Universidad de San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ingeniería Mecánica Eléctrica

DISEÑO DE SISTEMA DE MONITOREO DE PRODUCCIÓN POR MEDIO DEL ALMACENAMIENTO DE DATOS EN UN DISPOSITIVO DE ALMACENAMIENTO USB

Manuel Alejandro Díaz Oliva
Asesorado por el Ing. Juan Carlos Sánchez Meyer

Guatemala, agosto de 2013

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

FACULTAD DE INGENIERÍA

DISEÑO DE SISTEMA DE MONITOREO DE PRODUCCIÓN POR MEDIO DEL ALMACENAMIENTO DE DATOS EN UN DISPOSITIVO DE ALMACENAMIENTO USB

TRABAJO DE GRADUACIÓN

PRESENTADO A LA JUNTA DIRECTIVA DE LA
FACULTAD DE INGENIERÍA
POR

MANUEL ALEJANDRO DÍAZ OLIVA
ASESORADO POR EL ING. JUAN CARLOS SÁNCHEZ MEYER

AL CONFERÍRSELE EL TÍTULO DE

INGENIERO ELECTRÓNICO

GUATEMALA, AGOSTO DE 2013

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA

FACULTAD DE INGENIERÍA

NÓMINA DE JUNTA DIRECTIVA

DECANO	Ing. Murphy Olympo Paiz Recinos
VOCAL I	Ing. Alfredo Enrique Beber Aceituno
VOCAL II	Ing. Pedro Antonio Aguilar Polanco
VOCAL III	Inga. Elvia Miriam Ruballos Samayoa
VOCAL IV	Br. Walter Rafael Véliz Muñoz
VOCAL V	Br. Sergio Alejandro Donis Soto
SECRETARIO	Ing. Hugo Humberto Rivera Pérez

TRIBUNAL QUE PRACTICÓ EL EXAMEN GENERAL PRIVADO

DECANO	Ing. Murphy Olympo Paiz Recinos
EXAMINADOR	Ing. Enrique Edmundo Ruiz Carballo
EXAMINADOR	Ing. Julio César Solares Peñate
EXAMINADOR	Ing. Marvin Marino Hernández Fernández
SECRETARIO	Ing. Hugo Humberto Rivera Pérez

HONORABLE TRIBUNAL EXAMINADOR

En cumplimiento con los preceptos que establece la ley de la Universidad de San Carlos de Guatemala, presento a su consideración mi trabajo de graduación titulado:

DISEÑO DE SISTEMA DE MONITOREO DE PRODUCCIÓN POR MEDIO DEL ALMACENAMIENTO DE DATOS EN UN DISPOSITIVO DE ALMACENAMIENTO USB

Tema que me fuera asignado por la Dirección de la Escuela de Ingeniería Mecánica Eléctrica, con fecha 15 de diciembre de 2012.

Manuel Alejandro Díaz Oliva

Guatemala, 22 de noviembre de 2011

Ingeniero
Carlos Guzmán
Coordinador de Ingeniería Mecánica Eléctrica
Escuela de Ingeniería Mecánica Eléctrica
Universidad de San Carlos de Guatemala
Ciudad Universitaria, zona 12

Estimado Señor Coordinador.

Por este medio me dirijo a usted para presentarle el trabajo de graduación titulado: **Diseño de sistema de monitoreo de producción por medio del almacenamiento de datos en un dispositivo de almacenamiento USB**, desarrollado por el estudiante Manuel Alejandro Díaz Oliva.

A mi juicio cumple con los objetivos planteados, con un contenido interesante, útil y de actualidad. Por lo tanto el autor de este trabajo de graduación y yo, como asesor, nos hacemos responsables por el contenido y conclusiones del mismo.

Me es grato informarle que el presente trabajo de graduación me es satisfactorio, por lo que me permito someterlo a su consideración y aprobación.

Sin otro particular, me suscribo a usted,

Atentamente,

Juan C. Sánchez

Ing. Juan Carlos Sánchez Meyer
Colegiado No. 8496

Juan Carlos Sánchez Meyer
Ingeniero Electrónico
Colegiado No. 8496

UNIVERSIDAD DE SAN CARLOS
DE GUATEMALA

FACULTAD DE INGENIERIA

Ref. EIME 06. 2012
Guatemala, 02 de FEBRERO 2012.

Señor Director
Ing. Guillermo Antonio Puente Romero
Escuela de Ingenierìa Mecànica Elèctrica
Facultad de Ingenierìa, USAC.

Señor Director:

Me permito dar aprobación al trabajo de Graduación titulado: DISEÑO DE SISTEMA DE MONITOREO DE PRODUCCIÓN POR MEDIO DEL ALMACENAMIENTO DE DATOS EN UN DISPOSITIVO DE ALMACENAMIENTO USB, del estudiante Manuel Alejandro Díaz Oliva, que cumple con los requisitos establecidos para tal fin.

Sin otro particular, aprovecho la oportunidad para saludarle.

Atentamente,
ID Y ENSEÑAD A TODOS

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA
DIRECCION ESCUELA DE INGENIERIA MECANICA ELECTRICA
FACULTAD DE INGENIERIA

Ing. Carlos Eduardo Guzmán Salazar
Coordinador de Electrónica

CEGS/sro

UNIVERSIDAD DE SAN CARLOS
DE GUATEMALA

FACULTAD DE INGENIERIA

REF. EIME 29. 2012.

El Director de la Escuela de Ingeniería Mecánica Eléctrica, después de conocer el dictamen del Asesor, con el Visto Bueno del Coordinador de Área, al trabajo de Graduación del estudiante; MANUEL ALEJANDRO DÍAZ OLIVA titulado: DISEÑO DE SISTEMA DE MONITOREO DE PRODUCCIÓN POR MEDIO DEL ALMACENAMIENTO DE DATOS EN UN DISPOSITIVO DE ALMACENAMIENTO USB procede a la autorización del mismo.

Ing. Guillermo Antonio Puente Romero

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA
DIRECCION ESCUELA DE INGENIERIA MECANICA ELECTRICA
FACULTAD DE INGENIERIA

GUATEMALA, 04 DE JUNIO 2,012.

Universidad de San Carlos
de Guatemala

Facultad de Ingeniería
Decanato

DTG. 580 .2013

El Decano de la Facultad de Ingeniería de la Universidad de San Carlos de Guatemala, luego de conocer la aprobación por parte del Director de la Escuela de Ingeniería Mecánica Eléctrica, al Trabajo de Graduación titulado: **DISEÑO DE SISTEMA DE MONITOREO DE PRODUCCIÓN POR MEDIO DEL ALMACENAMIENTO DE DATOS EN UN DISPOSITIVO DE ALMACENAMIENTO USB,** presentado por el estudiante universitario **Manuel Alejandro Díaz Oliva,** autoriza la impresión del mismo.

IMPRÍMASE:

Ing. Murphy Olympo Paiz Recinos
Decano

Guatemala, 20 de agosto de 2013

UNIVERSIDAD DE SAN CARLOS DE GUATEMALA
DECANO
FACULTAD DE INGENIERIA

/gdech

ACTO QUE DEDICO A:

Dios	Por ser la fuerza que impulsa mi vida.
Mis padres	Magda Rendón y José Díaz por su amor y apoyo incondicional.
Mi familia	Por compartir esta etapa de la vida conmigo.
Mis amigos	Por ser ese tesoro que Dios ha puesto a mi lado.

AGRADECIMIENTOS A:

La Universidad de San Carlos de Guatemala	Por ser la fuente de parte del conocimiento que me permite servir a mi país.
Facultad de Ingeniería	Por haberme abierto las puertas.
Ing. Juan Carlos Sánchez	Por ser una importante influencia en mi vida.
ACISA	Por darme la oportunidad de desarrollar mi potencial e ingenio.

ÍNDICE GENERAL

ÍNDICE DE ILUSTRACIONES

FIGURAS

TABLAS

LISTA DE SÍMBOLOS

Símbolo	Significado
B	Bit
<sp>	Carácter espacio libre ASCII
<cr>	Carácter retorno de carro ASCII
<tb>	Carácter tabulador ASCII
bcd	Decimal codificado binario
Hz	Hertzio o Hertz, unidad de medida de frecuencia
Hr	Horas
bm	Mapa de bits
Ohm	Ohmio, unidad de medida de resistencia
w	Palabra
Pf	Pico faradio, medida de capacitancia
s	Segundo, unidad de medida de tiempo

V	Voltio, unidad de medida de la tensión eléctrica

GLOSARIO

ASCII	Código estándar americano para el intercambio de información.
Baudio	Unidad de medida que representa la cantidad de veces que cambia el estado de una señal en un período de tiempo. En este caso es igual a 1 bit/s.
Bit	Representación mínima digital, puede tomar únicamente dos valores 1 o 0.
BOMS	Dispositivo únicamente de almacenamiento masivo.
Byte	Conjunto de ocho bits.
CPU	Unidad de procesamiento central.
DCE	Equipo de comunicación de datos.
Driver	Es un programa de computadora que permite a estas comunicarse con dispositivos de hardware externo.
DTE	Equipo terminal de datos.
EPROM	Memoria de sólo lectura programable y borrable.

EndPoint	Se define como una única porción capaz de ser direccionada del dispositivo USB, la cual es fuente o sumidero de la información durante la comunicación entre el anfitrión y el dispositivo.
Ethernet	Estándar para la comunicación entre ordenadores.
E-FLASH	Se refiere a la memoria ROM del VNC1L.
FIFO	Primero que entra, primero que sale.
Firmware	Es el intermediario entre las órdenes externas que recibe el dispositivo y su electrónica.
Half-duplex	En comunicaciones se refiera a un canal de comunicación que permite la comunicación en ambas direcciones, pero en una sola a la vez.
Handshaking	Es un proceso automatizado de negociación que establece de forma dinámica los parámetros de un canal de comunicaciones establecido entre dos entidades antes de que comience la comunicación normal por el canal.
Hardware	Corresponde a todas las partes tangibles de un sistema informático.
HTL	Lógica de alto umbral.

Hub	Es un dispositivo que permite centralizar el cableado de una red y poder ampliarla. Esto significa que dicho dispositivo recibe una señal y repite esta señal emitiéndola por sus diferentes puertos.
MCU	Unidad principal de control.
MP3	Es un formato de compresión de audio digital patentado que usa un algoritmo con pérdida para conseguir un menor tamaño de archivo.
MODEM	Es un dispositivo que permite la transmisión y recepción de información binaria (es decir, datos de ordenador) a través de un medio analógico.
NPU	Unidad de procesamiento numérico.
NRZ	Código de transmisión en el cual el potencial de la línea de transmisión no retorna a cero entre la transmisión de bits 1 consecutivos.
NULL MODEM	Es un método para conectar dos terminales usando un cable serie RS-232.
PLC	Control lógico programable.
PLL	Lazo de seguimiento de fase.
RAM	Memoria de acceso aleatorio.

ROM	Memoria de sólo lectura.
RS-232	Es una interfaz que designa una norma para el intercambio de una serie de datos binarios entre un DTE y un DCE.
RS-485	Es un estándar de comunicación al igual que el RS-232. Se caracteriza por la transmisión multipunto diferencial.
SDRAM	Memoria de acceso aleatorio dinámico sincronizado.
Software	Es el equipamiento lógico intangible que contiene las instrucciones de operación de un equipo.
SPI	Interface serial periférica.
String	Tipo de variable de programa que permite almacenar cadenas de caracteres.
TTL	Lógica transistor transistor.
UART	Transmisor y receptor asíncrono universal.
USB	Puerto serial universal.
Word	Es el conjunto de 16 bits.

RESUMEN

Se describen en este trabajo de graduación las características principales de la estructura y la operación de los dispositivos seleccionados para formar parte del sistema de monitoreo de producción, así como, el criterio usado para la selección de estos dispositivos.

El capítulo 1 presenta los conceptos básicos del USB y se enfoca en la explicación de las principales características del protocolo USB.

El capítulo 2 describe la estructura y operación del dispositivo anfitrión USB VNC1L. Este es el dispositivo usado como base para la implementación del sistema de monitoreo de producción. Describe cada uno de los bloques funcionales que lo forman, así como, las interfaces de comunicación con las que cuenta el dispositivo y las características y comandos principales del firmware (interprete de comandos) que se utiliza en esta aplicación.

El capítulo 3 describe los componentes principales y las formas generales de direccionamiento y programación de un control lógico programable (PLC) genérico. En la parte final del capítulo se exponen las características de interés del PLC FESTO FEC-FC-XX.

El capítulo 4 describe el diseño del sistema de monitoreo de producción, empezando por la descripción de la aplicación, seguido de los diagramas de bloques del sistema, diagramas de flujo del programa de adquisición de datos, descripción de la interface HMI realizada en MS Excel y finalizando con un análisis del costo de implementación del sistema de monitoreo.

OBJETIVOS

General

Desarrollar un sistema de monitoreo de producción de costo accesible que pueda ser implementado en corto tiempo y permita almacenar los datos de producción de manera que facilite su análisis y acceso.

Específicos

1. Mostrar de manera ordenada el diseño del sistema de monitoreo de producción y la interfaz de usuario.

2. Ser un punto de partida para el lector interesado en el desarrollo de su propio sistema de monitoreo de producción.

3. Presentar un sistema de monitoreo de producción accesible en cuanto a costo, tiempo de implementación y complejidad, para la pequeña y mediana industria Guatemalteca.

4. Ejemplificar el uso de MS Excel como herramienta de desarrollo de aplicaciones basada en el lenguaje de programación Visual Basic.

INTRODUCCIÓN

En la actualidad la industria guatemalteca busca hacer más eficientes sus procesos y aumentar así la productividad de sus máquinas. El punto de partida para mejorar en los aspectos anteriormente mencionados es conocer los puntos de mejora dentro del proceso de producción. Para esto es necesario realizar un análisis, el cual se elabora basándose en la información de producción para lo cual es muy útil obtener datos veraces de la producción individual de cada máquina de forma certera y ordenada (por ejemplo, velocidad de producción, tiempos muertos, etcétera), de manera que estos puedan ser analizados rápida y fácilmente. Esta información debe ser accesible en todo momento y debe ser guardada en un formato tal que esta pueda ser analizada de manera fácil y rápida.

El sistema de monitoreo de producción por medio del almacenamiento de datos en un dispositivo de almacenamiento USB pretende ser una solución de costo accesible y práctica, que permita a la pequeña y mediana industria guatemalteca, tener un sistema de monitoreo de producción que puede ser implementado en un tiempo muy corto.

Tener información sobre la velocidad de producción, tiempo de operación y producción total de una máquina permite a la empresa, desde conocer los tiempos muertos en la producción de la máquina y tomar medidas correctivas, hasta tener un control de inventario del producto.

1. USB (BUS SERIAL UNVERSAL)

1.1. Características del bus serial universal

El bus serial universal, más conocido por sus siglas en inglés USB (Universal Serial Bus), ha sido una de las innovaciones más importantes dentro del mundo de las computadoras. Su facilidad de uso, velocidad, confiabilidad, versatilidad, bajo costo y su bajo consumo de potencia lo convierten en una de las opciones más utilizadas para diseñar interfaces entre dos dispositivos electrónicos. Debido a las características antes mencionadas esta interface permite la comunicación entre diversos dispositivos sin dar lugar a las distintas limitaciones y frustraciones que se tenían con las interfaces anteriores, como el puerto paralelo o el puerto serial.

1.1.1. Beneficios para el usuario

Desde el punto de vista del usuario el mayor beneficio que se tiene al utilizar el USB radica en su facilidad de uso.

Se necesita únicamente una interface para comunicarse con muchos tipos de dispositivos, esto debido a la versatilidad del USB, que en lugar de contar con muchos tipos diferentes de conectores y hardware para cada tipo de periférico, cuenta con una sola interface que puede ser usada por muchos dispositivos.

Una de las características más interesantes que distingue al USB, es su capacidad de configurarse automáticamente. Cuando se inserta un dispositivo USB en el puerto, el sistema automáticamente intenta buscar el driver adecuado para realizar la comunicación con el dispositivo periférico.

El hecho de ser dispositivos que en su mayoría son capaces de operar únicamente con el suministro eléctrico del puerto, los convierte en dispositivos de fácil conexión a los cuales no les es necesario una fuente externa de energía eléctrica.

La interface de comunicación USB maneja tres velocidades de transmisión, baja (*slow*) 1,5 Mbytes/segundo, plena (*full*) 12 Mbytes/segundo y alta (*high*) 480 Mbytes/segundo. Esto permite realizar la transferencia de datos sumamente rápido, lo cual es muy útil cuando se requiere compartir grandes cantidades de información.

Una de las características más destacadas de la interface USB es la robustez y confiabilidad del protocolo. Las especificaciones técnicas de los dispositivos USB y sus cables con blindaje hacen de esta interface un sistema poco ruidoso, en donde se elimina la mayoría de interferencias que podrían provocar errores en la transmisión o recepción de datos.

Este protocolo permite verificar si existen errores en los datos recibidos, además de notificar al emisor de los datos para que realice una retransmisión.

El USB es una interface que a pesar de ser más compleja que sus antecesores, es sumamente económica lo cual es resultado de su estandarización.

1.1.2. Beneficios para el desarrollador

Una de las mayores ventajas desde el punto de vista del desarrollador es la estandarización del hardware y el protocolo en el USB, lo que evita al desarrollador preocupares por la construcción de cables y el desarrollo del software para el chequeo de errores.

El USB fue desarrollado por un equipo de personas que incluyen diseñadores de hardware, programadores especializados en software para dispositivos y programadores especializados en software de computadoras, lo cual permitió que el resultado fuese la creación de un protocolo flexible, el cual es soportado por varios controladores y sistemas operativos.

El USB permite usar cuatro tipos de transferencias y tres velocidades de transmisión diferentes, lo cual lo hace compatible con muchos tipos de periféricos.

Dependiendo del tipo de transferencia el protocolo USB puede garantizar ancho de banda o tiempo de transmisión. Por ejemplo, si se desea transmitir audio, se desea que los datos se transfieran rápidamente y la pérdida de una pequeña cantidad de datos no será significativa, pero si se está transfiriendo una instrucción o archivo, el sistema no necesita que la transmisión sea inmediata, por lo tanto, se puede usar un ancho de banda menor y un algoritmo de corrección de error.

Al ser una interface de transmisión que especifica los requerimientos de los dispositivos y protocolos, el desarrollador puede usar estas especificaciones en lugar de tener que reinventar todo desde el inicio.

1.1.3. Características técnicas

La interface USB no incluye una señal física de reloj dentro de la transmisión, por ello, usa un formato de transmisión serial asíncrono.

La señales de transmisión son señales diferenciales en donde el estado del bit lo define la diferencia de potencial existente entre las señales D+ y D - , los dos estados se conocen como estado J y estado K. Esto reduce la probabilidad de que existan errores en la lectura del bit ya que el ruido introducido en la comunicación afectará ambas señales y la lectura de la diferencia de potencial entre las dos señales no se verá afectada.

La transmisión de datos entre dispositivos se realiza en modo half-duplex, lo que indica que el control del canal de comunicación puede ser tomado solamente por un dispositivo a la vez.

La tabla I muestra las características técnicas principales de la interface USB y se realiza una comparación con las interfaces de computadoras más comunes.

Tabla I. **Comparación entre las interfaces de computadoras más populares**

Interfase	**Formato**	**Numero de dispositivos (máximo)**	**Distancia (màxima, pies)**	**Velocidad (máximo, bits/sec)**	**Aplicación**
USB	serial asíncrono	127	16 (hasta 96 pies con 5 hubs)	1,5M, 12M, 480M	Ratones, teclados, memorias, audio, impresoras, otros peiféricos.
Ethernet	serial	1024	1600	10G	Redes de comunicación
IEEE-1394b (Fire Wire 800)	serial	64	300	3.2G	Video, almacenamiento masivo
IEEE-488 (GPIB)	paralelo	15	60	8M	Instrumentación
IrDA	serial asíncrono infrarojo	2	6	16M	Impresoras, computadoras de mano para configuración.
I^2C	serial síncrono	40	18	3.4M	Comunicación de microcontroladores
Microwire	serial síncrono	8	10	2M	Comunicación de microcontroladores
Puerto de impresora paralelo	paralelo	2	10-30	8M	Impresoras, escaners
RS-232 (EIA/TIA-232)	serial asincrono	2	50-100	20k	Modem, ratones, intrumentacón
RS-485 (EIA/TIA-485)	serial asincrono	32 cargas (hasta 256 dispositivos con algun hardware)	4000	10M	Adquisiciòn de datos y sistemas de control
SPI	serial síncrono	8	10	2.1M	Comunicación de microcontroladores

Fuente: AXELSON, Jan. USB Complete. p. 3.

1.1.4. Componentes del bus

Los componentes físicos del USB son los circuitos, conectores y cables que se encuentran entre el anfitrión (*host*) y uno o más dispositivos. Este permite la comunicación entre el dispositivo USB y el anfitrión.

1.1.4.1. Anfitrión

El anfitrión puede ser una PC o cualquier otra computadora o dispositivo que contenga un controlador USB anfitrión y un root hub. El controlador anfitrión le da formato a los datos para que sean transmitidos a través del bus y traduce los datos recibidos a un formato que pueda ser entendido por los componentes del Sistema Operativo. Este también realiza otras funciones relacionadas con la administración de la comunicación en el bus. El root hub tiene una o más conexiones para la conexión de dispositivos periféricos. El root hub en conjunto con el controlador anfitrión se encargan de detectar los dispositivos periféricos que se conectan o se remueven del bus, además de transmitir las peticiones del controlador anfitrión hacia los dispositivos periféricos y los datos de los dispositivos periféricos hacia el controlador anfitrión.

1.1.4.2. Dispositivos

Los dispositivos son los periféricos o los hubs adicionales que son conectados al bus. Los dispositivos según las especificaciones USB son divididos en funciones, hubs y dispositivos compuestos. Estos últimos cuentan con un hub y una o más funciones; y el anfitrión los maneja como si fuesen dispositivos independientes dentro del bus.

Una función es un dispositivo que provee una capacidad al anfitrión, por ejemplo, un teclado, unas bocinas, una memoria USB u otros. Es importante mencionar que un dispositivo puede tener una o más funciones, por ejemplo, un teléfono celular.

Un hub es un dispositivo que cuenta con una conexión de entrada para comunicarse con el anfitrión y una o más conexiones de salida para conectar dispositivos periféricos o incluso otro dispositivo hub. Las conexiones de salida del hub representan nuevos puertos USB.

1.1.4.3. Puerto USB

En general un puerto en una computadora es una localidad direccionable que está disponible para conectar dispositivos adicionales.

El puerto USB difiere de muchos otros puertos debido a que todos los puertos en el bus comparten una sola ruta de acceso hacia el anfitrión y no pueden ser directamente direccionables. Cada conexión en el bus representa un puerto USB, pero a diferencia de otros puertos como el RS-232, todos los dispositivos comparten el ancho de banda del bus y únicamente un dispositivo a la vez o el anfitrión puede transmitir información.

1.2. El anfitrión USB (Host)

Cada anfitrión y sus dispositivos tienen diferentes tareas asignadas. La mayor parte del manejo de la comunicación es realizada por el anfitrión, pero un dispositivo debe tener la capacidad para responder al mensaje y a otros eventos del bus enviados por el anfitrión y el hub al cual están conectados.

1.2.1. Tareas del anfitrión USB

Como se mencionó anteriormente el hardware del anfitrión está formado por un controlador anfitrión USB y un root hub con uno o más puertos los cuales se encargan de realizar las tareas del anfitrión.

El anfitrión está a cargo del bus. El anfitrión debe saber que dispositivos están conectados al bus y las características de cada dispositivo. El anfitrión también debe hacer el mejor intento para asegurar que todos los dispositivos en el bus puedan enviar y recibir datos cuando lo necesiten. El bus puede tener muchos dispositivos, cada uno con diferentes requerimientos y todos esperando para transmitir datos al mismo tiempo.

Por fortuna, el hardware del controlador anfitrión y sus distintos firmwares hacen mucho del trabajo en la administración del bus. Las aplicaciones no tienen que preocuparse acerca de los detalles específicos de la comunicación entre el anfitrión y los dispositivos a través del USB. Todo lo que deben hacer las aplicaciones es enviar y recibir datos usando funciones estándar del Sistema Operativo las cuales son accesibles desde casi todos los lenguajes de programación.

1.2.1.1. Detección de dispositivos

Al energizarse el sistema, los hubs notifican al anfitrión acerca de todos los dispositivos conectados al bus. En un proceso llamado enumeración, el anfitrión asigna una dirección y solicita información adicional a cada dispositivo. Después de energizarse el sistema, cuando se remueva o conecte un dispositivos el anfitrión enumerará al nuevo dispositivo o removerá al dispositivo desconectado de su lista de dispositivos disponibles.

1.2.1.2. Manejo del flujo de datos

Múltiples dispositivos periféricos podrían querer transferir datos al mismo tiempo. El controlador anfitrión divide el tiempo disponible en segmentos llamados frames o microframes y le asigna a cada transmisión una porción de estos.

A las transferencias que deben ocurrir a una tasa (Bytes/segundo) específica se les garantiza el tiempo que necesitan en cada frame. Durante el proceso de enumeración, el driver del dispositivo solicita el ancho de banda que él necesitará para transferencias en las cuales se deba garantizar el tiempo de transmisión. Si el ancho de banda solicitado no está disponible, el anfitrión no permite que se inicie la comunicación. Entonces el driver debe solicitar un ancho de banda menor o esperar hasta que el ancho de banda solicitado esté disponible. Las transferencias que no necesitan garantizar su tiempo de transmisión usan las porciones restantes del frame y deben esperar si el bus está ocupado.

1.2.1.3. Chequeo de error

Cuando se transfieren datos, el anfitrión agrega bits para el chequeo de error. Cuando los dispositivos reciben una transferencia realizan cálculos en los datos y los comparan contra los bits de chequeo de error. Si el resultado no concuerda, el dispositivo deja de indicar que recibió los datos. Con esto el anfitrión reconoce que los datos deben ser reenviados.

El USB también soporta tipos de transferencia que no permiten la retransmisión de datos, debido a la necesidad de mantener constante la tasa de transmisión. De manera similar el anfitrión chequea los datos recibidos desde los dispositivos.

El anfitrión puede recibir instrucciones que no sean soportadas por el dispositivo. El anfitrión puede entonces indicar al driver del problema y, el driver indicarle a la aplicación, de manera que se tome una acción apropiada.

1.2.1.4. Proveer de alimentación eléctrica

El cable USB tiene una línea de +5 voltios y una de referencia. Algunos dispositivos toman su energía eléctrica totalmente del puerto y otros cuentan con una alimentación eléctrica externa.

1.2.1.5. Intercambiar datos con periféricos

Todas las tareas antes descritas hacen posible que el anfitrión realice su trabajo principal, el cual es intercambiar datos con los dispositivos periféricos.

1.2.2. Proceso de enumeración

Antes de establecer una comunicación con los dispositivos conectados al bus, el anfitrión necesita reconocer al dispositivo. El proceso de enumeración incluye la asignación de una dirección al dispositivo, la lectura de los descriptores, la asignación y carga del driver adecuado para el dispositivo y la selección de la configuración que especifica los requerimientos de potencia, los endpoints y otros detalles.

Una de las tareas del hub es la detección de todos los dispositivos que se desconectan o conectan a sus puertos. Cada hub cuenta con un endpoint tipo entrada (*IN*) para reportar estos eventos al anfitrión. Cuando se inicializa el sistema el anfitrión pregunta al root hub por cualquier dispositivo conectado a él, incluyendo otros hubs y dispositivos conectados a estos otros. Luego de la inicialización el anfitrión continúa preguntando periódicamente al root hub para conocer si se ha desconectado o conectado algún dispositivo al sistema.

Cuando el anfitrión está reconociendo un nuevo dispositivo el hub establece la ruta de comunicación entre estos. Entonces es cuando el anfitrión procede a enumerar al dispositivo enviando transferencias de control al endpoint 0 de los dispositivos. Para una enumeración exitosa el dispositivo debe responder a cada solicitud del anfitrión retornando la información solicitada y realizando otras acciones si son solicitadas.

En una aplicación típica, el firmware contiene la información que el anfitrión solicita en el proceso de enumeración y una combinación de hardware y firmware decodifica y responde las solicitudes de información. Algunos controladores pueden manejar completamente la enumeración por medio de hardware, sin ayuda de un firmware.

1.2.2.1. Pasos del proceso de enumeración

Las especificaciones USB definen seis estados en los cuales puede estar un dispositivo. Durante la enumeración, un dispositivo pasa a través de cuatro de estos seis estados: energizado, predeterminado, direccionado y configurado.

- El usuario conecta el dispositivo al puerto: el sistema se enciende con un dispositivo que se encuentra ya conectado. El dispositivo puede estar conectado directamente al root hub en el anfitrión o puede estarlo al puerto de un hub conectado al root hub. El hub brinda energía al dispositivo, entonces este pasa al estado energizado.

- El hub detecta el dispositivo: el hub monitorea los potenciales existentes en las líneas de comunicación de cada uno de sus puertos. Este cuenta con una resistencia a referencia (0 voltios) en cada una de las líneas de comunicación D+ y D-. Un dispositivo cuenta con una resistencia a 5 voltios en una de sus terminales de comunicación, en D+ para dispositivos de velocidad plena o en D- para dispositivos de baja velocidad. Un dispositivo de alta velocidad se conecta en primera instancia como un dispositivo de velocidad plena. Cuando se conecta un dispositivo al puerto la resistencia a 5 voltios permite que en una de las líneas de comunicación D+ o D- según el tipo de dispositivo, aparezca un estado alto o 5 voltios, permitiéndole al hub detectar que un dispositivo ha sido conectado.

- El anfitrión reconoce al nuevo dispositivo: cada hub indica por medio de una interrupción al anfitrión si ha ocurrido algún evento en el hub o en algún puerto, si fue así, indica en qué puerto ocurrió. Luego de esto el anfitrión envía solicitudes de información al hub para conocer más de los eventos. La información retornada le indica al anfitrión si un dispositivo ha sido conectado al hub.

- El hub detecta si el dispositivo es de baja o plena velocidad: justo antes que el hub reinicie el dispositivo, determina si este es de baja o plena velocidad examinando los potenciales en las dos líneas de comunicación. El hub envía esta información al anfitrión hasta que este realice una nueva solicitud de información.

- El hub reinicia al dispositivo: cuando el anfitrión ha reconocido al nuevo dispositivo envía una solicitud pidiendo al hub que reinicia el puerto. La señal de reinicio es un caso especial en la cual las dos señales D+ y D- se encuentran en estado lógico bajo. El hub envía la señal de reinicio únicamente al nuevo dispositivo.

- El anfitrión reconoce si un dispositivo inicializado en plena velocidad soporta trabajar en modo de alta velocidad: para detectar si un dispositivo soporta trabajar en modo de alta velocidad se usan dos estados especiales de la señal. En el estado de indicación J, solamente la línea D+ es manejada y en el estado de indicación K, solamente se maneja la línea D –. Durante el proceso de reinicio, los dispositivos que soportan el modo de alta velocidad envían una indicación K. El hub detecta la indicación y responde con una serie de indicaciones K y J. Al detectar el patrón enviado por el hub el dispositivo cambia sus parámetros para operar a alta velocidad.

- El hub establece una ruta de acceso entre el dispositivo y el anfitrión: el anfitrión verifica que el dispositivo se halla reiniciado correctamente enviando una solicitud de estatus. La información recibida indica si el dispositivo ya fue reiniciado o se encuentra en el proceso. Al finalizar el proceso el dispositivo se encuentra en estado predeterminado y listo para responder a las transferencias de control enviadas a su endpoint 0.

- El anfitrión envía una solicitud de descriptor para obtener el tamaño de paquete máximo soportado por el endpoint 0: el anfitrión envía una solicitud de descriptor a la dirección 0, endpoint 0, la cual es la dirección de un dispositivo recién conectado. Este dato se encuentra entre los 8 bytes del descriptor del dispositivo.

- El anfitrión asigna una dirección al dispositivo: este asigna una única dirección al dispositivo enviando una solicitud para ajustar la dirección. El dispositivo termina la etapa de status usando la dirección predeterminada y luego implementa la nueva dirección. El dispositivo se encuentra ahora en estado direccionado.

- El anfitrión reconoce las características del dispositivo: el anfitrión envía una solicitud de descriptor a la nueva dirección asignada al dispositivo para leer su descriptor. En este momento es cuando el anfitrión obtiene el descriptor completo del dispositivo.

- El anfitrión asigna y carga el driver adecuado para el dispositivo: luego de conocer las características del dispositivo a través de su descriptor, el anfitrión busca el driver que mejor se ajuste a las características de este para establecer una comunicación entre ellos.

- El driver del anfitrión selecciona una configuración: en este paso el driver del dispositivo realiza una solicitud de configuración enviando una solicitud para ajustar la configuración junto con el número deseado de configuración. El dispositivo lee la solicitud y habilita la configuración solicitada. El dispositivo se encuentra ahora en estado configurado y la interface del dispositivo se encuentra totalmente habilitada.

1.2.3. Descriptores

Los descriptores del USB son estructuras de datos o bloques de información con un formato, que permiten al anfitrión aprender acerca de un dispositivo.

Todos los dispositivos USB deben responder a las solicitudes del anfitrión para obtener el descriptor de estos. El dispositivo debe almacenar información en el descriptor y responder a la solicitud.

1.2.3.1. Tipos de descriptores

El anfitrión utiliza transferencias de control para solicitar los descriptores del dispositivo. Conforme progresa el proceso de enumeración, los descriptores solicitados corresponden mayormente a elementos más puntuales del dispositivo. Primero corresponden a todo el dispositivo, luego a cada configuración, cada interface de la configuración y finalmente a cada endpoint de las interfaces.

A excepción de los dispositivos compuestos, cada dispositivo cuenta únicamente con un descriptor de dispositivo, que contiene información acerca del dispositivo como un todo y especifica el número de configuraciones que soporta. Cada dispositivo también cuenta con uno o más descriptores de configuración que contienen información acerca de la administración de la potencia y el número de interfaces soportadas por la configuración. Cada descriptor de interface especifica cero o más descriptores de endpoint*s* que contienen la información necesaria para comunicarse con él. Cada descriptor de endpoint contiene información acerca de cómo el endpoint transfiere la información.

Los dispositivos que pueden operar en velocidad plena y alta soportan dos tipos de descriptores adicionales: Dispositivo_Clasificado y Otras_configuraciones_de_velocidad.

Además, de los descriptores estándar, un dispositivo puede contener descriptores específicos de la clase o el proveedor del dispositivo. Estos descriptores ofrecen una manera estructurada para proveer información más detallada acerca del dispositivo.

Tabla II. **Tipos de descriptores estándar**

bTipoDeDescriptor	Tipo de Descriptor	Requerido
01h	Dispositivo	Sí,
02h	Configuración	Sí,
03h	*String*	No,
04h	Interfase	Sí,
05h	*EndPoint*	No, Si el dispositivo usa únicamente el Endpoint 0
06h	Dispositivo_Clasificado	Sí, Para dispositivos que soportan velocidad plena y alta. No soportado por los demas
07h	Otras_configuraciones_de_velocidad	Sí, Para dispositivos que soportan velocidad plena y alta. No soportado por los demas
08h	Interface_de_potencia	No, Soporta el manejo de la potencia a nivel interfase

Fuente: AXELSON, Jan. USB Complete. p. 95.

Cada descriptor contiene un valor hexadecimal que identifica su tipo. Este valor se muestra en la primera columna de la tabla II.

Cada descriptor consiste en una serie de campos. La mayoría de los nombres de los campos utilizan prefijos para indicar el formato o el contenido de los datos del campo, b = byte, w = palabra (Word), bm = mapa de bits (Bitmap), bcd = decimal codificado binariamente, i = índice, id = identificador. Por ejemplo, en la tabla III, se observan los campos del descriptor de dispositivo.

En este trabajo no se incluirá una explicación detallada de los tipos de descriptores y sus campos ya que no es el objetivo de este. A continuación se encuentra una descripción de los tipos de descriptores principales.

Tabla III. **Campos del descriptor de dispositivo**

No.	Campo	Tamaño (Bytes)	Descripción
0	bLenght	1	Tamaño del descriptor
1	bDescriptorType	1	Número de tipo de descriptor
2	bcdUSB	2	Número de la versión de la especificación USB
4	bDeviceClass	1	Código de clase
5	bDeviceSubclass	1	Código de subclase
6	bDeviceProtocol	1	Código de protocolo
7	bMaxPacketSize0	1	Máximo tamaño de paquete para el enpoint 0
8	idVendor	2	Identificación del proveedor
10	idProduct	2	Identificación del producto
12	bcdDevice	2	Número de la versión del dispositivo
14	iManufacturer	1	Índice hacia el descriptor string del manufacturador
15	iProduct	1	Índice hacia el descriptor string del producto
16	iSerialNumber	1	Índice hacia el descriptor string que contiene el número de serie del dispositivo
17	bNumConfiguration	1	Número de posible configuraciones

Fuente: AXELSON, Jan. USB Complete. p. 97.

- Descriptor de dispositivo: este contiene la información básica acerca del dispositivo. Este descriptor es el primero en ser leído por el anfitrión al conectar un dispositivo. El descriptor contiene información que el anfitrión necesita saber para recibir información adicional del dispositivo.

  Este descriptor cuenta con 14 campos. La tabla III lista los campos en el orden en que ocurren en el descriptor y provee una pequeña descripción de la información que conllevan. En este se incluye información propia, del dispositivo, su configuración y cualquier clase de dispositivo a la que pertenece.

- Descriptor de dispositivo_clasificado: los dispositivos que soportan trabajar a plena y alta velocidad deben tener un descriptor de dispositivo_clasificado. Cuando un dispositivo cambia de velocidad, algunos campos en el descriptor de dispositivo deben cambiar. Este descriptor contiene los valores de estos campos correspondientes a la velocidad que no está siendo usada en ese momento. En otras palabras, el contenido de los campos del descriptor de dispositivo y el descriptor_clasificado se intercambian dependiendo de la velocidad que está siendo usada. En la tabla IV se muestran los campos de este descriptor.

Tabla IV. **Campos del descriptor de dispositivo_clasificado**

No.	Campo	Tamaño (Bytes)	Descripción
0	bLenght	1	Tamaño del descriptor
1	bDescriptorType	1	Número de tipo de descriptor
2	bcdUSB	2	Número de la versión de la especificación USB
4	bDeviceClass	1	Código de clase
5	bDeviceSubclass	1	Código de subclase
6	bDeviceProtocol	1	Código de protocolo
7	bMaxPacketSize0	1	Máximo tamaño de paquete para el enpoint 0
8	bNumConfiguration	1	Número de posible configuraciones
9	Reservado	1	Para uso futuro

Fuente: AXELSON, Jan. USB Complete. p. 100.

- Descriptor de configuración: después de obtener el descriptor de dispositivo, el anfitrión puede obtener los descriptores de configuración, interface y endpoints.

  Cada dispositivo tiene al menos una configuración que especifica los detalles y habilidades del dispositivo. Un dispositivo de múltiples usos puede tener múltiples configuraciones. Únicamente una configuración puede estar activa a la vez. Cada configuración requiere un descriptor. Este descriptor contiene información acerca del manejo de la potencia y el número de interface soportadas. Cada descriptor de configuración cuenta con descriptores subordinados, incluyendo uno o más descriptores de interface y descriptores de endpoints opcionales.

Tabla V. **Campos del descriptor de configuración**

No.	Campo	Tamaño (Bytes)	Descripción
0	bLenght	1	Tamaño del descriptor
1	bDescriptorType	1	Número de tipo de descriptor
2	wTotalLength	2	Número de bytes del descriptor de configuración y sus subordinados
4	bNumInterfaces	1	Número de interfases en la configuración
5	bConfigurationValue	1	Identificador para las solicitudes de obtener y ajustar configuración
6	iConfiguration	1	Índice del descriptor string para la configuración
7	bmAttributes	1	Ajustes de ahorro de potencia y opción wakeup
8	bMaxpower	1	Potencia màxima requerida, expresada en mA

Fuente: AXELSON, Jan. USB Complete. p. 102.

- Descriptor de Otras_config_de_velocidad: este descriptor al igual que el descriptor de dispostivo_clasificado es usado únicamente en dispositivos que soportan la operación en plena y alta velocidad. La estructura de este descriptor es idéntica a la del descriptor de configuración con la diferencia que en este, el valor de los campos corresponde a la operación en la velocidad actualmente inactiva. La estructura de este descriptor y el descriptor de configuración se muestran en la tabla V.

- Descriptor de interface: este descriptor provee información acerca de la función que desempeña el dispositivo. El descriptor contiene información de la clase, subclase y protocolo, además del número de endpoints que la interface usa.

  Una configuración puede tener múltiples interfaces que pueden estar activas al mismo tiempo.

El anfitrión obtiene el descriptor de interface solicitando el descriptor de configuración para la configuración a la cual pertenece la interface. La tabla VI muestra la estructura del descriptor.

Tabla VI. **Campos del descriptor de interface**

No.	Campo	Tamaño (Bytes)	Descripción
0	bLenght	1	Tamaño del descriptor
1	bDescriptorType	1	Número de tipo de descriptor
2	bInterfaceNumber	1	Número de identificación de la interfase
3	bAlternateSetting	1	Valor usado para seleccionar otros ajustes
4	bNumEndpoints	1	Número de endpoints soportados, sin contar el endpoint 0
5	bInterfaceClass	1	Código de clase
6	bInterfaceSubclass	1	Código de subclase
7	bInterfaceProtocol	1	Código de protocolo
8	iInterface	1	Índice hacia el descriptor string para la interfase

Fuente: AXELSON, Jan. USB Complete. p. 108.

Uno de los campos de interés es el campo de clase, este campo define a qué clase de interface pertenece el dispositivo. En la tabla VII se encuentra el código hexadecimal para las clases de interface estándar.

Tabla VII. **Códigos de las clases de interface**

Código de Clase (Hex)	Descripción
01	Audio
02	Interfase de comunicación
03	Interfase hombre máquina
05	Física
06	Imagen
07	Impresora
08	Unidad de almacenamiento masivo
09	Hub
0A	Interfase de datos
0B	SmartCard
0D	Contenido de seguridad
0E	Video
DC	Dispositvos de diagnóstico
E0	Controlador Wireless
FE	Aplicaciónes específicas
FF	Especícado por el proveedor

Fuente: AXELSON, Jan. USB Complete. p. 109.

- Descriptor de endpoint: cada endpoint especificado en un descriptor de interface tiene un descriptor de endpoint. El endpoint 0 no tiene un descriptor debido que este debe ser soportado por todos los dispositivos. El descriptor contiene información del tamaño de paquete máximo que el endpoint puede recibir y las especificaciones USB que definen todos los demás aspectos del endpoint. Al igual que con el descriptor de interface el anfitrión obtiene los descriptores, solicitando el descriptor de configuración al cual pertenece el endpoint.

Tabla VIII. **Campos del descriptor de endpoint**

No.	Campo	Tamaño (Bytes)	Descripción
0	bLenght	1	Tamaño del descriptor
1	bDescriptorType	1	Número de tipo de descriptor
2	bEndpointAddress	1	Número de Endpoint y dirección
3	bmAtrributes	1	Tipo de transferencia soportada
4	wMaxPacketSize	2	Tamaño máximo del paquete soportado
5	wbInterval	1	Máximo intervalo para verificiaciòn de interupciones

Fuente: AXELSON, Jan. USB Complete. p. 110.

- Descriptor String: estos contienen textos con descripciones. La especificación USB 2.0 define descriptores que pueden contener índices para varios strings, incluyendo strings que describen al fabricante, producto, número de serie, configuración e interface. La tabla IX muestra los campos del descriptor y su propósito.

Tabla IX. **Campos del descriptor string**

No.	Campo	Tamaño (Bytes)	Descripción
0	bLenght	1	Tamaño del descriptor
1	bDescriptorType	1	Número de tipo de descriptor
2	bSTRING o wLANGID	Varía	Para indicar los lenguajes en los que está disponible el string

Fuente: AXELSON, Jan. USB Complete. p. 110.

1.3. Transferencias

Las transferencias dentro de la especificación del USB se definen como el proceso de llevar a cabo una solicitud de comunicación. En este apartado se expondrán los conocimientos esenciales de las transferencias y sus cuatro tipos.

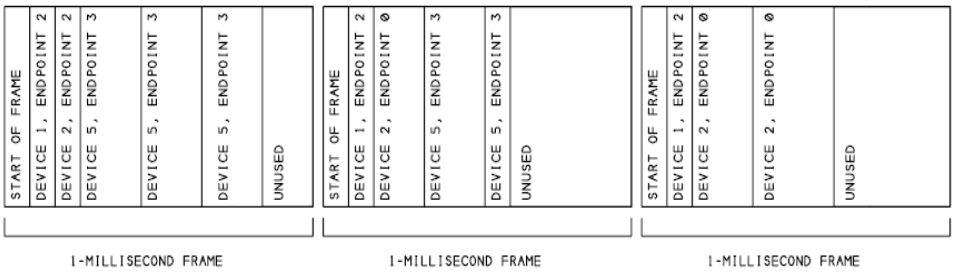
La comunicación dentro del USB se puede dividir en dos categorías: las comunicaciones usadas para realizar el proceso de enumeración y las comunicaciones usadas para la aplicación del dispositivo. Durante la enumeración, el firmware del dispositivo responde a una serie de solicitudes estándar provenientes del anfitrión. El dispositivo debe identificar cada solicitud, retornar la información requerida y tomar otras acciones especificadas por la solicitud.

Después que el anfitrión ha intercambiado la información necesaria para la enumeración con el dispositivo y el driver ha sido asignado y cargado, empiezan las comunicaciones para la aplicación, las cuales se relacionan con el propósito del dispositivo. Cada transferencia de datos usa uno de los cuatro tipos de transferencias establecidos: control, interrupción, *bulk* o isócrona.

Las dos líneas del USB llevan datos desde y a todos los dispositivos conectados al bus. Estas líneas conllevan una señal diferencial simple, cuya dirección cambia según lo necesite el anfitrión.

El anfitrión busca que las transferencias ocurran lo más rápido posible. Para ello, el anfitrión divide el tiempo de transmisión en trozos llamados frames (cuando se utiliza velocidad baja o plena) o microframes (cuando se utiliza velocidad alta). El anfitrión asigna una porción de cada frame o microframe a cada transferencia. Cada frame cuenta con un período de 1 milisegundo. Para trabajar a velocidad alta cada frame se divide en trozos de 125 microsegundos formando microframes como se ilustra en la figura 1.

Figura 1. **Frames USB**



Fuente: AXELSON, Jan. USB Complete. p. 35.

Cada transferencia consiste de una o más transacciones. Dependiendo de la manera en que el anfitrión organice las transacciones y la velocidad de respuesta de los dispositivos, una transacción puede estar contenida en su totalidad en un solo frame o microframe o pueden estar distribuidas a lo largo de múltiples frames o microframes.

Debido a que toda la información comparte una sola ruta, cada transacción debe incluir una dirección de dispositivo que identifique el destino de la transacción. Cada dispositivo tiene una dirección única asignada por el anfitrión y toda la información viaja desde o hacia el anfitrión. Cada transacción comienza cuando el anfitrión envía un bloque de información que incluye la dirección del dispositivo receptor y su posición específica, llamada endpoint.

Es importante mencionar que toda información enviada por el dispositivo es resultado de la recepción de solicitudes o paquetes enviados por el anfitrión.

1.3.1. Endpoints

Toda la información en el bus viaja hacia o desde el endpoint de un dispositivo. Un endpoint se define como una única porción direccionable del dispositivo USB, la cual es fuente o sumidero de la información durante la comunicación entre el anfitrión y el dispositivo. La información almacenada en el endpoint puede ser información recibida o información para enviar, la cual está esperando ser transmitida. El anfitrión también cuenta con buffers que almacenan temporalmente la información recibida y la información a enviar, pero es importante mencionar que el anfitrión no cuenta con endpoints. En lugar de esto el anfitrión siempre es el inicio y final para la comunicación con los endpoints de los dispositivos.

Una dirección de un endpoint consiste en dos campos, un número de endpoint y un sentido. El número puede ser un valor entre 0 a 15. El sentido es definido desde la perspectiva del anfitrión, un endpoint de entrada (IN) provee información para enviar al anfitrión y un endpoint de salida (OUT) almacena los datos enviados por el anfitrión. Un endpoint configurado para transferencias de control debe transmitir información en ambos sentidos, así que los endpoints de control consisten de un par de endpoints IN y OUT que comparten el mismo número de endpoint.

Cada transacción comienza con un paquete que contiene un número de endpoint y un código que indica el sentido del flujo de información y si se está iniciando una transferencia de control. Los códigos son IN, OUT y Setup.

Una transacción de ajuste es similar a la transacción de salida debido a que la información viaja del anfitrión al dispositivo, pero este es un caso especial debido a que una transacción de ajuste inicia una transferencia tipo control.

1.3.2. Conductos

Antes de que pueda ocurrir una transferencia, el anfitrión y el dispositivo deben establecer un conducto para la comunicación. Un conducto dentro de la especificación USB es una asociación entre un endpoint del dispositivo y el software del controlador anfitrión. Este establece un conducto de comunicación durante el proceso de enumeración. Si el dispositivo es removido del bus, el anfitrión remueve los conductos que ya no son de utilidad.

Con el propósito de clasificar los conductos según el tipo de transferencia que conlleva, se definen dos tipos dependiendo si la información viaja en unas o ambas direcciones. Las transferencias de control usan conductos tipo mensaje, en estos cada transferencia comienza con una transacción de ajuste que contiene una solicitud. Para completar la transferencia, el anfitrión y el dispositivo deben intercambiar datos e información de estado o el dispositivo debería únicamente enviar la información de estado. Cada transferencia de control tiene al menos una transacción que envía información en ambos sentidos. Todas las otras transferencias utilizan conductos tipo cadena. En este tipo de conducto los datos no contienen una estructura definida dentro de la especificación USB.

El anfitrión o el dispositivo simplemente aceptan cualquier tipo de datos que les lleguen. En este tipo de conducto la información viaja en un solo sentido durante cada transacción.

Es importante mencionar que todos los dispositivos USB cuentan con un conducto de control predeterminado el cual usa el endpoint 0.

1.3.3. Tipos de transferencias

El USB está diseñado para trabajar con muchos tipos de periféricos los cuales tienen diferentes requerimientos de velocidad de transmisión, tiempo de respuesta y corrección de errores. Cada uno de los cuatro tipos de transferencias de datos cubren los diferentes requerimientos de los dispositivos. En la tabla X se encuentra un resumen de las características y uso típico de cada tipo de transferencia.

Tabla X. **Tipos de transferencias**

Tipo de Transferencia	Control	Bulk	Interrupción	Isochronous
Uso típico	Identificación y configuración	Impresoeas, scanner	Ratones, teclados	Transmisión de audio, video
Requerida?	sí	no	no	no
Permitidas a velocidad baja	sí	no	si	no
Bytes de datos/milisegundo por Transferencia, máximo posible por conducto.(Velocidad alta)*	15,872 (treinta y un transacciones de 64 bytes/mictoframe	53,248 (trece transacciones de 512 bytes/microframe	24,576 (tres transacciones de 1024 bytes/microframe	24,576 (tres transacciones de 1024 bytes/microframe
Bytes de datos/milisegundo por Transferencia, máximo posible por conducto.(Velocidad plena)*	832 (trece transacciones de 64 bytes/frame	1,216 (diecinueve transacciones de 64 bytes/frame	64 (una transaccion de 64 bytes/frame	1,023 (una transaccion de 1,023 bytes/frame
Bytes de datos/milisegundo por Transferencia, máximo posible por conducto.(Velocidad baja))*	24 (tres transacciones de 8 bytes/frame	no permitida	66 (una transaccion de 64 bytes/frame	no permitida
Dirección del flujo de datos	Entrada y Salida	Entrada ó Salida	Entrada ó Salida	Entrada ó Salida
Ancho de banda reservado para la transferencia. (percent)	10 a baja/plena velocidad, 20 a alta velocidad (mínimo)	none	90 a baja/ plena velocidad, 80 a alta velocida	
Corrección de error?	Sí	Sí	Sí	No
Conductos tipo ?	mensaje	stream	stream	stream
Velocidad de transmisión garantizada?	no	no	no	sí
*** Se asume que la transferencia usa el tamaño máximo de paquete**				

Fuente: AXELSON, Jan. USB Complete. p. 41.

1.3.3.1. Transferencia de control

Es el único tipo de transferencia para la cual las especificaciones USB definen su función. Este tipo de transferencia le permite al anfitrión obtener información acerca de un dispositivo, asignarle una dirección, seleccionar la configuración y realizar algunos otros ajustes. Las transferencias de control podrían también enviar solicitudes definidas por el fabricante, las cuales envían o reciben información para cualquier propósito.

Todos los dispositivos deben soportar transferencias de control a través de un conducto predeterminado en el endpoint 0. Cada transferencia de control debe tener una etapa de ajuste y una etapa de estado. La etapa de datos es opcional.

En la etapa de ajuste, el anfitrión comienza una transacción de ajuste enviando información acerca de la solicitud. El paquete token contiene un PID (identificador de paquete) que identifica el tipo de transferencia, en este caso, de control. El paquete de datos contiene información acerca de la solicitud, incluyendo el número de solicitud, si la transferencia cuenta o no con una etapa de datos y si es así, en qué dirección viajará la información. La especificación USB 2.0 define 11 solicitudes estándar.

Cuando está presente la etapa de datos en una transferencia de control, esta consiste en una o más transacciones, las cuales pueden ser de salida o entrada. Dependiendo de la solicitud, el anfitrión o el periférico podrían ser la fuente de estas transacciones, pero todos los paquetes de datos en esta etapa van en la misma dirección.

La etapa de estado consiste de una transacción de entrada o de salida, también llamada transacción de estado. En la etapa de estado el dispositivo reporta el éxito o fracaso de las etapas anteriores.

El controlador anfitrión reserva una porción del ancho de banda del bus para las transferencias de control: 10% cuando se trabaja a baja o plena velocidad y 20% cuando se trabaja a alta velocidad. Si la trasferencia de control no necesita mucho ancho de banda, lo sobrante puede ser usado por otro tipo de transferencia.

1.3.3.2. Transferencia masiva de datos (BulK)

Este tipo de transferencia es usada para transferencias de grandes cantidades de información en las cuales la velocidad de transmisión no es crítica, como por ejemplo, la transferencia de un archivo a la impresora, la transferencia de datos de un scanner o el accesar a un archivo de un dispositivo USB. Para estas aplicaciones tener una velocidad de transmisión constante es bueno, pero no es crítico, la información puede esperar si es necesario. Si el bus está ocupado, las transferencias de este tipo son retardadas, pero si el bus está libre la transferencia es sumamente rápida. Lo anterior permite la transmisión de grandes cantidades de datos sin saturar el bus.

Una transferencia de este tipo consiste de una o más transacciones de entrada o salida. Esta es una transferencia de un sólo sentido, si se quisiera transmitir datos en ambos sentidos se necesitarían un conducto y una transferencia separados, uno para cada sentido. Si la cantidad de datos no cabe dentro de un sólo paquete, el anfitrión completa la transferencia usando varias transacciones.

Este tipo de transferencia permite la detección de errores de transmisión. Si el dispositivo no retorna la confirmación de paquete recibido, el anfitrión realiza hasta dos intentos por enviarlos, si no recibe respuesta lo toma como un error y pide la retransmisión de los datos.

Este tipo de transferencia solamente es soportado por dispositivos que operen a velocidad plena o alta.

1.3.3.3. Transferencia de interrupción

Este tipo de transferencia es usada por dispositivos que deben recibir la atención del anfitrión periódicamente. Para los dispositivos que solamente operan a baja velocidad este es el único medio de transmisión de datos. Sus aplicaciones típicas incluyen teclados, consolas de juego, etcétera.

Las transferencias de interrupción a pesar de ser similares a una interrupción común, son atendidas hasta que el anfitrión realiza una lectura del bus, de cualquier forma estas garantizan que el anfitrión solicitará o enviará los datos lo más rápido que le sea posible.

Esta transferencia es soportada por cualquiera de las tres velocidades de transmisión usadas en el USB.

La estructura de una transferencia de interrupción es igual a la estructura de una tipo bulk. Este tipo de transferencia no garantiza una velocidad de transmisión, si no únicamente garantiza que el tiempo o período de transferencia entre cada transacción no será mayor al máximo establecido.

1.3.3.4. Transferencia isócrona

La transferencia isócrona, es un tipo de transferencia de datos en tiempo real la cual es útil cuando la información debe ser transmitida a una velocidad constante y pueden ser tolerados errores ocasionales. A velocidad alta, una transferencia isócrona puede transferir más datos por frame que una transferencia tipo interrupción, pero con esta no hay posibilidad de una retransmisión de los datos en caso de haber recibido algunos datos erróneos.

Algunas de las aplicaciones más comunes de este tipo de transferencia son la reproducción de audio o video en tiempo real. No siempre que se quiera reproducir audio o video se tiene que usar una transferencia isócrona, por ejemplo, se podría usar una transferencia de datos masivos (bulk) para enviar el archivo y luego que ya se encuentre en el dispositivo este pueda ser reproducido a la velocidad apropiada.

Una transferencia isócrona es una manera de asegurar que un gran bloque de datos será transferido rápidamente, aún cuando el tráfico de datos en el bus es grande.

El término isócrono significa que los datos transmitidos tendrán una velocidad de transmisión fija, con un número definido de bytes transferidos en cada frame o microframe, cosa que ninguno de los otros tipos de transferencia garantiza.

Este tipo de transferencia es soportada únicamente a velocidad plena y alta.

1.4. Transacciones

Cada transferencia consiste de una o más transacciones y cada transacción consiste de uno, dos o tres paquetes.

Los tres tipos de transacciones son definidos por su propósito y la dirección del flujo de datos. Una transacción está definida por las especificaciones USB como el reparto de un servicio hacia un endpoint. Un servicio puede ser el anfitrión enviando información al dispositivo o este mismo solicitando y recibiendo información del dispositivo.

Cada transacción incluye una identificación, chequeo de errores, bits de estado e información de control, así como, cualquier información que se desee intercambiar. Una transferencia completa puede ser distribuida en múltiples frames o microframes, pero una transacción debe ser completada sin interrupciones.

1.4.1. Fases de una transacción

Cada transacción cuenta con tres fases o partes que pueden ocurrir en secuencia: token, data y handshake. Cada fase consiste de uno o dos paquetes. Cada paquete es un bloque de información con un formato definido. Todos los paquetes comienzan con un identificador (PID) que contiene información acerca del paquete.

Dependiendo de la transacción, el PID puede ser seguido de una dirección de endpoint, datos, información de estado o un número de trama, junto con bits para el chequeo de errores de transmisión.

En la fase token de una transacción, el anfitrión inicia una comunicación, la cual realiza enviando un paquete token. El PID indica el tipo de transacción, la cual puede ser de ajuste, entrada, salida o inicio de trama.

En la fase data, el anfitrión o el dispositivo puede transmitir cualquier tipo de información enviando paquetes de datos. El PID incluye lo que se conoce en inglés como data-toggle, que es un dato que se usa para secuenciar la transmisión de los datos a manera de evitar duplicar o perder información cuando la información es enviada en varios paquetes.

En la fase handshake, el anfitrión y el dispositivo envían información acerca de su estado, en lo que se conoce como paquetes de handshake. El PID contiene un código de estado (ACK, NAK, STALL o NYET). Esta fase también es conocida como fase de estado y los paquetes son llamados paquetes de estado.

La fase token tiene un uso adicional. El paquete token puede contener un marcador de inicio de trama (SOF), el cual es una referencia de tiempo que es transmitida cada 1 milisegundo a velocidad plena y cada 125 microsegundos a velocidad alta.

1.4.2. Handshaking

Al igual que otras interfaces, el USB usa información de estado y control para ayudar a manejar el flujo de información en el bus, a manera de realizar la transmisión de información lo más rápido posible.

El USB realiza el handshaking por medio de software a diferencia de otras interfaces como la serial, la cual lo realiza por medio hardware. Un código indica el éxito o fallo de todas las transacciones excepto cuando se realiza una transferencia isócrona. A parte de la transferencia de control, la etapa de estado le permite dar a conocer el éxito o fallo de la transferencia completa.

Los códigos de estado son ACK, NAK, STALL, NYET y ERR. Estos códigos son enviados por el dispositivo o el anfitrión a manera de indicar el estado en el que se encuentran. La ausencia de un código de estado en la comunicación indica un problema más serio. La tabla XI muestra los códigos de estado y su significado.

Tabla XI. **Códigos de estado**

Código	Indicación
ACK	Indica que el anfitrión o el dispositivo han recivido la información sin errores.
NAK	Indica que el dispositivo está ocupado o no tiene información para retornar.
STALL	Esta puede tener tres diferentes significados, solicitud de control no soportada, solicitud de control reprobada, o endpoint dañado.
NYET	Usado únicamente en dispositivos que operen a velocidad alta. Es usada por el anfitrión para conocer si el dispositivo está listo para recibir la información, si no lo está, este envía el codigo NYET.
ERR	Es usado únicamente por dispositivos que operan a velocidad alta. Este indica que el dispositivo no envió el código de estado esperado.

Fuente: AXELSON, Jan. USB Complete. p. 50.

2. DISPOSITIVO ANFITRIÓN USB VNC1L DE FTDI

El dispositivo VNC1L, es un controlador anfitrión integrado fabricado por FTDI chips, el cual maneja completamente el protocolo USB. Este dispositivo cuenta con 2 puertos USB que pueden operar a velocidad baja y plena. Soporta la suspensión y reactivación para el ahorro de energía, además de ser alimentado eléctricamente a través del bus o de una fuente independiente. La figura 2 muestra el dispositivo VNC1L.

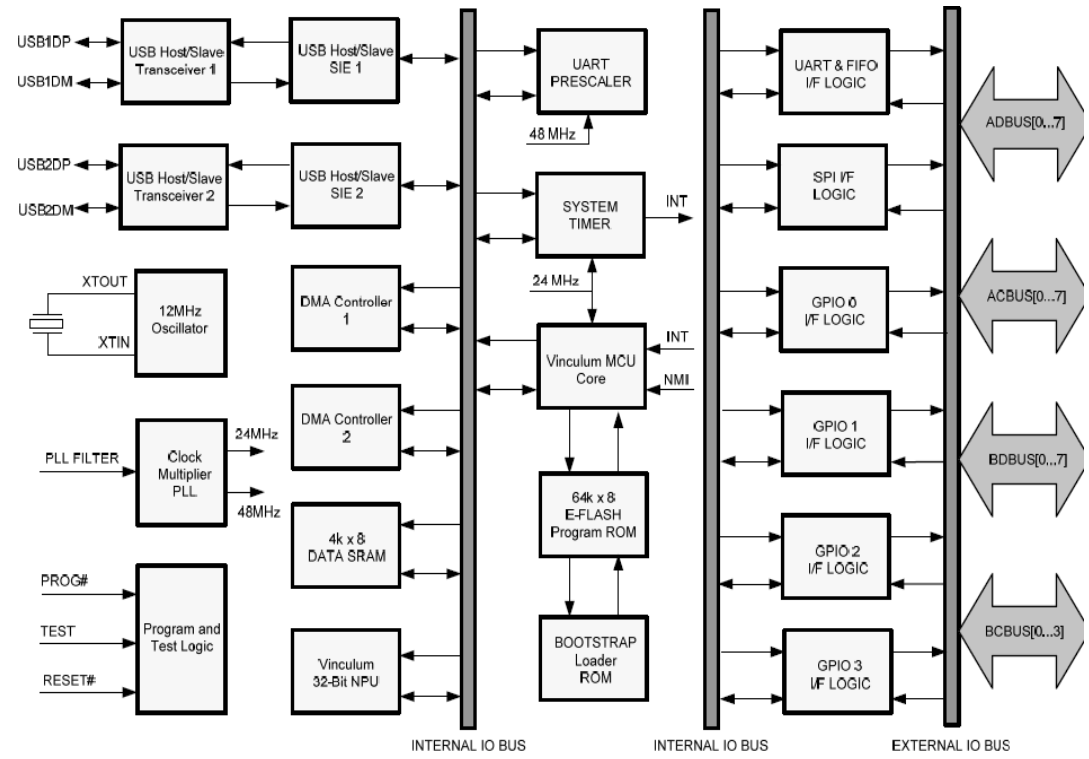
Figura 2. **FTDI Chip VNC1L-1A**

Fuente: Hoja de datos VNC1L-1A. p. 1.

2.1. Diagrama de bloques simplificado

En la figura 3 se muestra el diagrama de bloques simplificado del VNC1L-1A. Los bloques principales son los transreceptores, el motor de la interface USB (SIE), el oscilador, el multiplicador de reloj PLL, programación y pruebas de lógica, controlador DMA, información de SDRAM, procesador numérico NPU, temporizador del sistema, núcleo MCU, ROM de programa, Bootstrap loader ROM, Lógica UART y FIFO y GPIO.

Figura 3. **Diagrama de bloques simplificado VNC1L-1A**



Simplified Block Diagram

Fuente: hoja de datos VNC1L-1A. p. 6.

2.1.1. Descripción de los bloques funcionales

Transreceptores USB: los dos transreceptores proveen la interface física del USB, soportando los estándares USB 1.1 y USB 2.0. Son soportadas las velocidades de trabajo, baja y plena. Estos transreceptores también incluyen las resistencias pull-up o pull-down que son requeridas para operar en modo anfitrión o esclavo.

Motor de la interface USB (SIE): estos bloques manejan la conversión de paralelo a serial y serial a paralelo de la capa física del USB. Estos incluyen paquetes de bits de relleno, generación del frame USB y el chequeo de error.

Oscilador: este bloque se encarga de generar la señal de reloj de referencia para el multiplicador de frecuencia PLL desde un cristal externo de 12 megahertzios.

Multiplicador de reloj PLL: este toma la señal del oscilador (12 megahertzios) y genera señales de referencia de 24 y 48 megahertzios, las cuales son usadas por el bloque de motor de interface, el núcleo, el temporizador del sistema y los bloques del UART.

Programación y pruebas de lógica: este bloque permite programar la memoria E-FLASH. Cuando la señal PROG# es puesta a nivel bajo y el dispositivo es reiniciado, la memoria es accesada por medio del bootstrap loader ROM el cual contiene código que permite a la memoria E-Flash ser programada por medio de comandos enviados a través de la interface UART.

Controlador DMA: los controladores DMA hacen más versátil al VNC1L permitiendo que la información de los motores de la interface USB SIE, UART, FIFO e interfaces SPI pueda ser transmitida entre ellos, a través de la SRAM con una intervención mínima del núcleo.

SRAM de datos: este bloque actúa como la memoria de datos para el núcleo, pensando que esta también puede ser accesada al núcleo por medio de los controladores DMA.

NPU (Coprocesador numérico): las operaciones que usan aritmética de 32 bits, como los cálculos relacionados al sistema de archivos FAT, son realizados por este bloque.

UART Prescaler: este bloque provee el reloj principal para transmitir información a través del bloque UART. El baudrate del UART puede ser ajustado desde 300 baudios hasta 1 megabaudio.

Temporizador del sistema: provee una interrupción regular al firmware del VNC1L.

Núcleo MCU: este núcleo del procesador está basado en una arquitectura de 8-bit propia de FTDI. Soporta programas de hasta 64 kilobytes para información y 256 bytes de espacio para las entradas y salidas.

E-Flash memoria del programa ROM: el VNCL1L cuenta con una memoria integrada E-Flash de 64 kilobytes la cual es la encargada de almacenar el programa o firmware del dispositivo. No se necesitan voltajes especiales para la programación de la memoria ya que son proveídos internamente por el chip.

Bootstrap Loader ROM: este es un pequeño bloque de memoria ROM (512 x 8 bits) que realiza un bypass a la memoria E-Flash cuando la señal PROG# está en estado 0 y la conecta al UART. Esto permite programar completamente la E-Flash por medio de la interface UART. Todo dispositivo VNC1L debe ser programado primeramente a través de la interface UART, luego puede ser programado a través del puerto USB.

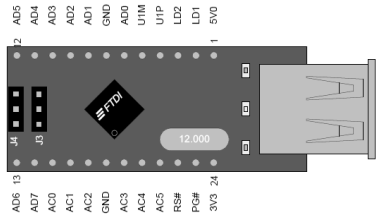
Bloques GPIO: son pines de I/O de uso general. No todos los pines I/O están disponibles para ser configurados por el usuario.

2.2. VDIP

El VDIP es módulo de desarrollo para el circuito integrado VNC1L el cual se encuentra embebido en el VDIP. Este cuenta con un regulador de 3,3 volteos el cual alimenta al VNC1L.

En la figura 4 se muestra el diagrama de pines del dispositivo VDIP. La función de cada uno de los pines se detalla en la tabla XII.

Figura 4. **Diagrama de pines VDIP 1**



Fuente: hoja de datos VDIP 1 FTDI's CHIPS. p. 3.

Tabla XII. **Descripción de pines**

No. Pin	*Nombre*	*Nombre en el PCB*	*Tipo*	*Descripción*
1	5V0	5V0	Entrada de poder	Módulo de alimentación de 5V. Este pin provee la alimentación de 5V para el dispositivo USB y también los 3.3V para la alimentación del VNC1L, por medio de un regulador de voltaje de 3.3V colocado en la placa.
2	LED1	LD1	Salida	Indicador de tráfico de datos en el puerto USB 1. Este pin está conectado al led verde de la placa. Este pin pernite colocar un led indicador de tráfico externo a la placa
3	LED2	LD2	Salida	Indicador de tráfico de datos en el puerto USB 2. Este pin está conectado al led verde de la placa. Este pin pernite colocar un led indicador de tráfico externo a la placa
4	USBD1P	U1P	I/O	Puerto USB anfitrión/esclavo - Señal de datos positiva con resistencia pull up/ pul down integrada. El módulo cuenta con una resistencia de 27 Ohm en serie. Los pines pueden ser usados junto con el pin cinco para proveer un nuevo puerto USB.
5	USBD1M	UIM	I/O	Puerto USB anfitrión/esclavo - Señal de datos negativa con resistencia pull up/ pul down integrada. El módulo cuenta con una resistencia de 27 Ohm en serie. Los pines pueden ser usados junto con el pin cuatro para proveer un nuevo puerto USB.
6	ADBUS0	AD0	I/O	5V información bidireccional/ bus de control, AD bit 0
7	GND	GND	PWR	Pin de alimentación de tierra del módulo
8	ADBUS1	AD1	I/O	5V información bidireccional/ bus de control, AD bit 1
9	ADBUS2	AD2	I/O	5V información bidireccional/ bus de control, AD bit 2
10	ADBUS3	AD3	I/O	5V información bidireccional/ bus de control, AD bit 3
11	ADBUS4	AD4	I/O	5V información bidireccional/ bus de control, AD bit 4
12	ADBUS5	AD5	I/O	5V información bidireccional/ bus de control, AD bit 5
13	ADBUS6	AD6	I/O	5V información bidireccional/ bus de control, AD bit 6
14	ADBUS7	AD7	I/O	5V información bidireccional/ bus de control, AD bit 7
15	ACBUS0	AC0	I/O	5V información bidireccional/ bus de control, AC bit 0
16	ACBUS1	AC1	I/O	5V información bidireccional/ bus de control, AC bit 1
17	ACBUS2	AC2	I/O	5V información bidireccional/ bus de control, AC bit 2
18	GND	GND	PWR	Pin de alimentación de tierra del módulo
19	ACBUS3	AC3	I/O	5V información bidireccional/ bus de control, AC bit 3
20	ACBUS4	AC4	I/O	5V información bidireccional/ bus de control, AC bit 4
21	ACBUS5	AC5	I/O	5V información bidireccional/ bus de control, AC bit 5
22	RESET#	RS#	Input	Puede ser usada por un dispositivo externo para resetear el VNC1L. Este puede ser usado en conjunto con PROG# y la interfase UART/FIFO/SPI para programar el firmware en el VNC1L
23	PROG#	PG#	Input	Este pin es usado en combinación con RESET# y la interfase UART/FIFO/SPI para programar el firmware en el VNC1L
24	3V3	3V3	PWR Output	Salida de 3.3V de la placa base del VDIP1

Fuente: hoja de datos VDIP 1 FTDI's CHIPS. p. 4.

2.3. Monitor de comandos

El dispositivo VNC1L tiene 2 puertos USB, así como, una interface de comunicación UART, SPI o FIFO. La interface de comunicación puede ser seleccionada por medio de puentes que pueden ser configurados en el dispositivo.

El firmware del VNC1L permite al monitor de comandos ser activado en cualquiera de las interfaces o incluso uno de los puertos USB. La función principal del monitor de comandos es permitir a los dispositivos comunicarse por medio de las interfaces UART, FIFO o SPI con dispositivos USB esclavos.

Los dispositivos esclavos más comunes son dispositivos de almacenamiento masivo USB e impresoras.

La comunicación y el control del VNC1L es realizada por medio de instrucciones que son enviadas al monitor de comandos el cual se encarga de interpretarlos.

2.3.1. Modos de operación

El VNC1L puede cambiar entre dos modos de operación, los cuales cambian el método para comunicarse con los dispositivos esclavos USB. En el modo comando, el VNC1L interpreta y ejecuta los comandos; en el modo de datos la información que es recibida en el monitor es conducida directamente hacia otro dispositivo.

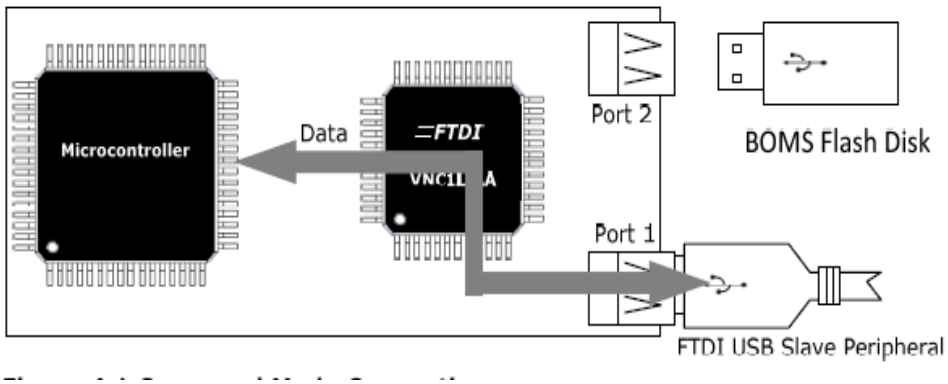
Las señales DATAREQ# y DATAACK# permiten cambiar de modo de operación. Si se está usando la interface serial UART estas señales se encuentran en los pines de la interface DTR# y DSR#. Si se está usando la interface FIFO y SPI se encuentran en los pines 45 y 46.

2.3.1.1. Modo comando

Cuando el firmware del VNC1L inicia, se encuentra en modo comando. En este modo la señal DATAACK# es alta y DATAREQ# deberá ser mantenida alta para permanecer en modo comando.

Mientras se encuentra en modo comando, es posible configurar la interface serial UART, BOMS, etcétera.

Figura 5. **Esquema en modo comando**



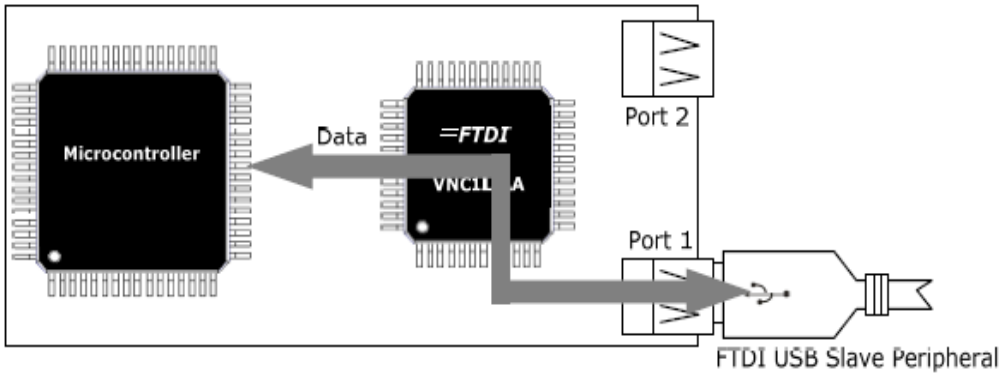
Fuente: Future Technology Devices International Ltd. Vinculum Firmware Manual. p. 14.

2.3.1.2. Modo de datos

Para cambiar a modo de datos la señal DATAREQ# deberá ser baja. Una vez la señal DATAACK# pasa a ser baja, cualquier dato enviado al puerto del monitor será enviado al dispositivo USB esclavo conectado ha cualquiera de los puertos USB. Cualquier información recibida desde el esclavo USB será repetida al monitor del puerto.

En este modo el VNC1L ignora los datos y simplemente los pasa entre los puertos USB y el puerto del monitor. La figura 6 ilustra la operación.

Figura 6. **Esquema en modo de datos**



Fuente: Future Technology Devices International Ltd. Vinculum Firmware Manual. p. 15.

2.3.2. Firmwares estándares

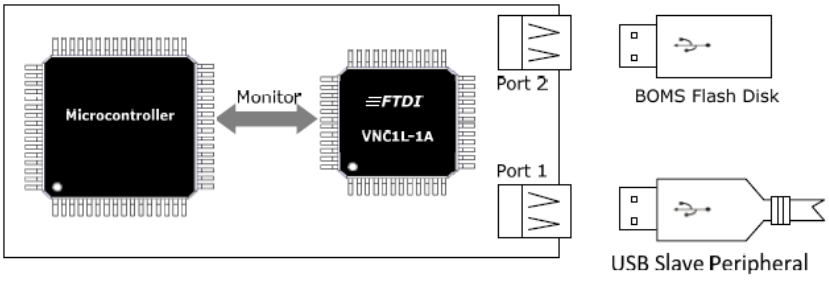
El firmware se refiere al software de bajo nivel contenido dentro de la memoria ROM del VNC1L. Este software permite la operación del dispositivo y define las capacidades y características de su operación.

El VNC1L ofrece la posibilidad de trabajar con 5 diferentes firmwares estándar, los cuales están especializados para realizar una tarea específica.

2.3.2.1. VDAP

Este firmware permite a un microcontrolador operar como un anfitrión USB permitiendo la conexión de un dispositivo de almacenamiento masivo en el puerto 2 y otro periférico en el puerto 1. Este firmware es el que se utilizará para el desarrollo del sistema de monitoreo de producción. La figura 7 ilustra la operación del VNC1L con el firmware VDAP.

Figura 7. **VNC1L en conjunto con un microcontrolador**



Fuente: Future Technology Devices International Ltd. Vinculum Firmware Manual. p. 9.

2.3.2.2. VMSC

El firmware VMSC es similar al VDAP, pero en él se introducen nuevos comandos que permiten la reproducción de archivos MP3 por medio de un dispositivo decodificador. Los datos son enviados al decodificador de MP3 VLSI VS1003 a través de un bus SPI y el control de la reproducción es realizada por medio de un microcontrolador, como se ilustra en la figura 8.

Figura 8. **VNC1L conectado a un decodificador MP3**

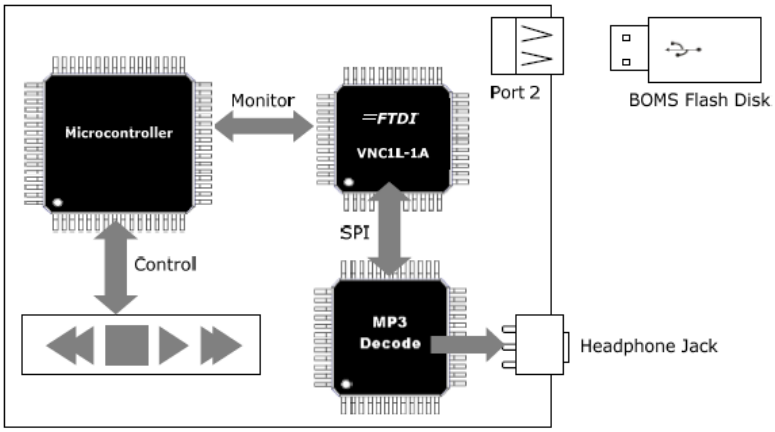


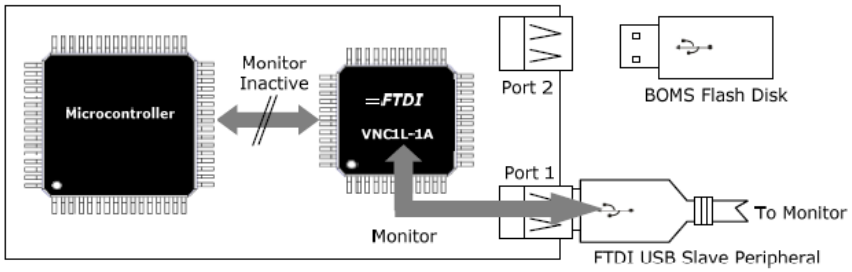
Figure 3.2 VNC1L Connected to MP3 Decoder and Microcontroller

Fuente: Future Technology Devices International Ltd. Vinculum Firmware Manual. p. 10.

2.3.2.3. VDIF

El firmware VDIF está diseñado para trabajar de manera similar al VDAP hasta que un periférico esclavo USB de FTDI es detectado en el puerto USB 1. Cuando esto ocurre, el monitor de comandos cesa de estar activo en la interface principal y es activado en el puerto donde fue conectado el dispositivo periférico esclavo, como se ilustra en la figura 9. Esto permite el acceso a un dispositivo de almacenamiento masivo por medio de un dispositivo periférico esclavo USB.

Figura 9. **VNC1L con monitor en el dispositivo esclavo FTDI**

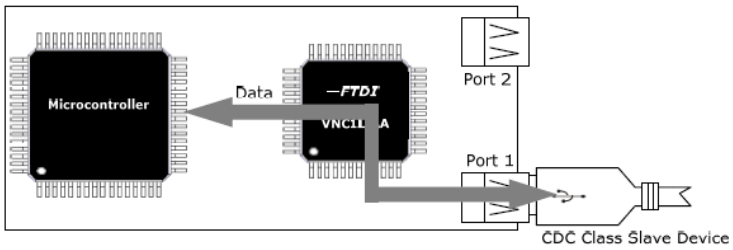


Fuente: Future Technology Devices International Ltd. Vinculum Firmware Manual. p. 10.

2.3.2.4. VCDC

Ciertos dispositivos de comunicación seleccionados pueden ser conectados al VNC1L, para que este realice la función de MODEM para el microcontrolador, permitiendo al microcontrolador comunicarse con dispositivos de comunicación externos que cuenten con una interface USB. La figura 10 muestra la conexión a un dispositivo CDC (dispositivo clase comunicación).

Figura 10. **VNC1L conectado a un dispositivo CDC**

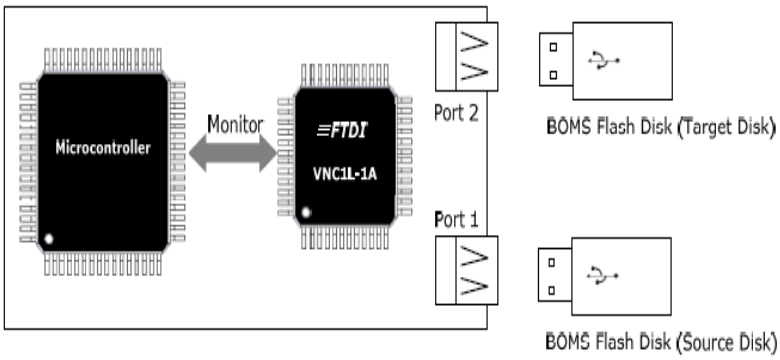


Fuente: Future Technology Devices International Ltd. Vinculum Firmware Manual. p. 11.

2.3.2.5. VDFC

El firmware VDFC permite accesar a cualquiera de dos dispositivos de almacenamiento masivo (BOMS) conectados al VNC1L, además de facilitar la transferencia de datos entre dispositivos. La figura 11 ilustra su función.

Figura 11. **VNC1L con dos dispositivos de almacenamiento**



Fuente: Future Technology Devices International Ltd. Vinculum Firmware Manual. p. 11.

Todos estos firmwares permiten al VNC1L-A realizar una o varias tareas para una aplicación específica. Para realizar estas tareas es necesario que un control externo se comunique con el dispositivo anfitrión para indicarle lo que se desea realizar.

Para establecer la comunicación entre el VNC1L-A y el control se puede usar una puerto UART, uno SPI o uno FIFO. Estos son las interfaces del firmware.

2.4. Interfaces del firmware

Existen tres opciones de interfaces del firmware para el monitor de comandos, UART (serial asíncrona), FIFO (paralela) y SPI (serial síncrona). Las conexiones de los pines usados para seleccionar el modo de interface que se utilizará son mostradas en la tabla XIII.

Tabla XIII. **Combinaciones para la selección de interfaces**

Pin No.		Modo
47	46	
Alto	Alto	UART
Alto	Bajo	SPI
Bajo	Alto	FIFO
Bajo	Bajo	UART

Fuente: hoja de datos VDIP 1 FTDI's CHIPS. p. 4.

Para el desarrollo del sistema se ha escogido el uso de la interface UART debido a que es una de las interfaces más comunes por medio de las cuales se puede establecer una comunicación con un controlador lógico programable.

Debido a lo antes mencionado, únicamente se hará un enfoque en la interface UART.

2.4.1. UART

El nombre UART proviene de sus siglas en inglés (Universal Asyhncronos Reciver/Transmitter), esta interface transmite bit a bit la información que se desea transmitir, a diferencia del puerto paralelo (FIFO) que transmite varios bits a la vez. A diferencia de la interface SPI (interface serial para periféricos) la cual cuenta con una señal de reloj que permite la sincronización de los dispositivos participantes en la comunicación, el puerto serial cumple con un protocolo de transmisión asíncrono, debido que dentro de las señales de transmisión no se cuenta con una señal específica de sincronización, generalmente conocida como reloj.

La transmisión de datos se realiza enviando varias cadenas de bits, cada trama o cadena de bits inicia con un bit de inicio el cual se utiliza para sincronizar al emisor con el receptor, luego de esto se transmiten 7 u 8 bits de datos y algunas veces un bit más conocido como bit de paridad. Para saber que la trama ha finalizado se envía uno o dos bits de finalización indicando al receptor que el primer dato fue transmitido.

La interface UART del VNC1L implementa un puerto serial asíncrono estándar con bits de control de flujo. Puede operar desde 300 baudios hasta 1 megabaudio.

La transmisión usa un formato NRZ (señal sin retorno a cero) y consiste de 1 bit de inicio, 7 u 8 bits de datos, un bit de paridad opcional y uno o dos bits de final de trama. Cuando se envía información, el bit menos significativo se transmite de primero.

La velocidad de transmisión, los ajustes del control de flujo, el número de bits de datos, paridad y final de trama pueden ser configurados por medio del monitor de comandos del firmware.

El VNC1L cuenta con un reloj interno para sincronización de la transmisión de datos a través de la interface UART. Cada cierto número de ciclos de reloj envía un bit. El número de ciclos de espera entre el envío de cada bit puede ser modificado para modificar la velocidad de transmisión.

Para configurar la velocidad de transmisión del puerto serial se debe enviar el comando “SBD’<sp><divisor (3 bytes) primero el LSB><cr>” este comando “SBD” le indica al VNC1L que los siguientes 3 bytes serán el divisor de reloj que determinará los ciclos de espera entre el envío de cada bit. Es importante mencionar que el VNC1L sólo soporta ciertos valores ya establecidos como divisores, lo cual se explica en el capítulo 4 en la sección de comandos aplicados VDAP.

La interface UART utiliza ocho señales, 2 para transmisión, 1 de referencia y 5 para control de flujo de información cuando se conecta a un dispositivo DCE (equipo de comunicación de datos). En este sistema se conectarán directamente dos dispositivos DTE (equipo terminal de datos) el VNC1L y un controlador lógico programable, por lo tanto, se utilizarán únicamente 3 señales RX, TX y SG; para lo cual se debe asegurar que la velocidad de transmisión de datos de ambos dispositivos sea la misma. Las señales de control de flujo de datos son usadas cuando se tiene comunicación con dispositivos DCE como un MODEM y las velocidades de transmisión de datos entre los dispositivos y la salida del MODEM no son las mismas. La descripción de cada señal se muestra en la tabla XIV.

Tabla XIV. **Asignación y descripción de señales de interface UART del VDIP**

Pin No.	*Nombre*	*Nombre en el Circuito Impreso*	*Tipo*	*Función seleccionando la interfase UART*	*Descripción*
6	ADBUS0	AD0	I/O	TXD	Salida asíncrona de transmisión de datos.
8	ADBUS1	AD1	I/O	RXD	Entrada asíncrona de recepción de datos.
9	ADBUS2	AD2	I/O	RTS#	Salida de control - Solicitud de envío de datos. Esta línea informa al MODEM que el dispositivo está listo para intercambiar información.
10	ADBUS3	AD3	I/O	CTS#	Entrada de control - Libre para enviar datos. Esta línea indica que el MODEM está listo para intercambiar información.
11	ADBUS4	AD4	I/O	DTR#	Salida de control - Terminal de datos lista. Esta línea indica al MODEM que el dispositivo está listo para establecer una comunicación.
12	ADBUS5	AD5	I/O	DSR#	Entrada de control - Datos listos para enviar. Esta línea indica al MODEM que el dispositivo está listo para establecer una comunicación.
13	ADBUS6	AD6	I/O	DCD#	Entrada de control - Detección de acarrreo.
14	ADBUS7	AD7	I/O	RI#	Entrada de control - puede ser usada para activar el host USB despues de estar suspendido.
15	ACBUS0	AC0	I/O		
16	ACBUS1	AC1	I/O		
17	ACBUS2	AC2	I/O		
19	ACBUS3	AC3	I/O		
20	ACBUS4	AC4	I/O		

Fuente: elaboración propia.

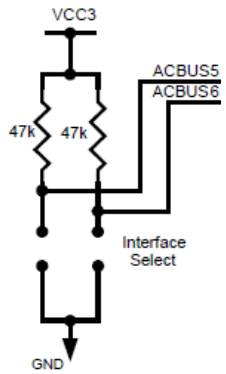
2.5. Firmware VDAP

El firmware VDAP está diseñado para permitir al dispositivo VNC1L actuar como una interface entre un dispositivo de almacenamiento masivo USB colocado en el puerto USB 2 y la interface de comunicación 0 del VNC1L. Este firmware permite configurar externamente la interface comunicación como un puerto UART, paralelo FIFO o SPI, usando puentes para la selección. Los dispositivos conectados a la interface de comunicación pueden enviar comandos los cuales permiten realizar operaciones en el dispositivo de almacenamiento masivo conectado al puerto USB 2. El puerto de comunicación el cual es configurado para recibir estos comandos es conocido como el monitor de comandos.

2.5.1. Selección del puerto del monitor de comando

El firmware VDAP configura el puerto monitor de comandos como UART, paralelo FIFO o SPI según sea seleccionado en los puentes de los pines ACBUS5 y ACBUS6. La configuración de los puentes es mostrada en la figura 12. Si no está conectado ningún puente el sistema configurará el puerto monitor de comandos como una interface UART. La tabla XV muestra la configuración de los puentes según la interface a usar.

Figura 12. **Configuración de puentes para la selección de interface**



Fuente: Future Technology Devices International Ltd. VDAP Firmware Manual. p. 3.

Tabla XV. **Combinaciones para la selección de interfaces**

ACBUS6 (pin 47)	*ACBUS5 (pin 46)*	*Mode*
Pull-Up	Pull-Up	Serial UART
Pull-Up	Pull-Down	SPI
Pull-Down	Pull-Up	Parallel FIFO
Pull-Down	Pull-Down	Serial UART

Fuente: Future Technology Devices International Ltd. VDAP Firmware Manual. p. 4.

Como se mencionó anteriormente, la interface seleccionada para el monitor de comandos puede operar en modo de comando o en modo de datos, realizando la función de un MODEM. El modo de comando es usado para comunicarse con el VNC1L mientras que el modo de datos es usado para comunicarse con un dispositivo esclavo conectado en el puerto USB 1. Para seleccionar el modo de operación se utilizan dos señales adicionales DATAACK# y DATAREQ# las cuales deben ser configuradas como se explicó en el inciso 2.3.1.

2.5.2. Comandos del monitor

El monitor de comandos cuenta con un conjunto de comandos o instrucciones las cuales puede interpretar y realizar. Puede reconocer dos tipos de comandos, los extendidos, los cuales son una serie de caracteres ASCII y están orientados para el control desde una terminal o los comandos cortos o hexadecimales, los cuales están orientados para el control desde un microprocesador. A través de comandos se puede seleccionar el tipo de comandos que se desea utilizar ya sean extendidos o cortos.

Al enviar el comando desde la terminal o el microprocesador al monitor de comandos, este busca interpretarlo, si no lo reconoce o hay algún problema con la instrucción que desea realizar este devuelve un mensaje de error. Si logra realizar la instrucción enviada sin problema este devuelve la información solicitada o un indicador de que se realizó con éxito la instrucción.

2.5.2.1. Listado de comandos

La tabla XVI muestra de una forma ordenada y de fácil comprensión el conjunto de comandos que el firmware VDAP es capaz de reconocer.

Tabla XVI. **Listado de comandos**

Comandos Extendidos ASCII	*Comandos cortos Hexadecimales*	*Función del comando*	*Respuesta*
Cambio entre set de comandos extendidos y set de comandos cortos			
'SCS' <cr>	$10,$0D	Cambia a set de comandos cortos	Retorna el indicador '>',$0D para indicar que el dispositivo está en modo de comados cortos
'ECS'<cr>	$11,$0D	Cambia set de comandos extendidos	Retorna el indicador 'D:\>',$0D para indicar que el dispositivo está en modo de comados extendidos
'E'<cr>	'E'<cr>	Eco Envía el carácter para que este mismo le sea enviado	Retorna 'E' para sincronización
'e'<cr>	'e'<cr>	Eco Envía el carácter para que este mismo le sea enviado	Retorna 'E' para sincronización
Consulta de dispositivo disponible			
<cr>	$0D	Cheque si existe un dispositivo conectado	Si se encuentra en modo de comandos extendidos retorna 'D:\>',$0D o 'no disk',$0D si existe o no un dispositivo conectado respectivamente. Si se encuentra en modo de comandos cortos retorna '>',$0D o 'ND',$0D si existe o no un dispositivo conectado respectivamente.
Operaciones con directorios			
'DIR'<cr>	$01,$0D	Lista el directorio actual	Retorna una lista de los nombres de los archivos y directorios. Cada nombre es terminador por $0D. Los nombres de los directorios son seguidos de <sp> 'DIR' y términados con $0D.
'DIR'<sp> <nombre><cr>	$01,$20, <name>,$0D	Lista el nombre del archivo seguido por el tamaño de éste. Usarlo antes de leer un archivo para conocer cuantos bytes se esperan.	$0D, <nombre><sp>< tamaño en hexadecimal (4 bytes) LSB primero> $0D
'DLD'<sp> <nombre><cr>	$05,$20,<name>, $0D	Borra el directorio <nombre>	Si se encuentra en modo de comandos cortos retorna '>' $0D Si se encuentra en modo de comandos extendidos retorna 'D:\>'$0D
'MKD'<sp> <nombre><cr>	$07,$20, <name>,$0D	Crea el directorio <nombre>	Si se encuentra en modo de comandos cortos retorna '>' $0D Si se encuentra en modo de comandos extendidos retorna 'D:\>'$0D
'CD'<sp> <nombre><cr>	$02,$20,<name> $0D	Cambiar de direcotrio <nombre>	Si se encuentra en modo de comandos cortos retorna '>' $0D Si se encuentra en modo de comandos extendidos retorna 'D:\>'$0D
'CD'<sp>'..'<cr>	$02,$20,$2E,$2E,$0D	Moverse a un directorio arriba	Si se encuentra en modo de comandos cortos retorna '>' $0D Si se encuentra en modo de comandos extendidos retorna 'D:\>'$0D

Continuación de la tabla XVI.

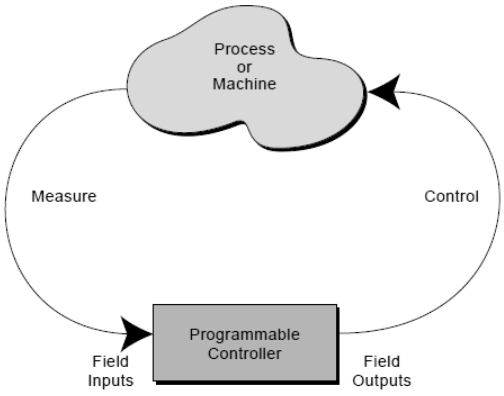
Operaciones con archivos			
'RD'<sp> <nombre><cr>	$04,$20,<name> $0D	Lee el archivo <nombre>	Retorna al monitor el archivo completo en binario. El tamaño debe ser consultado antes con el comando 'DIR'<sp><nombre><cr> a manera de conocer el número de bytes esperado.Seguido de '>'$0D o 'D:\>'$0D según sea el modo de comandos cortos o extendidos respectivamente
'RDF'<sp> <tamaño en hex(4 bytes) ><cr>	$0B,$20,size in hex(4 bytes) , $(	Lee los datos de <tamaño en hex(4 bytes)> del archivo actualmente abierto	Retorna la cantidad de datos solicitada Seguido de '>'$0D o 'D:\>'$0D según sea el modo de comandos cortos o extendidos respectivamente
'DLF'<sp> <nombre><cr>	$07,$20,<nombre> $0D	Borra el archivo <nombre>	Borra el archivo del diretorio actual y libera los sectores FAT. Retorna '>'$0D o 'D:\>'$0D según sea el modo de comandos cortos o extendidos respectivamente
'WRF'<sp> <tamaño en hex(4 bytes) ><cr> < bytes de datos de tamaño><cr>	$08,$20,tamaño in hex(4 bytes) , $0D $data,$0D	Escribe los datos de <tamaño en hex(4bytes)> al final del archivo actualmente abierto	Retorna '>'$0D o 'D:\>'$0D según sea el modo de comandos cortos o extendidos respectivamente
'OPW'<sp> <nombre><cr>	$09,$20, <nombre>,$0D	Abre un archivo para ser escrito	Retorna '>'$0D o 'D:\>'$0D según sea el modo de comandos cortos o extendidos respectivamente
'OPR'<sp> <nombre><cr>	$0E,$20, <nombre>,$0D	Abre un archivo para ser leido	Retorna '>'$0D o 'D:\>'$0D según sea el modo de comandos cortos o extendidos respectivamente
'CLF'<sp> <nombre><cr>	$0A,$20, <nombre>,$0D	Cierra un archivo	Retorna '>'$0D o 'D:\>'$0D según sea el modo de comandos cortos o extendidos respectivamente
'REN'<sp> <nombre orig> <sp> <nombre nuevo><cr>	$0C,$20, <nomre orig>,$20, <nombre nuevo> <cr>	Renombra un archivo o directorio	Retorna '>'$0D o 'D:\>'$0D según sea el modo de comandos cortos o extendidos respectivamente
'FS'<cr>	$12,$0D	Retorna el espacio libre en el disco en bytes.	Retorna <espacio libre en 4 hex(4Bytes) el LSB al inicio> $0D
Configuración del UART			
'SBD'<sp><divisor (3 bytes) primero el LSB t ><cr>	$14, $20,divisor (3 bytes) primero el LSB >,$0D	Configura el Baud Rate	Retorna '>'$0D o 'D:\>'$0D según sea el modo de comandos cortos o extendidos respectivamente
Administración de la energía			
'SUD'<cr>	$15,$0D	Suspende el dispostivo de almacenamiento masivo para conservar energía. El disco será despertado automáticamente la próxima vez que un comando de disco sea enviado a él	Retorna '>'$0D o 'D:\>'$0D según sea el modo de comandos cortos o extendidos respectivamente
'WKD'<cr>	$16,$0D	Despierta el dispositivo de almacenamiento masivo	Retorna '>'$0D o 'D:\>'$0D según sea el modo de comandos cortos o extendidos respectivamente
'SUM'<cr>	$17,$0D	Suspende el monitor de comados y para los relojes.	Retorna '>'$0D o 'D:\>'$0D según sea el modo de comandos cortos o extendidos respectivamente

Fuente: elaboración propia.

3. CONTROLES LÓGICOS PROGRAMABLES (PLC)

Los controladores lógicos programables, también llamados controladores programables o PLC, son dispositivos de estado sólido de la familia de las computadoras, los cuales utilizan circuitos integrados en lugar de dispositivos electromecánicos para realizar tareas de control. Estos pueden controlar secuencias, temporizado, conteo, aritmética, manipulación de datos y comunicación en máquinas y procesos industriales. La figura 13 presenta un diagrama conceptual de la aplicación de un PLC.

Figura 13. **Diagrama conceptual de la aplicación de un PLC**

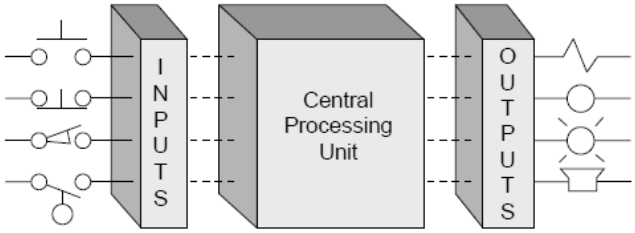


Fuente: L.A. Bryam/E.A.Bryan. Programable Controllers Theory and Implmentation. p. 4.

3.1. Principio de operación

Un PLC consiste de dos secciones básicas, la unidad central de procesamiento (CPU) y la interface de entradas/salidas del sistema.

Figura 14. **Diagrama de bloques del PLC**



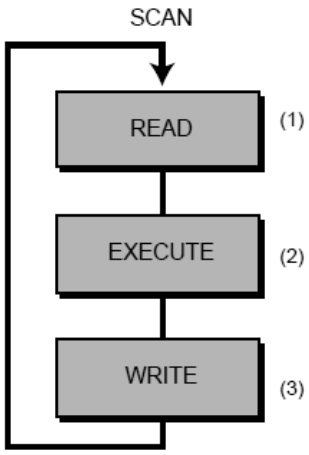
Fuente: L.A. Bryam/E.A.Bryan. Programable Controllers Theory and Implmentation. p. 10.

La unidad central de procesamiento es la encargada de gobernar todas las actividades del controlador programable. Este se encuentra compuesto por tres secciones, el procesador, la memoria del sistema y la alimentación de potencia del sistema.

La interface de entradas/salidas se encuentra conectada físicamente a los dispositivos de campo de la máquina, estos se dividen en dos grupos, elementos de entrada (sensores, pulsadores, pantallas, etcétera los cuales brindan información sobre el estado del proceso controlado y los elementos de salida (válvulas, relés, contactores, etcétera) los cuales ejercen una acción que afecta de manera directa o indirectamente al proceso.

Durante la operación el PLC realiza tres acciones, la lectura o chequeo de los elementos de entrada, ejecuta el programa almacenado en la memoria del sistema y escribe o actualiza la interface de salidas. Este proceso se ejecuta secuencialmente y es conocido como scanning.

Figura 15. **Ciclo SCAN**



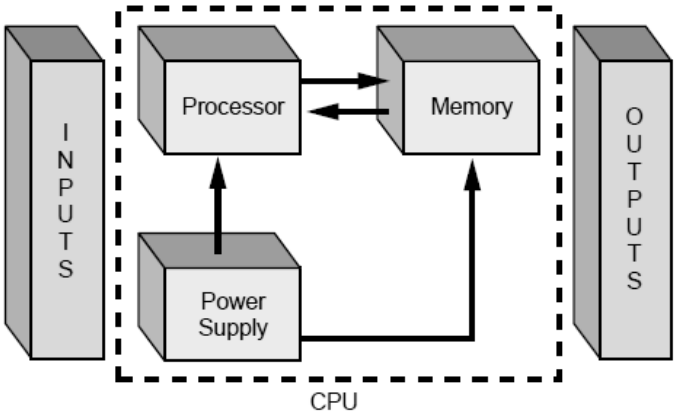
Fuente: L.A. Bryam/E.A.Bryan. Programable Controllers Theory and Implmentation. p. 11.

3.2. Componentes principales del CPU

El CPU es considerado el cerebro del sistema, está formado por tres componentes principales, el procesador, el sistema de memoria y la fuente de potencia.

La arquitectura del CPU puede variar de un fabricante a otro, pero en general, la mayoría de los CPU contienen una organización de tres componentes como se muestra en la figura 16. La herramienta de programación no es considerada como parte del CPU, pero esta completa la arquitectura al ser el medio de comunicación entre el programador y el CPU.

Figura 16. **Diagrama de bloques del CPU**



Fuente: L.A. Bryam/E.A.Bryan. Programable Controllers Theory and Implmentation. p. 82.

El CPU alberga todos los componentes necesarios que forman la inteligencia del sistema. Todos los componentes operan de manera conjunta, el procesador ejecuta las instrucciones de programa almacenadas en la memoria, mientras que el sistema de alimentación de potencia proporciona todos los niveles de voltaje necesarios para asegurar la correcta operación del procesador y los componentes de memoria.

3.2.1. Procesador

La función principal del procesador es comandar y regir completamente las actividades de todo el sistema. Este se encarga de interpretar y ejecutar una colección de programas del sistema conocidos como firmware. Este es almacenado permanentemente en el sistema y es considerado como parte del controlador. Al ejecutarlo el procesador puede realizar todas las funciones de control, procesamiento y comunicación.

El firmware lleva a cabo la comunicación entre el PLC y el usuario a través del dispositivo de programación. Este también ejecuta la comunicación con elementos periféricos, como dispositivos de monitoreo de campo; lee la información de diagnóstico correspondiente a los módulos de entradas/ salidas y la memoria; y se comunica con una interface de operación.

Los procesadores utilizados en el PLC son categorizados de acuerdo al tamaño o longitud de la palabra. Las longitudes estándares son de 8, 16 y 32 bits. La longitud de la palabra está directamente relacionada con la velocidad de procesamiento del procesador. Ya que mientras más grande sea la longitud de la palabra el sistema podrá manejar mayor cantidad de información en cada ciclo.

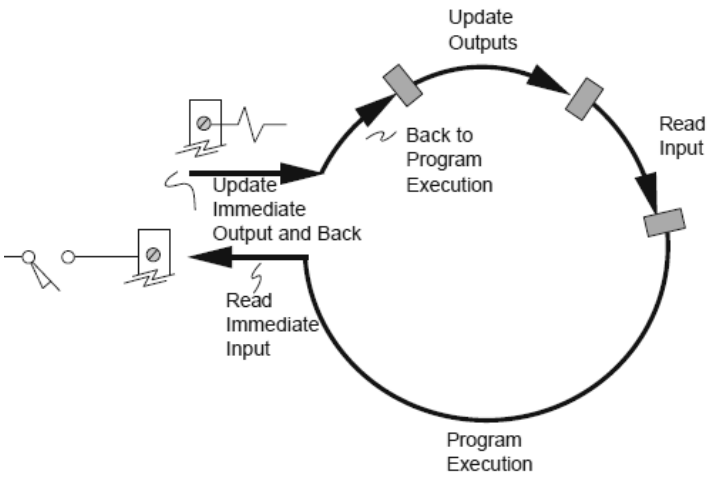
3.2.1.1. Ciclo de revisión

Un controlador básicamente realiza tres operaciones, la lectura de los dispositivos de entrada, ejecuta el programa de control y activa los dispositivos de salida, de acuerdo con la lógica programada. Este último proceso en realidad ocurre en dos pasos. Primero, luego que el procesador ejecuta la lógica interna programada, activa cada una de las salidas internas (mapeadas en memoria). La activación de estas salidas internas no significa la activación de las salidas físicas o interface de salidas. Luego que el procesador ha terminado de evaluar, todo el programa procede a realizar la actualización de las salidas físicas. Todo este proceso es llamado ciclo de revisión del sistema (system scan). Al terminar el ciclo de revisión el procesador activa una señal interna llamada señal de fin de revisión (EOS).

El tiempo que le toma al procesador realizar el ciclo de revisión es llamado tiempo de ciclo. Y este tiempo depende del tamaño en memoria del programa realizado por el programador. Para aplicaciones de alto dinamismo el tiempo de ciclo se vuelve un parámetro crítico por lo cual programas más eficientes deben ser implementados.

El método de revisión común, revisando el estado de las entradas al final de cada ciclo de revisión puede ser inadecuado para la lectura de algunas señales de entrada extremadamente rápidas. Algunos controladores cuentan con instrucciones de software para leer una entrada o escribir una salida inmediatamente, algo como una rutina de interrupción. La figura 17 muestra el ciclo de revisión para este caso particular.

Figura 17. **Ciclo de revisión con interrupciones**



Fuente: L.A. Bryam/E.A.Bryan. Programable Controllers Theory and Implmentation. p. 89.

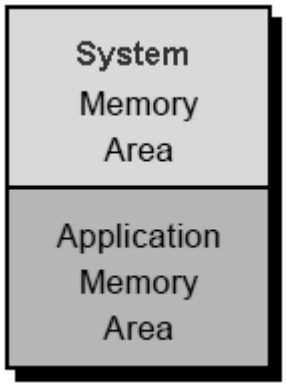
3.2.2. Sistema de memoria/Interface IO

El sistema de memoria es el área dentro del CPU del controlador donde son guardadas y ejecutadas por el procesador todas las secuencias de instrucciones o programas.

3.2.2.1. Secciones de la memoria

El sistema de memoria está dividido en dos secciones, la sección de memoria del sistema y la sección de memoria de aplicación.

Figura 18. **Secciones del sistema de memoria**



Fuente: L.A. Bryam/E.A.Bryan. Programable Controllers Theory and Implmentation, p. 110.

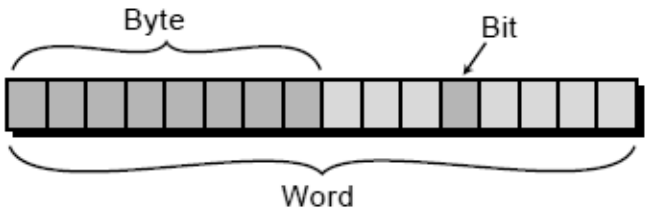
La sección de memoria del sistema está formada por el firmware del controlador, el cual dirige todas las actividades del controlador y se encarga de la ejecución de los programas de control y la comunicación con dispositivos periféricos. En sección es donde se encuentran almacenadas las instrucciones disponibles para la programación (instrucciones de bits, bloques de funciones de transferencia, operaciones multibit, instrucciones matemáticas, etcétera). Esta sección de memoria no es accesible por el usuario. La memoria de ejecución se encuentra almacenada en la memoria no volátil del sistema por lo general una EPROM.

La sección de memoria de aplicación provee el área de almacenamiento para el programa realizado por el usuario. Esta se compone de varias secciones, cada una teniendo una función y uso específico. La memoria de aplicación se encuentra almacenada ya sea en una RAM con alimentación de respaldo (para evitar que se pierda la información) o un arreglo entre memoria RAM e EPROM, lo cual evita que se pierda el programa aún en ausencia total de alimentación eléctrica.

3.2.2.2. Estructura y capacidad de la memoria

El sistema de memoria de un controlador puede ser pensado como un arreglo bidimensional de bits, en el cual cada fila del arreglo corresponde a una locación de memoria, la cual está formada por un conjunto de bits. La longitud de cada locación o número de bits que la forman, está determinada por la arquitectura del procesador, las longitudes más comunes son 8,16 y 32 bits. A la longitud de cada locación de memoria también se le llama tamaño de la palabra, aunque el término palabra se utiliza para designar una agrupación de 16 bits.

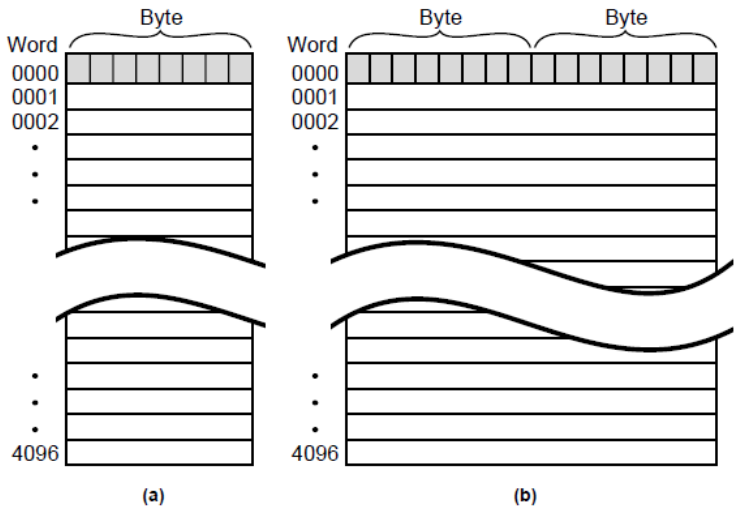
Figura 19. **Locación de memoria**



Fuente: L.A. Bryam/E.A.Bryan. Programable Controllers Theory and Implmentation. p.115.

La capacidad de la memoria del controlador es un dato importante cuando se escoge un controlador, por lo general, esta capacidad se da en función de kilobytes, por ejemplo, en la figura 20a se muestra un sistema de memoria con una capacidad de 4 kilobytes mientras que en la figura 20b se muestra un sistema de memoria con capacidad para 8 kilobytes ya que la longitud de la palabra es de 2 bytes o 16 bits.

Figura 20. **Estructura de la memoria**



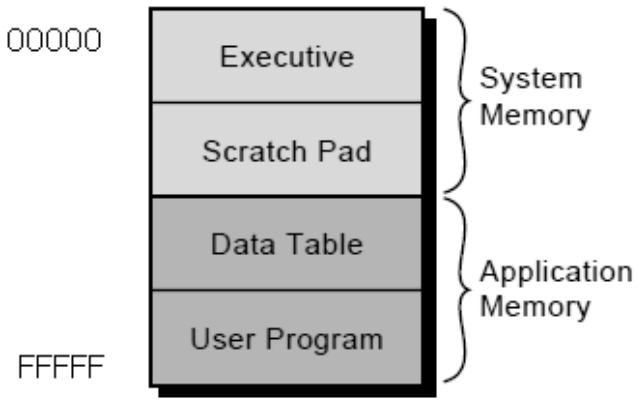
Fuente: L.A. Bryam/E.A.Bryan. Programable Controllers Theory and Implmentation. p. 115.

Un término de interés es la utilización de memoria, el cual se refiere al número de locaciones de memoria requeridas para almacenar cada tipo de instrucción. Este dato es dado generalmente para cada instrucción de software por el fabricante.

3.2.2.3. Organización de la memoria

Como se mencionó anteriormente el sistema de memoria se encuentra dividido en dos secciones, la memoria del sistema y la memoria de aplicación. La figura 21 muestra de manera simplificada lo que se conoce como mapa de memoria, el cual muestra no sólo lo que es almacenado en la memoria, sino también donde se almacena la información, de acuerdo con las locaciones específicas llamadas direcciones.

Figura 21. **Mapa de memoria**



Fuente: L.A. Bryam/E.A.Bryan. Programable Controllers Theory and Implmentation. p. 119.

A pesar de que dos diferentes controladores raramente tienen mapas de memoria idénticos, se pueden divisar 4 regiones básicas, área de memoria de administración, área de memoria de cálculo, área de tabla de datos y el área de programa. Las dos primeras pertenecen a la sección de memoria del sistema y no pueden ser accesadas por el usuario.

El área de tabla de datos almacena todos los datos asociados con el programa de control, como valores de inicialización de timers/contadores, variables y otros usados por el programa de control o el CPU. Esta sección también almacena la información del estado de las entradas (luego de ser leídas) y el estado de las salidas (luego de ser escritas).

El área de programa de usuario almacena el conjunto de instrucciones ingresadas por el usuario.

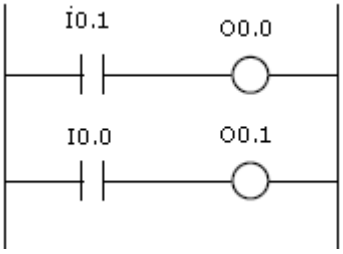
3.2.2.4. Direccionamiento

El término direccionamiento se refiere a relacionar una variable dentro del programa del controlador con una locación de memoria específica en el área de tabla de datos. Las variables dentro del programa se pueden direccionar hacia una entrada, salida, registro, palabras de inicialización de temporizadores o contadores, etcétera.

Por ejemplo, si se programa en lenguaje de escalera, cada contacto o bobina es una variable del programa y el programa necesita saber que locación de memoria guarda el estado de esta variable, por lo tanto, se le debe indicar.

En la figura 22 se muestra un ejemplo. Los contactos están direccionados a la entrada 0,1 y 0,0 y las bobinas a las salidas 0,0 y 0,1. Como se observa en la figura para direccionar una variable no se utiliza la dirección física en el mapa de memoria, sino se usa un operando el cual facilita la comprensión del programa. Cada operando tiene asignada una dirección en memoria específica.

Figura 22. **Direccionamiento**



Fuente: elaboración propia.

3.2.3. Interfaces de comunicación serial

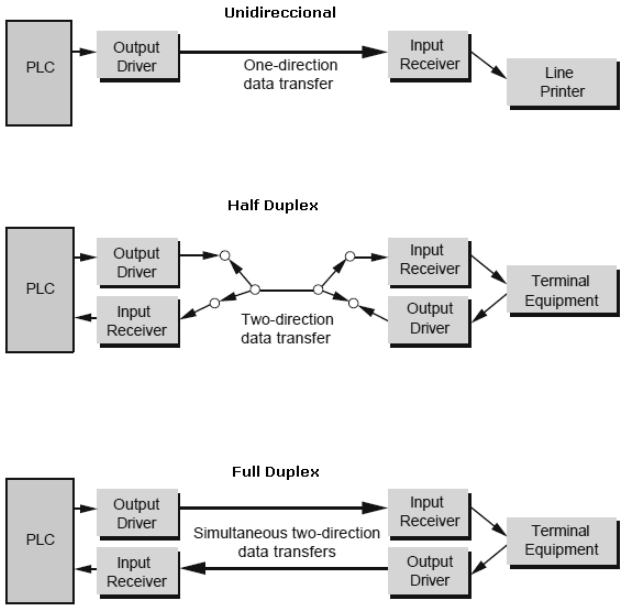
La mayoría de dispositivos periféricos de comunicación utilizan transmisión serial de datos. La transmisión serial permite al equipo periférico, como terminales remotas, modems, pantallas e impresoras, recibir información en forma de caracteres ASCII.

Los estándares más comunes para la comunicación serial son el RS-232 y el lazo de corriente de 20 miliamperios. Otros controladores utilizan estándares como el RS-422 y RS-485, los cuales mejoran y brindan una mayor flexibilidad y fiabilidad en la transmisión.

La comunicación entre los equipos puede ser realizada de manera unidireccional o bidireccional. Si un equipo es estrictamente emisor y otro el receptor ocurre una transmisión unidireccional y una sola línea de señal es necesaria. Cuando los equipos operan como emisores y receptores ocurre una comunicación bidireccional. Hay dos maneras de llevar a cabo una comunicación bidireccional, half duplex, en donde la información puede ser enviada en cualquier dirección, pero únicamente en una dirección a la vez y full duplex, en la cual la comunicación puede fluir en ambas direcciones simultáneamente, ya que se asignan dos líneas de señal para la transmisión, una asignada únicamente como entrada de datos y otra únicamente como salida de datos.

La figura 23 ilustra los tres modos de comunicación mencionados.

Figura 23. **Modos de comunicación**



Fuente: L.A. Bryam/E.A.Bryan. Programable Controllers Theory and Implmentation. p . 262.

Algunos módulos especiales de entrada/salida proporcionan al controlador la capacidad de comunicarse con dispositivos periféricos, como computadoras, otros controladores, dispositivos de campo, etcétera. La información entre el controlador y los periféricos generalmente se transmite por medio de caracteres ASCII o protocolos de red de buses de campo.

Las interfaces ASCII, reciben y transmiten datos alfanuméricos entre los dispositivos periféricos y el controlador, los cuales forman comandos que pueden ser interpretados por las dos partes. Generalmente, los dispositivos que cuentan con una interface ASCII cuentan con un intérprete de comandos el cual es capaz de reconocer una serie de comandos establecidos por el fabricante. Los dispositivos que comúnmente utilizan esta interface son impresoras, pantallas y monitores de video.

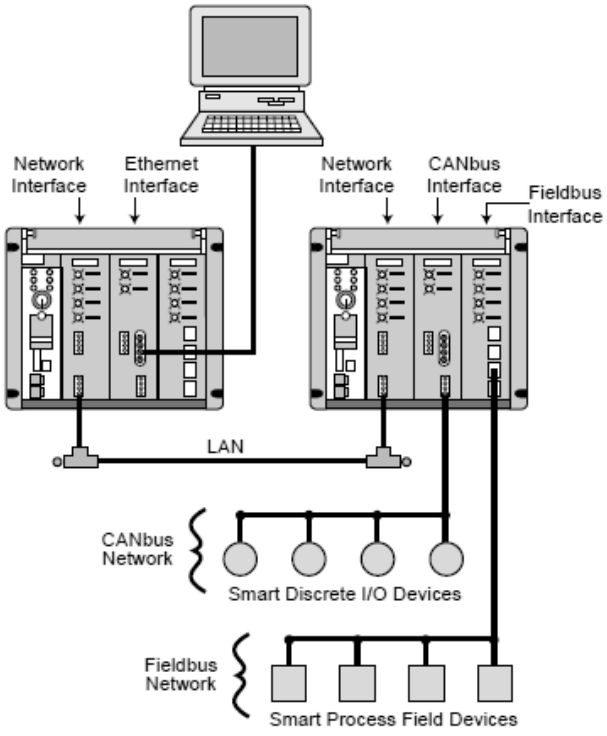
Las interfaces ASCII pueden ser completas, en las cuales se incluye un procesador y memoria RAM dedicados al procesamiento de la transmisión de datos o ser básica, en las cuales el procesamiento de datos es realizado por el mismo procesador que ejecuta el programa, lo cual limita la velocidad de transmisión ya que el ciclo de escaneo del programa define la velocidad de transmisión máxima.

Generalmente, el intercambio de información se realiza por medio de una comunicación RS-232, RS-485 o RS-422.

Las interfaces de red permiten al PLC y otro dispositivo inteligente comunicar y trasladar información por medio de una red local de alta velocidad. Al ser redes estandarizadas permiten que dispositivos de múltiples fabricantes puedan pertenecer a la red.

Las redes más comunes son CANbus, Profibus y Ethernet. Un módulo de interface de red permite todas las conexiones de comunicación y protocolos necesarios para asegurar que el mensaje es transmitido correctamente a través de la red. La interface de recepción acepta la transmisión, traslada la información al CPU y si es necesario envía un comando al dispositivo de campo. La velocidad y protocolo de comunicación varia dependiendo de la red utilizada.

Figura 24. **Interfaces de Red del PLC**



Fuente: elaboración propia.

3.2.3.1. RS-232C

El estándar EIA RS-232C define la interface entre los equipos de datos y los equipos de comunicación que utilizan transmisión serial para el intercambio de datos. El estándar define los detalles eléctricos y mecánicos de la comunicación. Una interface completa RS-232C consiste en 25 líneas, la cual incluye todas las posibles señales para realizar la comunicación unidireccional y bidireccional. Sin embargo, muchas de estas líneas son de uso especializado y unas pocas están indefinidas, la mayoría de periféricos requieren únicamente tres o cinco líneas para operar correctamente por lo tanto utilizan conectores de nueve pines (DB-9) en lugar de 25 pines (DB-25). La descripción de las líneas relevantes se mostró en el capítulo 2 en la tabla XIV.

El estándar RS-232C establece ciertas características eléctricas, como:

- Los valores de voltaje para un 0 lógico deben estar en el rango de +5 a +15 voltios; para un 1 lógico deben estar entre -15 y -5 voltios.

- La distancia máxima del cable de comunicación es de 15 metros.

- La impedancia de carga en el punto de terminación debe estar entre 3 000 y 7 000 Ohmios con una capacitancia no mayor a 2 500 picofaradios.

- Los voltajes entre -3 y 3 voltios no están definidos.

4. SISTEMA DE MONITOREO DE PRODUCCIÓN

Para realizar el sistema de monitoreo de producción se utilizan dos elementos principales, un dispositivo anfitrión USB (VDIP) y un controlador lógico programable marca FESTO (FEC-FCXX).

Se optó por el uso de un controlador lógico programable, debido a su robustez en cuanto a la operación en ambientes industriales. A diferencia de un microcontrolador, con el cual también se puede realizar este sistema, un PLC presenta ciertas ventajas como, alto desempeño, robustez, fácil montaje y compatibilidad con equipos industriales.

La siguiente aplicación será realizada para monitorear la producción de las máquinas empacadoras en una línea de producción de frituras. Esta línea cuenta con diez máquinas empacadoras, las cuales son el punto final de la línea de producción.

En las siguientes páginas se darán a conocer los aspectos relevantes de la aplicación que sean necesarios para el diseño del sistema.

4.1. Diagrama esquemático de la aplicación

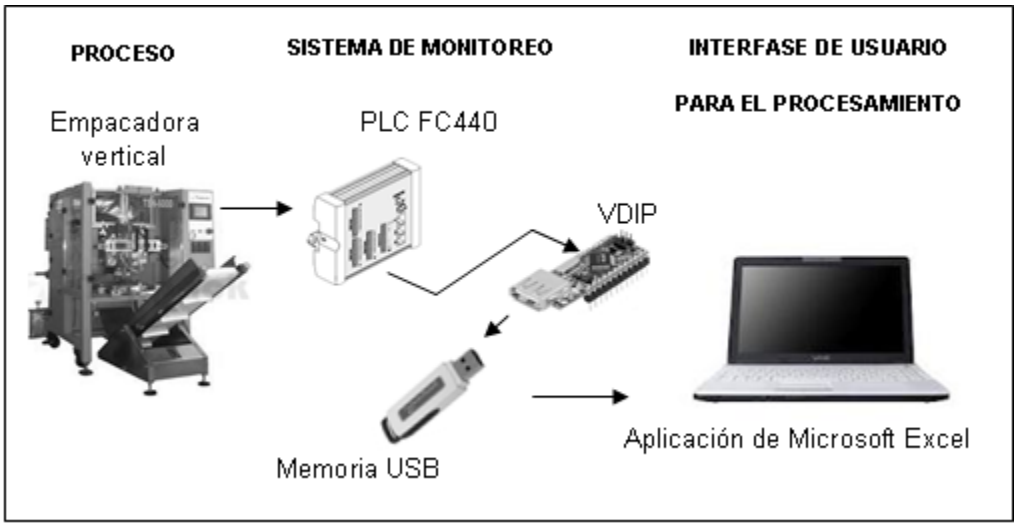
El sistema de monitoreo de producción obtiene la información de producción directamente de la máquina que se desea monitorear, a esta sección se le nombrará proceso.

La sección del sistema de monitoreo está constituida por tres componentes, el controlador programable FEC-FC440, el dispositivo anfitrión USB VDIP y la unidad de almacenamiento masivo. Esta se encarga de tomar, procesar y almacenar los datos de producción.

La última sección es la interface de usuario para el procesamiento de datos, la cual permite al analista de datos obtener de manera rápida, sencilla, ordenada y gráfica los datos más relevantes de producción, lo cual facilita el análisis de estos.

A continuación, en la figura 25 se muestra el diagrama esquemático de la aplicación del sistema de monitoreo de producción.

Figura 25. **Esquema de aplicación**

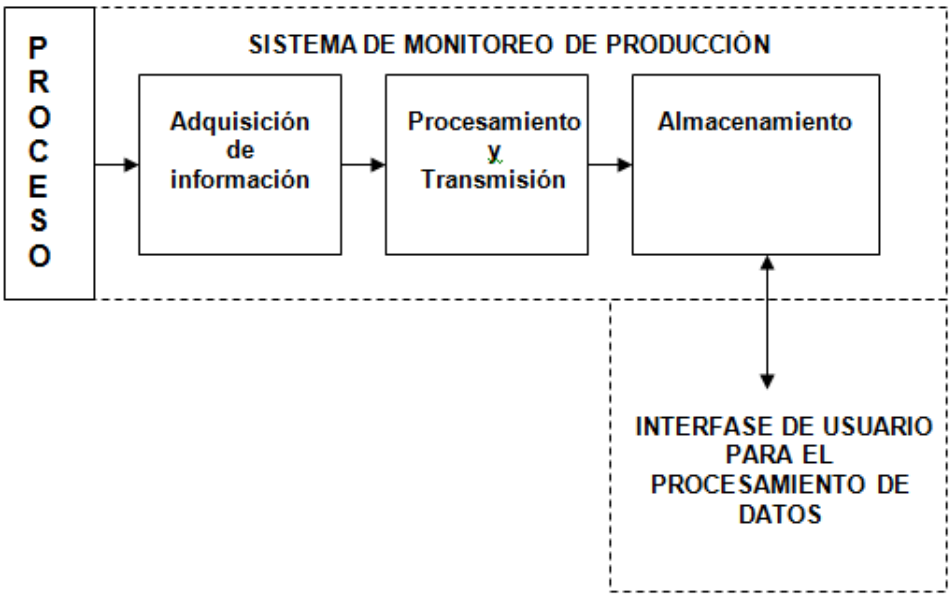


Fuente: elaboración propia.

4.2. Diagrama de bloques de la aplicación

La figura 26 muestra el diagrama de bloques de la aplicación. Este consta de tres bloques principales, proceso, sistema de monitoreo de producción e interface de usuario para el procesamiento de datos.

Figura 26. **Diagrama de bloques de sistema**



Fuente: elaboración propia.

4.2.1. Proceso

El sistema de monitoreo de producción se puede implementar en cualquier proceso del cual se pueda obtener una indicación discreta de las unidades producidas, la cual es fácil de obtener en la mayoría de procesos de producción.

Para esta aplicación se decidió monitorear la salida del proceso de empaque de frituras ya que es el último proceso apto para el monitoreo producción que se realiza antes del embalaje para distribución.

El sistema de monitoreo de producción presentado puede ser utilizado prácticamente en cualquier proceso, realizando pequeñas modificaciones en la etapa de adquisición de datos.

4.2.2. Sistema de monitoreo de producción

Como se observa en el diagrama de bloques mostrado en la figura 26 el sistema de monitoreo de producción se subdivide en tres secciones las cuales se exponen a continuación.

4.2.2.1. Adquisición de información

Lo primero que se debe resolver y el punto de inicio para el diseño del sistema es, ¿Qué información concerniente a la producción se quiere obtener? ¿Cómo se puede obtener esta información?, estas dos preguntas dependen totalmente de la aplicación. Abajo se listan algunos aspectos relacionados con la producción, que según la experiencia son de interés.

- Producción: se refiere a la cantidad de producto producido, este valor podría ser estimado por turno, día, mes, etcétera.

- Producción defectuosa: se refiere a la cantidad de producto que por alguna razón es rechazado.

- Tiempo de operación: se refiere al tiempo en el cual se ha producido la producción.

- Tiempo muerto: se refiere al tiempo que la máquina en producción permanece inactiva. Debido a paros, cambios de operarios, etcétera.
- Velocidad de producción: generalmente se conoce como la cantidad de golpes por minuto que una máquina realiza. La palabra golpe se refiere a un ciclo completo de máquina en el cual produce uno o varios productos.

Generalmente son de interés, la velocidad real (sin tomar en cuenta el producto defectuoso), la velocidad promedio, la velocidad máxima y la velocidad mínima de producción.

- Cantidad de paros: se refiere a la cantidad de paros de máquina realizados en producción.

Para obtener información del proceso es necesario obtener señales de retroalimentación. Los datos anteriores pueden ser obtenidos si se conoce, cuándo la máquina está activada, cuándo se encuentra detenida, una indicación de golpe de máquina y una indicación de producto defectuoso. Con estos datos en la siguiente etapa del sistema se puede obtener la información de producción deseada.

Generalmente, para obtener los datos antes mencionados se montan elementos de retroalimentación externos a la máquina, con el fin de no interferir en ningún momento operación normal de la máquina. Estos elementos son microswitch, sensores ópticos, inductivos, capacitivos, etcétera.

El principio de detección a utilizar depende de la aplicación. Si luego de un estudio de la máquina se encuentran puntos en los cuales se pueden tomar estas señales de retroalimentación (señal del controlador, sensores propios de la máquina, contactos auxiliares, etcétera) y no ponen en riesgo la operación de la máquina, se puede realizar de esta manera.

Para esta aplicación se requiere conocer, la producción, la producción defectuosa, la velocidad real de producción y la eficiencia de la producción.

Las señales de retroalimentación serán tomadas directamente de la máquina. Estas máquinas cuentan con una señal de descarga, la cual indica que se ha dosificado producto en el empaque y una señal de cierre, la cual indica que el empaque ha sido sellado. Una señal de descarga siempre es seguida de una señal de cierre pero una señal de sellado no siempre es precedida por una señal de descarga, en cuyo caso se trata de un producto defectuoso. De esta manera se puede realizar un conteo de unidades de producto producido y unidades de producto defectuoso, lo cual permite conocer la velocidad de producción de la máquina y la eficiencia de esta.

Estas señales son señales con niveles de voltaje TTL (0 y 5 voltios) y el controlador (PLC) únicamente trabaja con señales con niveles de voltaje HTL (0 y 24 voltios). Por lo anteriormente mencionado es necesario el diseño de un circuito de acoplamiento de señal que convierta la señal TTL a HTL. Se debe mencionar que estas máquinas alcanzan hasta doscientos golpes por minuto por lo cual el sistema de acoplamiento debe soportar esta frecuencia de activación.

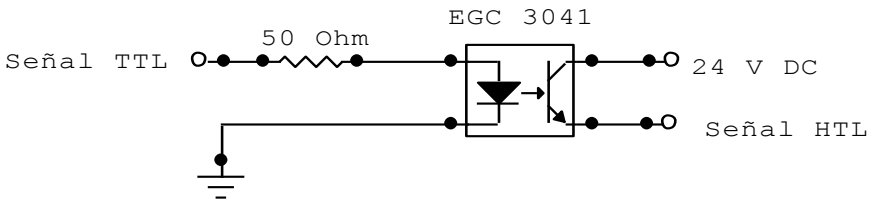
- Circuito de acoplamiento TTL-HTL

Debido a la frecuencia de activación deseada el uso de dispositivos de conmutación mecánica como lo son los relés electromecánicos resultaría inapropiado ya que el ciclo de activación es demasiado largo para conmutar a la frecuencia deseada. Debido a lo antes mencionado se utilizarán optoacopladores, específicamente el EGC 3 041, el cual permite operar a frecuencias tan altas como 9 000 golpes por minuto.

El optoacoplador está formado por un diodo emisor de luz LED y un fototransistor, al excitar el LED este emite una señal luminosa hacia el fototransistor el cual opera como una válvula de paso de corriente que al recibir

una señal luminosa permite el flujo de corriente entre la terminal colector y la terminal emisor. La figura 27 muestra el circuito de acoplamiento.

Figura 27. **Circuito de acoplamiento de señales TTL a HTL**



Fuente: elaboración propia.

4.2.2.2. Procesamiento y transmisión

Esta etapa del sistema es realizada por medio del controlador programable (PLC). Esta se puede dividir en tres subetapas, monitoreo, procesamiento y transmisión de la información. El controlador programable debe estar continuamente monitoreando las señales de retroalimentación de la máquina las cuales debe procesar periódicamente para obtener la información de producción requerida. Por último, debe transmitir por medio del puerto serial la información hacia la siguiente etapa para que esta sea almacenada por medio del dispositivo anfitrión USB.

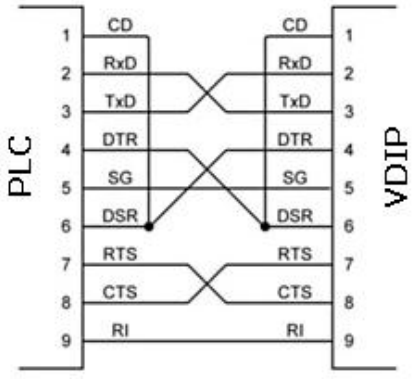
Para esta aplicación se utilizará un controlador FESTO FC400 el cual cuenta con un puerto serial (TTL). La frecuencia máxima de conmutación de las entradas es de 200 hertzios lo cual le permite operar en esta aplicación.

- Configuración del cable de comunicación serial

Para comunicar el controlador programable y el dispositivo anfitrión USB se puede utilizar una configuración de conexión NULL MODEM, en esta configuración las señales de recepción y transmisión de datos se encuentran cruzadas, es decir, la terminal de recepción de datos de un extremo conecta directamente con la señal de transmisión del otro y viceversa. Como se mencionó en el capítulo 2 dado que se comunicarán directamente dos equipos terminales de datos (DTE) se pueden obviar las señales de control de flujo de datos y trabajar únicamente con las señales de transmisión de datos TXD, RXD y referencia. Para ello, se tiene que asegurar que el puerto serial de los dos equipos esté configurado a la misma velocidad de transmisión. Para esta aplicación la velocidad de transmisión será 9 600 baudios.

La figura 28 muestra el diagrama esquemático del cable de comunicación serial a usar.

Figura 28. **Cable de comunicación serial NULL MODEM**



Fuente: elaboración propia.

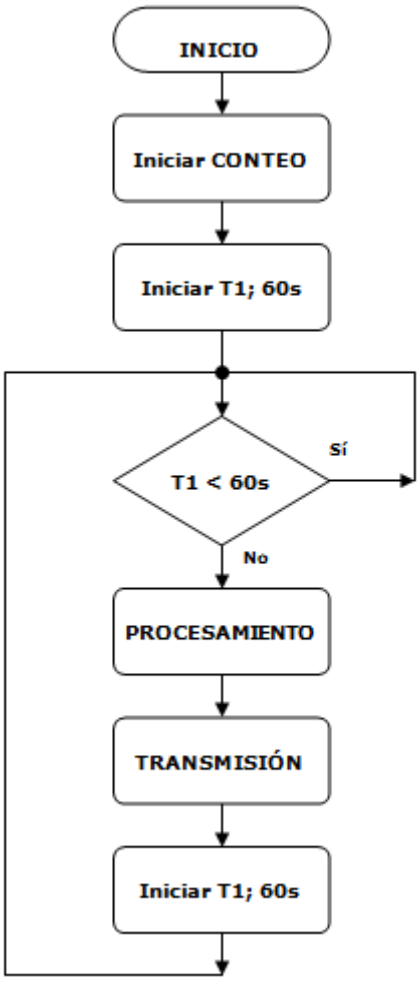
- Programa del Controlador Lógico Programable

El programa del controlador se encarga de tomar la información de las señales de descarga y cierre de la máquina, procesarlas y transmitir la información de producción y comandos necesarios hacia el dispositivo anfitrión USB VDIP.

Para esta aplicación se utilizará un único directorio y se procederá a crear un archivo mensual de producción. Este contendrá las unidades producidas, las unidades de producto defectuoso y la velocidad real de producción de cada minuto de operación. También se debe incluir la hora, fecha y turno al cual corresponde cada dato adquirido.

A continuación se presenta el programa del controlador en forma de diagrama de flujo de manera que este sea de fácil entendimiento, sin importar cual sea el lenguaje de programación dominado por el lector o si incluso si este no está familiarizado con la programación. En el apéndice podrá encontrar el programa realizado para el controlador FESTO FC400.

Figura 29. **Rutina principal del programa**



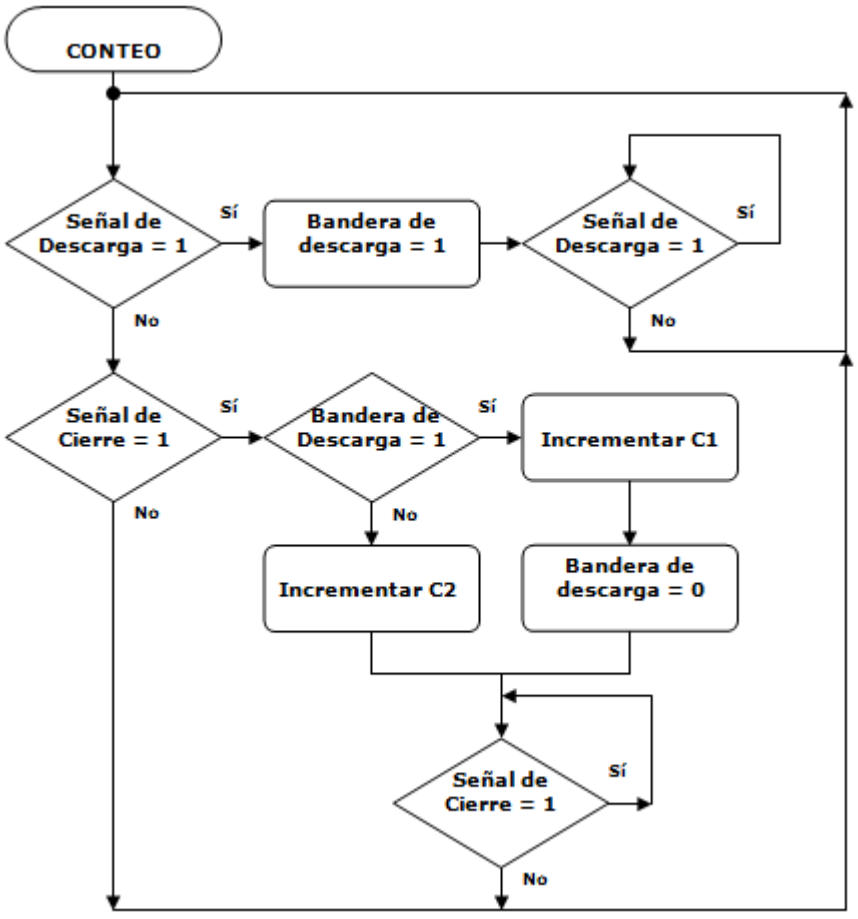
Fuente: elaboración propia.

La figura 29 muestra la rutina principal del programa, primeramente se inicia la subrutina conteo que se encarga de monitorear las señales de indicación de descarga y cierre de la máquina. Esta subrutina permanece activa paralelamente a la rutina principal.

A continuación se inicia el conteo de tiempo a través del temporizador T1, el cual define el ciclo de lectura de datos recopilados en la secuencia conteo, en este caso de 60 segundos.

Luego de esto se ejecutan cíclicamente las rutinas de procesamiento y transmisión. El intervalo de ejecución es de 60 segundos.

Figura 30. **Subrutina de conteo**



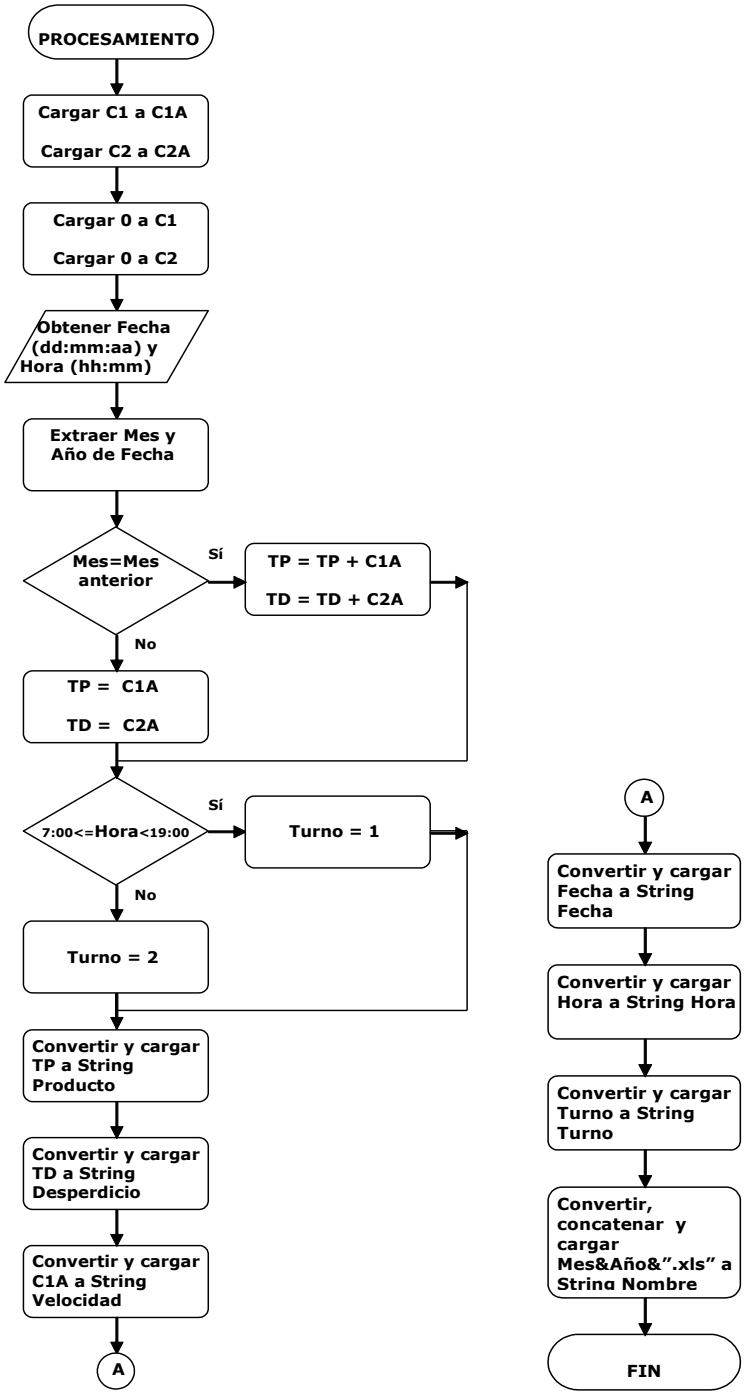
Fuente: elaboración propia.

La figura 30 muestra la subrutina conteo, como se mencionó anteriormente esta subrutina continua corriendo paralelamente a la rutina principal dado que esta se encarga de monitorear las señales de cierre y descarga de la máquina empacadora.

La subrutina espera por la activación de alguna de las dos señales cierre o descarga en un ciclo permanente. Para esta aplicación, como se mencionó anteriormente, una señal de descarga siempre es seguida de una señal de cierre, pero una señal de cierre si puede darse sin que se haya dado una señal de descarga, lo cual se define como producto defectuoso.

Si se activa la señal de descarga se activa la bandera o marca de descarga y se retorna al ciclo de espera al desactivarse la señal de descarga. Si se activa la señal de cierre y la bandera o marca de descarga está activa, se aumenta el contador C1, el cual almacena la cantidad de producto producido. Luego se procede a desactivar la bandera o marca de descarga y se reinicia el ciclo si se ha desactivado la señal de cierre. Si se activa la señal de cierre y no está activa la bandera o marca de descarga, lo cual indica que no ha sucedido descarga de producto, se procede a aumentar el contador C2, el cual almacena la cantidad de producto defectuoso producido. Luego si se desactiva la señal de cierre se reinicia el ciclo.

Figura 31. **Subrutina de procesamiento**



Fuente: elaboración propia.

La figura 31 muestra la subrutina de procesamiento, esta subrutina se encarga de tomar los datos adquiridos por la subrutina Conteo y procesarlos de manera que la subrutina de transmisión pueda enviar los datos al VDIP.

Lo primero que se realiza en esta rutina es la carga de los contadores de producto y producto defectuoso C1 y C2 respectivamente a contadores auxiliares C1A y C2A respectivamente. Inmediatamente después se procede a reiniciar el valor de los contadores C1 y C2 para que tomen los datos del siguiente minuto. Luego procede a consultar la fecha y la hora del controlador para saber a qué hora corresponden estos datos.

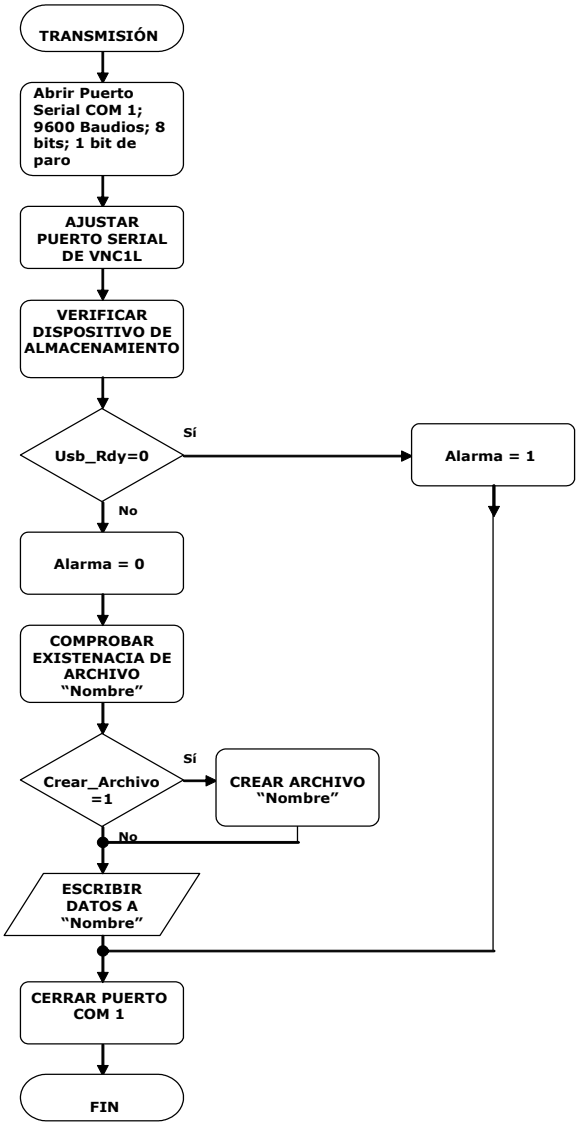
Si el mes de la fecha corresponde con el mes de la fecha anterior se procede a acumular el valor de C1A y C2A en las variables TP (Total producido) y TD (Total defectuoso) respectivamente y, si no corresponden las fechas se procede a reiniciar el valor de TP y TD con los valores actuales de C1A y C2A respectivamente.

Seguidamente se evalúa a qué turno pertenece el dato de producción dependiendo de la hora a la que fue tomado el dato, de 7:01 a 19:00 horas corresponde al turno 1 y de 19:01 a 7:00 horas del siguiente día corresponde al turno 2.

De último se procede convertir y almacenar los datos TP, TD, velocidad (en este caso CA1, unidades por minuto), fecha, hora y turno a strings, para que estas puedan ser enviadas luego al VDIP por medio del puerto serial del controlador.

Antes de terminar la subrutina se crea el nombre del archivo concatenando el mes, el año y la extensión de los archivos de excel “.xls”. Por ejemplo, si el mes es enero y el año 2011 el nombre del archivo sería “enero2011.xls”.

Figura 32. **Subrutina de transmisión**



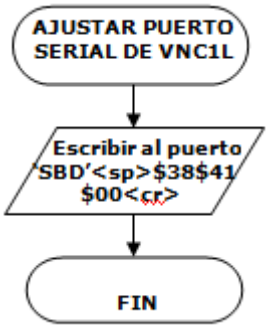
Fuente: elaboración propia.

La figura 32 muestra la subrutina de transmisión, esta se encarga de transmitir los datos y comandos necesarios para que el dispositivo anfitrión USB almacene los datos dentro del dispositivo de almacenamiento USB.

Al iniciar la subrutina se abre y configura el puerto de comunicación serial a usar. Lo cual se debe realizar con la mayoría de controladores programables. Para esta aplicación se ajustará a 9 600 baudios, configurado para 8 bits y 1 bit de paro ya que es la configuración predeterminada en puerto de comunicación serial del VDIP.

La figura 33 muestra la rutina auxiliar Ajustar Puerto Serial VNC1L, esta se encarga de enviar el comando ‘SBD’ el cual se encarga de configurar la velocidad del puerto serial como se explica más adelante.

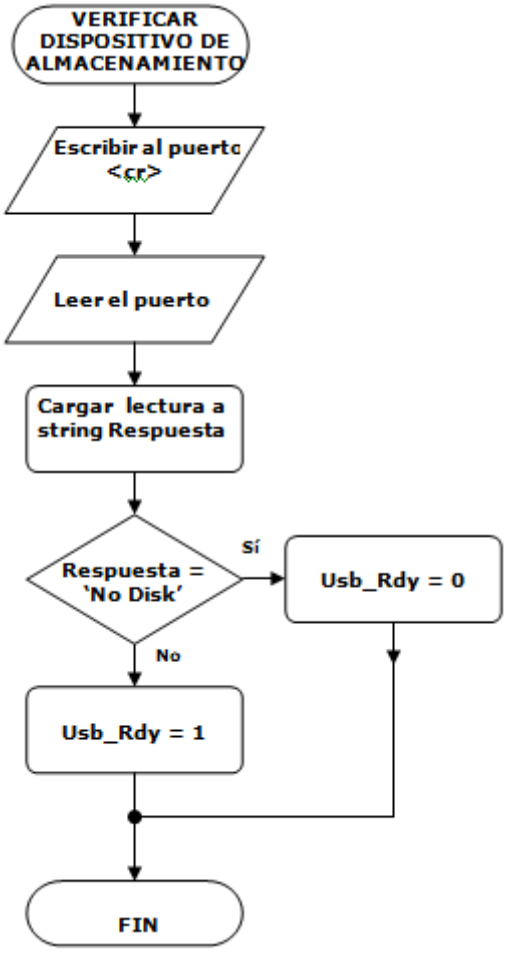
Figura 33. **Rutina auxiliar para ajustar puerto serial VNC1L**



Fuente: elaboración propia.

Luego de esto se procede a verificar si el dispositivo de almacenamiento masivo se encuentra conectado al VDIP, si este se encuentra conectado y listo, se prosigue con la subrutina, de lo contrario se activa una alarma y se cierra el puerto del controlador y se termina la rutina, de manera que los datos de ese minuto se pierden. La figura 34 muestra la rutina auxiliar para verificar dispositivo de almacenamiento. Esta rutina envía el comando <cr> al VDIP el cual devuelve No disk si no hay un dispositivo de almacenamiento conectado al VDIP o “D:\\” si lo hay. Dependiendo del dato recibido se activa o desactiva la bandera Usb_Rdy.

Figura 34. **Rutina auxiliar para verificar dispositivo de almacenamiento**

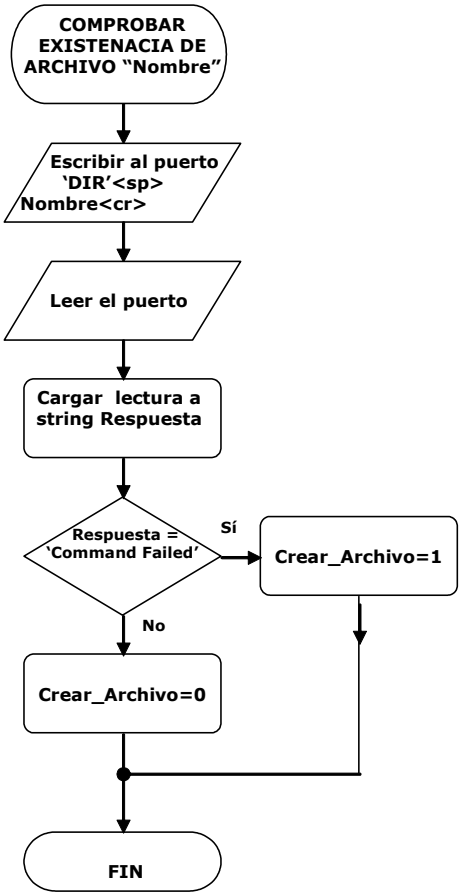


Fuente: elaboración propia.

Si el dispositivo de almacenamiento está listo, se procede a desactivar la alarma, en dado caso hubiese sido activada en el ciclo anterior. Luego se procede a comprobar la existencia del archivo de almacenamiento Nombre. Dado que el nombre del archivo está relacionado con el mes en que se está tomando el dato, cuando se realice un cambio de mes, la rutina auxiliar Comprobar Existencia de Archivo Nombre no podrá encontrar el archivo activando la bandera o marca crear archivo. Esto permite la creación de un archivo mensual. La figura 35 muestra la rutina auxiliar Comprobar Existencia de Archivo Nombre. Esta envía el comando 'DIR' en conjunto con el nombre del archivo.

Si el archivo no existe el VDIP retorna 'Command Failed'. Por lo tanto, si se obtiene esta respuesta se procede a activar la bandera o marca Crear_Archivo.

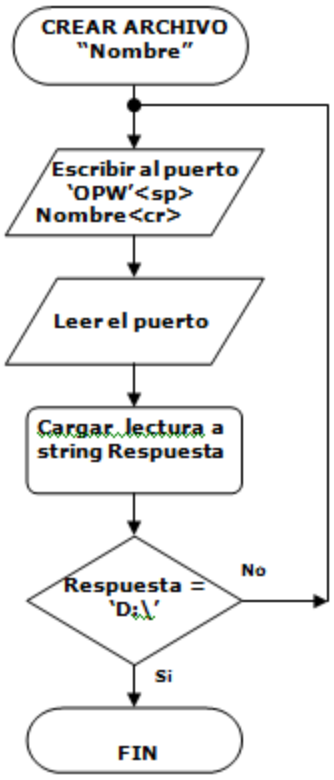
Figura 35. **Rutina auxiliar para comprobar existencia de archivo**



Fuente: elaboración propia.

Si se activó la bandera Crear_Archivo. Se corre la rutina auxiliar Crear Archivo Nombre, que se muestra en la figura 36. Esta envía el comando OPW seguido del nombre del archivo a crear al VDIP. Este comando abre o crea un archivo de texto para escritura. Si el comando se ejecutó con éxito el VDIP devuelve "D:\\".

Figura 36. **Rutina auxiliar para crear archivo nombre**



Fuente: elaboración propia.

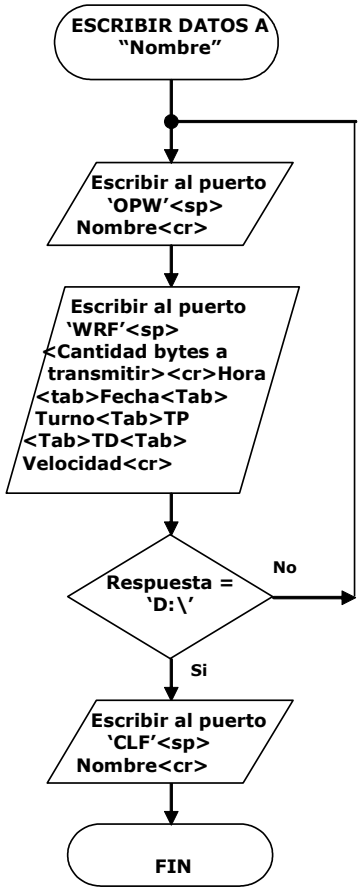
La última rutina auxiliar dentro de la subrutina transmisión se encarga de escribir los datos dentro del controlador, para lo cual primeramente abre el archivo con el comando, ‘OPW’ y luego escribe los datos con el comando ‘WRF’. Este se envía acompañado de la cantidad de bytes que se enviarán y los strings que se desean escribir.

Ya que se desea que estos archivos sean desplegados en un archivo de excel, es importante notar que inmediatamente después de cada dato se escribe un tabulador y al finalizar la fila de datos se envía un retorno de carro para dejar el cursor en la siguiente línea para escribir los siguientes datos. La figura 37 muestra la rutina auxiliar Escribir Datos a Nombre.

Luego de esta rutina auxiliar la subrutina procede a cerrar el puerto quedan así finalizadas.

Al finalizar la subrutina de transmisión, la rutina principal reinicia el temporizador T1 para luego iniciar de nuevo el ciclo.

Figura 37. **Rutina auxiliar para escribir datos a nombre**



Fuente: elaboración propia.

4.2.2.3. Almacenamiento

Para almacenar los datos obtenidos anteriormente, dentro del dispositivo de almacenamiento USB se utilizó el anfitrión USB VDIP1. Este se encarga de interpretar los comandos enviados por el controlador, ejecutar la acción solicitada y emitir una respuesta o indicación del éxito en la ejecución de la acción solicitada. El firmware VDAP del VDIP permite a través de una serie de comandos sencillos la escritura y lectura de datos de un dispositivo de almacenamiento USB.

- Comandos VDAP aplicados

Consulta de dispositivo disponible <cr>: <cr> corresponde al carácter ASCII $0D el cual es conocido como retorno de carro. Al enviar este comando el VDIP chequea si existe algún dispositivo USB conectado al puerto. Si lo hay devuelve el directorio raíz 'D:\' seguido de un retorno de carro $0D, de lo contrario devuelve 'No disk' seguido de $0D.

Consulta de tamaño de archivo 'DIR'<sp><nombre><cr>: <sp> corresponde al carácter ASCII $20 el cual es conocido como espacio. Al enviar este comando el VDIP busca el archivo "nombre" dentro del directorio actual. Si lo encuentra retorna el tamaño del archivo seguido de un retorno de carro $0D, de lo contrario devuelve 'Command Failed' seguido de un retorno de carro $0D.

Configurar velocidad de UART 'SBD'<sp><divisor (3 bytes) primero el LSB t ><cr>: al enviar este comando el VDIP configura la velocidad de transmisión del puerto según el divisor enviado, la tabla XVII muestra los distintos divisores y velocidades posibles. La velocidad predeterminada de fábrica es de 9 600 baudios.

Tabla XVII. **Velocidades de transmisión del VDIP**

Baudios	1er. Byte	2do. Byte	3er. Byte
300	$10	$27	$00
600	$88	$13	$00
1200	$C4	$09	$00
2400	$E2	$04	$00
4800	$71	$02	$00
9600	$38	$41	$00
19200	$9C	$80	$00
38400	$4E	$C0	$00
57600	$34	$C0	$00
115200	$1A	$00	$00
230400	$0D	$00	$00
460800	$06	$40	$00
921600	$03	$80	$00
1000000	$03	$00	$00
1500000	$02	$01	$00
2000000	$01	$02	$00
3000000	$00	$03	$00

Fuente: Future Technology Devices International Ltd. VDAP Firmware Manual. p. 8.

Crear o abrir archivo Nombre.xls ‘OPW’<sp>Nombre.xls<cr>: al enviar este comando el VDIP abre el archivo Nombre.xls para escritura, si no existe el archivo este se encarga de crearlo.

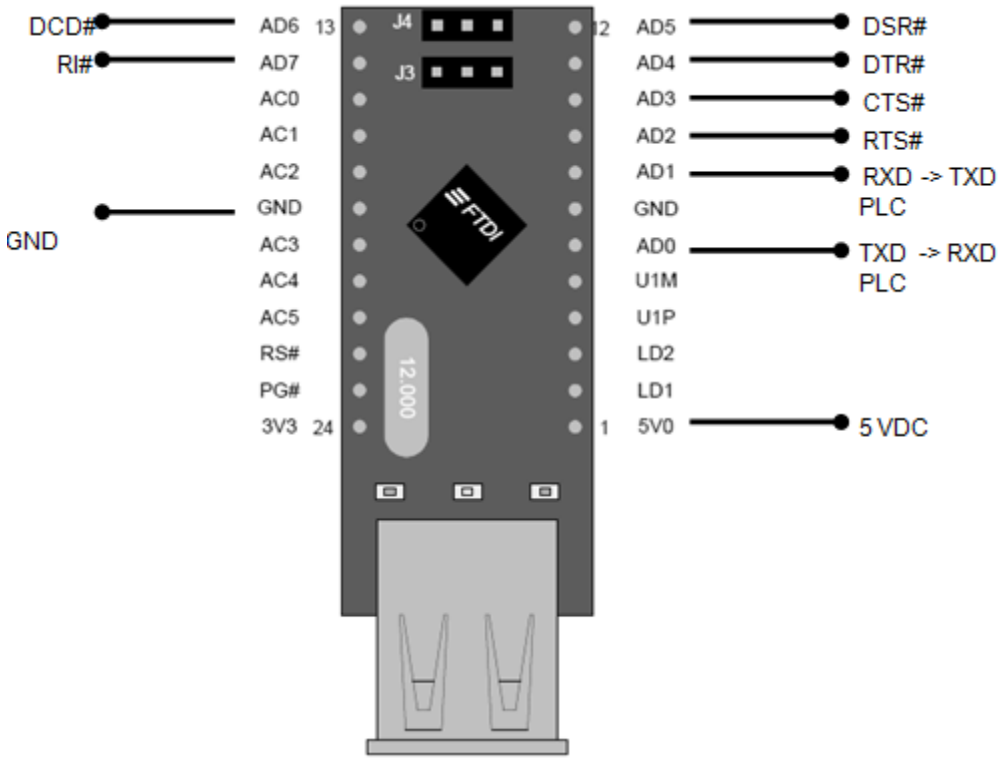
Escribir al archivo ‘WRF’<sp> <tamaño en hex(4 bytes) ><cr> < bytes de datos de tamaño><cr>: al enviar este comando el VDIP escribe los datos en el archivo abierto para escritura. Los datos son escritos al final del archivo.

Cerrar archivo ‘CLF’<sp>Nombre.xls<cr>: al enviar este comando el VDIP cierra el archivo Nombre.xls. Si se desea volver a escribir al archivo será necesario abrirlo de nuevo.

- Esquema de conexión del VDIP

En la figura 38 se muestra el esquema de conexión del VDIP.

Figura 38. **Esquema de conexión del VDIP**



Fuente: elaboración propia.

El VDIP cuenta con terminales para montaje en placa impresa, lo cual permite insertarlo en un tablero de pruebas fácilmente.

Los puentes de J1 y J2 permiten seleccionar la interface de comunicación deseada.

4.2.2.4. Interface de usuario

Como herramienta para el análisis de los datos obtenidos por el sistema de monitoreo de producción se presenta una herramienta desarrollada en Microsoft Excel.

Microsoft Excel es un software para el manejo de hojas de cálculo electrónicas ampliamente conocido que usualmente forma parte del paquete básico de Microsoft Office el cual se encuentra instalado en la mayoría de computadoras de la industria. Este software cuenta con una herramienta de desarrollo de aplicaciones basada en el lenguaje de programación Visual Basic, lo cual le brinda un gran potencial y adaptabilidad para el uso en aplicaciones de ingeniería.

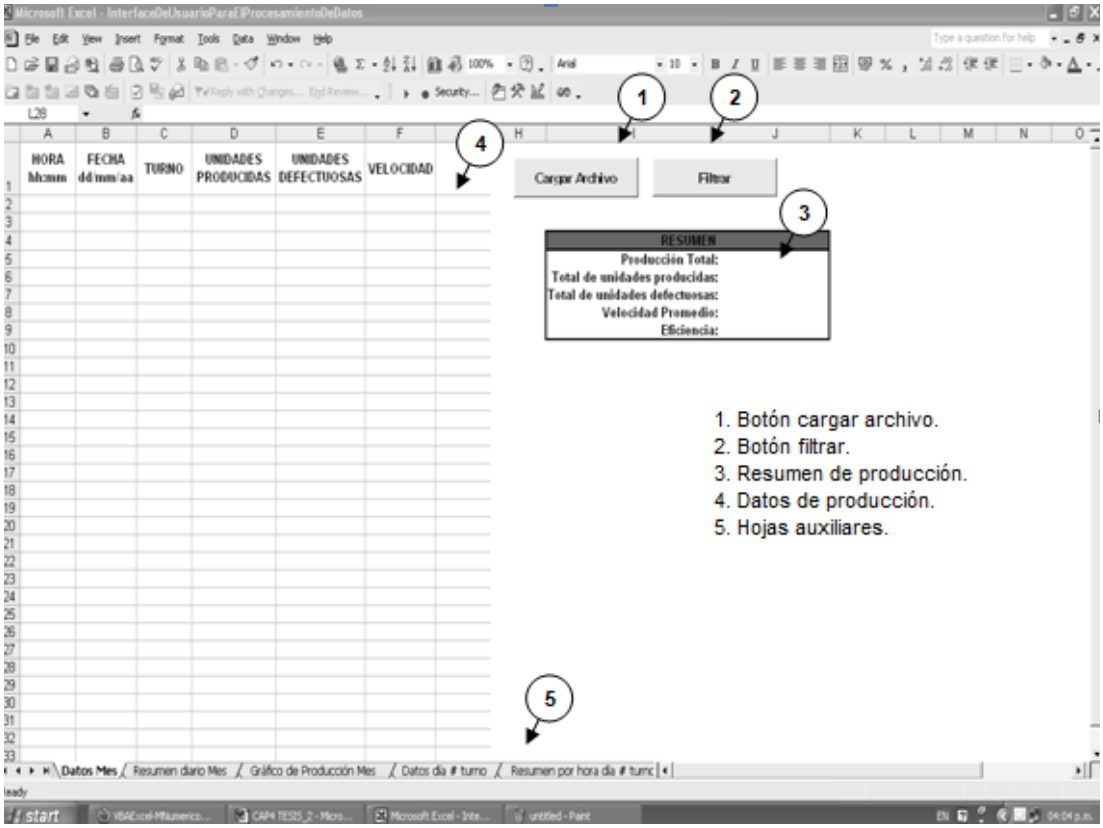
La siguiente herramienta se piensa como una forma sencilla y rápida para realizar un análisis del comportamiento de la producción de la máquina. Los datos y forma de análisis presentados por la herramienta van en función de las necesidades definidas para esta aplicación, sin embargo, esta herramienta puede ser fácilmente adaptada a otras necesidades.

A continuación, se presenta la descripción de la herramienta, en el apéndice se podrá encontrar la aplicación desarrollada.

- Descripción de interface de usuario

La herramienta cuenta con dos botones de operación como se muestra en la figura 39.

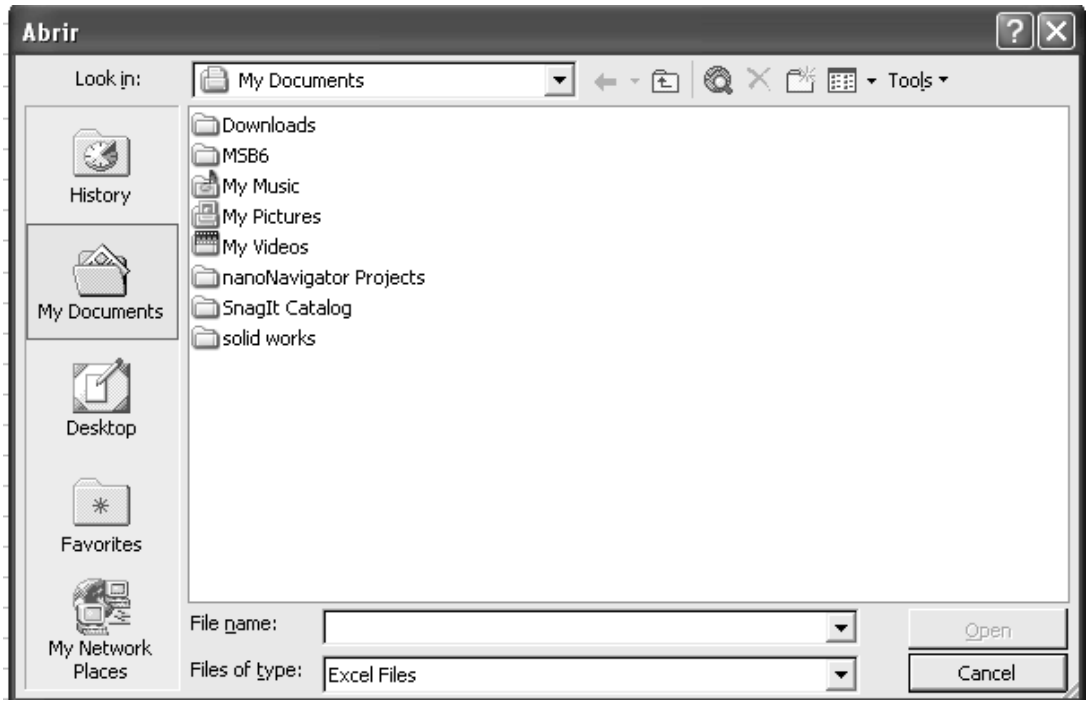
Figura 39. **Herramienta de procesamiento de datos**



Fuente: elaboración propia.

Botón cargar archivo: este permite buscar y cargar el archivo de mes que se desea procesar. Al presionarlo se despliega un explorador de archivos como se muestra en la figura 40. Al abrir el archivo, la herramienta procesará automáticamente los datos mostrando el resumen de producción en la hoja de datos de mes, el resumen diario en la hoja resumen diario del mes y la gráfica de producción del archivo del mes seleccionado en la hoja de gráfico de producción del mes.

Figura 40. **Explorador de archivos**



Fuente: elaboración propia.

La hoja de datos de mes muestra los siguientes campos:

- Hora: la cual indica a qué momento corresponden los datos.
- Fecha: la cual indica a qué día del mes corresponden los datos.
- Turno: para esta aplicación se dividirá la jornada de producción en dos turnos, el turno 1 corresponde al turno de 7:00 a 19:00 horas y el turno 2 corresponde al turno de 19:00 a 7:00 horas del día siguiente.
- Unidades producidas: indica el total de unidades producidas hasta ese momento.
- Unidades defectuosas: indica el total de unidades defectuosas producidas hasta ese momento.
- Velocidad: indica la velocidad de producción real de la máquina durante ese minuto. La indicación se da en unidades por minuto.

Figura 41. **Campos mostrados en la hoja de datos del mes**

	A	B	C	D	E	F
1	**HORA hh:mm**	**FECHA dd/mm/aa**	**TURNO**	**UNIDADES PRODUCIDAS**	**UNIDADES DEFECTUOSAS**	**VELOCIDAD**
2	00:01	01/09/2011	2	0	2	98
3	00:02	01/09/2011	2	98	2	100
4	00:03	01/09/2011	2	198	2	100
5	00:04	01/09/2011	2	298	64	38
6	00:05	01/09/2011	2	336	66	98
7	00:06	01/09/2011	2	434	67	99

Fuente: elaboración propia.

En esta hoja también se muestra el cuadro de resumen de producción, mostrado en la figura 42, el cual muestra los siguientes datos:

- Producción total: la cual es la suma de las unidades producidas y las unidades defectuosas.

- Total de unidades producidas: la cual indica el total de producto correctamente producido.

- Total de unidades defectuosas: la cual indica el total de producto defectuoso.

- Velocidad promedio: la cual indica la velocidad real promedio, en unidades por minuto, registrada durante el mes procesado.

- Eficiencia: la cual indica la relación entre la producción total y el total de unidades producidas.

Figura 42. **Cuadro de resumen de producción**

RESUMEN		
Producción Total:	3,991,743	Unidades
Total de unidades producidas:	3,931,200	Unidades
Total de unidades defectuosas:	60,543	Unidades
Velocidad Promedio:	91	Unidades/min
Eficiencia:	98.48	%

Fuente: elaboración propia.

La hoja de resumen diario muestra los siguientes campos:

- Fecha: la cual indica a qué día del mes corresponden los datos.
- Unidades producidas: indica el total de unidades producidas este día.
- Unidades defectuosas: indica el total de unidades defectuosas producidas este día.
- Velocidad: indica la velocidad de producción real de la máquina durante ese día. La indicación se da en unidades por minuto.

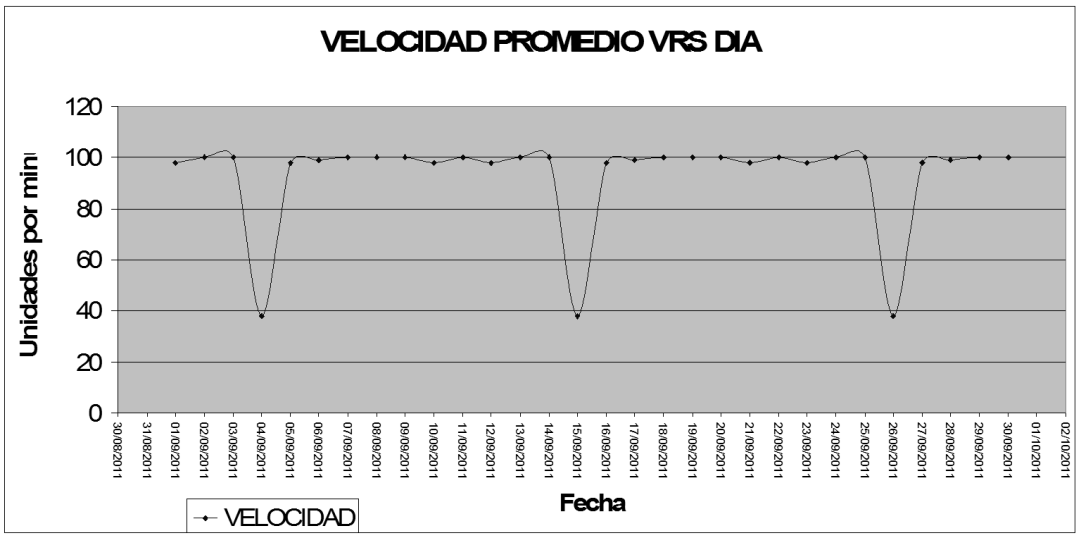
Figura 43. **Campos mostrados en la hoja de resumen diario**

	A	B	C	D
1	FECHA dd/mm/aa	UNIDADES PRODUCIDAS	UNIDADES DEFECTUOSAS	VELOCIDAD
2	01/09/2011	128,800	2,000	98
3	02/09/2011	127,596	1,999	100
4	03/09/2011	129,300	3,989	100
5	04/09/2011	128,800	1,200	38

Fuente: elaboración propia.

La hoja de gráfico de producción muestra en forma gráfica los datos tabulados en el resumen diario, como se muestra en la figura 44.

Figura 44. **Gráfico de producción**



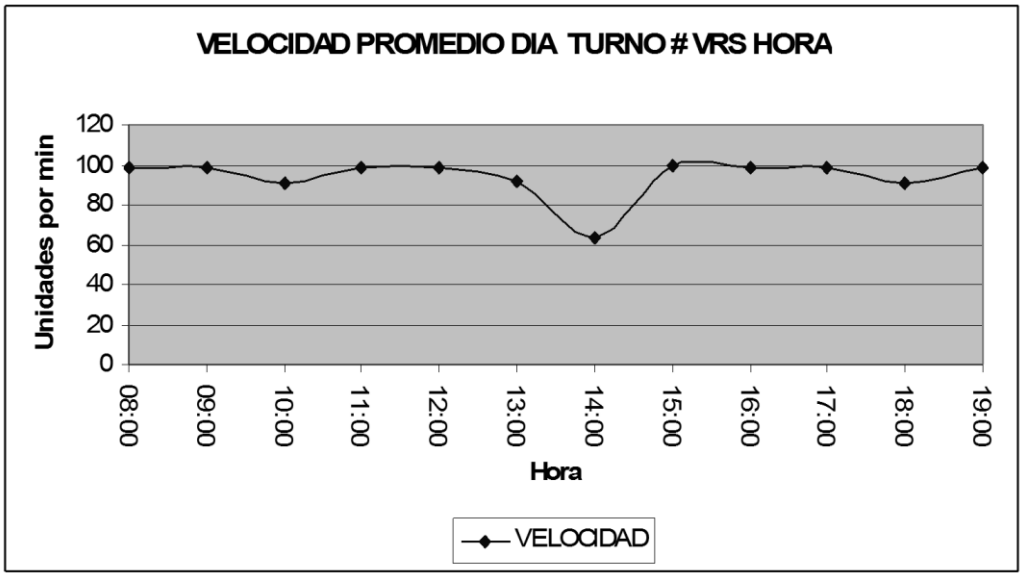
Fuente: elaboración propia.

Botón filtrar: este despliega una ventana la cual permite seleccionar el día y el turno a analizar como se muestra en la figura 45. Al presionar filtrar la información de este turno es cargada en la hoja de datos del día y turno seleccionado. Además un resumen por hora del turno se muestra en la hoja de resumen por hora. La hoja de gráfico de producción muestra de forma gráfica los datos tabulados en el resumen por hora como se observa en la figura 46.

Figura 45. **Ventana filtrar**

Fuente: elaboración propia.

Figura 46. **Gráfico de producción por turno**



Fuente: elaboración propia.

La hoja de datos de día y de turno muestra el cuadro de resumen de producción del turno, como se muestra en la figura 47, el cual muestra los siguientes datos:

- Producción total: es la suma de las unidades producidas y las unidades defectuosas en el turno.

- Total de unidades producidas: indica el total de producto correctamente producido en el turno.

- Total de unidades defectuosas: indica el total de producto defectuoso en el turno.

- Velocidad promedio: indica la velocidad real promedio, en unidades por minuto, registrada durante el turno.

- Eficiencia: indica la relación entre la producción total y el total de unidades producidas en el turno.

Figura 47. **Cuadro de resumen de producción de turno**

RESUMEN día TURNO #		
Producción Total:	68,520	Unidades
Total de unidades producidas:	65,520	Unidades
Total de unidades defectuosas:	3,000	Unidades
Velocidad Promedio:	91	Unidades/min
Eficiencia:	95.62	%

Fuente: elaboración propia.

Como se puede observar esta herramienta permite extraer de forma fácil y sencilla la información de los datos adquiridos por el sistema de monitoreo de producción mostrando la información de manera ordenada facilitando el análisis de los datos.

4.3. Costo del sistema de monitoreo de producción

Para el desarrollo de esta aplicación se tomaron en cuenta dos tipos de hardware diferentes, los cuales tienen diferentes prestaciones y costos. En la tabla XVIII se muestra el costo de implementación utilizando el controlador FESTO FEC -FC400.

Tabla XVIII. **Costo del sistema con controlador FESTO FC-400**

Cantidad	Descripción	Precio unitario Q	Precio total Q
Servicios			
4	Instalación (hora)	275,00	1 100,00
16	Programación (hora)	275,00	4 400,00
1	Puesta en marcha (hora)	275,00	275,00
Equipos			
1	Controlador FESTO FC-440	6 008,66	6 008,66
1	Dispositivo anfitrión USB VDIP	182,40	182,40
		TOTAL	11 966,06

Fuente: elaboración propia.

En la tabla XIX se muestra el costo de implementación utilizando el controlador Phoenix Contact NLC-050.

Tabla XIX. **Costo del sistema con controlador Phoenix Contact NLC-050**

Cantidad	Descripción	Precio unitario Q	Precio total Q
Servicios			
4	Instalación (hora)	275,00	1 100,00
16	Programación (hora)	275,00	4 400,00
1	Puesta en marcha (hora)	275,00	275,00
Equipos			
1	Controlador nano NLC-050	1 883,70	1 883,70
1	Modulo RS-232 para NLC-050	627,90	627,90
1	Dispositivo anfitrión USB VDIP	182,40	182,40
		TOTAL	8 469,00

Fuente: elaboración propia.

CONCLUSIONES

1. El sistema de monitoreo de producción desarrollado es una solución económicamente viable para la pequeña industria guatemalteca.

2. Es posible disminuir el costo de implementación del sistema de monitoreo de producción, utilizando el hardware de control propio de la línea de producción a monitorear que cuente con una interface de comunicación serial.

3. El sistema de monitoreo de producción presentado es un sistema flexible el cual puede ser modificado para obtener más información de producción como son los tiempos muertos, paros de emergencias e incluso llevar un registro de la razón de los paros.

4. Este sistema de monitoreo de producción puede ser implementado en cualquier máquina en la cual pueda ser adquirida una señal, que indique la producción de una unidad o lote de producto.

5. El dispositivo anfitrión VDIP de FTDI es un dispositivo de fácil configuración y manejo, que permite manipular, crear y eliminar el contenido del dispositivo de almacenamiento USB a través de comandos transmitidos en forma serial.

6. MS Excel, es una herramienta versátil que permite desarrollar diversas aplicaciones especializadas de interacción con el usuario de manera económica sencilla y rápida.

RECOMENDACIONES

1. Utilizar como hardware de adquisición de datos un controlador de gama baja como el NLC-055 de Phoenix Contact u otro similar.

2. Tomar en cuenta la frecuencia máxima de conmutación de las entradas del controlador usado para la adquisición de datos. La velocidad de producción (unidades por segundo) debe ser menor a este valor.

3. Utilizar controladores con entradas rápidas para realizar el monitoreo de líneas de producción en las cuales la velocidad de producción pasa de las 10 unidades por segundo.

BIBLIOGRAFÍA

1. AXELON, Jan. *USB Complete: everything you need to develop USB Peripherals*. 3rd ed. USA: Lake View, 2005. 572 p.

2. BRYAN, L. *Programable Controllers Theory and Implementation*. 2nd ed. USA: Industrial Text Company, 1997. 506 p.

3. Festo AG. *FST Manual 525369*. Alemania: Festo AG, 2005. 115 p.

4. Future Technology Devices International. *Viniculum Firmware User Manual*. United Kingdom: Future Technology Devices International, 2008. 78 p.

5. HYDE, John. *USB Design by Example*. USA: Intel, 1999. 348 p.

APÉNDICES

Apéndice 1. **Programa realizado para controlador FEC FC440**

```
FST - MONITOR (Sistema de monnitoreo de producción) - FEC Standard
Controller Settings
------------------------------------------------------------------------------

Run Mode

   Autostart:                          NO
   Run/Stop Input:                     NO
                                       0.0
   Stop Program:                       0
   Reset Programs:                     YES

   Error Output:                       NO
                                       0.0
   Error Program:                      0
   Reset Outputs:                      YES


Drives

   Project Drive:                      B
   Start-up Drive:                     B
   Kernel Drive:                       A


CI Options

   Controller CI port:                  internal COM port
   Special Parameters:                 <none>


Password

   Password:                           <none>


Download

   Download Sources:                   NO
   Download Modified Driver Files:     YES
   Delete Project before Download:     NO
   Stop Programs during Download:      NO
```

Continuación del apéndice 1.

```
PST - MONITOR (Sistema de monnitoreo de producción) - FEC Standard
PST Project
```

Programs

* Program 0 (V1) - Rutina Principal
* Program 1 (V1) - Subrutina de Conteo
* Program 2 (V1) - Subrutina de Procesamiento
* Program 3 (V1) - Subrutina de Transmisión
* Program 4 (V1) - Rutina Auxiliar - Ajustar puerto serie de VDIP
* Program 5 (V1) - Rutina Auxiliar - Verificar memoria USB
* Program 6 (V1) - Rutina Auxiliar - Comprobar existenca de archivo
* Program 7 (V1) - Rutina Auxiliar - Crear Archivo
* Program 8 (V1) - Rutina Auxiliar - Escribir datos

CMPs

* CMP 0 (V1) - F13 - Interrogar FECHA
* CMP 1 (V1) - F12 - Interrogar HORA
* CMP 2 (V1) - FPI2F - Conversion de entero a foltante
* CMP 3 (V1) - FPF2A - Conversión de flotante a string
* CMP 4 (V1) - STRAPPND - Agregar un caracter al string
* CMP 5 (V1) - STRCPY - Copiara un string a otro
* CMP 6 (V1) - STRCAT- Concatenar dos strings
* CMP 7 (V1) - OPENCOM - Abrir el puerto serial con 9600 baud, 8 bits, no p
* CMP 8 (V1) - CLOSECOM - Cerrar puerto COM
* CMP 9 (V1) - PRINTCOM - Escribir al puerto serie delimitador <CR>
* CMP 10 (V1) - PUTCOM - Escribir caracter ASCII al puerto serie
* CMP 11 (V1) - READCOM - Leer puerto serie
* CMP 12 (V1) - STRICMP - Compara dos strings 0 = iguales
* CMP 13 (V1) - STRCLR - Borrar string

CFMs

* CFM 0 (V1) - Inicio

Continuación del apéndice 1.

FST - MONITOR (Sistema de monnitoreo de producción) - FBC Standard
IO Configuration

IO Module	Switch	IW	OW
PC440	0	0	0

Continuación del apéndice 1.

```
PST - MONITOR (Sistema de monnitoreo de producción) - FEC Standard
Driver Configuration
--------------------------------------------------------------------------------
Name                Number    Uses      Description

COMEXT              8         -         Serial Communication
FPMATHDR            46        -         FpMath Driver
STRINGS             3         -         String Datatype


Options for 'COMEXT'

  <none>


Options for 'FPMATHDR'

  <none>


Options for 'STRINGS'

  Reserved Memory in Bytes:             2000
  Number of Strings:                    256
  Initial Value File [.TXT]:            StrInit
  DOS Character Encoding:               N
```

Continuación del apéndice 1.

FST - MONITOR (Sistema de monnitoreo de producción) - FEC Standard
Allocation List

Operand	Symbol	Comment
O0.0	Alarma	Alarma de fallo en memoria
OW0		
I0.0	Descarga	Señal de descarga
I0.1	Cierre	Señal de cierre
F0.0	F_Proc	Bandera - Procesamiento realizado
F0.1	F_trans	Bandera - Transmision realizada
F0.2	F_Decarga	Bandera - Descarga realizada
F0.3	F_ajustar	Puerto de VDIP ajustado
F0.4	F_verif	Memoria USB verificada
F0.5	Usb_Rdy	Memoria USB lista
F0.6	F_Comp	Existensia de archivo comprobada
F0.7	Crear_A	Crear archivo
F0.8	F_Creado	Archivo creado
F0.9	F_Escrito	Datos escritos
FW0		
FW100	AUX	Bandera Auxiliar
FU32		
FU33		
FU34		
P1		Rutina de conteo
P2		Rutina de procesamiento
P3		Rutina de transmisión
P4		
P5		
P6		
P7		
P8		
R0	C1A	Auxiliar Conatdor de producto bueno
R1	C2A	Auxiliar Contador de producto malo
R2	año	
R3	mes	
R4	dia	
R5	mes_ant	mes del dato antes almacenado
R6	hora	
R7	minuto	
R8	TPB	Total de producto bueno
R9	TD	Total de daspardicio
R10	Turno	Turno
T1		Temporizador de 1 minuto
C1		Contador de producto bueno
C2		Contador de producto malo
CW1		
CW2		

Continuación del apéndice 1.

PST - MONITOR (Sistema de monnitoreo de producción) - FEC Standard
Strings

#	String
0	
1	
2	
3	
4	
5	
6	
7	
8	
9	
10	
11	
12	
13	SBD
14	No Disk
15	DIR
16	Command Failed
17	OPW
18	D:\
19	WRF
20	CLF

Continuación del apéndice 1.

```
FST - MONITOR (Sistema de monnitoreo de producción) - FEC Standard
Properties of CMP  0 (V1)
-------------------------------------------------------------------

Type:               CMP
Number:             0
Version:            1
Comment:            F13 - Interrogar FECHA

File:               C:\FST4\Projects\MONITOR\CZ0B00V1.FEC2.OBJ
Size:               452 bytes
Date:               26 November 1998 10:45:24

Language:           C (compiled)
Version:            2.10


-------------------------------------------------------------------

Realtime clock: get date

-------------------------------------------------------------------

Input Parameters


Output Parameters

FU32: Year (1980 to 2099)
FU33: Month (1 to 12)
FU34: Day (1 to 31)
FU35: Day of week (0=Sunday, 6=Saturday)

-------------------------------------------------------------------
```

Continuación del apéndice 1.

```
FST - MONITOR (Sistema de monnitoreo de producción) - FEC Standard
Properties of CMP  1 (V1)
-----------------------------------------------------------------

Type:               CMP
Number:             1
Version:            1
Comment:            F12 - Interrogar HORA

File:               C:\FST4\Projects\MONITOR\CZ0B01V1.FEC2.OBJ
Size:               446 bytes
Date:               07 March 2001 12:51:04

Language:           C (compiled)
Version:            2.10


-----------------------------------------------------------------

Realtime clock: get time

-----------------------------------------------------------------

Input Parameters


Output Parameters

FU32: Hour (0 to 23)
FU33: Minute (0 to 59)
FU34: Second (0 to 59)
FU35: 1/100 Seconds (0 to 99)

-----------------------------------------------------------------
```

Continuación del apéndice 1.

```
FST - MONITOR (Sistema de monnitoreo de producción) - FEC Standard
Properties of CMP  2 (V1)
-------------------------------------------------------------

Type:               CMP
Number:             2
Version:            1
Comment:            FPI2F - Conversion de entero a foltante

File:               C:\FST4\Projects\MONITOR\CZ0B02V1.FEC2.OBJ
Size:               482 bytes
Date:               20 April 2001 20:34:04

Language:           C (compiled)
Version:            2.10


-------------------------------------------------------------

16-bit integer to float conversion

-------------------------------------------------------------

Input Parameters

FU32: 16-bit signed integer
FU33: Range value

Output Parameters

FU32: Status: 0 = Ok
FU33: Bytes 3,4 of the float IEEE-754 number
FU34: Bytes 1,2 of the float IEEE-754 number

-------------------------------------------------------------

13 November 2011 20:52:32                                  Page   9
```

Continuación del apéndice 1.

```
FST - MONITOR (Sistema de monnitoreo de producción) - FEC Standard
Properties of CMP  3 (V1)
------------------------------------------------------------------

Type:               CMP
Number:             3
Version:            1
Comment:            FPF2A -  Conversión de flotante a string

File:               C:\FST4\Projects\MONITOR\CZ0B03V1.FEC2.OBJ
Size:               534 bytes
Date:               27 October 2002 14:12:50

Language:           C (compiled)
Version:            2.10


------------------------------------------------------------------

Float to string conversion

------------------------------------------------------------------

Input Parameters

FU32: Bytes 3,4 of fp number
FU33: Bytes 1,2 of fp number
FU34: Number of digits after decimal comma
FU35: Number of FST string for the result

Output Parameters

FU32: Status: 0 = Ok

------------------------------------------------------------------
```

Continuación del apéndice 1.

```
FST - MONITOR (Sistema de monnitoreo de producción) - FEC Standard
Properties of CMP  4 (V1)
------------------------------------------------------------

Type:               CMP
Number:             4
Version:            1
Comment:            STRAPPND - Agregar un caracter al string

File:               C:\FST4\Projects\MONITOR\CZ0B04V1.FEC2.OBJ
Size:               558 bytes
Date:               16 March 1998 11:55:56

Language:           C (compiled)
Version:            2.10


------------------------------------------------------------

Add character to string

------------------------------------------------------------

Input Parameters

FU32: Number of the string
FU33: Character to add

Output Parameters

FU32: 0 if successful, otherwise error

------------------------------------------------------------
```

Continuación del apéndice 1.

```
FST - MONITOR (Sistema de monnitoreo de producción) - FEC Standard
Properties of CMP  5 (V1)
```

```
Type:               CMP
Number:             5
Version:            1
Comment:            STRCPY - Copiara un string a otro

File:               C:\FST4\Projects\MONITOR\CZ0B05V1.FEC2.OBJ
Size:               502 bytes
Date:               11 January 1999 19:41:58

Language:           C (compiled)
Version:            2.10


---------------------------------------------------------------

Copy one string into another string

---------------------------------------------------------------

Input Parameters

FU32: Number of the source string
FU33: Number of the destination string

Output Parameters


---------------------------------------------------------------
```

Continuación del apéndice 1.

```
FST - MONITOR (Sistema de monnitoreo de producción) - FEC Standard
Properties of CMP  6 (V1)
-------------------------------------------------------------------

Type:               CMP
Number:             6
Version:            1
Comment:            STRCAT- Concatenar dos strings

File:               C:\FST4\Projects\MONITOR\CZ0B06V1.FEC2.OBJ
Size:               692 bytes
Date:               11 January 1999 19:41:46

Language:           C (compiled)
Version:            2.10


-------------------------------------------------------------------

Combine two strings in a third string

-------------------------------------------------------------------

Input Parameters

FU32: Number of the first source string
FU33: Number of the second source string
FU34: Number of the destination string

Output Parameters

FU32: 0 if successful, otherwise error

-------------------------------------------------------------------
```

Continuación del apéndice 1.

```
FST - MONITOR (Sistema de monnitoreo de producción) - FEC Standard
Properties of CMP  7 (V1)
-------------------------------------------------------------------------------

Type:               CMP
Number:             7
Version:            1
Comment:            OPENCOM -  Abrir el puerto serial con 9600 baud, 8 bits, no p

File:               C:\FST4\Projects\MONITOR\CZ0B07V1.FEC2.OBJ
Size:               788 bytes
Date:               28 March 2000 14:39:02

Language:           C (compiled)
Version:            2.10


---------------------------------------------------------------

Open serial interface with 9600, 8, N

---------------------------------------------------------------

Input Parameters

FU32: Serial interface

Output Parameters

FU32: 0 = DONE; 1 = ERROR

---------------------------------------------------------------
```

Continuación del apéndice 1.

```
FST - MONITOR (Sistema de monnitoreo de producción) - FEC Standard
Properties of CMP  8 (V1)
```

Type:	CMP
Number:	8
Version:	1
Comment:	CLOSECOM - Cerrar puerto COM
File:	C:\FST4\Projects\MONITOR\CZ0B08V1.FEC2.OBJ
Size:	466 bytes
Date:	28 March 2000 14:35:06
Language:	C (compiled)
Version:	2.10

```
------------------------------------------------------------

Close a previously opened serial port

------------------------------------------------------------

Input Parameters

FU32: Serial port

Output Parameters

FU32: 0 = DONE; 1 = ERROR

------------------------------------------------------------
```

Continuación del apéndice 1.

```
FST - MONITOR (Sistema de monnitoreo de producción) - FEC Standard
Properties of CMP  9 (V1)
-------------------------------------------------------------

Type:               CMP
Number:             9
Version:            1
Comment:            PRINTCOM - Escribir al puerto serie delimitador <CR>

File:               C:\FST4\Projects\MONITOR\CZ0B09V1.FEC2.OBJ
Size:               682 bytes
Date:               28 March 2000 14:39:18

Language:           C (compiled)
Version:            2.10


-------------------------------------------------------------

Write an FST string to a serial interface

-------------------------------------------------------------

Input Parameters

FU32: Serial interface
FU33: Number of FST string to be sent

Output Parameters

FU32: 0 = DONE; 1 = ERROR

-------------------------------------------------------------
```

Continuación del apéndice 1.

```
PST - MONITOR (Sistema de monnitoreo de producción) - FEC Standard
Properties of CMP 10 (V1)
-------------------------------------------------------------

Type:               CMP
Number:             10
Version:            1
Comment:            PUTCOM - Escribir caracter ASCII al  puerto serie

File:               C:\FST4\Projects\MONITOR\CZ0B10V1.FEC2.OBJ
Size:               442 bytes
Date:               28 March 2000 14:39:34

Language:           C (compiled)
Version:            2.10


-------------------------------------------------------------

Write character to a serial interface

-------------------------------------------------------------

Input Parameters

FU32: Serial interface
FU33: Char. to be sent  (0 to 255)

Output Parameters

FU32: 0 = DONE; 1 = ERROR

-------------------------------------------------------------
```

Continuación del apéndice 1.

```
PST - MONITOR (Sistema de monnitoreo de producción) - FEC Standard
Properties of CMP 11 (V1)
-------------------------------------------------------------

Type:               CMP
Number:             11
Version:            1
Comment:            READCOM - Lear puerto seria

File:               C:\PST4\Projects\MONITOR\CZOB11V1.FEC2.OBJ
Size:               720 bytes
Date:               12 August 2002 12:07:16

Language:           C (compiled)
Version:            2.10


-------------------------------------------------------------

Read characters to a delimiter, save to PST string

-------------------------------------------------------------

Input Parameters

FU32: Serial interface
FU33: Number of PST string for the character string
FU34: Maximum length
FU35: Delimiter  (0 to 255)

Output Parameters

FU32: 0 = DONE; 1 = ERROR; -1 = NOTHING RECEIVED

-------------------------------------------------------------
```

Continuación del apéndice 1.

```
FST - MONITOR (Sistema de monnitoreo de producción) - FEC Standard
Properties of CMP 12 (V1)
```

Type:	CMP
Number:	12
Version:	1
Comment:	STRICMP - Compara dos strings 0 = iguales
File:	C:\FST4\Projects\MONITOR\CZ0B12V1.FEC2.OBJ
Size:	550 bytes
Date:	11 January 1999 19:42:12
Language:	C (compiled)
Version:	2.10

```
---------------------------------------------------------------

Compare two strings character by character

---------------------------------------------------------------

Input Parameters

FU32: Number of the first string
FU33: Number of the second string

Output Parameters

FU32: 0 if successful, otherwise error
FU33: 1 if >; 0 if identical; -1 if <

---------------------------------------------------------------
```

Continuación del apéndice 1.

```
FST - MONITOR (Sistema de monnitoreo de producción) - FEC Standard
Properties of CMP 13 (V1)
-------------------------------------------------------------

Type:               CMP
Number:             13
Version:            1
Comment:            STRCLR - Borrar string

File:               C:\FST4\Projects\MONITOR\CZ0B13V1.FEC2.OBJ
Size:               410 bytes
Date:               11 January 1999 19:41:54

Language:           C (compiled)
Version:            2.10


---------------------------------------------------------------

Delete a string

---------------------------------------------------------------

Input Parameters

FU32: Number of the string

Output Parameters

FU32: 0 if successful, otherwise error

---------------------------------------------------------------
```

Continuación del apéndice 1.

```
FST - MONITOR (Sistema de monnitoreo de producción) - FEC Standard
Properties of CFM  0 (V1)
-------------------------------------------------------------------------------

Type:               CFM
Number:             0
Version:            1
Comment:            Inicio

File:               C:\FST4\Projects\MONITOR\CZ0P00V1.awl
Size:               62 bytes
Date:               13 November 2011 20:22:55

Language:           Statement List
Last Compiled:      13 November 2011 20:24:58
```

Continuación del apéndice 1.

```
FST - MONITOR (Sistema de monnitoreo de producción) - FBC Standard
CFM  0 (V1) - Inicio
-------------------------------------------------------------------------------

0001
0002  THEN
0003        LOAD          V0
0004          TO          OW0
0005          TO          FW0
0006          TO          R0            'C1A: Auxiliar Conatdor de producto bueno
0007          TO          R1            'C2A: Auxiliar Contador de producto malo
0008          TO          R2            'año:
0009          TO          R3            'mes:
0010          TO          R4            'dia:
0011          TO          R5            'mes_ant: mes del dato antes almacenado
0012          TO          R6            'hora:
0013          TO          R7            'minuto:
0014
```

Continuación del apéndice 1.

```
FST - MONITOR (Sistema de monnitoreo de producción) - FEC Standard
Properties of Program  0 (V1)
-------------------------------------------------------------------------------

Type:               Program
Number:             0
Version:            1
Comment:            Rutina Principal

File:               C:\FST4\Projects\MONITOR\CZ0P00V1.awl
Size:               400 bytes
Date:               13 November 2011 19:40:31

Language:           Statement List
Last Compiled:      13 November 2011 20:24:58
```

Continuación del apéndice 1.

```
PST - MONITOR (Sistema de monnitoreo de producción) - FBC Standard
Program  0 (V1) - Rutina Principal
0001 "" Rutina de Inicialización
0002 STEP Inicio
0003  THEN
0004        CPM 0                          'Inicio
0005
0006 "" Rutina de conteo
0007 STEP Conteo
0008  THEN  SET            P1             'Rutina de conteo
0009
0010 "" Iniciar temporizador
0011 STEP Temp
0012  THEN  SET            T1             'Temporizador de 1 minuto
0013          WITH         V60
0014
0015 "" Evaluar temporizador
0016 STEP
0017  IF             N     T1             'Temporizador de 1 minuto
0018  THEN  JMP TO Procesa
0019
0020 "" Rutina de procesamiento
0021 STEP Procesa
0022  THEN  RESET          F_Proc         'F0.0: Bandera - Procesamiento realizado
0023        SET            P2             'Rutina de procesamiento
0024  IF                   F_Proc         'F0.0: Bandera - Procesamiento realizado
0025  THEN  RESET          P2             'Rutina de procesamiento
0026        JMP TO Transm
0027
0028 "" Rutina de transmisión
0029 STEP Transm
0030  THEN  RESET          F_trans        'F0.1: Bandera - Transmision realizada
0031        SET            P3             'Rutina de transmisión
0032  IF                   F_trans        'F0.1: Bandera - Transmision realizada
0033  THEN  RESET          P3             'Rutina de transmisión
0034        JMP TO Temp
0035
0036
0037
```

Continuación del apéndice 1.

```
FST - MONITOR (Sistema de monnitoreo de producción) - FEC Standard
Properties of Program  1 (V1)
-------------------------------------------------------------------------------

Type:               Program
Number:             1
Version:            1
Comment:            Subrutina de Conteo

File:               C:\FST4\Projects\MONITOR\CZ0P01V1.awl
Size:               537 bytes
Date:               13 November 2011 17:15:12

Language:           Statement List
Last Compiled:      13 November 2011 20:24:58
```

Continuación del apéndice 1.

```
FST - MONITOR (Sistema de monnitoreo de producción) - FBC Standard
Program  1 (V1) - Subrutina de Conteo
-------------------------------------------------------------------------------
0001 "" Esperar por señal de descarga
0002 STEP Espera
0003 "" Si se da la señal de descarga
0004  IF                  Descarga       'I0.0: Señal de descarga
0005  THEN
0006        JMP TO Desc
0007 "" Si se da la señal de cierre
0008  IF                  Cierra         'I0.1: Señal de cierre
0009  THEN  JMP TO Cier
0010
0011 STEP Desc
0012  THEN
0013 ""Levantar bandera de descarga realizada
0014        SET           F_Decarga      'F0.2: Bandera - Descarga realizada
0015
0016 "" Esperar por desactivacion de señal de descarga
0017 STEP
0018  IF            N     Descarga       'I0.0: Señal de descarga
0019  THEN  JMP TO Espera
0020
0021
0022 STEP Cier
0023 ""Si se dio una señal de descarga y luego la de cierre
0024  IF                  F_Decarga      'F0.2: Bandera - Descarga realizada
0025  THEN  INC           C1             'Contador de producto bueno
0026 ""Si se dio una señal de cierre sin descarga
0027  IF            N     F_Decarga      'F0.2: Bandera - Descarga realizada
0028  THEN  INC           C2             'Contador de producto malo
0029
0030 STEP
0031 ""Esperar por desactivación de señal de cierre
0032  IF            N     Cierra         'I0.1: Señal de cierre
0033  THEN  JMP TO Espera
0034
-------------------------------------------------------------------------------
13 November 2011 20:52:32                                        Page  26
```

Continuación del apéndice 1.

```
FST - MONITOR (Sistema de monnitoreo de producción) - FBC Standard
Properties of Program  2 (V1)
-------------------------------------------------------------------------------

Type:               Program
Number:             2
Version:            1
Comment:            Subrutina de Procesamiento

File:               C:\FST4\Projects\MONITOR\CZ0P02V1.aw1
Size:               2407 bytes
Date:               13 November 2011 20:25:01

Language:           Statement List
Last Compiled:      13 November 2011 20:25:01
```

Continuación del apéndice 1.

```
FST - MONITOR (Sistema de monnitoreo de producción) - FBC Standard
Program  2 (V1) - Subrutina de Procesamiento
------------------------------------------------------------------------------

0001 STEP
0002  THEN  LOAD            CW1
0003          TO            C1A             'R0: Auxiliar Conatdor de producto bueno
0004        LOAD            CW2
0005          TO            C2A             'R1: Auxiliar Contador de producto malo
0006        RESET           C1              'Contador de producto bueno
0007        RESET           C2              'Contador de producto malo
0008 ""Extraer Fecha y Hora
0009        LOAD            mes             'R3:
0010          TO            mes_ant         'R5: mes del dato antes almacenado
0011        CMP 0                           'F13 - Interrogar FECHA
0012        LOAD            FU32
0013          TO            año             'R2:
0014        LOAD            FU33
0015          TO            mes             'R3:
0016        LOAD            FU34
0017          TO            dia             'R4:
0018        CMP 1                           'F12 - Interrogar HORA
0019        LOAD            FU32
0020          TO            hora            'R6:
0021        LOAD            FU33
0022          TO            minuto          'R7:
0023 ""Comparar mes
0024 STEP
0025  IF                    mes             'R3:
0026          =             mes_ant         'R5: mes del dato antes almacenado
0027  THEN
0028        LOAD            TPB             'R8: Total de producto bueno
0029          +             C1A             'R0: Auxiliar Conatdor de producto bueno
0030          TO            TPB             'R8: Total de producto bueno
0031        LOAD            TD              'R9: Total de desperdicio
0032          +             C2A             'R1: Auxiliar Contador de producto malo
0033          TO            TD              'R9: Total de desperdicio
0034  OTHRW
0035        LOAD            TPB             'R8: Total de producto bueno
0036          TO            C1A             'R0: Auxiliar Conatdor de producto bueno
0037        LOAD            C1A             'R0: Auxiliar Conatdor de producto bueno
0038          TO            TPB             'R8: Total de producto bueno
0039        LOAD            C2A             'R1: Auxiliar Contador de producto malo
0040          TO            TD              'R9: Total de desperdicio
0041
0042 ""Verificar Turno
0043 STEP
0044  IF              (     V7
0045          <=            hora      )     'R6:
0046        AND       (     hora            'R6:
0047          <             V19       )
0048  THEN
0049        LOAD            V1
0050          TO            Turno           'R10: Turno
0051  OTHRW
0052        LOAD            V2
0053          TO            Turno           'R10: Turno
0054
0055 STEP
0056  THEN
0057 "" Limpiar strings
0058        CMP 13                          'STRCLR - Borrar string
0059          WITH          V0
0060        CMP 13                          'STRCLR - Borrar string
0061          WITH          V1
0062        CMP 13                          'STRCLR - Borrar string
0063          WITH          V2
0064        CMP 13                          'STRCLR - Borrar string
0065          WITH          V3
0066        CMP 13                          'STRCLR - Borrar string
0067          WITH          V4
0068        CMP 13                          'STRCLR - Borrar string
0069          WITH          V5
0070        CMP 13                          'STRCLR - Borrar string
0071          WITH          V6
0072        CMP 13                          'STRCLR - Borrar string
0073          WITH          V7
0074        CMP 13                          'STRCLR - Borrar string
0075          WITH          V8
0076        CMP 13                          'STRCLR - Borrar string
0077          WITH          V9
0078        CMP 13                          'STRCLR - Borrar string

------------------------------------------------------------------------------
13 November 2011 20:52:32                                          Page  28
```

Continuación del apéndice 1.

```
FST - MONITOR (Sistema de monnitoreo de producción) - FBC Standard
Program  2 (V1) - Subrutina de Procesamiento
------------------------------------------------------------------------------
0079          WITH          V10
0080        CMP 13                        'STRCLR - Borrar string
0081          WITH          V11
0082        CMP 13                        'STRCLR - Borrar string
0083          WITH          V12
0084        CMP 13                        'STRCLR - Borrar string
0085          WITH          V13
0086        CMP 13                        'STRCLR - Borrar string
0087          WITH          V14
0088        CMP 13                        'STRCLR - Borrar string
0089          WITH          V15
0090        CMP 13                        'STRCLR - Borrar string
0091          WITH          V16
0092        CMP 13                        'STRCLR - Borrar string
0093          WITH          V17
0094        CMP 13                        'STRCLR - Borrar string
0095          WITH          V18
0096        CMP 13                        'STRCLR - Borrar string
0097          WITH          V19
0098        CMP 13                        'STRCLR - Borrar string
0099          WITH          V20
0100
0101 ""Cargar TPB a string 0
0102
0103        LOAD            TPB           'R8: Total de producto bueno
0104          TO            AUX           'FW100: Bandera Auxiliar
0105        CMP 2                         'FPI2F - Conversion de entero a foltante
0106          WITH          FW100         'AUX: Bandera Auxiliar
0107          WITH          V0
0108        LOAD            FU33
0109          TO            FU32
0110        LOAD            FU34
0111          TO            FU33
0112        CMP 3                         'FPF2A -  Conversión de flotante a string
0113          WITH          FU32
0114          WITH          FU33
0115          WITH          V0
0116          WITH          V0            "String 0 - TPB
0117
0118
0119 ""Cargar TD a string 1
0120
0121        LOAD            TD            'R9: Total de desperdicio
0122          TO            AUX           'FW100: Bandera Auxiliar
0123        CMP 2                         'FPI2F - Conversion de entero a foltante
0124          WITH          FW100         'AUX: Bandera Auxiliar
0125          WITH          V0
0126        LOAD            FU33
0127          TO            FU32
0128        LOAD            FU34
0129          TO            FU33
0130        CMP 3                         'FPF2A -  Conversión de flotante a string
0131          WITH          FU32
0132          WITH          FU33
0133          WITH          V0
0134          WITH          V1            "String 1 - TD
0135
0136 ""Cargar C1A (Velocidad) a string 2
0137
0138        LOAD            C1A           'R0: Auxiliar Conatdor de producto bueno
0139          TO            AUX           'FW100: Bandera Auxiliar
0140        CMP 2                         'FPI2F - Conversion de entero a foltante
0141          WITH          FW100         'AUX: Bandera Auxiliar
0142          WITH          V0
0143        LOAD            FU33
0144          TO            FU32
0145        LOAD            FU34
0146          TO            FU33
0147        CMP 3                         'FPF2A -  Conversión de flotante a string
0148          WITH          FU32
0149          WITH          FU33
0150          WITH          V0
0151          WITH          V2            "String 2 - Velocidad
0152
0153 ""Cargar año a string 3
0154
0155        LOAD            año           'R2:
0156          TO            AUX           'FW100: Bandera Auxiliar
------------------------------------------------------------------------------
13 November 2011 20:52:32                                          Page  29
```

Continuación del apéndice 1.

```
FST - MONITOR (Sistema de monnitoreo de producción) - FBC Standard
Program  2 (V1) - Subrutina de Procesamiento
------------------------------------------------------------------------------

0157        CMP 2                       'FPI2F - Conversion de entero a foltante
0158          WITH          FW100       'AUX: Bandera Auxiliar
0159          WITH          V0
0160        LOAD            FU33
0161          TO            FU32
0162        LOAD            FU34
0163          TO            FU33
0164        CMP 3                       'FPF2A -  Conversión de flotante a string
0165          WITH          FU32
0166          WITH          FU33
0167          WITH          V0
0168          WITH          V3          "String 3 - año
0169
0170 ""Cargar mes a string 4
0171
0172        LOAD            mes         'R3:
0173          TO            AUX         'FW100: Bandera Auxiliar
0174        CMP 2                       'FPI2F - Conversion de entero a foltante
0175          WITH          FW100       'AUX: Bandera Auxiliar
0176          WITH          V0
0177        LOAD            FU33
0178          TO            FU32
0179        LOAD            FU34
0180          TO            FU33
0181        CMP 3                       'FPF2A -  Conversión de flotante a string
0182          WITH          FU32
0183          WITH          FU33
0184          WITH          V0
0185          WITH          V4          "String 4 - mes
0186
0187 ""Cargar dia a string 5
0188
0189        LOAD            dia         'R4:
0190          TO            AUX         'FW100: Bandera Auxiliar
0191        CMP 2                       'FPI2F - Conversion de entero a foltante
0192          WITH          FW100       'AUX: Bandera Auxiliar
0193          WITH          V0
0194        LOAD            FU33
0195          TO            FU32
0196        LOAD            FU34
0197          TO            FU33
0198        CMP 3                       'FPF2A -  Conversión de flotante a string
0199          WITH          FU32
0200          WITH          FU33
0201          WITH          V0
0202          WITH          V5          "String 5 - dia
0203
0204 ""Cargar hora a string 6
0205
0206        LOAD            hora        'R6:
0207          TO            AUX         'FW100: Bandera Auxiliar
0208        CMP 2                       'FPI2F - Conversion de entero a foltante
0209          WITH          FW100       'AUX: Bandera Auxiliar
0210          WITH          V0
0211        LOAD            FU33
0212          TO            FU32
0213        LOAD            FU34
0214          TO            FU33
0215        CMP 3                       'FPF2A -  Conversión de flotante a string
0216          WITH          FU32
0217          WITH          FU33
0218          WITH          V0
0219          WITH          V6          "String 6 - hora
0220
0221 ""Cargar minuto a string 7
0222
0223        LOAD            minuto      'R7:
0224          TO            AUX         'FW100: Bandera Auxiliar
0225        CMP 2                       'FPI2F - Conversion de entero a foltante
0226          WITH          FW100       'AUX: Bandera Auxiliar
0227          WITH          V0
0228        LOAD            FU33
0229          TO            FU32
0230        LOAD            FU34
0231          TO            FU33
0232        CMP 3                       'FPF2A -  Conversión de flotante a string
0233          WITH          FU32
0234          WITH          FU33

------------------------------------------------------------------------------
13 November 2011 20:52:32                                          Page  30
```

Continuación del apéndice 1.

```
FST - MONITOR (Sistema de monnitoreo de producción) - FEC Standard
Program  2 (V1) - Subrutina de Procesamiento
------------------------------------------------------------------------------
0235          WITH          V0
0236          WITH          V7              "String 7 - minuto
0237
0238 ""Cargar Turno a string 8
0239
0240        LOAD            Turno           'R10: Turno
0241          TO            AUX             'FW100: Bandera Auxiliar
0242        CMP 2                           'FPI2F - Conversion de entero a foltante
0243          WITH          FW100           'AUX: Bandera Auxiliar
0244          WITH          V0
0245        LOAD            FU33
0246          TO            FU32
0247        LOAD            FU34
0248          TO            FU33
0249        CMP 3                           'FPF2A -  Conversión de flotante a string
0250          WITH          FU32
0251          WITH          FU33
0252          WITH          V0
0253          WITH          V8              "String 8 - Turno
0254
0255 ""Cargar Fecha a string 9
0256        CMP 5                           'STRCPY - Copiara un string a otro
0257          WITH          V5
0258          WITH          V9
0259        CMP 4                           'STRAPPND - Agregar un caracter al string
0260          WITH          V9
0261          WITH          V47             "Caracter '/'
0262        CMP 6                           'STRCAT- Concatenar dos strings
0263          WITH          V9
0264          WITH          V4
0265          WITH          V9
0266        CMP 4                           'STRAPPND - Agregar un caracter al string
0267          WITH          V9
0268          WITH          V47             "Caracter '/'
0269        CMP 6                           'STRCAT- Concatenar dos strings
0270          WITH          V9
0271          WITH          V3
0272          WITH          V9
0273
0274 ""Cargar Tiempo a string 10
0275        CMP 5                           'STRCPY - Copiara un string a otro
0276          WITH          V6
0277          WITH          V10
0278        CMP 4                           'STRAPPND - Agregar un caracter al string
0279          WITH          V10
0280          WITH          V58             "Caracter ':'
0281  IF                    minuto          'R7:
0282          <             V10
0283  THEN
0284        CMP 4                           'STRAPPND - Agregar un caracter al string
0285          WITH          V10
0286          WITH          V0              "Caracter '0'
0287        CMP 6                           'STRCAT- Concatenar dos strings
0288          WITH          V10
0289          WITH          V7
0290          WITH          V10
0291
0292  OTHRW
0293
0294        CMP 6                           'STRCAT- Concatenar dos strings
0295          WITH          V10
0296          WITH          V7
0297          WITH          V10
0298
0299 "" Cargar nombre a string 11
0300 STEP
0301  THEN
0302        CMP 5                           'STRCPY - Copiara un string a otro
0303          WITH          V4
0304          WITH          V11
0305
0306        CMP 4                           'STRAPPND - Agregar un caracter al string
0307          WITH          V10
0308          WITH          V95             "Caracter '_'
0309
0310        CMP 6                           'STRCAT- Concatenar dos strings
0311          WITH          V11
0312          WITH          V3
------------------------------------------------------------------------------
13 November 2011 20:52:32                                          Page  31
```

Continuación del apéndice 1.

```
FST - MONITOR (Sistema de monnitoreo de producción) - FEC Standard
Program  2 (V1) - Subrutina de Procesamiento
-------------------------------------------------------------------------------
0313          WITH          V11
0314
0315        CMP 4                         'STRAPPND - Agregar un caracter al string
0316          WITH          V11
0317          WITH          V46           "Caracter '.'
0318
0319        CMP 4                         'STRAPPND - Agregar un caracter al string
0320          WITH          V11
0321          WITH          V78           "Caracter 'x'
0322
0323        CMP 4                         'STRAPPND - Agregar un caracter al string
0324          WITH          V11
0325          WITH          V108          "Caracter '1'
0326
0327        CMP 4                         'STRAPPND - Agregar un caracter al string
0328          WITH          V11
0329          WITH          V73           "Caracter 's'
0330
0331        SET             F_Proc        'F0.0: Bandera - Procesamiento realizado
0332
0333
0334
0335
0336
0337
0338
-------------------------------------------------------------------------------
13 November 2011 20:52:32                                          Page  32
```

Continuación del apéndice 1.

```
PST - MONITOR (Sistema de monnitoreo de producción) - FBC Standard
Properties of Program  3 (V1)
-------------------------------------------------------------------------------

Type:               Program
Number:             3
Version:            1
Comment:            Subrutina de Transmisión

File:               C:\PST4\Projects\MONITOR\CZ0P03V1.awl
Size:               552 bytes
Date:               13 November 2011 18:13:36

Language:           Statement List
Last Compiled:      13 November 2011 20:24:58
```

Continuación del apéndice 1.

```
PST - MONITOR (Sistema de monnitoreo de producción) - FEC Standard
Program  3 (V1) - Subrutina de Transmisión
0001 "" Abrir el puerto serial EXT
0002 STEP
0003  THEN  CMP 7                          'OPENCOM -  Abrir el puerto serial con 9600 baud, 8 bits, no p
0004          WITH          V1
0005  IF                    FU32
0006          =             V0
0007  THEN
0008        JMP TO ajustar
0009
0010 "" Ajustar puerto del VDIP
0011 STEP ajustar
0012  THEN  RESET           F_ajustar       'P0.3: Puerto de VDIP ajustado
0013        SET             P4
0014  IF                    F_ajustar       'P0.3: Puerto de VDIP ajustado
0015  THEN  RESET           P4
0016
0017 "" Verificar memoria USB
0018 STEP
0019  THEN  RESET           F_verif         'P0.4: Memoria USB verificada
0020        SET             P5
0021  IF                    F_verif         'P0.4: Memoria USB verificada
0022  THEN  RESET           P5
0023
0024 STEP
0025  IF            N       Usb_Rdy         'P0.5: Memoria USB lista
0026  THEN  SET             Alarma          'O0.0: Alarma de fallo en memoria
0027        JMP TO cerrarCOM
0028  OTHRW
0029        RESET           Alarma          'O0.0: Alarma de fallo en memoria
0030
0031 "" Comprobar existencia de archivo
0032 STEP
0033  THEN  RESET           F_Comp          'P0.6: Existensia de archivo comprobada
0034        SET             P6
0035  IF                    F_Comp          'P0.6: Existensia de archivo comprobada
0036  THEN  RESET           P6
0037
0038 STEP
0039  IF                    Crear_A         'P0.7: Crear archivo
0040  THEN  JMP TO crearAr
0041  OTHRW
0042        JMP TO escribir
0043
0044 STEP crearAr
0045  THEN  RESET           F_Creado        'P0.8: Archivo creado
0046        SET             P7
0047  IF                    F_Creado        'P0.8: Archivo creado
0048  THEN  RESET           P7
0049
0050 STEP escribir
0051  THEN  RESET           F_Escrito       'P0.9: Datos escritos
0052        SET             P8
0053  IF                    F_Escrito       'P0.9: Datos escritos
0054  THEN  RESET           P8
0055
0056 STEP cerrarCOM
0057  THEN  CMP 8                           'CLOSECOM - Cerrar puerto COM
0058          WITH          V1
0059  IF                    FU32
0060          =             V0
0061  THEN  SET             F_trans         'P0.1: Bandera - Transmision realizada
0062
```

Continuación del apéndice 1.

```
FST - MONITOR (Sistema de monnitoreo de producción) - FEC Standard
Properties of Program  4 (V1)
-------------------------------------------------------------------------------

Type:               Program
Number:             4
Version:            1
Comment:            Rutina Auxiliar - Ajustar puerto serie de VDIP

File:               C:\FST4\Projects\MONITOR\CZ0P04V1.awl
Size:               862 bytes
Date:               13 November 2011 20:48:53

Language:           Statement List
Last Compiled:      13 November 2011 20:24:58
```

Continuación del apéndice 1.

```
FST - MONITOR (Sistema de monnitoreo de producción) - FEC Standard
Program  4 (V1) - Rutina Auxiliar - Ajustar puerto serie de VDIP
0001 "" Contenido de Strings
0002 "" 0) TPB                  11) Nombre
0003 "" 1) TD                   12) String Auxiliar
0004 "" 2) Velocidad            13) 'SBD ' (contiene el comando SBD y un espacio <sp>)
0005 "" 3) Año                  14) 'No Disk'
0006 "" 4) Mes                  15) 'DIR ' (contiene el comando DIR y un espacio <sp>)
0007 "" 5) Dia                  16) 'Command Failed'
0008 "" 6) Horas                17) 'OPW ' (contiene el comando OPW y un espacio <sp>)
0009 "" 7) Minutos              18) 'D:\'
0010 "" 8) Turno□               19) 'WRF ' (contiene el comando WRF y un espacio <sp>)
0011 "" 9) Fecha□               20) 'CLF ' (contiene el comando CLF y un espacio <sp>)
0012 "" 10)Tiempo
0013
0014 "" Cargar comando a Auxiliar
0015 STEP
0016  THEN
0017         CMP 13                          'STRCLR - Borrar string
0018           WITH          V12
0019         CMP 5                           'STRCPY - Copiara un string a otro
0020           WITH          V13
0021           WITH          V12
0022         CMP 4                           'STRAPPND - Agregar un caracter al string
0023           WITH          V12
0024           WITH          V56             "Hexadecimal $38
0025         CMP 4                           'STRAPPND - Agregar un caracter al string
0026           WITH          V12
0027           WITH          V65             "Hexadecimal $41
0028         CMP 4                           'STRAPPND - Agregar un caracter al string
0029           WITH          V12
0030           WITH          V0              "Hexadecimal $00
0031 "" Enviar comando
0032         CMP 9                           'PRINTCOM - Escribir al puerto serie delimitador <CR>
0033           WITH          V1
0034           WITH          V12
0035  IF                     FU32
0036           =             V0
0037  THEN  SET              F_ajustar       'F0.3: Puerto de VDIP ajustado
0038
```

Continuación del apéndice 1.

```
FST - MONITOR (Sistema de monnitoreo de producción) - FEC Standard
Properties of Program  5 (V1)
-------------------------------------------------------------------------------

Type:               Program
Number:             5
Version:            1
Comment:            Rutina Auxiliar  - Verificar memoria USB

File:               C:\FST4\Projects\MONITOR\CZ0P05V1.awl
Size:               960 bytes
Date:               13 November 2011 20:48:54

Language:           Statement List
Last Compiled:      13 November 2011 20:24:58
```

Continuación del apéndice 1.

```
FST - MONITOR (Sistema de monnitoreo de producción) - FEC Standard
Program  5 (V1) - Rutina Auxiliar  - Verificar memoria USB
-------------------------------------------------------------------------------

0001 "" Contenido de Strings
0002 "" 0) TPB                  11) Nombre
0003 "" 1) TD                   12) String Auxiliar
0004 "" 2) Velocidad            13) 'SBD ' (contiene el comando SBD y un espacio <sp>)
0005 "" 3) Año                  14) 'No Disk'
0006 "" 4) Mes                  15) 'DIR ' (contiene el comando DIR y un espacio <sp>)
0007 "" 5) Dia                  16) 'Command Failed'
0008 "" 6) Horas                17) 'OPW ' (contiene el comando OPW y un espacio <sp>)
0009 "" 7) Minutos              18) 'D:\'
0010 "" 8) Turno□               19) 'WRF ' (contiene el comando WRF y un espacio <sp>)
0011 "" 9) Fecha□               20) 'CLF ' (contiene el comando CLF y un espacio <sp>)
0012 "" 10)Tiempo
0013
0014 ""Enviar comando
0015 STEP
0016  THEN  CMP 10                        'PUTCOM - Escribir caracter ASCII al  puerto serie
0017          WITH          V1
0018          WITH          V13           " Caracter <CR>
0019  IF                    FU32
0020          =             V0
0021  THEN  JMP TO respuesta
0022
0023 ""Leer respuesta
0024 STEP respuesta
0025  THEN
0026        CMP 13                        'STRCLR - Borrar string
0027          WITH          V12
0028        CMP 11                        'READCOM - Leer puerto serie
0029          WITH          V1
0030          WITH          V12
0031          WITH          V10
0032          WITH          V13
0033  IF                    FU32
0034          =             V-1
0035  THEN  JMP TO respuesta
0036  IF                    FU32
0037          =             V0
0038  THEN  JMP TO comparar
0039
0040 ""Comparar respuesta
0041 STEP comparar
0042  THEN  CMP 12                        'STRICMP - Compara dos strings 0 = iguales
0043          WITH          V12
0044          WITH          V14
0045  IF                    FU32
0046          =             V0
0047  THEN  SET             Usb_Rdy       'F0.5: Memoria USB lista
0048  OTHRW
0049        RESET           Usb_Rdy       'F0.5: Memoria USB lista
0050
0051 STEP
0052  THEN  SET             F_verif       'F0.4: Memoria USB verificada
0053

-------------------------------------------------------------------------------
13 November 2011 20:52:32                                          Page  38
```

Continuación del apéndice 1.

```
PST - MONITOR (Sistema de monnitoreo de producción) - FBC Standard
Properties of Program  6 (V1)
-------------------------------------------------------------------------------

Type:               Program
Number:             6
Version:            1
Comment:            Rutina Auxiliar - Comprobar existenca de archivo

File:               C:\PST4\Projects\MONITOR\CZ0P06V1.awl
Size:               1015 bytes
Data:               13 November 2011 20:48:55

Language:           Statement List
Last Compilad:      13 November 2011 20:24:58
```

Continuación del apéndice 1.

```
FST - MONITOR (Sistema de monnitoreo de producción) - FEC Standard
Program  6 (V1) - Rutina Auxiliar - Comprobar existenca de archivo
-------------------------------------------------------------------------------
0001 "" Contenido de Strings
0002 "" 0) TPB                  11) Nombre
0003 "" 1) TD                   12) String Auxiliar
0004 "" 2) Velocidad            13) 'SBD ' (contiene el comando SBD y un espacio <sp>)
0005 "" 3) Año                  14) 'No Disk'
0006 "" 4) Mes                  15) 'DIR ' (contiene el comando DIR y un espacio <sp>)
0007 "" 5) Dia                  16) 'Command Failed'
0008 "" 6) Horas                17) 'OPW ' (contiene el comando OPW y un espacio <sp>)
0009 "" 7) Minutos              18) 'D:\'
0010 "" 8) Turno□               19) 'WRF ' (contiene el comando WRF y un espacio <sp>)
0011 "" 9) Fecha□               20) 'CLF ' (contiene el comando CLF y un espacio <sp>)
0012 "" 10)Tiempo
0013
0014
0015 "" Cargar comando
0016 STEP
0017  THEN  CMP 13                          'STRCLR - Borrar string
0018          WITH          V12
0019        CMP 5                          'STRCPY - Copiara un string a otro
0020          WITH          V15
0021          WITH          V12
0022        CMP 6                          'STRCAT- Concatenar dos strings
0023          WITH          V12
0024          WITH          V11
0025          WITH          V12
0026
0027 ""Enviar comando
0028        CMP 9                          'PRINTCOM - Escribir al puerto serie delimitador <CR>
0029          WITH          V1
0030          WITH          V12
0031  IF                    FU32
0032          =             V0
0033  THEN  JMP TO respuesta
0034
0035 ""Leer respuesta
0036 STEP respuesta
0037  THEN
0038        CMP 13                         'STRCLR - Borrar string
0039          WITH          V12
0040        CMP 11                         'READCOM - Leer puerto serie
0041          WITH          V1
0042          WITH          V12
0043          WITH          V20
0044          WITH          V13
0045  IF                    FU32
0046          =             V-1
0047  THEN  JMP TO respuesta
0048  IF                    FU32
0049          =             V0
0050  THEN  JMP TO comparar
0051
0052 ""Comparar respuesta
0053 STEP comparar
0054  THEN  CMP 12                         'STRICMP - Compara dos strings 0 = iguales
0055          WITH          V12
0056          WITH          V16
0057  IF                    FU32
0058          =             V0
0059  THEN  SET             Crear_A        'P0.7: Crear archivo
0060  OTHRW
0061        RESET           Crear_A        'P0.7: Crear archivo
0062
0063 STEP
0064  THEN  SET             F_Comp         'P0.6: Existensia de archivo comprobada
0065
-------------------------------------------------------------------------------
13 November 2011 20:52:32                                          Page  40
```

Continuación del apéndice 1.

```
PST - MONITOR (Sistema de monnitoreo de producción) - FBC Standard
Properties of Program  7 (V1)
-------------------------------------------------------------------------------

Type:               Program
Number:             7
Version:            1
Comment:            Rutina Auxiliar - Crear Archivo

File:               C:\PST4\Projects\MONITOR\CZ0P07V1.awl
Size:               1023 bytes
Data:               13 November 2011 20:48:55

Language:           Statement List
Last Compilad:      13 November 2011 20:24:58
```

Continuación del apéndice 1.

```
FST - MONITOR (Sistema de monnitoreo de producción) - FEC Standard
Program  7 (V1) - Rutina Auxiliar - Crear Archivo
-------------------------------------------------------------------------------
0001 "" Contenido de Strings
0002 "" 0) TPB                11) Nombre
0003 "" 1) TD                 12) String Auxiliar
0004 "" 2) Velocidad          13) 'SBD ' (contiene el comando SBD y un espacio <sp>)
0005 "" 3) Año                14) 'No Disk'
0006 "" 4) Mes                15) 'DIR ' (contiene el comando DIR y un espacio <sp>)
0007 "" 5) Dia                16) 'Command Failed'
0008 "" 6) Horas              17) 'OPW ' (contiene el comando OPW y un espacio <sp>)
0009 "" 7) Minutos            18) 'D:\'
0010 "" 8) Turno□             19) 'WRF ' (contiene el comando WRF y un espacio <sp>)
0011 "" 9) Fecha□             20) 'CLF ' (contiene el comando CLF y un espacio <sp>)
0012 "" 10)Tiempo
0013
0014
0015 "" Cargar comando
0016 STEP comando
0017  THEN  CMP 13                          'STRCLR - Borrar string
0018          WITH          V12
0019        CMP 5                          'STRCPY - Copiara un string a otro
0020          WITH          V17
0021          WITH          V12
0022        CMP 6                          'STRCAT- Concatenar dos strings
0023          WITH          V12
0024          WITH          V11
0025          WITH          V12
0026
0027 ""Enviar comando
0028        CMP 9                          'PRINTCOM - Escribir al puerto serie delimitador <CR>
0029          WITH          V1
0030          WITH          V12
0031  IF                    FU32
0032          =             V0
0033  THEN  JMP TO respuesta
0034
0035 ""Leer respuesta
0036 STEP respuesta
0037  THEN
0038        CMP 13                         'STRCLR - Borrar string
0039          WITH          V12
0040        CMP 11                         'READCOM - Leer puerto serie
0041          WITH          V1
0042          WITH          V12
0043          WITH          V10
0044          WITH          V13
0045  IF                    FU32
0046          =             V-1
0047  THEN  JMP TO respuesta
0048  IF                    FU32
0049          =             V0
0050  THEN  JMP TO comparar
0051
0052 ""Comparar respuesta
0053 STEP comparar
0054  THEN  CMP 12                         'STRICMP - Compara dos strings 0 = iguales
0055          WITH          V12
0056          WITH          V18
0057  IF                    FU32
0058          =             V0
0059  THEN  JMP TO fin
0060  OTHRW
0061        JMP TO comando
0062
0063 STEP fin
0064  THEN  SET             F_Creado       'F0.8: Archivo creado
0065
0066
-------------------------------------------------------------------------------
13 November 2011 20:52:32                                          Page  42
```

Continuación del apéndice 1.

```
FST - MONITOR (Sistema de monnitoreo de producción) - FEC Standard
Properties of Program  8 (V1)
-------------------------------------------------------------------------------

Type:                Program
Number:              8
Version:             1
Comment:             Rutina Auxiliar - Escribir datos

File:                C:\FST4\Projects\MONITOR\CZ0P08V1.awl
Size:                1458 bytes
Date:                13 November 2011 20:48:38

Language:            Statement List
Last Compiled:       13 November 2011 20:48:38
```

Continuación del apéndice 1.

```
FST - MONITOR (Sistema de monnitoreo de producción) - FEC Standard
Program  8 (V1) - Rutina Auxiliar - Escribir datos
-------------------------------------------------------------------------------

0001 "" Contenido de Strings
0002 "" 0) TPB                11) Nombre
0003 "" 1) TD                 12) String Auxiliar
0004 "" 2) Velocidad          13) 'SBD ' (contiene el comando SBD y un espacio <sp>)
0005 "" 3) Año                14) 'No Disk'
0006 "" 4) Mes                15) 'DIR ' (contiene el comando DIR y un espacio <sp>)
0007 "" 5) Dia                16) 'Command Failed'
0008 "" 6) Horas              17) 'OPW ' (contiene el comando OPW y un espacio <sp>)
0009 "" 7) Minutos            18) 'D:\'
0010 "" 8) Turno□             19) 'WRF ' (contiene el comando WRF y un espacio <sp>)
0011 "" 9) Fecha□             20) 'CLF ' (contiene el comando CLF y un espacio <sp>)
0012 "" 10)Tiempo
0013
0014
0015 "" Cargar comando
0016 STEP comando
0017  THEN  CMP 13                        'STRCLR - Borrar string
0018          WITH          V12
0019        CMP 5                        'STRCPY - Copiara un string a otro
0020          WITH          V19
0021          WITH          V12
0022        CMP 4                        'STRAPPND - Agregar un caracter al string
0023          WITH          V12
0024          WITH          V0           "Hexadecimal $00
0025        CMP 4                        'STRAPPND - Agregar un caracter al string
0026          WITH          V12
0027          WITH          V35          "Hexadecimal $23
0028
0029 ""Enviar comando
0030        CMP 9                        'PRINTCOM - Escribir al puerto serie delimitador <CR>
0031          WITH          V1
0032          WITH          V12
0033  IF                    FU32
0034          =             V0
0035  THEN  JMP TO datos
0036
0037 ""Preparar datos para escribir
0038
0039 STEP datos
0040  THEN
0041        CMP 13                       'STRCLR - Borrar string
0042          WITH          V12
0043        CMP 5                        'STRCPY - Copiara un string a otro
0044          WITH          V10
0045          WITH          V12
0046        CMP 4                        'STRAPPND - Agregar un caracter al string
0047          WITH          V12
0048          WITH          V9           "Caracater ascii <Tab>
0049        CMP 6                        'STRCAT- Concatenar dos strings
0050          WITH          V12
0051          WITH          V9
0052          WITH          V12
0053        CMP 6                        'STRCAT- Concatenar dos strings
0054          WITH          V12
0055          WITH          V8
0056          WITH          V12
0057        CMP 6                        'STRCAT- Concatenar dos strings
0058          WITH          V12
0059          WITH          V0
0060          WITH          V12
0061        CMP 6                        'STRCAT- Concatenar dos strings
0062          WITH          V12
0063          WITH          V1
0064          WITH          V12
0065        CMP 6                        'STRCAT- Concatenar dos strings
0066          WITH          V12
0067          WITH          V2
0068          WITH          V12
0069
0070 ""Escribir datos
0071        CMP 9                        'PRINTCOM - Escribir al puerto serie delimitador <CR>
0072          WITH          V1
0073          WITH          V12
0074  IF                    FU32
0075          =             V0
0076  THEN  JMP TO datos
0077
0078

-------------------------------------------------------------------------------
13 November 2011 20:52:32                                          Page  44
```

Continuación del apéndice 1.

```
FST - MONITOR (Sistema de monnitoreo de producción) - FEC Standard
Program  8 (V1) - Rutina Auxiliar - Escribir datos
-------------------------------------------------------------------------------

0079
0080 ""Leer respuesta
0081 STEP respuesta
0082  THEN
0083         CMP 13                          'STRCLR - Borrar string
0084           WITH          V12
0085         CMP 11                          'READCOM - Leer puerto seria
0086           WITH          V1
0087           WITH          V12
0088           WITH          V10
0089           WITH          V13
0090  IF                     FU32
0091           =             V-1
0092  THEN  JMP TO respuesta
0093  IF                     FU32
0094           =             V0
0095  THEN  JMP TO comparar
0096
0097 ""Comparar respuesta
0098 STEP comparar
0099  THEN  CMP 12                          'STRICMP - Compara dos strings 0 = iguales
0100           WITH          V12
0101           WITH          V18
0102  IF                     FU32
0103           =             V0
0104  THEN  JMP TO fin
0105  OTHRW
0106         JMP TO comando
0107
0108 ""Cargar comando
0109 STEP fin
0110  THEN  CMP 13                          'STRCLR - Borrar string
0111           WITH          V12
0112         CMP 5                           'STRCPY - Copiara un string a otro
0113           WITH          V20
0114           WITH          V12
0115         CMP 6                           'STRCAT- Concatenar dos strings
0116           WITH          V12
0117           WITH          V11
0118           WITH          V12
0119
0120 ""Enviar comando
0121         CMP 9                           'PRINTCOM - Escribir al puerto seria delimitador <CR>
0122           WITH          V1
0123           WITH          V12
0124  IF                     FU32
0125           =             V0
0126  THEN
0127         SET             F_Escrito      'F0.9: Datos escritos
0128
0129
0130

-------------------------------------------------------------------------------
13 November 2011 20:52:32                                          Page  45
```

Apéndice 2. **Interface de usuario para el procesamiento de datos en VB**

```
Sheet1 - 1


Private Sub CommandButton1_Click()


On Error GoTo Error
ArchivoParaAbrir = Application.GetOpenFilename("Excel Files(*.xls),*.xls", , "Abrir")
Workbooks.Open(ArchivoParaAbrir).Activate
NombreArchivo = ActiveWorkbook.Name
ActiveSheet.Range("A1:F44500").Select
Selection.Copy
Workbooks("InterfaceDeUsuarioParaElProcesamientoDeDatos.xls").Activate
Sheets("Datos Mes").Activate
ActiveSheet.Range("A2").Select
ActiveSheet.Paste
Workbooks(NombreArchivo).Close


FechaIni = Range("B2").Text
FechaIni = CDate(FechaIni)
NoDiasMes = Day(DateSerial(Year(FechaIni), Month(FechaIni) + 1, 0))
FechaIni = CStr(FechaIni)
FechaIni = Right(FechaIni, 8)
FechaIni2 = FechaIni

a = 2
j = 2
For i = 1 To NoDiasMes
Sheets("Datos Mes").Activate

If i < 10 Then
FechaAux = "0" & i & FechaIni
Else
FechaAux = i & FechaIni
End If
FechaAux = CDate(FechaAux)

VelAcum = 0
Nodatos = 0
Do While Cells(j, 2) = FechaAux
VelAcum = VelAcum + Cells(j, 6).Value
Nodatos = Nodatos + 1
j = j + 1
Loop

If Nodatos <> 0 Then
VelMedia = VelAcum / Nodatos
Else
VelMedia = 0
End If
VelMedia = Round(VelMedia, 0)
TotalUProducidas = ActiveSheet.Range("D" & j).Value
TotalUDesperdicio = ActiveSheet.Range("E" & j).Value

Sheets("Resumen diario mes").Activate
ActiveSheet.Range("A" & a).Value = FechaAux
ActiveSheet.Range("D" & a).Value = VelMedia
ActiveSheet.Range("B" & a).Value = TotalUProducidas
ActiveSheet.Range("C" & a).Value = TotalUDesperdicio
a = a + 1


Next i

Sheets("Datos Mes").Activate
UltimaFila = ActiveSheet.UsedRange.Rows.Count
ActiveSheet.Range("J6").Value = ActiveSheet.Range("D" & UltimaFila).Value
ActiveSheet.Range("J7").Value = ActiveSheet.Range("E" & UltimaFila).Value
ActiveSheet.Range("J5").Value = ActiveSheet.Range("D" & UltimaFila).Value + ActiveSheet.Range("E" & Ul
timaFila).Value
ActiveSheet.Range("J8").Value = Round(ActiveSheet.Range("D" & UltimaFila).Value / (UltimaFila - 1), 2)
```

Continuación del apéndice 2.

```
Sheet1 - 2

ActiveSheet.Range("J9").Value = Round((ActiveSheet.Range("J6").Value / ActiveSheet.Range("J5").Value
* 100, 2)


Sheets("Resumen diario mes").Activate


    Charts.Add
    ActiveChart.ChartType = xlXYScatterSmooth
    ActiveChart.SetSourceData Source:=Sheets("Resumen diario Mes").Range( _
        "A1:A31,D1:D31"), PlotBy:=xlColumns
    ActiveChart.Location Where:=xlLocationAsObject, Name:= _
        "Gráfico de Producción Mes "
    With ActiveChart
        .HasTitle = True
        .ChartTitle.Characters.Text = "VELOCIDAD PROMEDIO VRS DIA"
        .Axes(xlCategory, xlPrimary).HasTitle = True
        .Axes(xlCategory, xlPrimary).AxisTitle.Characters.Text = "Fecha"
        .Axes(xlValue, xlPrimary).HasTitle = True
        .Axes(xlValue, xlPrimary).AxisTitle.Characters.Text = "Velocidad U/min"
    End With
    With ActiveChart.Axes(xlCategory)
        .HasMajorGridlines = False
        .HasMinorGridlines = False
        .MinimumScaleIsAuto = True
        .MaximumScaleIsAuto = True
        .MinorUnit = 1
        .MajorUnit = 1
        .Crosses = xlAutomatic
        .ReversePlotOrder = False
        .ScaleType = xlLinear
        .DisplayUnit = xlNone
    End With
    With ActiveChart.Axes(xlValue)
        .HasMajorGridlines = True
        .HasMinorGridlines = False
    End With
    Application.CommandBars("Stop Recording").Visible = False
    ActiveChart.Axes(xlValue).Select
    With ActiveChart.Axes(xlValue)
        .MinimumScale = 0
        .MaximumScaleIsAuto = True
        .MinorUnitIsAuto = True
        .MajorUnitIsAuto = True
        .Crosses = xlAutomatic
        .ReversePlotOrder = False
        .ScaleType = xlLinear
        .DisplayUnit = xlNone
    End With
    ActiveChart.HasLegend = True
    ActiveChart.Legend.Select
    Selection.Position = xlTop

Sheets("Datos Mes").Activate
ActiveSheet.Range("A2").Select


Error:
End Sub

Private Sub CommandButton2_Click()
Load UserForm1
UserForm1.Show
End Sub

Private Sub Worksheet_Activate()

End Sub

Private Sub Worksheet_SelectionChange(ByVal Target As Range)

End Sub
```

Continuación del apéndice 2.

```
UserForm1 - 1

Private Sub CommandButton1_Click()

Sheets("Datos Mes").Activate
ActiveSheet.Range("A2").Select

fecha = ComboBox1.Value & FechaIni2
fecha = CDate(fecha)
j = 1
Do While Cells(j, 2) <> fecha
j = j + 1
Loop

MsgBox (j)

Do While Cells(j, 3) = 2
j = j + 1
Loop

MsgBox (j)

If ComboBox1.Text = 1 Then
RangoI = j
End If

Do While Cells(j, 3) = 1
j = j + 1
Loop

MsgBox (j)

If ComboBox1.Text = 1 Then
RangoF = j - 1
End If

If ComboBox1.Text = 2 Then
RangoI = j
End If


Do While Cells(j, 3) = 2
j = j + 1
Loop

MsgBox (j)

If ComboBox1.Text = 2 Then
RangoF = j - 1
End If

ActiveSheet.Range("A" & RangoI & ":" & "F" & RangoF).Select
Selection.Copy

Sheets("Datos día # turno").Activate
ActiveSheet.Range("A2").Select
ActiveSheet.Paste

Sheets("Resumen por hora día # turno").Activate
UltimaFila = ActiveSheet.UsedRange.Rows.Count
ActiveSheet.Range("J6").Value = ActiveSheet.Range("D" & UltimaFila).Value
ActiveSheet.Range("J7").Value = ActiveSheet.Range("E" & UltimaFila).Value
ActiveSheet.Range("J5").Value = ActiveSheet.Range("D" & UltimaFila).Value + ActiveSheet.Range("E" & Ul
timaFila).Value
ActiveSheet.Range("J8").Value = Round(ActiveSheet.Range("D" & UltimaFila).Value / (UltimaFila - 1), 2)
ActiveSheet.Range("J9").Value = Round((ActiveSheet.Range("J6").Value / ActiveSheet.Range("J5").Value)
* 100, 2)


   Charts.Add
    ActiveChart.ChartType = xlXYScatterSmooth
    ActiveChart.SetSourceData Source:=Sheets("Resumen por hora día # turno").Range( _
        "A1:A25,D1:D25"), PlotBy:=xlColumns
    ActiveChart.Location Where:=xlLocationAsObject, Name:= _
        "Gráfico de Producción Mes "
```

Continuación del apéndice 2.

```
UserForm1 - 2

    With ActiveChart
        .HasTitle = True
        .ChartTitle.Characters.Text = "VELOCIDAD PROMEDIO VRS HORA"
        .Axes(xlCategory, xlPrimary).HasTitle = True
        .Axes(xlCategory, xlPrimary).AxisTitle.Characters.Text = "Hora"
        .Axes(xlValue, xlPrimary).HasTitle = True
        .Axes(xlValue, xlPrimary).AxisTitle.Characters.Text = "Velocidad U/min"
    End With
    With ActiveChart.Axes(xlCategory)
        .HasMajorGridlines = False
        .HasMinorGridlines = False
        .MinimumScaleIsAuto = True
        .MaximumScaleIsAuto = True
        .MinorUnit = 1
        .MajorUnit = 1
        .Crosses = xlAutomatic
        .ReversePlotOrder = False
        .ScaleType = xlLinear
        .DisplayUnit = xlNone
    End With
    With ActiveChart.Axes(xlValue)
        .HasMajorGridlines = True
        .HasMinorGridlines = False
    End With
    Application.CommandBars("Stop Recording").Visible = False
    ActiveChart.Axes(xlValue).Select
    With ActiveChart.Axes(xlValue)
        .MinimumScale = 0
        .MaximumScaleIsAuto = True
        .MinorUnitIsAuto = True
        .MajorUnitIsAuto = True
        .Crosses = xlAutomatic
        .ReversePlotOrder = False
        .ScaleType = xlLinear
        .DisplayUnit = xlNone
    End With
    ActiveChart.HasLegend = True
    ActiveChart.Legend.Select
    Selection.Position = xlTop

Sheets("Datos día # turno").Activate
ActiveSheet.Range("A2").Select



End Sub


Private Sub CommandButton2_Click()
Unload UserForm1
End Sub



Private Sub UserForm_Activate()
For x = 1 To NoDiasMes

ComboBox1.AddItem Str(x)

Next
For x = 1 To 2

ComboBox2.AddItem Str(x)

Next

End Sub
```