Implementación de un dispositivo pasivo para una red Profibus-DP, basado en microcontrolador[1]

Implementation of a passive device for a Profibus-DP net-work, based on microcontroller

Recibido: 02-02-2015 Aceptado: 04 -05-2015

Asfur Barandica López[2]
Carlos Julio Ramírez Borrero[3]
Melani Gisella Viveros Moreno[4]

Resumen

Se presenta el desarrollo de una estación pasiva para entradas y salidas de propósito general, con capacidad de comunicación mediante el protocolo Profibus-DP. La implementación se basa en un microcontrolador STM32, respetando las especificaciones de las capas física y de enlace de datos definidas para el protocolo. Se describe la arquitectura hardware y software de la estación pasiva, así como las pruebas a las que fue sometida para verificar su funcionamiento. Tales pruebas involucran equipos industriales de reconocidos fabricantes, para demostrar la capacidad de interoperabilidad del dispositivo.

Palabras clave: esclavo Profibus-DP; buses de dispositivos; redes para automatización; comunicaciones industriales.

Abstract

This paper presents the development of a passive station with communication capability through the Profibus-DP protocol, for general purpose inputs and outputs. The implementation is based on an STM32 microcontroller, and it is fully compatible with the specifications of the physical and data link layers defined for the protocol. A detailed hardware and software architecture of the passive station, as well as the tests performed to verify its operation, are also described in this paper. These tests involve equipment from known industrial manufacturers to demonstrate the interoperability capability of the device.

Keywords: Profibus-DP slave; device level networks; automation networks; industrial communications.

[1] Desarrollado por el Grupo de investigación en Percepeción y Sistemas Inteligentes de la Universidad del Valle (Cali-Colombia), que viene haciendo desarrollos con el protocolo Profibus desde 2009.

[2] Colombiano. Msc. Ingeniería con énfasis en electrónica, profesor Universidad del Valle Cali, Colombia. Grupo de investigación en Percepción y Sistemas Inteligentes.
e-mail: asfur.barandica@correounivalle.edu.co.

[3] Colombiano. Ingeniero Electrónico - Postobón S.A.
e-mail: carlosjulior@gmail.com.

[4] Colombiana. Ingeniera Electrónica - ZTE Colombia S.A.S.
e-mail: melani.viveros@gmail.com.

Introducción

Profibus es, sin lugar a dudas, el protocolo de comunicación para equipos industriales con mayor número de nodos instalados en el mundo (Spiegel, 2008; Control Engineering, 2007; Lydon, 2008; Carlsson, 2015). En la región Andina son muchas las instalaciones automatizadas con esta red y las que se prevé serán instaladas en el futuro cercano. Un estudio realizado en 2008 encontró que Profibus compite con DeviceNet por el mercado de los *buses* de dispositivos en la industria del Valle del Cauca, pero que sus bondades son poco conocidas y explotadas (Barandica, 2008), lo cual genera la necesidad de capacitar personal en el diseño, operación y mantenimiento de redes de comunicación industrial.

Es claro que cuanto más profundos sean los conocimientos alrededor del protocolo, mayor será el retorno que se obtiene en la explotación de la red industrial y menores los tiempos de parada debidos a fallos y operaciones de expansión y mantenimiento. Por ello, al interior del grupo de investigación en Percepción y Sistemas Inteligentes, se definieron proyectos tendientes a formar ingenieros y a desarrollar equipos que aporten a la competitividad industrial mediante el uso adecuado de estas tecnologías. La capacidad de diseño de equipos con el protocolo Profibus implica el dominio de conocimientos profundos y permite ofrecer soluciones no convencionales, con buena relación costo/beneficio, a los usuarios de las redes Profibus.

La mayoría de las publicaciones relacionadas con Profibus en Colombia y los países vecinos se limitan a describir la interconexión de equipos de fabricantes extranjeros usando el protocolo (Córdoba y Sandoval, 2007; Otáñez, 2009; Echeverri y Grisales, 2013). Se han reportado implementaciones propias, como es el caso de Calderón (2004), donde además de un esclavo profibus, se desarrolló un software que hace las veces de maestro de la red; sin embargo, no hay un procedimiento confiable de validación, dado que todos los componentes del sistema son elaborados por el autor; en Cardona y Castañeda (2010), se construyó un esclavo Profibus y se validó convenientemente, aunque el desarrollo se apoyó en dos Circuitos Integrados de Aplicación Específica (ASIC) (UFC100-F1, AMIS-492x0) que reducen al mínimo el trabajo sobre la pila de comunicaciones de Profibus. Publicaciones recientes de países desarrollados indican que el tema abordado tiene relevancia y potencial comercial (Xu *et al.*, 2011; Yanjun *et al.*, 2007).

Se presenta la implementación de un dispositivo pasivo (esclavo) para una red Profibus-DP, la cual, a diferencia de las mencionadas anteriormente, se llevó a cabo sin utilizar circuitos integrados específicos para el protocolo, por lo que fue necesario desarrollar toda la pila de comunicaciones sobre una plataforma de procesamiento digital basada en el microcontrolador STM32. Se presenta también el procedimiento de validación que fue ejecutado para dar confiabilidad a los resultados.

El artículo se organizó de la siguiente manera: en la metodología se describen muy brevemente las características del protocolo y las especificaciones mínimas de cada tipo de dispositivo con capacidad Profibus, seguido por el diseño del esclavo y su definición, la selección del hardware y la arquitectura del software; las pruebas a las que fue sometido el prototipo y se consignan los resultados, las conclusiones y las perspectivas del trabajo.

Metodología

Inicialmente se abordó el estudio del protocolo Profibus-DP y se definieron las especificaciones del esclavo, para poder así establecer las características mínimas de la plataforma de procesamiento digital y del hardware del esclavo en general. Se llevó a cabo la implementación del esclavo y su conexión con un simulador de maestro para dar inicio a las pruebas básicas de comunicación. Se creó el archivo GSD usando el programa *GSD Editor5* desarrollado por Profibus *Nutzerorganisation* e.V. Para ello se tomó como ejemplo el archivo GSD del variador de velocidad Micromaster440. Los archivos GSD proporcionan una descripción clara y comprensiva de los dispositivos, especificando las características de comunicación en un formato definido. Finalmente se agregó el esclavo a una red Profibus para realizar las pruebas definitivas de comunicación.

A continuación se hace una breve descripción del protocolo, antes de mostrar el diseño.

Profibus-DP (Decentralized Peripherals)

Profibus-DP es un protocolo de comunicación entre equipos industriales que permite altas ratas de transferencia de datos de manera confiable. En una red

Profibus-DP se diferencian dos tipos de equipos: los maestros o estaciones activas y los esclavos o estaciones pasivas. Los maestros envían los datos de salida al esclavo y obtienen una respuesta que incluye sus datos de entrada.

Existen dos tipos de estaciones activas en el protocolo DP:

- **Maestro DP clase 1:** Se trata generalmente de los equipos que realizan el control central mediante el intercambio de información con los equipos distribuidos o esclavos.

- **Maestro DP clase 2:** Son los dispositivos usados para programación, configuración o diagnóstico. Son parametrizados en la puesta en marcha con el fin de especificar la configuración del sistema DP. Esta configuración suministra información como el número de dispositivos DP, direcciones de las estaciones del bus y de los dispositivos de entrada y salida.

Los dispositivos pasivos o esclavos son periféricos que generalmente realizan funciones de I/O, recogiendo el estado de los sensores y entregando valores a los actuadores del proceso.

La pila de comunicaciones del protocolo se organizó según el modelo de referencia OSI para interconexión de sistemas abiertos. Profibus define únicamente las capas física y de enlace de datos. La capa física basada en par trenzado de cobre, hace uso de la interfaz estandarizada RS-485, la cual permite usar velocidades de transferencia de hasta 10 Mbps y hasta 32 estaciones en un segmento; la transmisión es asíncrona con 8 bits de datos y paridad par. Alternativamente, el protocolo también acepta el uso de fibra óptica.

La capa de enlace de datos se encarga de hacer que el enlace sea fiable y proporciona los medios para activar, mantener y desactivar el enlace. También se encarga de la detección y control de errores. Los formatos de trama de Profibus se resumen en la Tabla 1. Una estación pasiva sólo debería estar en capacidad de interpretar tres de estos formatos (2, 3 y 5), pues los otros dos se usan para comunicación entre estaciones activas. Las transacciones de transferencia de datos incluyen la emisión de una trama de solicitud del maestro al esclavo, en la que se envían los datos de salida, y de una trama de respuesta del esclavo al maestro, en la cual se insertan los datos de entrada. Una descripción más completa del protocolo puede encontrarse en (Acromag, 2002).

Tabla 1. Tipos de tramas Profibus.

Tipo de trama	Campos usados de la trama											
	SD	LE	LEr	SD	DA	SA	FC	DSAP	SSAP	DU	FCS	ED
1. Búsqueda de nuevos maestros	10H				x	x	x				x	16H
2. Transferencia de datos	68H	x	x	68H	x	x	x	x	x	X	x	16H
3. Transferencia de datos	A2H				x	x	x			8x	x	16H
4. Paso de testigo entre maestros	DCH				x	x						16H
5. Reconocimiento corto	E5H											

SD: delimitador de inicio de trama

LE: byte de longitud (DA+SA+FC+DSAP+SSAP+DU)

LEr: byte de longitud repetido (el mismo contenido de LE)

DA: dirección del nodo destino

SA: dirección del nodo fuente

FC: byte de control de trama o código de función

DSAP: punto de acceso al servicio en el destino

SSAP: punto de acceso al servicio en la fuente

DU: unidad de datos. 8 bytes con SD=A2H y entre 1 y 244 bytes con SD=68H

FCS: byte de control de errores

ED: delimitador de fin de trama

Los campos en blanco no son usados. Así, la trama tipo 1 tiene siempre 6 bytes y la trama tipo 5, 1 byte. X indica que este campo contiene un número variable de bytes (1 a 244). 8x indica que este campo contiene 8 bytes.

Fuente: Los autores

La primera versión de Profibus se denominó DP-V0. En ella se definieron las especificaciones DP originales que permiten transferencia de datos cíclica entre maestro y esclavo, la parametrización y la lectura de datos de diagnóstico para una rápida localización de fallos. La versión DP-V1, que surgió como una extensión de DP-V0, agrega la comunicación acíclica de datos maestro/esclavo, lo que constituye un buen complemento para la modificación de parámetros, ya que permite transmisión en forma paralela de información del proceso (transmitida cíclicamente) y de los parámetros de configuración, transmitida acíclicamente y con menor prioridad; también añade el reconocimiento de alarmas. La versión DP-V2 adiciona a las versiones anteriores el intercambio de información esclavo/esclavo, permitiendo alcanzar altas velocidades en la sincronización de dispositivos sobre la marcha y sincronización de reloj.

Especificaciones del esclavo y requerimientos del hardware

Se especificó un esclavo con capacidad de manipulación de cuatro entradas y cuatro salidas digitales de propósito general. Con un *watchdog* para garantizar la robustez del diseño ante el ruido y las variables ambientales, y dotado de una interfaz serial asíncrona destinada a la comunicación en la red Profibus. Fue el primer desarrollo sobre este protocolo, se optó por una velocidad objetivo relativamente baja de 57,6 kbps.

Para una red que funcione a velocidades de hasta 57,6 kbps, se determinó que un microcontrolador de 8 bits con desempeño de 1 MIPS podría atender sin retardos las solicitudes del maestro. De igual manera, se consideró apropiado un tamaño de 32 kbytes de memoria para el almacenamiento del programa y una memoria RAM interna de al menos 2 kbytes para el manejo de las variables temporales generadas durante la ejecución del programa. Es indispensable la presencia de una memoria no volátil de al menos 256 bytes para el almacenamiento de los parámetros de configuración del nodo.

En vista de que ningún microcontrolador incluye el driver/receiver para el manejo de la interfaz RS-485, es imprescindible su colocación externa; además, siempre se recomienda el aislamiento eléctrico de las señales del bus RS-485, con el propósito de proteger el nodo ante posibles fallos en la red y viceversa. Cumpliendo las especificaciones de capa física de Profibus, se debe usar un conector DB9 hembra en el esclavo, para colocar las señales del bus RS-485 en el medio de transmisión.

Arquitectura hardware del esclavo

Como soporte hardware de la unidad pasiva se seleccionó la placa de desarrollo *ET-STM32 Stamp* que cuenta con un procesador ARM de 32 bits Cortex™-M3, CPU de 72 MHz de frecuencia máxima, el cual ofrece un rendimiento excepcional de cómputo (90 MIPS) y un avanzado sistema de respuesta a las interrupciones, memorias integradas de alta velocidad, memoria flash para programa de 512 kbytes, SRAM de hasta 64 kbytes y un conjunto interesante de unidades periféricas (ADC de 12 bits, temporizadores de 16 bits, subsistemas PWM, interfaces de comunicación I2C, SPI, I2S, SDIO, USB, CAN y USART). Dado que la tarjeta no dispone de EEPROM, se hizo una emulación de EEPROM usando la memoria Flash, como se describe en ST Microelectronics, 2009.

En la Figura 1 se muestra el diagrama en bloques del esclavo Profibus. La interfaz con la red Profibus se compone de dos etapas: aislamiento eléctrico y acople con el medio. El aislamiento se realizó con optoacopladores MCT6 para las señales de recepción (Rx), transmisión (Tx) y habilitación de transmisión de datos (DE). Estos dispositivos limitan la rata de bits a un máximo de 150 kbps, aunque existen otras referencias (6N137, VO2601) con las que se pueden lograr hasta 10 Mbps. La etapa de acople con el medio convierte las señales referidas a tierra del microcontrolador a señales diferenciales y adecúa los niveles de tensión e impedancias, a las definidas por la especificación RS-485; para este propósito se seleccionó el driver/receiver MAX485. Como parte de esta etapa también se añadió el conector DB9 hembra para la conexión al medio.

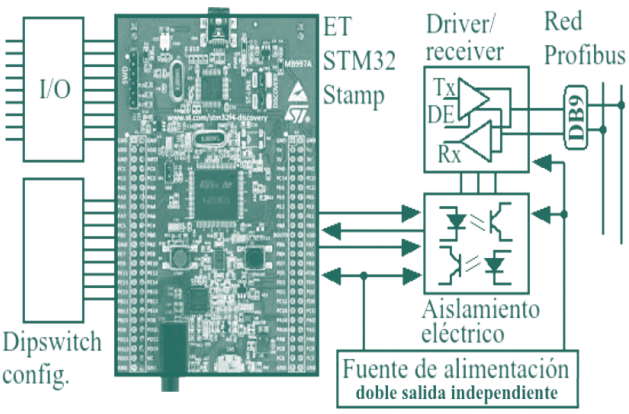

Figura 1. Diagrama de bloques del hardware del esclavo
Fuente: Los autores

Se colocó un conjunto de conmutadores tipo *dipswitch* para definir manualmente la dirección del esclavo y un led para confirmar al usuario que los procesos de parametrización y configuración se realizaron exitosamente; el led parpadea a diferentes velocidades según sea el estado en el que se encuentra, hasta quedar completamente encendido al alcanzar el estado *intercambio de datos*.

Arquitectura software

La implementación de la capa de enlace de datos se realizó a través de la programación del microcontrolador de la placa de desarrollo STM32F103. El programa se escribió en lenguaje C y se compiló en el ambiente de desarrollo *Keil uVision4,* que genera un archivo con extension *.hex.* El micrococontrolador posee un *bootlader* de fábrica, que permite que la descarga del archivo *.hex* a la plataforma se realice a través del puerto RS-232 del computador mediante el software *Flash Loader Demonstrator*.

En su vista más general, el programa desarrollado implementa la máquina de estados que rige el comportamiento del esclavo Profibus, la cual es presentada en la Figura 2 (a).

Al encenderse, después de un reset o tras la activación del temporizador *watchdog*, el esclavo estará en el estado *Power_ON/Reset*, durante el cual hace la inicialización de los recursos hardware y los parámetros en memoria que definen el comportamiento del nodo. Si la dirección de esclavo es válida (cualquier valor entre 0 y 125), la máquina de estados pasará automáticamente al estado ***Parametrización***; si la dirección no es válida, esperará por un mensaje de asignación de dirección. En el estado ***Parametrización***, el esclavo espera el telegrama del maestro, el cual identifica la estación que le ha sido asignada como maestro y el modo de operación. Una vez se reciben los parámetros, el esclavo pasa al estado ***Configuración_I/O***, en el cual el esclavo espera un telegrama que especifica el número de bytes de entrada y salida que deberán ser intercambiados en cada transacción posterior. En los estados ***Parametrización y Configuración_I/O*** serían inicializados los diferentes periféricos del microcontrolador para atender las demandas en cuanto a la funcionalidad del esclavo. Realizada la configuración, el esclavo queda en el estado ***Intercambio_datos***, en el que se pueden intercambiar datos de I/O, solicitar diagnósticos o modificar los parámetros y la configuración del esclavo.

Por ningún motivo el esclavo toma la iniciativa para enviar un mensaje; debe esperar siempre un mensaje de solicitud por parte de un maestro. El programa entonces permanece capturando y analizando las tramas que circulan por la red y ejecutando las acciones pertinentes, de acuerdo con el estado actual. La atención a los telegramas del maestro se efectúa siguiendo el diagrama de flujo de la Figura 2 (b).

El bloque ***Recepción_de_trama*** captura los caracteres recibidos por la Interfaz para Comunicaciones Síncronas y Asíncronas (USART) siguiendo la máquina de estados de la Figura 2 (c), que es controlada principalmente por la Interrupción por Recepción (RI) y el dato recibido en el Buffer de Recepción de la USART (SBUF). Se usa un contador para capturar el número de bytes esperado según el tipo de trama. Se generan errores si el delimitador de fin de trama no está en la posición esperada o si el byte de repetición de longitud no coincide con la longitud anunciada. Terminado el proceso de recepción de una trama, el bloque ***Trama_válida?*** hace la validación de la secuencia de verificación (FCS en la Tabla 1) y determina si la trama va dirigida al esclavo, en cuyo caso el bloque ***Análisis_de_trama*** hace un análisis de los campos restantes: SA, FC, DSAP, SSAP y DU.

Después de analizar la trama recibida, el bloque ***Generación_respuesta*** se encarga de construir la trama de respuesta y enviarla al maestro. Para ello, los diferentes campos de la trama de respuesta son almacenados en una variable tipo Array para luego ser transmitidos por la USART. El mecanismo de Acceso Directo a Memoria (DMA) controla por completo la transmisión directa de los datos del arreglo hacia el registro de datos de la USART, hasta que el contador del DMA llegue a cero; con ello se libera a la CPU de la gestión de interrupciones de transmisión cada que se envía un byte de la trama; de hecho, antes de entregar el control de la transmisión al DMA, las interrupciones la USART deben ser deshabilitadas (Hitex, 2009).

Finalizada la transmisión de una trama, el driver de transmisión debe deshabilitarse para permitir el uso del canal de comunicación por parte de las demás estaciones de la red y deben habilitarse las interrupciones en recepción generadas por la USART.

Pruebas y resultados

Las pruebas del esclavo se centraron en tres aspectos: capacidad para operar en el bus sin generar interferencias sobre otros dispositivos, conformidad con la especificación del protocolo y determinación de la velocidad máxima de operación.

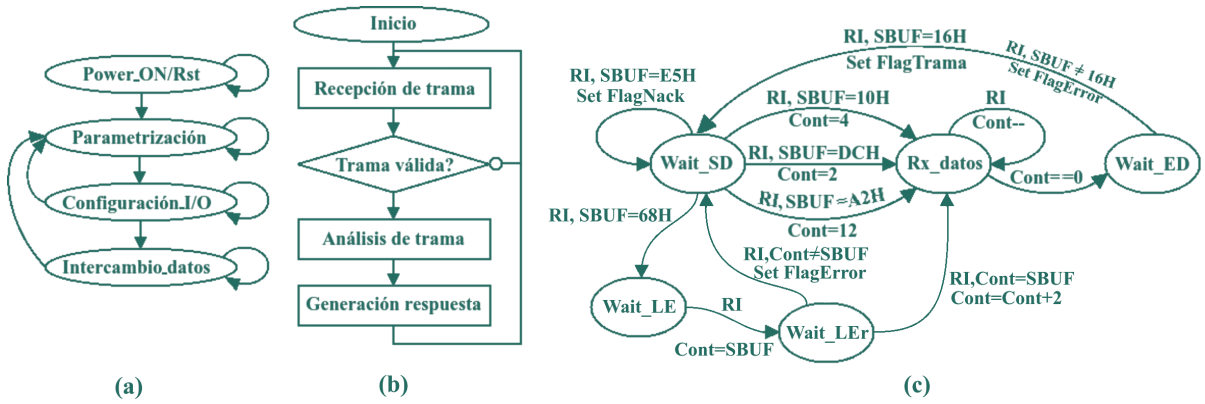

Figura 2. Diagramas de estados y de bloques del programa.
Fuente: Los autores

En sus primeras fases de desarrollo, el esclavo fue probado enviando tramas Profibus desde un computador portátil dotado de un convertidor USB/RS-485 (USOPTL4-LS de B y B Electronics), que simulaba el accionar de un maestro; se utilizó un programa para manipulación de puertos seriales (*232Analyzer*) que permite componer manualmente las tramas a ser enviadas y recibir las respuestas. En una fase posterior, se insertó el dispositivo en una red conformada por equipos industriales y software de reconocidos fabricantes; se configuró la red Profibus a 9600 bps con dos variadores de velocidad Micromaster 440 de Siemens y una tarjeta CP-5611, instalada en un slot PCI del PC que hace las veces de maestro de la red. El esquema completo se muestra en la Figura 3.

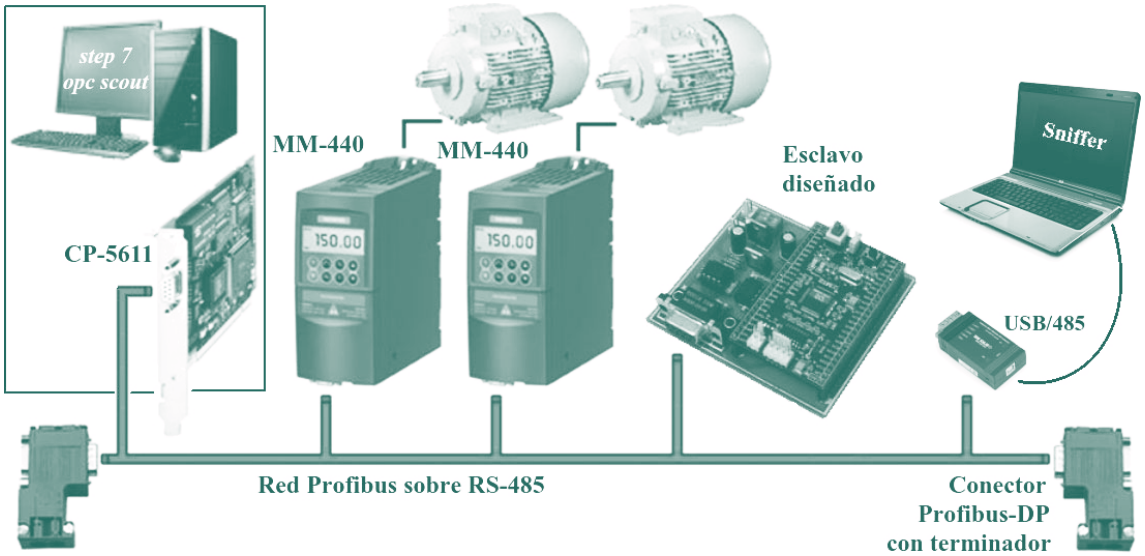

Figura 3. Esquemas de la red para las pruebas del esclavo.
Fuente: Los autores

Con la aplicación ***Step 7*** se realizó la configuración de la red (asignación de direcciones, velocidad de comunicación, definición del servidor OPC). Antes de agregar a la red el dispositivo esclavo basado en STM32, se cargó su archivo GSD con el fin de que el ***Step7*** lo reconociera e instalara fácilmente. Una vez configurada la red, se realizaron pruebas básicas de conectividad con los variadores, teniendo en cuenta que los Micromaster 440 usan el perfil Profidrive. El programa *232Analyzer* se usó en este caso a manera de *sniffer* de la red, para capturar el tráfico y comprobar las secuencias de tramas requeridas para la inicialización de los dispositivos por parte del maestro. Pudo verificarse que la operación de los variadores de velocidad no fue afectada por la presencia del esclavo diseñado, pues desde el computador se intercambiaron datos

con los variadores, por medio del software *OPC Scout* de Siemens.

Asimismo, la conformidad con la especificación del protocolo se da por satisfecha, teniendo en cuenta que el esclavo superó los estados iniciales hasta quedar en modo de intercambio de datos. Aunque el servidor OPC *Scout* permitió la visualización de todas las tramas, entregó únicamente la información del campo DU de las tramas que lo incluyen; por ello, el *sniffer* fue de gran utilidad, pues captura las tramas completas con cualquiera de los delimitadores, permitiendo verificar que no se presentaron reconocimientos negativos o intentos fallidos de comunicación por parte del maestro.

En la fase de operación o intercambio de datos, fue posible leer el estado de un conjunto de entradas digitales y alterar el estado de un conjunto de salidas digitales. De igual manera, se realizaron operaciones de cambio de dirección. Estos intercambios se llevaron a cabo mientras los demás equipos de la red estaban intercambiando datos activamente, lo cual asegura la interoperabilidad del esclavo desarrollado. La Figura 4 muestra la red completa implementada en *Step7* y los mensajes de comunicación exitosa del *OPC Scout*.

Finalmente se incrementó la velocidad de la red a 19,2 kbps sin que se produjeran cambios en su funcionamiento.

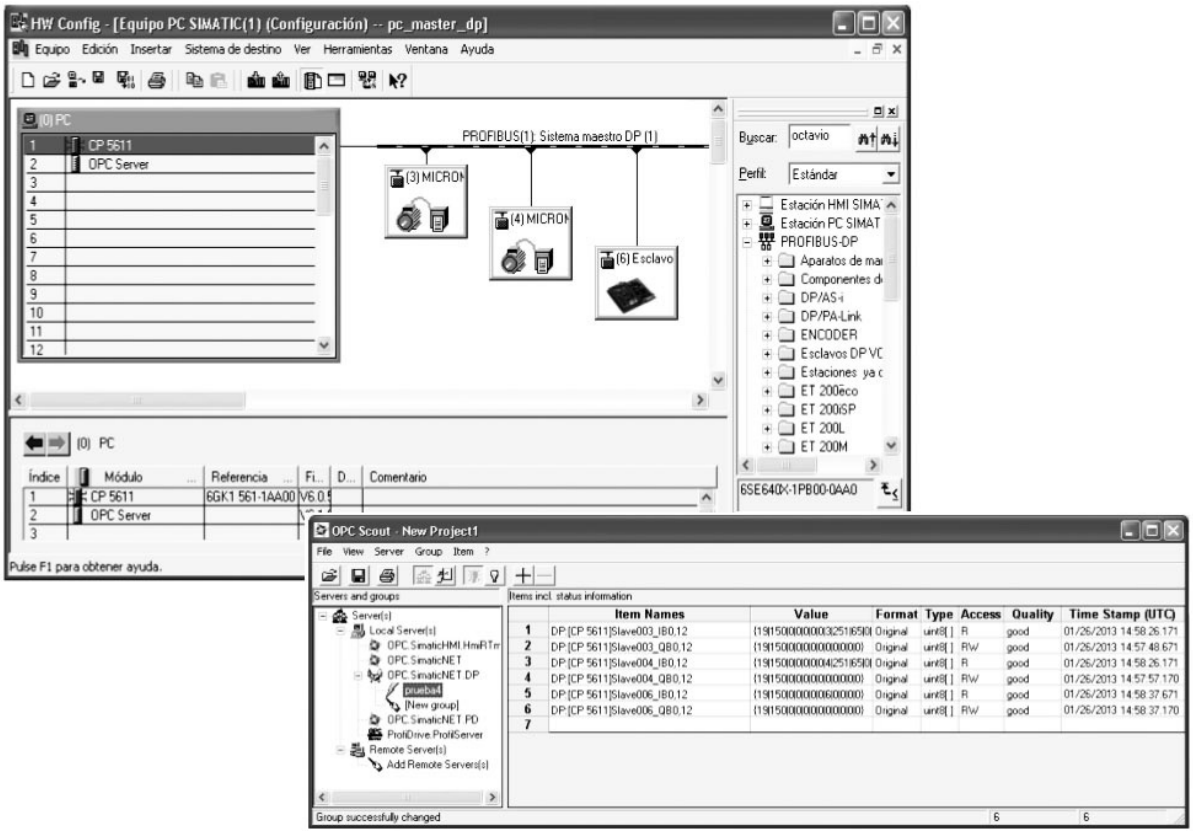

Figura 4. Captura de pantalla mostrando la red y mensajes intercambiados exitosamente.
Fuente: los autores

Conclusiones

Se realizó la implementación de una estación pasiva para una red Profibus-DP usando una plataforma microcontrolada STM32. Ello implica que la pila de comunicaciones tuvo que ser desarrollada completamente, a diferencia de las implementaciones que emplean circuitos ASIC. También fue necesario el desarrollo de un archivo tipo hoja de datos electrónica (archivo GSD), como condición imprescindible para que el maestro realizara la parametrización y configuración del esclavo. El esclavo fue insertado en una red conformada por equipos industriales, demostrando su capacidad de intercambiar datos con el maestro sin interferir en las comunicaciones con los demás

dispositivos de la red. En la red estaban presentes dos esclavos, un maestro y un equipo de diagnóstico (*snnifer*). Las pruebas fueron realizadas sucesivamente a 9600 y 19200 bps, sin que aparecieran problemas debidos al incremento en la velocidad.

El esclavo permitió el cambio de dirección, la parametrización, la configuración de I/O y el intercambio de datos usando las tramas y las funciones definidas por la especificación del protocolo.

Más que el diseño de un producto, este proyecto muestra la capacidad de ofrecer a la industria nacional soluciones que pueden mejorar los procesos productivos con costos inferiores a los que tendrían las opciones importadas que ofrece el mercado. Con él se ha demostrado la capacidad de diseño de la ingeniería colombiana y se han generado documentos y herramientas útiles para realizar nuevas implementaciones y capacitar personal en el área de las comunicaciones industriales, particularmente en el protocolo Profibus-DP. Se pretende que en una versión posterior, el esclavo incluya las capacidades *Freeze* y *Sync*, pueda intercambiar datos de variables analógicas y opere exitosamente hasta 500 kbps .

Referencias

Acromag Incorporated (2002*). Introduction to Profibus DP. BusWorksTM 900PB Series ProfiBus/RS485 Network I/O Modules Technical Reference, USA*. Recuperado de: http://www.acromag.com/sites/default/files/Acromag_Intro_ProfibusDP_698A.pdf.

Barandica A. (2008). *Una propuesta para la apropiación de la tecnología HART y su transferencia hacia el sector industrial*, (tesis de maestría) Universidad del Valle, Cali, Colombia.

Calderón Porras E. (2004). Implementación del protocolo Profibus de redes de comunicación industriales para la automatización de sistemas de manufactura, *Revista Colombiana de Tecnologías de Avanzada*, 2(4), 17-24.

Cardona C.M. y Castañeda A.M. (2010). *Convertidor 4-20mA a Profibus PA,* (tesis de http://www.automationworld.com/networking-amp-connectivity/pto-step-profinet-promotion).

Carlsson T. (2015). *HMS' view of the different networks on the market in 2015.* January 22. Recuperado de: http://www.anybus.com/readnews.asp?NID=177.

Castañeda Verano, A.M. y Cardona Barón, C.M. *Convertidor 4-mA a Profibus PA*, (Tesis de pregrado) Pontificia Universidad Javeriana, Bogotá, Colombia. Recuperado de: http://repository.javeriana.edu.co/bitstream/10554/7002/1/tesis451.pdf.

Control Engineering (2007). *Digital network: With 20 million nodes, Profibus is now mainstream technology*, Control Engineering, Agosto 14. Recuperado de: http://www.controleng.com/index.php? id=2805&tx_ttnews[tt_cHash=95bae4cd19569a06ea7e406e4cc9aa36.

Córdoba J.E. y Sandoval A.M. (2007). *Implementación de un sistema supervisorio sobre una red industrial Profibus-Ethernet en un caso de estudio*, (tesis de pregrado) Universidad del Cauca, Popayán, Colombia.

Echeverri S.M. y Grisales G.A. (2013). *Implementación de una red Profibus-DP en un sistema automatizado*, (tesis de pregrado) Universidad Tecnológica de Pereira, Pereira, Colombia. Recuperado de: http://repositorio.utp.edu.co/dspace/bitstream/11059/4353/1/6298E18.pdf.

Hitex (UK) Ltd. (2009). *The insider's guide to the STM32 ARM based microcontrollers.* Coventry, Reino Unido. Recuperado de: http://www.hitex.com/fileadmin/pdf/insiders-guides/stm32/isg-stm32-v18d-scr.pdf.

Lydon B. (2008). PROFIBUS Hits New Highs... over 25 million installed nodes!. *Rev. Automation.* Recuperado de: http://www.automation.com/library/articles-white-papers/articles-by-bill-lydon/pto-2008-annual-meeting.

Otáñez J.P. (2009*). Modernización de dos sierras de corte para el proceso de elaboración de tablas en la planta industrial Aglomerados Cotopaxi S.A.*, (tesis de pregrado) Escuela Politécnica Nacional, Quito, Ecuador. Recuperado de: http://bibdigital.epn.edu.ec/bitstream/15000/1437/1/CD-2135.pdf.

Spiegel R. (Septiembre 2008). PTO to step up Profinet promotion, *Rev. Automation World*. Recuperado de: http://www.automationworld.com/networking-amp-connectivity/pto-step-profinet-promotion.

ST Microelectronics (2009). *Application note AN2594: EEPROM emulation in STM32F10x microcontrollers*.Recuperado de: http://www.st.com/web/en/resource/technical/document/application_note/CD00165693.pdf.

Xu Hongyan, Wu Guichu, Chen Chong. (2011). *Based on STM32F103 Implement Profibus-DP Slave with High-speed Transmission*, International Conference on Uncertainty Reasoning and Knowledge Engineering, Bali, August 4-7, Vol. 1, 232-234. Recuperado de: http://www.ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6007805.

Yanjun Fang, Jingyu Liu, Bo Xi. (2007). *Development of Multifunctional Controller Based on PROFIBUS-DP*, IEEE International Conference on Automation and Logistics, August 18 - 21, Jinan, China, 144–148. Recuperado de: http://www.ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4338546.