Actas de las IX jornadas de computación reconfigurable y aplicaciones

Universidad de Alcalá. Departamento de Electrónica
Alcalá de Henares, 9-11 Septiembre, 2009

Título: Actas de las IX jornadas de computación reconfigurable y aplicaciones

I.S.B.N.: 978-84-8138-832-9
Depósito Legal: M-32241-2009

Editores:

D. Raúl Mateos Gil
D. Ignacio Bravo Muñoz

Departamento de Electrónica
Escuela Politécnica
Universidad de Alcalá
28871 Alcalá de Henares

Diseño de Cubierta:

Juan José Villadangos Pombar

Imagen de portada:

© Ivan Cholakov | Dreamstime.com

Impresión:

Servicio de Publicaciones de la Universidad de Alcalá

Escuela Politécnica
Universidad de Alcalá
28871 Alcalá de Henares

Este libro ha sido editado utilizando LaTeX.

Implementación sobre FPGA de un cliente SNTP de bajo coste y alta precisión

J. Viejo[1], J. Juan[1], E. Ostua[1], A. Millan[1], P. Ruiz-de-Clavijo[1], J. I. Villar[1], J. Quiros[1]

[1] Dpto. Tecnología Electrónica-Grupo ID2, E.T.S. Ingeniería Informática, US, Sevilla, España,
{julian, jjchico, ostua, amillan, paulino, jose, jquiros}@dte.us.es
http://www.dte.us.es/id2/

Abstract. Este trabajo presenta el diseño y la implementación sobre FPGA de un cliente SNTP compacto, de bajo coste y alta precisión, específico para entornos IEC 61850. Este módulo cliente es capaz de sincronizarse con un servidor SNTP y proporcionar información temporal precisa, pudiendo reemplazar en una amplio rango de aplicaciones industriales a receptores de GPS dedicados. En concreto, el dispositivo se ha empleado para realizar la sincronización de Unidades Terminales Remotas usadas comúnmente en sistemas de control industrial.

1. Introducción

La sincronización temporal de equipos electrónicos es necesaria en una gran cantidad de aplicaciones industriales, siendo un ejemplo típico la adquisición de datos por Unidades Terminales Remotas (RTU), donde el sellado de tiempo o *timestamping* es una tarea crítica en muchos casos. En este sentido, la norma industrial IEC 61850 [1] define el S*imple Network Time Protocol* (SNTP) [2] sobre una red *Ethernet* conmutada como una forma estándar de sincronizar un conjunto de subestaciones con un servidor de hora. El protocolo SNTP es una versión simplificada del protocolo *Network Time Protocol* (NTP) [3] que es comúnmente usado en servidores de Internet y *routers*. Ambos, SNTP y NTP, comparten el mismo protocolo de comunicación y formato de datos, siendo la principal diferencia que NTP usa algoritmos más complejos que aseguran una correcta sincronización con múltiples servidores en redes con latencias muy variables, lo cual es común en una red mundial como Internet. Por el contrario, el protocolo SNTP cubre la sincronización con un único servidor de hora y usa un algoritmo simplificado; así, en un entorno industrial controlado, para conseguir información temporal de cierta precisión, puede resultar apropiada la implementación de este protocolo como un sistema empotrado.
En este escenario, el servidor SNTP adquiere información de hora precisa mediante una referencia absoluta como un reloj preciso o un receptor de GPS estándar. Los clientes SNTP, localizados en las subestaciones, se sincronizarán con el servidor a través de la Red de Área Local (LAN), proporcionando a las unidades terminales remotas la información temporal y de sincronización que necesitan. De esta forma, evitamos tener que instalar referencias absolutas en las subestaciones y los problemas de cableado derivados de sacarlas al exterior.
Este trabajo se enmarca dentro del Proyecto PTC cuya finalidad es el desarrollo de una Plataforma Tecnológica Común (PTC) que facilite la implementación de la funcionalidad típicamente encontrada en Unidades Terminales Remotas usadas para el control de la red eléctrica. Un punto clave del proyecto es asegurar la sincronización de los equipos electrónicos

dentro del rango de pocos microsegundos. Esta sincronización se consigue mediante el uso de clientes y servidores SNTP implementados completamente en hardware. El diseño de esta plataforma cliente/servidor SNTP se ha dividido en tres fases principales:

1. Implementación de la pila básica de protocolos e integración con el controlador Ethernet. Al menos los siguientes protocolos de comunicación son necesarios: IP, UDP, BOOTP [4], ARP [5] y NTP.
2. Implementación del cliente SNTP: emisión de peticiones, procesado de respuestas, sincronización de la hora local y control de la deriva del reloj.
3. Implementación del servidor SNTP: sincronización con alguna referencia externa (GPS) y gestión de las peticiones de hora por parte de los clientes.

En este trabajo se describe el diseño e implementación del cliente SNTP y se muestran los resultados obtenidos una vez finalizado el mismo, ampliando el trabajo presentado en [6]. El resto de esta contribución está organizada como sigue: en la sección 2 se incluye un breve resumen de introducción al protocolo NTP/SNTP, en la sección 3 se especifican los requisitos del sistema y las especificaciones generales, en la sección 4 se describen los detalles del diseño e implementación del cliente SNTP hardware, en la sección 5 se incluyen los resultados obtenidos y, finalmente, en la sección 6 se discuten algunas conclusiones.

2. Operación Básica del 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 (T1). Cuando el servidor recibe la petición se genera una nueva marca de tiempo T2 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 (T3). Cuando el cliente recibe la respuesta también se anota la hora de llegada (T4). Con este conjunto de marcas de tiempo, el cliente puede calcular el *round trip time* (t_{rd}) y el *time offset* (t_{offset}). Asumiendo una conexión simétrica podemos definir estos dos tiempos de acuerdo a (1).

$$t_{rd} = (T4 - T1) - (T3 - T2), \; t_{offset} = \frac{(T2 - T1) + (T3 - T4)}{2} \qquad (1)$$

Usando el offset calculado, el cliente puede corregir su reloj local para ajustarse a la hora del servidor. Una implementación software del protocolo NTP/SNTP típicamente consigue un tiempo de sincronización del orden de un milisegundo con respecto al servidor [3]. En esta medida, hay 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 debe al intervalo de tiempo variable entre el instante en que la marca de tiempo se registra en el datagrama y el instante real en que éste abandona o alcanza el equipo. En las implementaciones software, estas marcas de tiempo son registradas por clientes/servidores software corriendo como una aplicación a nivel de usuario 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 softwa-

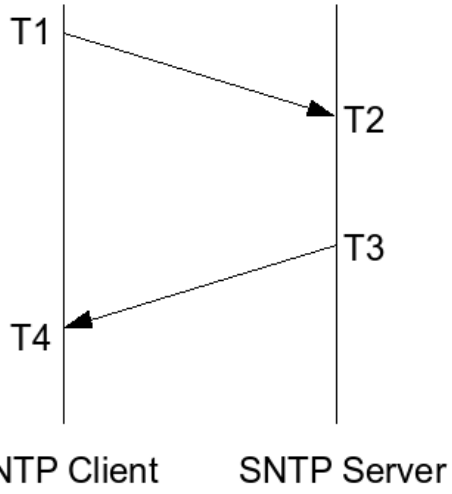

Fig. 1: Operación del protocolo NTP/SNTP

re. Por tanto, este error dependerá en gran medida de la carga del sistema, correcta implementación *sotfware*, etc. No obstante, algunos sistemas operativos como Linux o FreeBSD soportan todo el procesado en el kernel [7].

La precisión de sincronización del protocolo SNTP puede ser ampliamente mejorada si las operaciones del mismo son realizadas en las capas más bajas de la pila de protocolos [8], y en la operación de registrar las marcas de tiempo el sellado temporal se realiza en el hardware del dispositivo Ethernet tan pronto como los paquetes lleguen o abandonen la interfaz de red. De esta forma, la precisión puede alcanzar algunas decenas de microsegundos.

3. Especificaciones del Sistema

El principal objetivo de esta contribución es comentar los aspectos más significativos del diseño e implementación sobre dispositivos FPGA de una plataforma cliente/servidor SNTP de bajo coste, autónoma, compacta y de alta precisión para entornos IEC 61850, aunque no limitada a ellos. En la Fig. 2 se muestra un escenario típico donde emplear los clientes y servidores SNTP junto con las unidades terminales remotas; en este escenario, el servidor SNTP usará un receptor de GPS estándar como referencia de tiempo. Concretamente, ajustará su reloj interno haciendo uso de la señal de PPS (Pulse Per Second) y las tramas del protocolo NMEA-0183 [9] que llegan del GPS. Los clientes SNTP se sincronizarán con el servidor a través de la red de área local usando el protocolo SNTP y proporcionarán a la RTU la información temporal y de sincronización que necesitan (señal de PPS e información NMEA) a través de una interfaz serie, emulando un receptor de GPS.

Así, el cliente y el servidor SNTP deberían cumplir las siguientes especificaciones:

1. El cliente y el servidor operarán dentro una Red de Área Local Ethernet 10/100Mbps.
2. Tanto el cliente como el servidor se configurarán automáticamente usando el protocolo BOOTP, de forma que la configuración pueda ser centralizada en un único servidor de BOOTP.

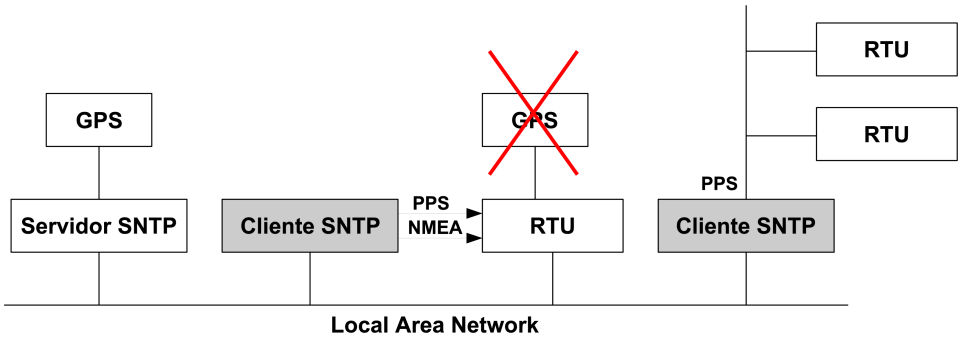

Fig. 2: Escenario típico donde emplear los clientes y servidores SNTP

3. En óptimas condiciones, la precisión del reloj local del cliente deberá estar dentro del rango de los 10µs con respecto a la hora del servidor: conexión directa a la red local (sin switches) y sellado de marcas de tiempo por el hardware del controlador Ethernet MAC. En típicas condiciones (conexión mediante un switch y sellado de las marcas de tiempo por software), la precisión debería estar siempre por debajo de 1ms.
4. Para la implementación del sistema se propone el uso de una FPGA de bajo coste tipo SPARTAN-3E para una tirada de pocas unidades, donde no debería ser necesario ningún requerimiento hardware adicional.
5. En cuanto al consumo de potencia como máximo el diseño deberá consumir 5W.

4. Diseño e implementación

En este apartado, se van a comentar los aspectos más importantes del diseño e implementación del cliente SNTP. En la Fig. 3 se muestra un diagrama de los módulos que forman este dispositivo; se pueden distinguir las siguientes partes: unidad de control, controlador Ethernet MAC, módulo cliente SNTP y módulo de generación de la señal de PPS y transmisión de tramas NMEA. A continuación, vamos a explicar brevemente la funcionalidad de cada uno de estos subsistemas.

La unidad de control se encarga de arbitrar las operaciones que realizan el resto de módulos, controlando las tareas que realizan cada uno en cada momento. Este módulo se ha modelado como una máquina de estados finita, codificada en el lenguaje de descripción de hardware *Verilog*, de acuerdo a la estructura descrita en [10]. Ésta define dos modos de operación:

1. Configuración. Cuando el cliente comienza a operar o tras un *reset* del sistema, se realiza un proceso automático de configuración de acuerdo al *Bootstrap Protocol* (BOOTP). Este proceso consiste en averiguar la dirección MAC e IP del cliente SNTP y una serie de parámetros de configuración como la dirección IP del servidor y la velocidad del puerto serie. El uso del protocolo BOOTP se debe a su simplicidad comparado al protocolo DHCP [11]. Esta característica hace que este protocolo sea la mejor opción para ser implementada en hardware, ya que las opciones avanzadas de DHCP no serían de utilidad para la aplicación en cuestión y sólo introduciría un gasto extra de recursos hardware y de tiempo de desarrollo.

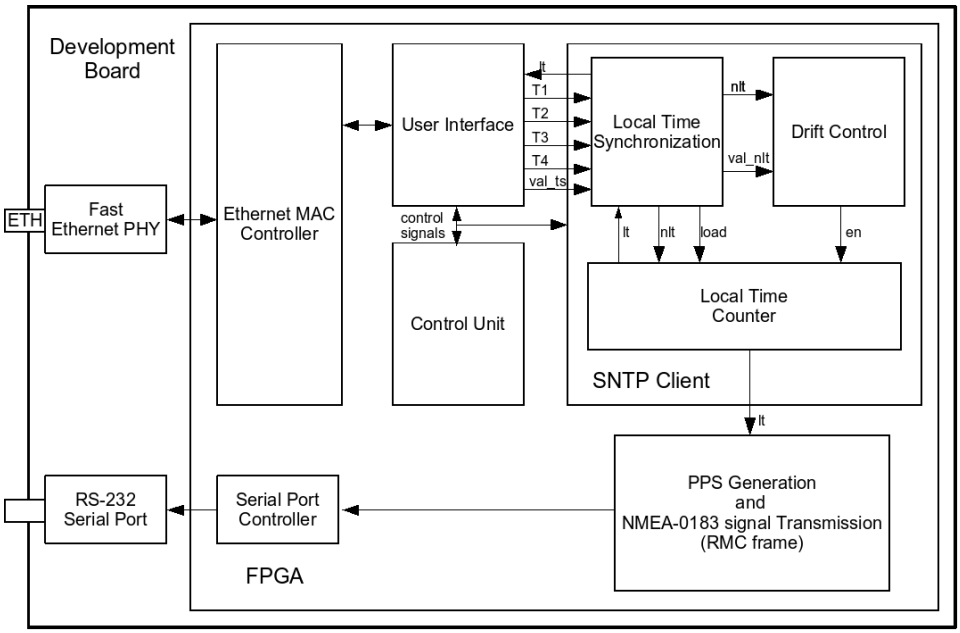

Fig. 3: Bloques que forman el cliente SNTP

2. Operación normal. Una vez que el proceso de configuración ha acabado, el cliente SNTP comienza a funcionar en el modo normal de operación. En este modo, el diseño realiza diferentes tareas que vamos a resumir a continuación. En primer lugar, el cliente necesita conocer la dirección MAC del servidor SNTP; de esta forma, el cliente implementa una versión simplificada del protocolo ARP (*Address Resolution Protocol*). En segundo lugar, el cliente debe transmitir paquetes de peticiones de hora (en formato NTP) en intervalos de tiempo programados. Finalmente, cuando el cliente recibe la respuesta a la petición de hora realizada, las marcas de tiempo son registradas, de forma que el módulo cliente puede calcular con estas marcas el desplazamiento y sincronizar su reloj local.

El controlador Ethernet MAC se encarga de controlar un dispositivo *Fast Ethernet PHY* estándar, permitiendo recibir y transmitir tramas Ethernet conforme a las especificaciones del estándar IEEE 802.3. La implementación de este módulo se ha realizado usando el *IP-core Tri-mode Ethernet MAC*. Este controlador es uno de los OpenCores disponibles en el portal web www.opencores.org. Este core dispone de una interfaz para aplicaciones de usuario que facilita el uso del controlador.

La interfaz de usuario con el controlador Ethernet MAC se ha desarrollado de acuerdo al documento de especificaciones [12]. Esta interfaz ha sido implementada como una máquina de estados finita codificada en *Verilog*, estando formada por los módulos de transmisión y recepción de paquetes y el de chequeo y registro de los parámetros de configuración. Por un lado, el módulo de transmisión es capaz de transmitir tres tipos de tramas Ethernet diferentes: peticiones de BOOTP, peticiones y respuestas de ARP y paquetes de petición de hora, los cuales son almacenados en una memoria RAM. Este módulo incluye un componente denominado actualizador de la memoria que se encarga de actualizar los campos de los diferentes paquetes antes de ser transmitidos. Por otro lado, el módulo de recepción es

capaz de identificar los siguientes datagramas: respuestas de BOOTP, peticiones y respuestas de ARP y respuestas a las peticiones de hora, descartando el resto de tramas Ethernet. Además, este bloque se encarga de extraer de los paquetes que recibe las marcas de tiempo, los parámetros de configuración, etc., y de generar señales de validación que serán utilizadas por la unidad de control para continuar con el procesado de la información recibida.

El módulo cliente se encarga de calcular el desplazamiento del reloj local usando las marcas de tiempo y de sincronizar la hora local a partir del offset calculado. Adicionalmente, el módulo estabilizador realiza un control de la deriva para mejorar la precisión del reloj local. Este módulo ha sido desarrollado usando la herramienta a nivel de sistemas System Generator for DSP, de acuerdo a la metodología presentada en [13]. La funcionalidad de este módulo consiste en calcular a partir del desplazamiento actual y el anterior un parámetro que permite modificar la frecuencia del reloj local, de forma que suavemente en sucesivas actualizaciones la deriva va convergiendo hacia cero.

Finalmente, el módulo de generación del PPS y de transmisión de tramas NMEA se encarga de generar una señal de sincronización (PPS+NMEA) que se enviará a la unidad terminal remota a través del puerto serie, emulando un receptor de GPS.

5. Resultados

En esta sección, se presentarán los resultados de simulación e implementación hardware.

5.1. Resultados de simulación

Para verificar que el diseño opera correctamente, se ha seguido el siguiente proceso de simulación. En la primera etapa, la simulación funcional se ha realizado usando Simulink y el simulador de Xilinx ISE Simulator. Para la generación de los estímulos de entrada se ha empleado el Blockset Source de Simulink. En una segunda etapa, para la verificación on-chip del cliente se ha empleado la herramienta ChipScope. De esta forma, se ha verificado la correcta transmisión y recepción de los diferentes tipos de paquetes: BOOTP, ARP y mensajes de petición de hora en formato NTP. En relación con el escenario mostrado en la Fig. 2, el cliente ha sido testado contra un servidor NTP software. Como se observa en las Fig. 4 y Fig. 5 la precisión de sincronización está dentro de los 10 microsegundos, limitado por el uso del servidor NTP software.

5.2. Resultados de implementación hardware

En este subapartado vamos a presentar los resultados de implementación hardware. Concretamente, vamos a analizar los recursos hardware de la FPGA empleados y la frecuencia de operación máxima conseguida. El diseño se ha implementado sobre una FPGA Spartan-3E XC3S500E. En la Tabla 1 se muestra el uso de recursos para el actual estado del desarrollo. Es necesario remarcar que aunque la implementación se ha finalizado, se está realizando un proceso de optimización con el que se pretende mejorar tanto los recursos de la FPGA empleados como la frecuencia de operación conseguida.

Recurso hardware	Uso (%)
Slices	3,820 (82%)
Slice Flip Flops	3,767 (40%)
4 input LUTs	5,504 (59%)
Bonded IOBs	37 (15%)
Block RAMs	4 (20%)
Máxima frecuencia	50.123 MHz

Tabla 1: Resultados de implementación hardware sobre una FPGA Spartan-3E XC3S500E.

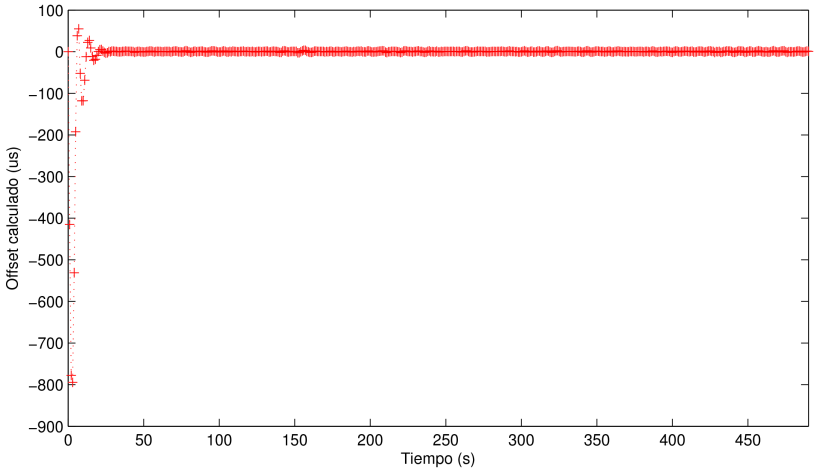

Fig. 4: Simulación con ChipScope. Convergencia de la hora local

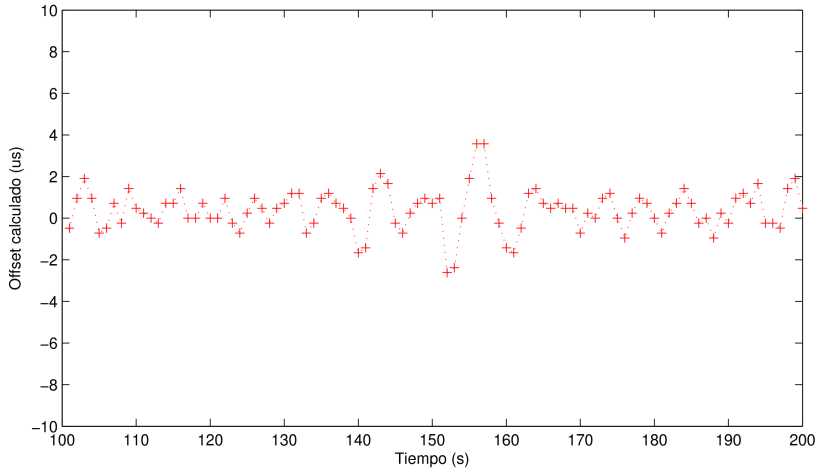

Fig. 5: Desplazamientos calculados para el intervalo de tiempo de 100 a 200

6. Conclusiones

En este trabajo se ha presentado el diseño de un cliente SNTP realizado completamente en hardware. Usando una metodología de diseño de alto nivel y programando el diseño en una FGPA estándar es posible producir una solución flexible, de alta precisión y bajo coste. El dispositivo diseñado puede ser empleado en entornos industriales como referencia temporal precisa para distribuir información de tiempo o para realizar marcado temporal, pudiendo sustituir a soluciones basadas en computador o a receptores de GPS específicos.

7. 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).

8. Referencias

1. IEC 61850 Communication Networks and Systems In Substations, Technical Committee 57. International Electrotechnical Commission.
2. D. L. Mills, Simple Network Time Protocol (SNTP) Version 4 for Ipv4, IPv6 and OSI, RFC 4330, Category: Informational. January 2006.
3. D. L. Mills, Network Time Protocol (Version 3) Specification, Implementation and Analysis, RFC 1305, Status: Draft Standard. March 2006.
4. B. Croft and J. Gilmore, Bootstrap Protocol (BOOTP), RFC 951, September 1985.
5. D. C. Plummer, An Ethernet Address Resolution Protocol, RFC 826, a.k.a. STD 37, November 1982.
6. 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, 2008 IEEE International Symposium on Industrial Electronics, ISIE 2008, Cambridge, United Kingdom, July 2008.
7. D. L. Mills and P. H. Kamp. The nanokernel, Proc. Precision Time and Time Interval (PTTI) Applications and Planning Meeting, Reston VA, November 2000.
8. T. Skeie, S. Johannessen, and Ø. Holmeide, Highly Accurate Time Synchronization over Switched Ethernet, 8th IEEE International Conference on Emerging Technologies and Factory Automation, ETFA , 2001, Antibes-Juan les Pins, France, October 2001.
9. NMEA 0183 V 4.00. http://www.nmea.org/content/nmea_standards/nmea_083_v_400.asp
10. C. E. Cummings, The Fundamentals of Efficient Synthesizable Finite State Machine Design using NC-Verilog and BuildGates, International Cadence Usergroup conference, ICU 2002, San Jose, California, September 2002.
11. R. Droms, Dynamic Host Configuration Protocol, RFC 2131, March 1997.
12. J. Gao, 10 100 1000 Mbps Tri-mode Ethernet MAC Specification, OPENCORES.ORG, January 2006.
13. J. Viejo, M. J. Bellido, A. Millan, E. Ostua, J. Juan, P. Ruiz-de- Clavijo, and D. Guerrero, Efficient design and implementation on FPGA of a MicroBlaze peripheral for processing direct electrical networks measurements, First IEEE Symposium on Industrial Embedded Systems, IES 2006, Antibes-Juan les Pins, France, October 2006.