigoOperario =
.rscmdPesosOperarios.Fields("CodigoOperario")
            i = i + 1
            .rscmdPesosOperarios.MoveNext
        Wend
        .rscmdPesosOperarios.Close
    End If
End With
'Una vez tengo a todos los operarios consulto los datos relativos a cada uno
Dim k As Integer
k = 1
For i = 1 To NOperarios
    With ConexionBd(iConexionActiva)
    'Quito la relación entre las tablas por si se ha eliminado a un operario que haya
intervenido
    'en la partida que igualmente salgan los datos
    'El nombre lo guardo después
        .Commands("cmdPesosOperarios").CommandText = "Select distinct * from
PesosOperarios Where CodigoOperario=" & Operarios(i).CodigoOperario & " and
NumeroPartida='" & Me.txtPartida.Text & "' and Orden=" & Me.txtOrden.Text & " order by
Id"
        .Commands("cmdPesosOperarios").Execute
        .rscmdPesosOperarios.Open
        If .rscmdPesosOperarios.RecordCount = 0 Then
            .rscmdPesosOperarios.Close
        Else
            NPesos(i) = .rscmdPesosOperarios.RecordCount
            ii = 1
            ReDim Operarios(i).Peso(NPesos(i))
            ReDim Operarios(i).PesoAcc(NPesos(i))
            ReDim Operarios(i).KgH(NPesos(i))
            ReDim Operarios(i).Hora(NPesos(i))
            ReDim Operarios(i).Fecha(NPesos(i))
            ReDim Operarios(i).TiempoEnLinea(NPesos(i))
            While Not .rscmdPesosOperarios.EOF
                Operarios(i).Peso(ii) = .rscmdPesosOperarios.Fields("Peso") / 100
                Operarios(i).PesoAcc(ii) = .rscmdPesosOperarios.Fields("PesoAcc") / 100
                Operarios(i).KgH(ii) = .rscmdPesosOperarios.Fields("KgH")
```

```
                Operarios(i).Hora(ii) = .rscmdPesosOperarios.Fields("Hora")
                'No tengo el tiempo que está en la línea. Lo saco de los kg/h que
                'lleva
                If Not (Operarios(i).PesoAcc(ii) = 0 Or Operarios(i).KgH(ii) = 0) Then
                    Operarios(i).TiempoEnLinea(ii) = CDate((Operarios(i).PesoAcc(ii) /
Operarios(i).KgH(ii)) / 24)
                End If
                If Operarios(i).TiempoEnLinea(ii) = "" Then
                    Operarios(i).TiempoEnLinea(ii) = "00:00:00"
                End If
                Operarios(i).Fecha(ii) = .rscmdPesosOperarios.Fields("Fecha")
                ii = ii + 1
                .rscmdPesosOperarios.MoveNext
            Wend
            .rscmdPesosOperarios.Close
        End If
    End With
Next i


'PARTE PESOS ENTRADA
For i = 1 To NOperarios
    With ConexionBd(iConexionActiva)
    'Quito la relación entre las tablas por si se ha eliminado a un operario que haya
intervenido
    'en la partida que igualmente salgan los datos
    'El nombre lo guardo después
        .Commands("cmdPesosOperarios").CommandText = "Select distinct * from
PesosEntradaOperarios Where CodigoOperario=" & Operarios(i).CodigoOperario & " and
NumeroPartida='" & Me.txtPartida.Text & "' and Orden=" & Me.txtOrden.Text & " order by
Id"
        .Commands("cmdPesosOperarios").Execute
        .rscmdPesosOperarios.Open
        If .rscmdPesosOperarios.RecordCount = 0 Then
            .rscmdPesosOperarios.Close
        Else
            NPesosEntrada(i) = .rscmdPesosOperarios.RecordCount
            ii = 1
            ReDim Operarios(i).PesoEntrada(NPesosEntrada(i))
            ReDim Operarios(i).PesoEntradaAcc(NPesosEntrada(i))
            While Not .rscmdPesosOperarios.EOF
                Operarios(i).PesoEntrada(ii) = .rscmdPesosOperarios.Fields("Peso")
                Operarios(i).PesoEntradaAcc(ii) = .rscmdPesosOperarios.Fields("PesoAcc")
                ii = ii + 1
                .rscmdPesosOperarios.MoveNext
            Wend
            .rscmdPesosOperarios.Close
        End If
    End With
Next i
```

```
'PARTE PESOS ENVASADO
For i = 1 To NOperariosEnvasado
    With ConexionBd(iConexionActiva)
    'Quito la relación entre las tablas por si se ha eliminado a un operario que haya
intervenido
    'en la partida que igualmente salgan los datos
    'El nombre lo guardo después
        .Commands("cmdPesosOperarios").CommandText = "Select distinct * from
PesosEnvasado Where CodigoOperario=" & OperariosEnvasado(i).CodigoOperario & " and
NumeroPartida='" & Me.txtPartida.Text & "' and Orden=" & Me.txtOrden.Text & " order by
Id"
        .Commands("cmdPesosOperarios").Execute
        .rscmdPesosOperarios.Open
        If .rscmdPesosOperarios.RecordCount = 0 Then
            .rscmdPesosOperarios.Close
        Else
            NPesosEnvasado(i) = .rscmdPesosOperarios.RecordCount
            ii = 1
            ReDim OperariosEnvasado(i).PesoEnvasado(NPesosEnvasado(i))
            ReDim OperariosEnvasado(i).PesoEnvasadoAcc(NPesosEnvasado(i))

            While Not .rscmdPesosOperarios.EOF
                OperariosEnvasado(i).PesoEnvasado(ii) =
.rscmdPesosOperarios.Fields("Peso")
                OperariosEnvasado(i).PesoEnvasadoAcc(ii) =
.rscmdPesosOperarios.Fields("PesoAcc")
                OperariosEnvasado(i).Cajas = .rscmdPesosOperarios.Fields("NumeroPesada")
                ii = ii + 1
                .rscmdPesosOperarios.MoveNext
            Wend
            .rscmdPesosOperarios.Close
        End If
    End With
Next i
'Ahora guardo el nombre de cada operario que ha intervenido
For i = 1 To NOperarios
    With ConexionBd(iConexionActiva)
        .Commands("cmdOperarios").CommandText = "Select * from Operarios Where
CodigoOperario=" & Operarios(i).CodigoOperario & ""
        .Commands("cmdOperarios").Execute
        .rscmdOperarios.Open
        If .rscmdOperarios.RecordCount = 0 Then
            Operarios(i).Nombre = ""
            .rscmdOperarios.Close
        Else
            Operarios(i).Nombre = .rscmdOperarios.Fields("Nombre")
            .rscmdOperarios.Close
        End If
    End With
Next i
'Ahora guardo el nombre de cada operario que ha intervenido en el envasado
```

```
For i = 1 To NOperariosEnvasado
    With ConexionBd(iConexionActiva)
        .Commands("cmdOperarios").CommandText = "Select * from Operarios Where
CodigoOperario=" & OperariosEnvasado(i).CodigoOperario & ""
        .Commands("cmdOperarios").Execute
        .rscmdOperarios.Open
        If .rscmdOperarios.RecordCount = 0 Then
            OperariosEnvasado(i).Nombre = ""
            .rscmdOperarios.Close
        Else
            OperariosEnvasado(i).Nombre = .rscmdOperarios.Fields("Nombre")
            .rscmdOperarios.Close
        End If
    End With
Next i
'Ahora obtengo el peso total de todos los operarios sumando el peso acumulado
'de cada uno de ellos
'También calculo la media de Kgh de todos los operarios
'Guardo los datos en la tabla PartidaOperariosTemp para cuando quiera imprimir

For i = 1 To NOperarios
    dPesoOperarios = dPesoOperarios + Operarios(i).PesoAcc(NPesos(i))
    dKghObtenidos = dKghObtenidos + Operarios(i).KgH(NPesos(i))
    If Operarios(i).PesoEntradaAcc(NPesosEntrada(i)) <> 0 Then
        Operarios(i).Merma = Format(1 - (Operarios(i).PesoAcc(NPesos(i)) /
Operarios(i).PesoEntradaAcc(NPesosEntrada(i))), "##0.0000") * 100
    Else
        Operarios(i).Merma = 100
    End If
    HorasTrabajadas = CStr(Operarios(i).TiempoEnLinea(NPesos(i)))
    DiferenciaTeorico = Format(CDbl(Operarios(i).KgH(NPesos(i)) -
Me.txtKghTeoricos.Text), "#0.00#")
    If DiferenciaTeorico >= 0 Then
        Nivel = "OK"
    Else
        Nivel = "--"
    End If
    With Data1
        .Commands("cmdPartidaOperariosTemp").CommandText = "Insert into
PartidaOperariosTemp (Codigo, Nombre, KilosEntrada, Merma, Producido, Kgh,
DiferenciaTeorico, HorasTrabajadas, Nivel) values ('" & Operarios(i).CodigoOperario &
"', '" & Operarios(i).Nombre & "', '" & Operarios(i).PesoEntradaAcc(NPesosEntrada(i)) &
"', '" & Operarios(i).Merma & "', '" & Operarios(i).PesoAcc(NPesos(i)) & "', '" &
Operarios(i).KgH(NPesos(i)) & "', '" & DiferenciaTeorico & "', '" & HorasTrabajadas &
"', '" & Nivel & "')"
        .Commands("cmdPartidaOperariosTemp").Execute
    End With
Next i
For i = 1 To 1
    With Data1
```

```
        .Commands("cmdPartidaOperariosTemp").CommandText = "Insert into
PartidaOperariosTemp (Codigo) values (' ')"
        .Commands("cmdPartidaOperariosTemp").Execute
    End With
Next i
If Data3.rscmdPartidaOperariosTemp.State = adStateOpen Then
Data3.rscmdPartidaOperariosTemp.Close
For i = 1 To 10
    Data3.Commands("cmdPartidaOperariosTemp").CommandText = "Select * from
PartidaOperariosTemp order by Codigo"
    Data3.Commands("cmdPartidaOperariosTemp").Execute
    Data3.rscmdPartidaOperariosTemp.Open
    Data3.rscmdPartidaOperariosTemp.Close
Next i
If Not dKghObtenidos = 0 Or Not NOperarios = 0 Then
    dKghObtenidos = Format(dKghObtenidos / NOperarios, "###0.00")
End If
Me.txtKgObtenidos.Text = dKghObtenidos
Me.txtPesoOperarios.Text = dPesoOperarios


'Ahora haré los cálculos para sacar la caída de porcentaje en los respectivos pesos
'Guardo los datos en la tabla PartidaOperariosTemp para cuando quiera imprimir
For i = 1 To NOperariosEnvasado
    dPesoOperariosEnvasado = dPesoOperariosEnvasado +
OperariosEnvasado(i).PesoEnvasadoAcc(NPesosEnvasado(i))
    With Data1
        .Commands("cmdPartidaOperariosTemp").CommandText = "Insert into
PartidaOperariosEnvasadoTemp (Codigo, Nombre, Kilos, Cajas) values ('" &
OperariosEnvasado(i).CodigoOperario & "', '" & OperariosEnvasado(i).Nombre & "', '" &
OperariosEnvasado(i).PesoEnvasadoAcc(NPesosEnvasado(i)) & "', '" &
OperariosEnvasado(i).Cajas & "')"
        .Commands("cmdPartidaOperariosTemp").Execute
    End With
Next i
For i = 1 To 1
    With Data1
        .Commands("cmdPartidaOperariosTemp").CommandText = "Insert into
PartidaOperariosEnvasadoTemp (Codigo) values (' ')"
        .Commands("cmdPartidaOperariosTemp").Execute
    End With
Next i
If Data3.rscmdPartidaOperariosEnvasadoTemp.State = adStateOpen Then
Data3.rscmdPartidaOperariosEnvasadoTemp.Close
For i = 1 To 10
    Data3.Commands("cmdPartidaOperariosEnvasadoTemp").CommandText = "Select * from
PartidaOperariosEnvasadoTemp order by Codigo"
    Data3.Commands("cmdPartidaOperariosEnvasadoTemp").Execute
    Data3.rscmdPartidaOperariosEnvasadoTemp.Open
    Data3.rscmdPartidaOperariosEnvasadoTemp.Close
Next i
```

```
'************************************************************************'
If Not Me.txtPesoCongelado.Text = 0 Then
    Me.lblPorcentaje_1.Caption = Format(100 - ((Me.txtPesoDescongelado.Text /
Me.txtPesoCongelado.Text) * 100), "###0.0") & " %"
Else
    Me.lblPorcentaje_1.Caption = "< 0 %"
End If
If Not Me.txtPesoDescongelado.Text = 0 Then
    Me.lblPorcentaje_2.Caption = Format(100 - ((Me.txtPesoOperarios.Text /
Me.txtPesoDescongelado.Text)) * 100, "###0.0") & " %"
Else
    Me.lblPorcentaje_2.Caption = "< 0 %"
End If
If Not Me.txtPesoOperarios.Text = 0 Then
    Me.lblPorcentaje_3.Caption = Format(100 - ((Me.txtPesoElaborado.Text /
Me.txtPesoOperarios.Text)) * 100, "###0.0") & " %"
Else
    Me.lblPorcentaje_3.Caption = "< 0 %"
End If


If Not Me.txtPesoCongelado.Text = 0 Then
    Me.lblPorcentaje_4.Caption = Format(100 - ((Me.txtPesoElaborado.Text /
Me.txtPesoCongelado.Text)) * 100, "###0.0") & " %"
Else
    Me.lblPorcentaje_4.Caption = "< 0 %"
End If
If Not Me.txtPesoCongelado.Text = 0 Then
    Me.lblPorcentaje_5.Caption = Format(100 - ((Me.txtPesoOperarios.Text /
Me.txtPesoCongelado.Text)) * 100, "###0.0") & " %"
Else
    Me.lblPorcentaje_5.Caption = "< 0 %"
End If
Me.lblPorcentaje_1.Visible = True
Me.lblPorcentaje_2.Visible = True
Me.lblPorcentaje_3.Visible = True
Me.lblPorcentaje_4.Visible = True
Me.lblPorcentaje_5.Visible = True
Me.Line1.Visible = True
Me.Line2.Visible = True
Me.Line3.Visible = True
Me.Line4.Visible = True
'Refresco el grid de los históricos
RefrescaGridHistoricos
'Refresco grid de los operarios
RefrescaGridOperarios
End Sub
```

BIBLIOGRAFIA


1. Microsoft Corporation. *Microsoft Visual Basic 6.0. Manual del programador*. Editorial McGrawHill. 1ª edición, Barcelona, 1998. ISBN: 8448120620.
2. González Rueda, Emilio. *Programación de autómatas SIMATIC S7-300. Lenguaje AWL*. Ediciones Ceysa. 1ª edición, Madrid, 2004. ISBN: 8486108519.
3. Hopcroft, John E.; Motwani, Rajeev; Ullman, Jeffrey D. *Introduction to automata theory, languages, and computation*. Addison Wesley. 2ª edición, 2003. ISBN: 0321210298.
4. Rivera Loaiza, Cuauhtémoc. Definiciones formales [fichero en línea]. Actualitzación 23 de mayo de 2000. http://www.fismat.umich.mx/~crivera/tesis/node26.html [última consulta: 29 de enero 2007].
5. Gea, Jose Manuel. Grafcet [fichero en línea]. Actualización 2005. http://www.automatas.org/redes/grafcet.htm [última consulta: 29 de enero 2007].
6. Download Electrical Training Courses – Power Distributions, Motors & Controls [fichero en línea]. http://www.sea.siemens.com/step/downloads.html [última consulta: 24 de marzo de 2006].
7. Técnicas de electrónica y automatismos, S.A. Utilcell: Células de carga [fichero en línea]. http://www.utilcell.es [última consulta: 18 de junio de 2006].
8. Grupo Epelsa: Balanzas, básculas y pesaje electrónico [fichero en línea]. http://www.grupoepelsa.com/productos/producto.php?did=2&gid=40&pid=186 [última consulta: 25 de junio de 2006].