iberchip

XVI workshop

Proceedings

Iguaçu Falls, Brazil
February 24-26, 2010

Promoted by

Co-Sponsored by

In-Cooperation with

Organized by

Supported by

iberchip
XVI workshop

Iguazu Falls, Brazil
February 23-25, 2010

Table of Contents

ORAL PRESENTATIONS

POSTER PRESENTATIONS

Implementación sobre FPGA de un cliente SNTP usando MicroBlaze

J. Quiros, J. Viejo, A. Muñoz, A. Millan, E. Ostua and J. I. Villar
Grupo ID2 (Investigación y Desarrollo Digital)
Departamento de Tecnología Electrónica - Universidad de Sevilla
E. T. S. Ing. Informática, Campus Universitario Reina Mercedes
41012 Sevilla (SPAIN)
Email: jquiros@dte.us.es, julian@dte.us.es, amrivera@dte.us.es, amillan@us.es,
ostua@dte.us.es, jose@dte.us.es

***Abstract*—En este trabajo se describe la implementación de un cliente SNTP sobre un dispositivo FPGA usando el microprocesador MicroBlaze. El objetivo es conseguir un diseño compacto, de bajo coste y alta precisión para ser integrado en las unidades terminales remotas (RTU) usadas en entornos industriales. El sistema se compone de una plataforma hardware y otra software diseñadas a medida que implementan las funciones necesarias para poder sincronizarse con un servidor NTP/SNTP a través de una red de área local. El dispositivo proporciona a la RTU información temporal precisa a través de un puerto serie, emulando el comportamiento de un receptor de GPS. Los resultados obtenidos demuestran una precisión en la sincronización del orden de decenas de microsegundos.**

I. Introducción

La sincronización temporal es crítica en los sistemas de control industrial. Un caso particular lo constituyen los sistemas de adquisición de datos a través de unidades terminales remotas (RTU). En este contexto, la norma IEC-61850 [6] establece el protocolo SNTP[1] [3] sobre Ethernet como un estándar en el proceso de sincronización temporal de los dispositivos que componen una red industrial.

El protocolo SNTP y el NTP[2] [1] (implementado por los sistemas operativos más comunes y gran parte de dispositivos de nivel 3 del modelo OSI[3]) comparten el protocolo de comunicación y el formato de datos. La diferencia entre ellos reside en el procesado que efectúan tras recibir los datos procedentes del servidor de sincronización. El NTP es un protocolo orientado a redes asimétricas donde no es posible controlar su carga y congestión, como por ejemplo la red Internet. Así, un cliente NTP utiliza la información procedente de varios servidores para calcular la latencia de la red y el desplazamiento de su reloj local respecto a los servidores y conseguir una correcta sincronización mediante un procesamiento de estos parámetros calculados. Con respecto al SNTP, uno de los entornos donde puede ser utilizado consiste en redes privadas donde sus latencias puedan ser estudiadas y controladas de forma que un único servidor es suficiente para conseguir una alta precisión. No obstante, ambos protocolos son completamente compatibles, pudiendo conectarse un servidor/cliente NTP con un cliente/servidor SNTP.

Este trabajo describe el diseño e implementación de un cliente SNTP muy compacto y de bajo coste para ser usado como sistema empotrado en unidades terminales remotas. El diseño se basa en el microprocesador Microblaze [10] de Xilinx[4] y puede ser integrado en un dispositivo FPGA de gama baja. El sistema es capaz de sincronizarse con un servidor NTP que opere en la misma red local y proporciona una datos NMEA[5] [4] a través de un puerto serie, emulando a un receptor de GPS. Estas características hacen que el dispositivo diseñado sea idóneo para suministrar información de sincronización precisa a unidades terminales remotas.

El documento se ha organizado de la siguiente forma: en primer lugar se describe el protocolo SNTP (sección II), a continuación se exponen las especificaciones del sistema (sección III), seguido de los detalles más relevantes de la arquitectura del sistema (sección IV). Por último se presentan los resultados obtenidos (sección V) y las conclusiones (sección VI).

II. Protocolo NTP/SNTP

La operación del protocolo NTP/SNTP es muy simple (Fig. 1). El cliente envía una petición al servidor mediante la emisión de un paquete UDP donde se incluye la hora de su reloj local (T_1). Cuando el servidor recibe la petición genera una nueva marca de tiempo (T_2) con la hora de recepción (dada por el reloj local del servidor). Después de procesar la petición, el servidor emite una respuesta incluyendo las dos marcas anteriores y la hora a la que la respuesta abandona el servidor (T_3). Cuando el cliente recibe la respuesta anota la hora de llegada (T_4).

Por lo tanto, el protocolo NTP/SNTP se basa en cuatro marcas de tiempo:

- Originate Timestamp (T_1). Hora en la que el cliente envía la petición al servidor.
- Receive Timestamp (T_2). Hora en la que el servidor recibe la petición del cliente.
- Transmit Timestamp (T_3). Hora en la que el servidor envía la respuesta al cliente.

[1]SNTP: Simple Network Time Protocol
[2]NTP: Network Time Protocol
[3]OSI: Open System Interconnection
[4]Xilinx: http://www.xilinx.com
[5]NMEA: National Marine Electronics Association

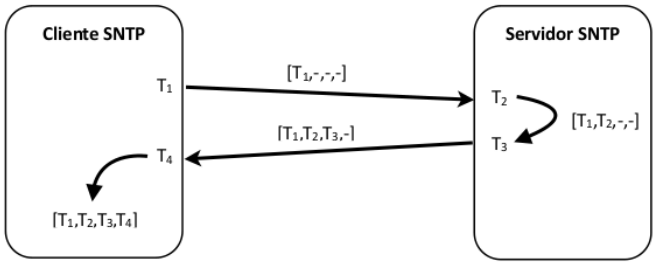

Figura 1. Operación del protocolo NTP/SNTP.

- Destination Timestamp (T_4). Hora en la que el cliente recibe la respuesta del servidor.

A partir de éstas, el cliente puede calcular el round trip time (t_{rd}) y el time offset (t_{offset}). Asumiendo una conexión simétrica podemos definir estos tiempos aplicando las siguientes fórmulas:

$$t_{rd} = (T_4 - T_1) - (T_3 - T_2)$$
$$t_{offset} = \frac{(T_2 - T_1) + (T_3 - T_4)}{2} \quad (1)$$

Usando el offset calculado, el cliente puede corregir su reloj local para ajustarse a la hora del servidor. En la sincronización de la hora local del cliente respecto a la del servidor existen, principalmente, dos fuentes de error. La primera es la asimetría en la comunicación de red, donde el tiempo de llegada de la petición al servidor difiere del tiempo de regreso de la respuesta al cliente. Esto se debe a latencias no predecibles en los equipos de la red, especialmente cuando se incrementa el número de dispositivos involucrados y se empiezan a detectar colisiones. La segunda fuente de error se produce al registrar las marcas de tiempo en el datagrama. Lo ideal sería que T_1/T_3 reflejaran exactamente la hora en la que el datagrama abandona el cliente/servidor y T_2/T_4 la hora en la que el servidor/cliente reciben el datagrama.

De los dos errores posibles, el primero es dependiente de las características de la red y, por lo tanto, independiente del cliente. No ocurre lo mismo con la segunda, que dependerá de la implementación del cliente y servidor. En este caso, el error producido será mayor cuanto mayor tiempo transcurra entre que las marcas son registradas en el datagrama y el instante real en que éste abandona o alcanza al dispositivo. Por lo tanto, la calidad de las distintas implementaciones posibles, en lo que se refiere a sincronización, se medirá en función de este tiempo.

Las implementaciones software de NTP típicamente consiguen tiempos de sincronización en el orden de un milisegundo con respecto al servidor. En estas implementaciones, las marcas de tiempo son registradas por clientes/servidores corriendo como una aplicación a nivel de usuario (Fig. 2), de forma que el error en la marca de tiempo dependerá del tiempo empleado en procesar el datagrama y en subir la pila de protocolos y capas software, que son dependientes de la carga del sistema,

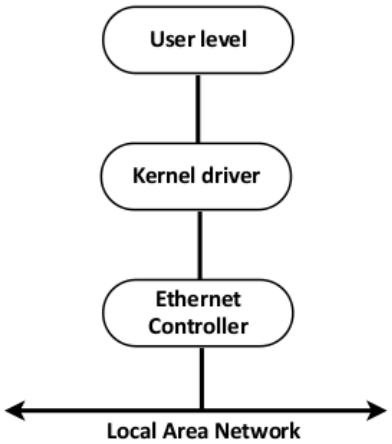

Figura 2. Capas donde NTP/SNTP puede ser implementado.

correcta implementación software, etc. No obstante, algunos sistemas operativos como Linux o FreeBSD [2] soportan todo el procesado en el kernel (Fig. 2), consiguiendo mejorar la precisión.

La precisión de sincronización del protocolo NTP puede ser ampliamente mejorada si la operación de registrar las marcas de tiempo se realiza en las capas más bajas [5], obteniendo la mejor precisión si el sellado temporal se realiza en el hardware del dispositivo Ethernet (Fig. 2) tan pronto como los paquetes lleguen o abandonen el interfaz de red. Esto último sólo es posible si se realiza una modificación del controlador Ethernet a nivel hardware y se desarrolla un driver específico.

Para corregir los errores en el proceso de sincronización, el estándar NTP plantea una serie de mecanismos avanzados que hacen posible su uso en redes de latencia variable y que se basan en el envío de mensajes a diferentes servidores de hora. Sin embargo, implementar este conjunto de mecanismos en hardware puede resultar inviable debido a su gran complejidad. No obstante, la versión simplificada de este protocolo (SNTP) si resulta adecuada para ser implementada completamente en hardware ya que sólo cubre la sincronización con un único servidor, usa un algoritmo simplificado y comparte el mismo protocolo de comunicación y formato de datos.

En esta línea, el Grupo ID2 ha desarrollado una implementación completamente hardware de un cliente SNTP [7]–[9], consiguiendo una precisión por debajo de los 10 microsegundos. Esta precisión se debe a que el marcado temporal se realiza en el hardware del controlador Ethernet MAC, así como el control de la deriva. No obstante, esta implementación presenta determinados aspectos inherentes a cualquier desarrollo hardware que hacen necesario la exploración de otras alternativas. Entre estos aspectos podemos citar una mayor complejidad en las fases de diseño, implementación, mantenimiento y configuración del sistema.

Estos puntos implícitos al diseño hardware y, por lo tanto, inevitables, hacen pensar en otras soluciones basadas en una combinación de hardware y software. Normalmente, estas soluciones se basan en el diseño de un SoPC (System On

Programmable Chip), consiguiendo una reducción bastante considerable en la complejidad del sistema, a la misma vez que facilita la inclusión de nuevas funcionalidades, como por ejemplo un servidor web. Este decremento será directamente proporcional al número de funcionalidades que se traspasen de la parte hardware a la software.

No obstante, el planteamiento de un sistema de este tipo no siempre es viable. Existe un factor determinante a la hora de optar entre un diseño u otro: el rendimiento, definido en este caso como la calidad en la precisión temporal del dispositivo. Una implementación hardware siempre ofrecerá un rendimiento superior (igual en el peor de los casos y, evidentemente, en igualdad de condiciones) que una hardware/software, ya que la primera elimina las latencias producidas en la pila de protocolos y características de implementaciones software comentadas anteriormente. Por lo tanto, únicamente cabe pensar en una solución hardware/software si su rendimiento es suficiente para la aplicación que se está desarrollando.

III. Especificaciones del sistema

El presente artículo describe el desarrollo de una implementación basada en MicroBlaze de un cliente SNTP, comparándolo con la versión completamente hardware desarrollada por el grupo ID2. En este sentido, ambos son sistemas de bajo coste, autónomos, compactos y de alta precisión, pensados para ser usados en entornos IEC-61850, aunque no necesariamente limitados a éstos.

El funcionamiento del cliente es sencillo. La unidad terminal remota a la que se encuentra conectado posee una interfaz preparada para el protocolo NMEA-0183 procedente de un GPS. Por lo tanto, el sistema debe emular el comportamiento de un GPS, proporcionando la información temporal y de sincronización que éstas necesitan (señal de PPS e información NMEA) a través de una interfaz serie. Por otro lado, el cliente se sincronizará con un servidor NTP/SNTP a través de una red de área local, que se mantiene sincronizado a su vez con un receptor de GPS.

Por lo tanto, el cliente debe cumplir las siguientes especificaciones:

1) El cliente operará dentro de una LAN Ethernet 10/100 Mbps.
2) Debe proporcionar una interfaz serie para ofrecer a las RTU la información que precisan.
3) Se propone implementar el sistema en una FPGA Spartan-3E, no siendo necesario ningún hardware adicional.
4) Es necesario proporcionar un procedimiento que permita configurar el sistema. En este sentido se implementará una aplicación web y servidor web.
5) En condiciones normales la precisión debería estar siempre por debajo de 1 ms.

IV. Arquitectura del cliente SNTP

La arquitectura global del sistema se divide en un diseño hardware, donde se ejecutará el código desarrollado y el diseño software, compuesto por el cógido a ejecutar.

A. Arquitectura hardware

El diseño hardware (Fig. 3) se basa en una solución SoPC (System on a Programmable Chip) que se programará en una FPGA. Este tipo de soluciones permiten implementar un sistema completo formado por un microprocesador, controladores, módulos IP y lógica programable para diseño a medida en un único dispositivo programable. Esta tecnología ofrece grandes ventajas en términos de funcionalidad, coste, área de la tarjeta y, lo que es más importante, flexibilidad. Se puede comenzar con un núcleo inicial e ir añadiendo módulos hasta obtener la arquitectura hardware final.

La placa empleada en el transcurso del proyecto es una Spartan-3E Starter Kit Board que incluye una FPGA Spartan-3E XC3S500E. El diseño del SoPC (Fig. 3) se basa en el procesador MicroBlaze y el bus PLB. El resto de periféricos soft-core utilizados, excepto el de corrección de la deriva que se describirá más adelante, se corresponden con IP Cores proporcionados por Xilinx junto con las herramientas que componen el ISE Design Suite. Entre ellos se incluyen controladores para memorias SDRAM (ejecución de software), memoria Flash (almacenamiento no volátil del sistema), Ethernet y puerto serie.

Por motivos de precisión y para evitar una sobrecarga del sistema, no es factible portar todo el sistema a una solución software. Debido a ello, ha sido necesario desarrollar un módulo soft-core que implemente esta funcionalidad en hardware: el periférico de corrección de la deriva (Fig. 4). Este periférico implementa la funcionalidad de reloj Hardware del sistema, la generación de la señal de sincronización de un pulso por segundo (PPS) y de la lógica de corrección de la deriva del reloj local. El reloj Hardware está en formato NTP y posee una precisión de 54 bits, de los que los 32 bits más significativos se destinan a la parte entera o número de segundos y los 22 menos significativos a la parte fraccionaria, siendo su precisión máxima de 2^{-22} segundos = 238 nanosegundos, suficientes para lograr una precisión de sincronización del orden de microsegundos. Existen dos modos de lectura: una lectura normal en la que la hora actual es transmitida y una lectura retardada en la que se transmite una hora almacenada previamente a través de un comando en un registro. Este último modo es necesario para reducir las latencias generadas en el procesamiento (subir pila de protocolos y tratamiento) de la información de sincronización recibida del servidor. La parte de control de la deriva se encarga de ajustar el reloj hardware local para corregir la diferencia existente entre el reloj local del cliente y el del servidor.

La integración de este periférico en el SoPC se logra a través del bus PLB, comportándose como una memoria, cualquier operación de modificación y consulta se corresponderá con una operación de escritura/lectura en el espacio de direcciones asociado. De este modo, su diseño se divide en dos partes bien diferenciadas (Fig. 4). En primer lugar la lógica que implementa la funcionalidad requerida y por otro lado una interfaz para su interconexión con el bus PLB.

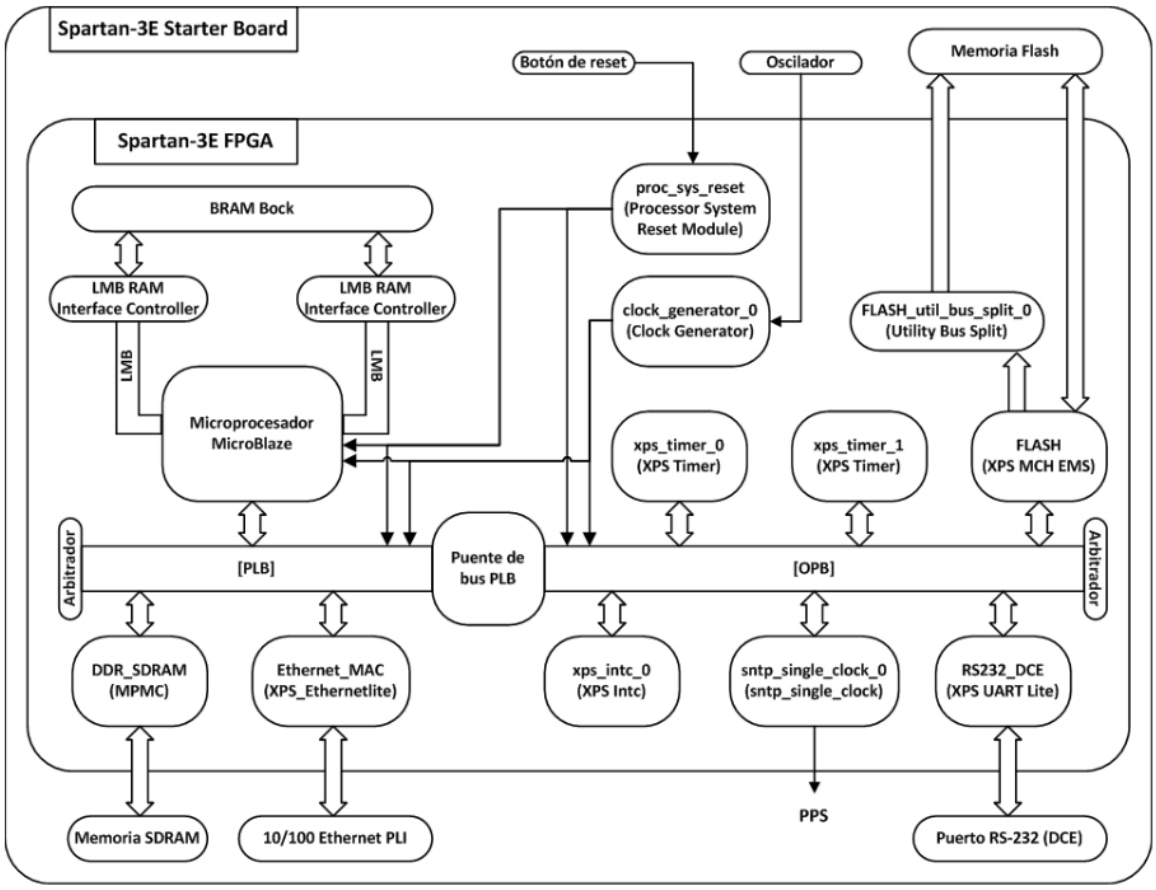

Figura 3. Diseño de arquitectura hardware.

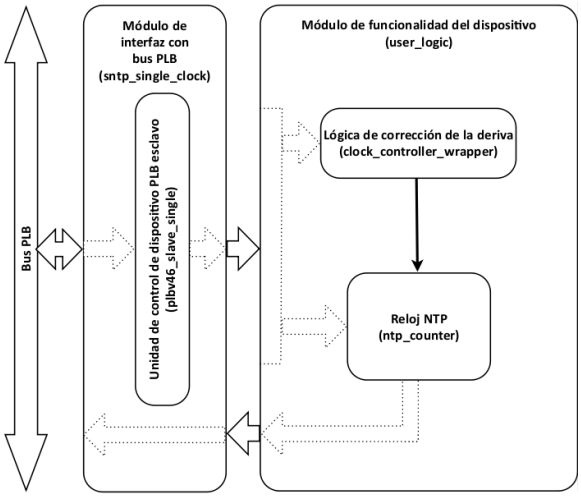

Figura 4. Esquema general del periférico.

B. Arquitectura software

Una vez definida la arquitectura hardware se ha de diseñar la solución software que se ejecutará sobre ella. Ésta se construirá tomando como punto de partida los drivers de los dispositivos soft-core que componen el sistema hardware y el Standalone BSP (Board Support Package). Además se han utilizado bibliotecas de funciones (MFS v1.00.a., LibXil Flash v1.00.a. y LWIP 1.3.0 Library v1.00.a.) que proporcionan parte de la funcionalidad que ha de ofrecer la solución final, entre las que se incluyen el manejo de la pila de protocolos TCP/IP y los drivers de controladores.

Las tareas del cliente SNTP se han implementado en software: generación y recepción de tramas SNTP y cálculo de los parámetros necesarios (t_{rd}, t_{offset} y parámetro corrector de la deriva). Para el control del periférico de corrección de la deriva se ha desarrollado el driver correspondiente. También se ofrece la posibilidad de configuración del sistema a través de una aplicación web basada en JavaScript, lo que conlleva el desarrollo de un servidor web y de la propia aplicación.

Se ha diseñado un sistema de arranque que posibilita la puesta en marcha del sistema a partir del contenido de una memoria Flash. Éste se compone de dos partes:

- Software en SoPC: En el inicio la FPGA se programa automáticamente con el diseño hardware y el programa software (almacenado en las BRAM) contenido en la Flash. Éste programa carga el software y los datos de configuración almacenados desde la FLASH a una memoria RAM, para proceder posteriormente a su ejecución.
- Software en PC: El contenido de la FLASH ha de ser

generado con un software desarrollado específicamente para ello. A partir del fichero .bit generado por el EDK, de los datos de configuración y del fichero .elf relativo al software se genera una imagen de la memoria en formato .mcs (Intel Hexadecimal File Format).

V. Resultados obtenidos

Los principales resultados de implementación hardware y verificación se detallan a continuación.

A. Resultados de implementación hardware

La FPGA utilizada para este proyecto ha sido una Spartan-3E XC3S500E. La tabla I muestra los resultados de la última implementación. No obstante, estos resultados son orientativos ya que pueden ser sometidos a distintos tipos de optimizaciones que minimicen los recursos necesarios.

Recursos	Valor
Slices Registers	5,247 (56 %)
Slices	4,399 (94 %)
4 input LUTs	5,993 (64 %)
16x1 RAMs	4
Bonded IOBs	98 (42 %)
DCMs	2 (50 %)
MULT18X18SIOs	4 (20 %)
Frecuencia máxima de operación	58.703MHz.

Tabla I
Resultados de implementación hardware en una Spartan-3E XC3S500E.

B. Resultados de verificación on chip

Como se expuso en las especificaciones del sistema (sección III), una de las motivaciones más importantes era determinar si la precisión del sistema desarrollado era inferior a la especificada (1 milisegundo) y comparar los resultados obtenidos con la implementación completamente hardware. La precisión queda determinada por la diferencia entre la señal de PPS procedente del receptor GPS y la generada por el cliente.

El escenario de pruebas usado para obtener los resultados de verificación on chip estaba constituido por el cliente SNTP descrito, un servidor SNTP hardware desarrollado por el grupo ID2, un ordenador y un switch al que se conectaron el resto de dispositivos. El ordenador era usado para configurar el cliente SNTP a través de la interfaz web propocionada por éste. Para medir la sincronización se comparó la señal PPS del cliente con la señal PPS que proporcionaba el receptor GPS al servidor hardware, obteniendo de ese modo la diferencia absoluta entre la señal original del receptor GPS y la del cliente SNTP.

La precisión conseguida en tales condiciones de funcionamiento es del orden de los 100 microsegundos y, por lo tanto, inferior al milisegundo. No obstante este resultado no se asemeja al conseguido con la implementación hardware. Esto es debido a que el marcado temporal no se realiza en el controlador Ethernet, sino que se realiza en el nivel de aplicación del modelo OSI. De ese modo, existe la posibilidad de mejorar considerablemente la precisión del sistema si el marcado temporal se realiza por hardware en el controlador Ethernet y se adapta la biblioteca de manejo de la pila TCP/IP (LWIP) para tener en cuenta esta nueva funcionalidad.

VI. Conclusiones

Este documento presenta el diseño sobre FPGA de un cliente SNTP basado en el microprocesador MicroBlaze, encuadrándose dentro del desarrollo de una plataforma de bajo coste y alta precisión dedicada a la sincronización de unidades terminales remotas.

De esta forma, en primer lugar se ha presentado un análisis de las diferentes alternativas disponibles a la hora de implementar un cliente SNTP y las ventajas e inconvenientes de cada una de ellas (sección II). Los aspectos más importantes que podemos destacar de este análisis son:

1) Por un lado, las implementaciones software disponibles consiguen una precisión del orden de 1ms. Esto se debe principalmente a que las marcas de tiempos son registradas por aplicaciones corriendo a nivel de usuario.
2) Por otro lado, el cliente SNTP completamente hardware desarrollado por el Grupo ID2, consigue una precisión de sincronización del orden de los 10 microsegundos. Esto se debe a que la marcas de tiempo son registradas en hardware por el controlador Ethernet MAC y a que todo el procesado se realiza mediante un diseño a medida basado en máquinas de estado finitas. Este último aspecto hace que las fases de diseño, implementación y mantenimiento de este tipo de sistemas resulte muy complejo, por lo que resulta necesario explorar otras alternativas.
3) Finalmente, en este trabajo se ha propuesto el diseño de un cliente SNTP como un SoPC basado en MicroBlaze como alternativa a las dos implementaciones anteriores. En este sentido, se pretende dotar de una mayor flexibilidad en las labores de diseño, implementación y mantenimiento del sistema, así como a la hora de incorporar nuevas funcionalidades. Además, en todo momento conservamos la posibilidad de realizar determinadas operaciones completamente en hardware cuando las especificaciones de precisión temporal así lo requieran.

En lo que respecta al diseño de este sistema empotrado, se ha desarrollado una plataforma hardware completamente operativa, así como la solución software que implementa gran parte de la funcionalidad del sistema.

Finalmente, se han realizado una serie de pruebas para medir la precisión de sincronización conseguida por esta implementación. Así, en el actual estado del desarrollo no se ha conseguido una precisión mejor que 100 microsegundos. Aunque dicha precisión difiere en un orden de magnitud respecto a la precisión conseguida por la implementación hardware, también podemos decir que mejora significativamente la precisión conseguida por las implementaciones software.

La siguiente fase del desarrollo se centrará en combinar ambas soluciones, integrando el cliente hardware en el software como un IP core. De ese modo, las operaciones críticas serían

realizadas completamente en hardware, manteniendo una precisión de 10 microsegundos y por otro lado se dispondría de la flexibilidad de incorporar nuevas funcionalidades al sistema. Este cambio se traduce en una adaptación del cliente hardware, en la modificación del IP Core Ethernet y en el desarrollo de los drivers correspondientes.

AGRADECIMIENTOS

Este trabajo ha sido parcialmente financiado por el Ministerio de Educación y Cultura del Gobierno Español a través de los proyectos TEC2007-61802/MIC (HIPER) y PROFIT-MITC TSI-020100-2008-258 (SEPIC).

REFERENCIAS

[1] D. L. Mills, "Network Time Protocol (Version 3) Specification, Implementation and Analysis," RFC 1305 (Draft Standard), Mar. 1992. [Online]. Available: http://www.ietf.org/rfc/rfc1305.txt

[2] D. L. Mills and P. H. Kamp, "The nanokernel," in *Proc. Precision Time and Time Interval (PTTI) Applications and Planning Meeting*, Reston VA (USA), Nov. 2000, pp. 423–430.

[3] D. Mills, "Simple Network Time Protocol (SNTP) Version 4 for IPv4, IPv6 and OSI," RFC 4330 (Informational), Jan. 2006. [Online]. Available: http://www.ietf.org/rfc/rfc4330.txt

[4] NMEA, "NMEA 0183 Standard," NMEA 0183 V 4.00, Jan. 2002.

[5] T. Skeie, S. Johannessen, and Ø. Holmeide, "Highly accurate time synchronization over switched Ethernet," in *Proc. 8th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, Antibes-Juan les Pins (France), Oct. 2001, pp. 195–204.

[6] I. E. C. Technical Committee 57, "IEC 61850 Communication Networks and Systems In Substations," IEC 61850 Edition 2 and other extensions, Jun. 2008.

[7] J. Viejo, J. Juan, M. J. Bellido, E. Ostua, A. Millan, P. Ruiz-de Clavijo, A. Muñoz, and D. Guerrero, "Design and implementation of a SNTP client on FPGA," in *Proc. 2008 IEEE International Symposium on Industrial Electronics (ISIE)*, Cambridge (United Kingdom), Jun. 2008, pp. 1971–1975.

[8] J. Viejo, J. Juan, E. Ostua, M. J. Bellido, A. Millan, A. Muñoz, and J. I. Villar, "Accurate and compact implementation of a hardware SNTP Client," in *Proc. 15th Iberchip Workshop (IWS)*, Buenos Aires (Argentina), Mar. 2009, pp. 504–509.

[9] J. Viejo, J. Juan, E. Ostua, A. Millan, P. Ruiz-de Clavijo, J. I. Villar, and J. Quiros, "Implementación sobre FPGA de un cliente SNTP de bajo coste y alta precisión," in *Proc. 9th Jornadas de Computación Reconfigurable y Aplicaciones (JCRA, Workshop on Reconfigurable Computing and Applications)*, Madrid (Spain), Sep. 2009, pp. 359–366.

[10] Xilinx, "MicroBlaze Processor Reference Guide. Embedded Development Kit EDK 10.1i." 2008.