IP core MAC Ethernet

Instituto Nacional de Tecnología Industrial

Rodrigo A. Melo, Salvador E. Tropea
(*rmelo@inti.gob.ar, salvador@inti.gob.ar*)

Instituto Nacional de Tecnología Industrial
Electrónica e Informática - UTIC - DESoL

SPL 2011

Introducción

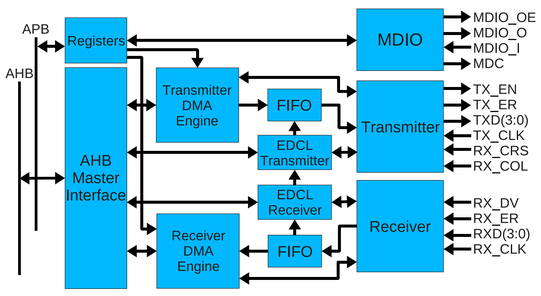
Nuestro equipo de trabajo precisaba un *core Ethernet*. La búsqueda de uno libre y descripto en **VHDL** destacó al **GReth**, perteneciente a la **GRLib**. Sin embargo, el área ocupada de **FPGA**, el complejo modo de uso y la única opción de utilización mediante bus **AMBA** no eran características deseadas.

En este trabajo presentamos un *core* **MAC** (*Media Access Controller*) *Ethernet* surgido de lo aprendido del estudio de **GReth**. Es compacto, fácil de utilizar y capaz de ser usado en **FPGAs** de cualquier fabricante.

El diseño fue simulado con herramientas de **Software Libre** y verificado en *hardware* utilizando una **FPGA** Virtex 4.

Core **GReth**

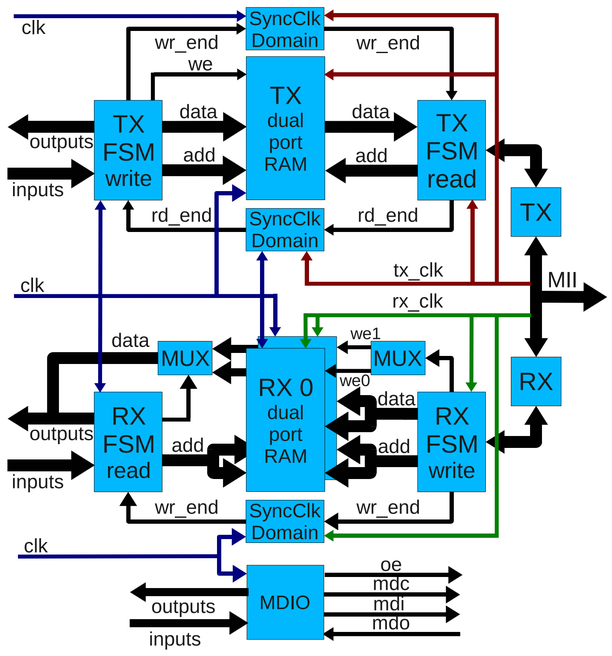
- Perteneciente a la **GRLib**, biblioteca de IP *cores* (licencia **GPL**).
- Implementa el estándar 802.3-2002.
- Provee interfaz entre un bus **AMBA** y una red *Ethernet*.
- Se conecta a un **PHY** externo mediante **MII** (*Media Independent Interface*) o **RMII** (*Reduced MII*) para intercambio de datos y **MDIO** (*Management Data Input/Output*) para acceder a configuración y estado.
- Descripta utilizando el llamado **Método de los dos procesos** (uno para la lógica combinacional y otro para secuencial).



El *core* es controlado mediante registros en un bus **AMBA APB**. Los datos se transfieren mediante un bus **AMBA AHB**, haciendo uso de descriptores con bits de control, estado, cantidad de *bytes* a transferir y punteros a la zona de memoria de datos.

Para el testeo del **GReth** se desarrollo un core denominado FakePHY que simula ser un **PHY**, y una biblioteca denominada AMBA Handler para el manejo de transacciones en los buses.

Core desarrollado: **MAC** *Ethernet*



En transmisión, una **FSM** escribe datos a una **FIFO**. Cuando finaliza, genera la señal *wr_end*, que luego de ser sincronizada, es leída por la **FSM** que toma datos de la **FIFO** y los transmite a través de **MII**.

La recepción es similar, pero para evitar pérdidas de paquetes, se implementó un esquema de múltiples **FIFO**s. La cantidad es configurable y su manejo depende exclusivamente del *core*.

Las **FIFO**s fueron implementada con **RAM** *dual port*.

El *core* presenta diversas configuraciones en base a *generics*, como permitir especificar la capacidad de almacenamiento de las **FIFO**, seleccionar cantidad de canales de recepción a utilizar, hacer uso del módulo **MDIO**, entre otras. Además, posee líneas de control para des/habilitar los canales de transmisióny recepción e interrupciones, indicar comunicación *half* o *full duplex*, especificar la dirección MAC y activar modo promiscuo.

En **transmisión**, se indica inicio y fin con señales independientes y se confirman los datos con una señal de escritura. Posee indicación de ocupado y provee información de errores de *overrun* de la memoria o alcance de límite de reintentos.

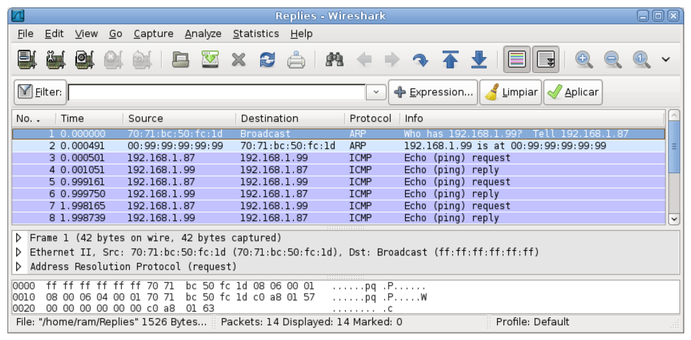
En **recepción**, se informan datos disponibles colocando una señal en estado alto, la cual se mantiene hasta la lectura de todos los datos, los cuales se confirman mediante la señal de lectura o se abortan para descartar el paquete. Los errores que señaliza son: *Overrun* de la memoria de datos; paquete recibido más corto/largo que el mínimo/máximo soportado por *Ethernet*; alineamiento o **CRC** erróneo; cantidad de datos recibidos diferente a los especificados en el campo *length*.

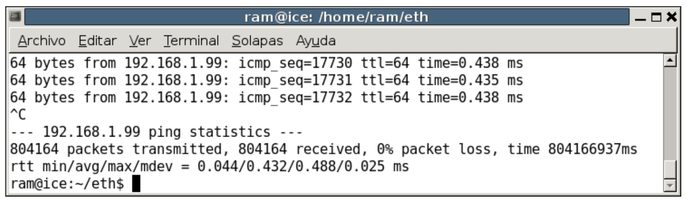
MDIO presenta características similares al **GReth**, pero una nueva interfaz.

Simulación

Se realizó un *testbench* y se utilizó *GHDL* para simular. Implementa procesos separados para verifica el funcionamiento de transmisión y recepción, e indicación de errores. Utiliza tres relojes que no son múltiplos entre sí.

Además, se desarrolló un *core* denominado *Replies*, que contesta peticiones **ARP** e **ICMP**, y se utilizó en un *testbench* junto a tramas *Ethernet* reales adquiridas con el *software wireshark*, para recrear la ejecución del comando *ping*.





Validación en *hardware*

Se utilizó:

- **FPGA** Virtex 4 (Xilinx) y *software* ISE WebPack 11.3 - L.57.
- Computadora con Debian GNU/Linux.
- *core Replies*.
- **PHY** externo DP83847 (National Semiconductor).

- Pruebas realizadas usando comunicación *full-duplex* de 100 Mb/s.
- Se ejecutó el comando *ping* por más de una semana sin que se presenten paquetes perdidos.
- Se utilizó *wireshark* para verificar la correcta conformación de paquetes.

Resultados

core **GReth**

Configuración	LUTs	FFs	Slices	BRAMs
Sin **MDIO**	1814	775	1099	2
Con **MDIO**	2011	834	1220	2

core **MAC**

Configuración	LUTs	FFs	Slices	BRAMs
1 RX sin **MDIO**	823	333	491	2
2 RX sin **MDIO**	872	341	516	3
2 RX con **MDIO**	1016	381	591	3

Conclusiones

- Implementación compacta. En configuraciones equivalentes utiliza menos del 50 % de área de **FPGA** que **GReth**.
- El *core* desarrollado es simple y no depende de utilizar un cierto bus.
- La utilización de lenguaje **VHDL** 93 estándar, permite un *core* sintetizable en **FPGA**s de cualquier fabricante.
- Las herramientas propuestas por el proyecto FPGALibre son adecuadas para un proyecto de estas características.

debian

LaTeX

OpenOffice.org

FPGA Libre