METODOLOGÍA DE DISEÑO PARA SISTEMAS COMPLEJOS EN FPGA

Luis Enrique Acosta
Universidad Tecnologica de Bolivar, Cartagena, Bolivar, Colombia, luis.spim@gmail.com

Jorge Duque
Universidad Tecnologica de Bolivar, Cartagena, Bolivar, Colombia, jduque@unitecnologica.edu.co

José Granados
Universidad Tecnologica de Bolivar, Cartagena, Bolivar, Colombia, josedgv@yahoo.com

Sergio Fiallo
Universidad Tecnologica de Bolivar, Cartagena, Bolivar, Colombia, serge512@gmail.com

RESUMEN

En este trabajo presenta el proceso de diseño hardware/software de un prototipo en una FPGA, empleando metodologías de diseño que permita el diseño, la documentación, la implementación en un lenguaje de descripción de hardware y verificación en una tarjeta de desarrollo para FPGA's. La metodología presentada es utilizada en el desarrollo de un una unidad terminal remota (RTU) la cual involucra sistemas basados en procesadores, interfaces de comunicaciones, interfaces análogas, interfaces digitales y sistemas operativos en tiempo real.

Palabras claves: RTU, FPGA, TopDown, RTOS, Vhdl.

ABSTRACT

This paper presents the design process, hardware / software prototype in an FPGA, using design methodologies that allow for the design, documentation, implementation in a hardware description language and verification by a development board for FPGA's. The methodology is used in the development of a remote terminal unit (RTU) which involves processor-based systems, communication interfaces, analog interfaces, digital interfaces and real-time operating systems.

Keywords: RTU, FPGA, TopDown, RTOS, Vhdl.

1. INTRODUCCIÓN

Con la aparición de las FPGA y CPLD, la incorporación de los lenguajes de descripción de Hardware o HDL y las herramientas de desarrollo para la Automatización de Diseños Electrónicos EDA, se ha cambiado la forma de diseñar los prototipos digitales. El uso de metodologías apropiadas permite el desarrollo de aplicaciones más complejas, dispositivos más seguros, en espacios reducidos, con bajos consumos de energía y alto rendimiento.

El presente trabajo aborda el diseño y la implementación de una RTU en una FPGA, mostrando todas las etapas de diseño, desde las especificaciones, pasando por el diseño de la arquitectura, la implementación y finalmente la puesta en marcha de la RTU. Adicional a la obtención del prototipo de la RTU se presenta una metodología de

diseño para el desarrollo de equipos de alta complejidad que implican sistemas basados en microprocesadores, interfaces de comunicaciones, interfaces análogas, interfaces digitales y sistemas operativos en tiempo real.

El éxito de un diseño depende de varios factores tales como: las metodologías de diseño, las herramientas de diseño, los componentes, el conocimiento y la experiencia que tenga el diseñador. La metodología permite establecer los procedimientos, técnicas y actividades para dar solución al problema de diseño, las herramientas permiten llevar a cabo la implementación del diseño, haciendo uso de elementos tangibles como recursos de hardware y recursos intangibles como puede ser el software y la experiencia tiene relación directa con el diseñador debido a que se pueden tomar decisiones precisas en determinadas eventualidades o circunstancias en todo el proceso de diseño.

2. UNIDAD TERMINAL REMOTA

Una RTU típica normalmente tiene; puertos de entrada digital, puertos de entrada análogos, puertos de salidas análogos, puertos de salida digital; en algunas arquitecturas permiten aumentar el número de entradas y salidas a través de módulos de expansión.

Son muchas las aplicaciones para la cual se utilizan estos equipos, como estaciones de monitoreo del clima, sistemas de monitoreo y control de presión, flujo, voltaje, y corriente en gasoductos, acueductos, pozos petroleros, subestaciones eléctricas; entre otros. Una RTU se conecta al equipo físicamente y lee los datos de los sensores de nivel, presión, flujo, luego esta información es enviada hasta la estación remota para ser procesados, el operador remoto puede controlar la apertura de la válvula, controlar la velocidad de la bomba etc. Algunos tipos de RTU permiten almacenar algoritmos de control, permitiéndoles ser autónomas. La comunicación puede ser desarrollada a través de cable o de forma inalámbrica, empleando protocolos como MODBUS RTU, o PROFIBUS, entre otros (Sklanny S. Behrends C, 2008). Existen tres características importantes en el diseño de la RTU, el hardware, el software y las comunicaciones.

Este articulo describe los pasos que se llevaron a cabo para el diseño e implementación de la RTU, la sección II describe las premisas de diseño que se tomaron para el desarrollo del prototipo, luego se muestra los módulos que componen la RTU, la sección III presenta la metodología y finalmente los resultados y conclusiones.

3. DISEÑO DEL PROTOTIPO.

3.1 ESPECIFICACIONES DEL DISEÑO.

Las especificaciones de diseño fueron elaboradas a partir de la selección de tres tipos de RTU comerciales de las cuales se evaluaron características como: procesador, periféricos I/O, comunicaciones y software de control, entre las distintas marcas, para determinar que especificaciones tendrá la RTU que está bajo diseño. La primera RTU es distribuida por Motorola, la segunda por SIXNET, y la tercera por ALLEN-BRADLEY.

TABLA 1: Tabla comparativa de tres marcas de RTU y las características más relevantes.

ESPECIFICACIONES	ALLEN-BRADLEY	MOTOROLA	SIXNET	PROYECTO
CPU				
Procesador	32Bit AMR	Power PC 32bit	Power PC 32bit	***NIOS 32BIT***
Frecuencia del porcesador	30Mhz	200Mhz		**100Mhz**
DRAM	1Mb SRAM	32Mb	32Mb	**4Mb**
PROGRAM MEMORY	2M Flash	16Mb	32Mb Flash	**16Mb**
ENTRADAS DIGITALES				
Eentradas Digietales	8	16	12 entradas	***8***
ENTRADAS ANALOGAS				
Resolución DAC	16 Bit	16bit	16Bit	***8BIT***
Canales	8 entradas	8	6 - 8 entradas	***8***
SALIDAS DIGITALES				
Salidas Digitales	4	8	4 - 8 Salidas	***4***
SALIDAS ANALOGAS				
Resolución DAC	16 bit	14bit	16Bits	***16BIT***
Canales	2 salidas	4	Minimo 2 canales	***2***
COMUNICACIONES				
Puerto seriales	2 (com1, com2)	2 (com1, com2)	2 (com1, com2)	***1 (COM1)***
Puerto Ethernet	10/100Mbits/s	10/100Mbits/s	10/100Mbits/s	***10/100Mb***
Protocolo	MODBUS TCP/UDP	MODBUS TCP/UDP	MODBUS TCP/UDP	***MODBUS SERIAL***
HMI				
Pantalla	Ninguna	Ninguna	Ninguna	***Informativa***
Teclado	Inicio/Parada	Inicio/Parada	No especificado	***Configuración***
SOTWARE DE CONTROL EMBEBIDO				
Tipo de sitema operativo	RTOS	RTOS		***RTOS***
Sistema Operativo	Desconocido	VX-Works	Embedded Linux	***µC/OS-II***

La Tabla 1 muestra las especificaciones de cada RTU, en la columna cinco está consignadas las especificaciones finales de la RTU que se está diseñando. Teniendo en cuenta los datos de la tabla podemos establecer las siguientes premisas de diseño:

1) *Recursos de hardware.* A partir de la tabla 1 y se planteó inicialmente un diagrama de bloques que permitiera visualizar los componentes de hardware que debía tener la RTU. La Fig. 1 muestra la primera aproximación.

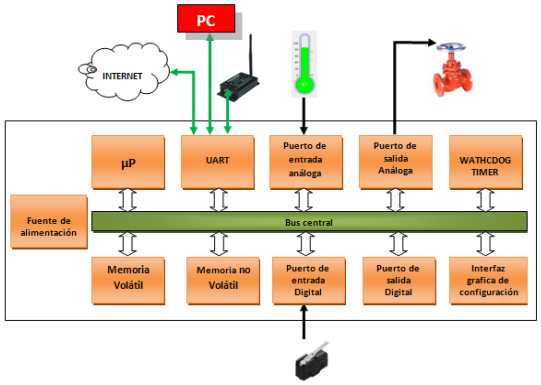

Figura 1. Diseño preliminar de la RTU.

Las arquitectura planteada permite tener una visión del tipo de FPGA a seleccionar, se escogió una FPGA Stratix II de altera, debido a que presenta un alto rendimiento a altas frecuencias, variedad de interfaces para la conexión de periféricos como memorias y dispositivos de comunicaciones, entre otros; una característica en común de las RTU's analizadas es su arquitectura de 32 bits, los sistemas desarrollo ofrecidos por el fabricante de FPGA permite la implementar diseños con esta característica. ALTERA

NIOS II es un Softcore (J. Labrose, 2002) con arquitectura de de 32 bits, completamente configurable en sus tres diferentes versiones, dependiendo los recursos que se quieran manejar y el rendimiento que se le quiera dar al procesador, Nios II/f (fast), Nios II/s (standard) y Nios II/e (economy). Asociado al procesador está el bus central, NIOS II emplea un bus llamado Avalon Fabric que permite interactuar con todos los recursos de hardware, además admite fácil integración con nuevos periféricos.

2) *Recursos de software.* Para escoger el software de control se tienen dos opciones, la primera, un programa propietario compuesto por un ciclo que permita explorar las entradas, actualizar las salidas y monitorear los puertos de comunicaciones, o emplear sistema operativo (Garcia L. Gonzales A. Moreno H. Jaquenod G., 2009) En el diseño del sistema se empleo un sistema operativo en tiempo real (RTOS), debido a que se hace necesario administrar los recursos de hardware y eventualmente los de software en la medida que se quiera aumentar el poder de procesamiento de la RTU, además los sistemas que están destinados a trabajar en procesos industriales deben ser robustos y confiables, un RTOS brinda esta característica, otra ventaja que proporciona un RTOS es la posibilidad de manejar varios procesos al tiempo esto permite aumentar el poder de la RTU debido a que pueden estar corriendo simultáneamente, algoritmos de control, protocolos de comunicaciones, algoritmos de cálculo etc. MicroC/Os-II[8] fue escogido por que cumple con las características anteriormente anotadas, además de soportar el procesador NIOS II.

3) *Comunicaciones.* Las comunicaciones fueron escogidas teniendo en cuenta la norma IEC 60870 para comunicaciones SCADA, donde el protocolo Modbus-RTU. cumple con dichas especificaciones. Adicional a esto TCP/IP se escogió como protocolo para ejecutar la tarea de diagnóstico empleándolo como otro medio de transmisión.

4) *Elección de la tarjeta de desarrollo.* para seleccionar la tarjeta de desarrollo se evaluaron los recursos de hardware que se plantearon preliminarmente en la Fig. 1 tales como recursos de memoria SDRAM y FLASH, los periféricos empleados como puertos de comunicaciones seriales, Ethernet y puertos de expansión disponibles. Con esta información se seleccionó el Kit de desarrollo de altera **NIOS II Development Kit - Stratix II Edition.**

5) *Herramientas de diseño.* Con la finalidad de mantener un balance Hardware/Software se diseñaron módulos a la medida se utilizó una metodlogía de diseño jerárquica de tal manera que se pudiera implementar en lenguaje de descripción de hardware VHDL. Entre los módulos diseñados estan el controlador ADC, controlador DAC y el modulo PWM.

   La descripción, síntesis, integración y simulación de los módulos se hace con la herramienta de QUARTUS II, esta herramienta dispone de SOPC Builder que permite la union de módulos a la medida y bloques de Propiedad Intelectual IP (Maxfield, Clive Max. 2004). (procesador NIOS-II, Controladores de memorias y periféricos), el software de la RTU se desarrollo en NIOS IDE, empleando lenguaje C++ y teniendo en cuenta las reglas para escribir código C, MISRAC, utilizado en el diseño de sistemas embebidos para procesos críticos (MISRAC stands).

   La tarjeta de expansión contiene las interfaces y los dispositivos que perminten adadaptar los niveles de voltaje del Kit de desarrollo con los sensores y actuadores que se conecten al dispositivo, se empleó la herramienta Altium Designer para el diseño de la PCB.

3.2 DISEÑO DEL HARDWARE DE LA RTU.

La arquitectura definitiva se muestra en la Fig. 2, el sistema está constituido por un procesador de 32bit NIOS-II, tres módulos a la medida: un ADC, un DAC y un módulo PWM; los demás módulos son IP cores que se encuentran integrados con el sistema de desarrollo, módulo de control para el panel de configuración, dos módulos de comunicaciones seriales, y un puerto de entrada y salida digital; además de los módulos mencionados anteriormente está el módulo JTAG UART que permite hacer depuración y programación del dispositivo, este puerto solo es utilizado en la etapa de diseño.

Los módulos a la medida fueron diseñados e implementados en VHDL de manera estructural.

1) *Controlador ADC.* El controlador ADC genera todas las señales necesarias para que el conversor análogo digital desarrolle el proceso de conversión, el conversor empleado es ADC0808, el controlador se encarga de controlar los ocho canales del ADC0808, la salida del módulo entrega un dato de 64bit en el cual se encuentran los datos de los 8 canales.

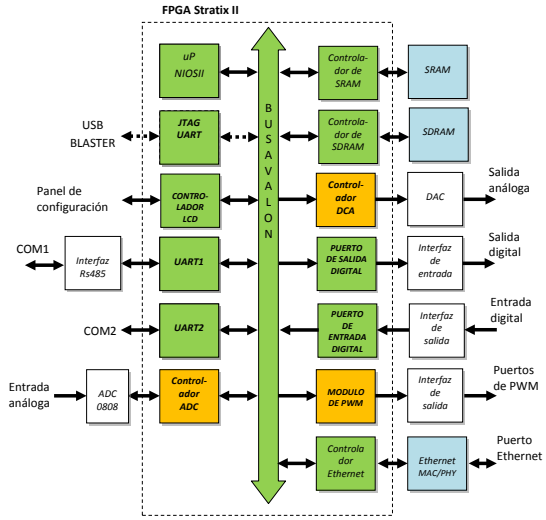

Figura 2. Arquitectura definitiva de la RTU.

2) *Controlador DAC.* El controlador DAC corresponde a un registro de 8bit el cual permite mantener a la salida el datos que va a ser convertido en una señal análoga. El dispositivo que se va a controlar es el DAC0808.
3) *Módulo PWM.* El módulo PWM se encarga de generar una señal cuadrada modulada, el valor del periodo y el ancho del pulso de la señal pude ser cargado a través de un dato de 32bits, la interfaz hacia el procesador NIOSII se da a travez del bus AVALON.

3.3 SISTEMA EMBEBIDO.

El sistema está compuesto por el procesador, las memorias de datos y de programa, módulos de comunicaciones, el bus Avalon y el controlador Ethernet; a continuación se describen en detalle los módulos:

1) *Procesador NIOS-II.* Obedece a una arquitectura Harvard, contiene un conjunto reducido de instrucciones RISC, todos sus registros internos son de 32bit. Se selecciono la versión estándar (Nios II/s) debido a que se queria tener una relación tamaño y rendimiento en la FPGA; esta version ofrece esta caracteristica.

2) *Bus Avalon.* Es una interfaz de comunicación diseñada para unificar interconexión de componentes dentro de los sistemas embebidos, permite comunicarse con los dispositivos empleando Mapeo de Memoria, a través de la herramienta SOPC Builder se puede interconectar cualquier dispositivo que emplee el Bus Avalon.

3) *Controlador Ethernet.* Permite establecer la conexión a Internet, emplea el periférico LAN91C111 que cumple dos funciones: control de acceso Ethernet (10/100Mbps) e interfaz física MAC/PHY que realiza la conexión con la tarjeta de desarrollo mediante un conector RJ-45. La comunicación con el procesador Nios II se realiza por medio del bus Avalon.

   La interfaz de diagnostico es una página web la cual emplea la librería *TCP/IP Niche Stack*, en cargada de brindar funciones de conexión a redes TCP/IP a través de *Socket*.

4. METODOLOGÍA

Con la finalidad de desarrollar un diseño confiable y seguro se empleo metodología de diseño que permita de manera sistemática llevar a cabo una serie de pasos para dar solución al problema de diseño.

4.1 METODOLOGÍA DE DISEÑO DE HARDWARE.

Para el diseño del hardware se usó una metodología de diseño jerárquica (Garcia L. Gonzales A. Moreno H. Jaquenod G., 2009), empleada para el diseño de sistemas de alta complejidad (W. Wolf. 2004). Esta metodología maneja dos enfoques que permiten afrontar el problema de diseño, el enfoque de arriba a – abajo (Top Down), en donde el dispositivo o la máquina que se va a diseñar se ve como un sistema general, este a su vez está compuesto subsistemas, estos subsistemas por otros y así sucesivamente; para el caso de método abajo – arriba (Bottom up) se parte de subsistemas que ya se encuentran definidos para así conformar sistemas más complejos y poder dar solución al diseño. En el proyecto se maneja un enfoque mixto debido a que se emplean algunos dispositivos ya definidos (Ip core CPU, controladores de memoria, puertos de comunicaciones etc.) para conformar dispositivos más complejos (enfoque abajo - arriba) y otros dispositivos que son diseñados a la medida (Módulos PWM, Puertos, interfaces para ADC) para esto se emplea el enfoque arriba - abajo, y que luego tienen que ser integrados al sistema que se está diseñando.

4.2 METODOLOGÍA DE PRUEBAS.

Al tener implementado un sistema Hardware/software se desarrollaron pruebas en el hardware y en el software:

1) Pruebas en el hardware. El uso de la metodología de diseño jerárquica permitió implementar los bloques de hardware separadamente, además permitió también la prueba de cada uno de los bloques de manera individual. Se desarrollaron testbench para cada bloque y se utilizó ModelSim como software de simulación.

2) Pruebas en el software. Las pruebas en el software se desarrollaron sobre el RTOS con los algoritmos corriendo en las diferentes tares del sistema operativo, el Websever, Modbus, periféricos entre otros.

5. RESULTADOS

Lugo de hacer la implementación del sistema en VHDL y la integración del sistema embebido con SOPC Builder, podemos mostrar un resumen de los recursos de hardware gastados en la FPGA así como también un análisis de potencia el cual permite tener un estimativo del consumo de la FPGA en la RTU.

TABLA 2: Análisis de recursos de la FPGA en el diseño de la RTU.

Recurso	% Utilizado
Elementos lógicos utilizados	11%
Total pines	49%
Bloques de memoria	23%
Bloques DSP	3%
PLLs	17%
DLL	50%

TABLA 3: Análisis de potencia disipada en la FPGA pàra el diseño completo.

Parámetro	Valor
Potencia	1657,01mw
Máximo T_A	50 °C
Flujo de aire	Ninguno
Máximo T_J	85 °C
Θ_{JA}	11,4 °C/W
Tj Calculado	*68,88 °C*

En Tabla 2 se puede observar que en promedio se usa solo un 26% de los recursos de la FPGA, por lo tanto es posible elegir una FPGA más pequeña para la implementación del diseño.
En la Tabla 3 se muestra la disipación de potencia para la FPGA, la variable Tj calculado (Temperatura de la unión) al compararla con el máximo Tj indica que el dispositivo no necesita un disipador de calor externo, la formula que emplea la herramienta para el cálculo de Tj es (1).

$$Tj = Ta + P \times \Theta JA \qquad (1)$$

Figura 3. Unidad Terminal Remota implementada.

La tarjeta de expansión se compuso de los diferentes dispositivos para ajustar los niveles de voltaje para ser trabajados en el kit de desarrollo, dotándose de entradas análogas de 0 – 5v, entradas digitales hasta 24v, salidas digitales tipo relé, salida analógica de 0 – 10v y salida PWM tipo transistor.

La Fig. 3 muestra la tarjeta de desarrollo Nios II Development Kit (RoHS) - Stratix II Edition (2S60N) junto con la tarjeta de entrada y salida.

6. CONCLUSIONES

Este artículo presentó el diseño e implementación en FPGA, la metodología de diseño empleada permitió desarrollar el diseño, la implementación y la verificación de los módulos individualmente, el dispositivo completo fue probado directamente en la tarjeta de desarrollo **Nios II Development Kit.**

La metodología de diseño empleada fue mixta debido a que el diseño hizo uso de bloques prediseñados IP para conformar dispositivos más complejos (enfoque Bottom up) y bloques a la medidas descritos en VHDL (enfoque Top Down) en los Modulo ADC, Modulo DCA, Modulo PWM.

El software empleado en la RTU se llevó a acabo a través de un RTOS. Las pruebas sobre el RTOS permitió ejecutar 9 tareas simultáneamente de las cuales podemos mencionar, dos tares propias del RTOS, dos tareas para el Webserver, una tarea para el protocolo Modbus, cuatro tareas para leer y escribir sobre los módulos a la medida. El uso del procesador con todas las tares funcionando son del 28%.

La verificación de las comunicaciones se llevo a cabo en dos fases, la primera a través de una comunicación MODBUS a 9600bps, efectuándose consultas del estado de las entradas y salidas de la RTU; la segunda, empleando una conexión TCP/IP de 100Mbps y a través de un navegador WEB, se pudo verificar el estado de las entradas y salidas del sistema como método alterno de comunicación, cumpliéndose así con las especificaciones propuestas.

Los recursos lógicos usados en la FPGA son 26%, la potencia disipada es del orden de 1657,01mw, con esta potencia disipada no fue necesario utilizar disipador de calor.

REFERENCIAS

Sklanny S. Behrends C. (2008). Sistemas Digitales de Control de Procesos. Tomo 1. El Galpón.

Maxfield, Clive Max. (2004). The design warrio`s guide to FPGAs: devices, tools and flows.

http://www.altera.com/products/ip/processors/nios2/ni2-index.html

Tanenbaum, Andrew S. Escalona García, Roberto, (2003). Sistemas operativos modernos segunda edicion. México D. F : Pearson.

J. Labrose (2002), MicroC/OS-II – The real Time Kernel, 2nd ed USA, CMP Books.

Garcia L. Gonzales A. Moreno H. Jaquenod G. (2009) Remote logic analyzer implemented on FPGA. Pontifica Universidad Javeriana. ISBN: 978-987-655-003-1.

W. Wolf. (2004). FPGA Based System Design, Prestice Hall. 2004.

MISRAC stands for "Motor Industry Software Reliability Association" C standards. http://www.misra.org.uk/

Autorización y Renuncia

Los autores autorizan a LACCEI para publicar el escrito en las memorias de la conferencia. LACCEI o los editores no son responsables ni por el contenido ni por las implicaciones de lo que esta expresado en el escrito.