UNIVERSIDAD DE
COSTA RICA

Proyecto I
Circuitos Digitales II

Andrés Alvarado, B30316
Alejandro Rojas Cordero, B36049
Fernando Gonzáles, B43023

Escuela de Ingeniería Eléctrica
Universidad de Costa Rica
San José, Costa Rica

I-2018

Índice

abstract

En este proyecto se diseñará la capa física (PHY) de una interfaz PCIe, dividiendo el funcionamiento en 3 grandes módulos, Parallel to Serial/Serial to Parallel, Muxes de Control y el Byte Striping tanto Tx como Rx.

1. Objetivos

1.1. Objetivo General

- Implementar la capa física de la interfaz PCIe.

1.2. Objetivos Específicos

- Realizar un estudio de mercado sobre dispositivos en los cuáles se pueda implementar el PHY del PCIe o de una interfaz USB.
- Verificar el correcto funcinamiento de cada módulo por separado para poder unirlos.
- Sintetizar el diseño en Yosys y comprobar su correcto funcionamiento.

2. Estudio de Mercado

2.1. Bloques funcionales de la capa PHY para una interfaz PCIe y una USB

2.1.1. PCIe

Se encontrarón diseños de Texas Instruments y de NXP, a continuación pordemos verlos para compararlos.

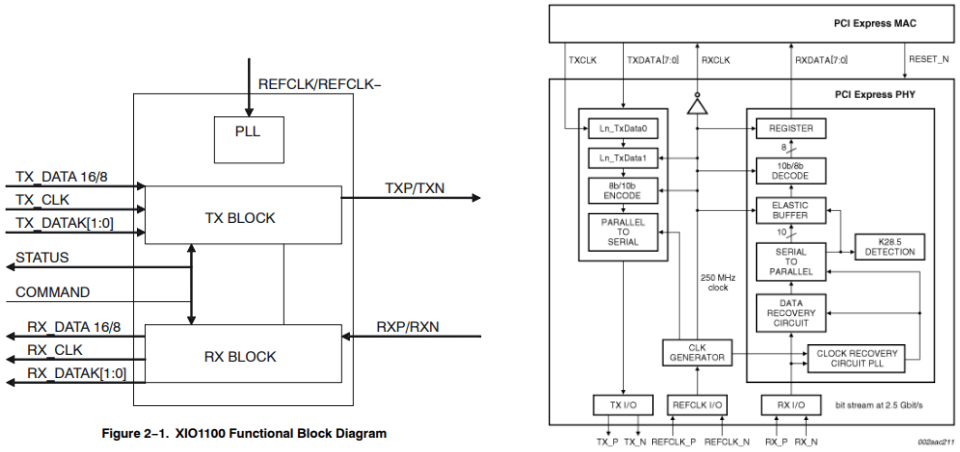

Figura 1: Diagramas de xio1100 de TI y de PX1011B de NXP

El diseño de TI, es una modificación de PHY del PCIe llamado PIPE, pero como vemos su diseño es muy parecido a lo que hace el nuestro, un bloque Tx y Rx que se comunican entre sí, con datos de 16 a 8 bits, por otro lado el diseño de NXP es prácticamente el mismo que utilizamos nosotros, con bloques de serial a paralelo y decodificadores de bits.

2.1.2. USB

Al igual que para el PCIe encontramos dos diseños de Phy para USB de Texas Instruments y de NXP que veremos a continuación.

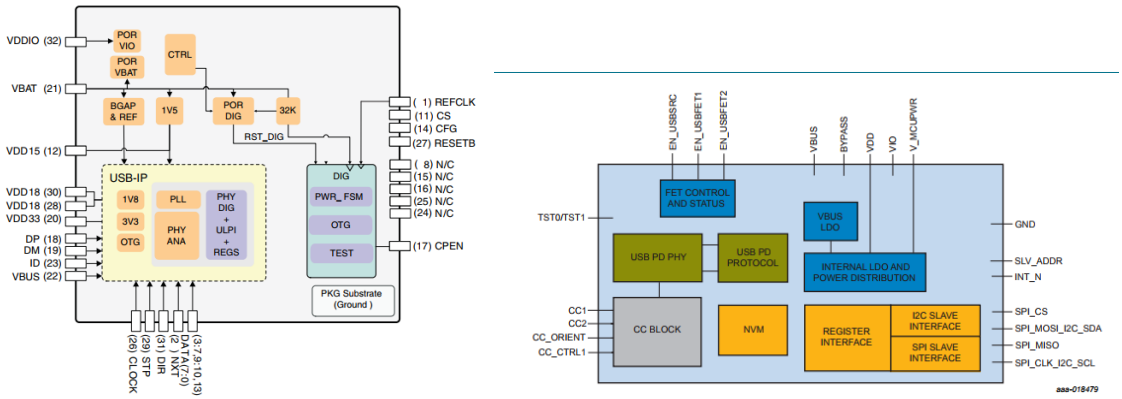

Figura 2: Diagramas de TUSB1210 de TI y de PTN5100D de NXP

Como podemos apreciar, entre los bloques PCIe y USB hay bastantes diferencias, en lo que son los protocolos que usan como el I2C, pero al final lo que tenemos son bloques que van a estar enviandi y recibiendo datos, por lo que se podrían implementar de una forma similar, adaptando los protocolos para cada unidad.

2.2. Posibilidad de hacer un diseño que combine la funcionalidad de ambas interfaces: PCIe y USB

Esto es algo que ya se ha realizado, de hecho NXP tiene el un procesador modelo MPC8308, el cuáñ combina varoas intefaces además de la PCIe y la USB, como podemos ver en la figura

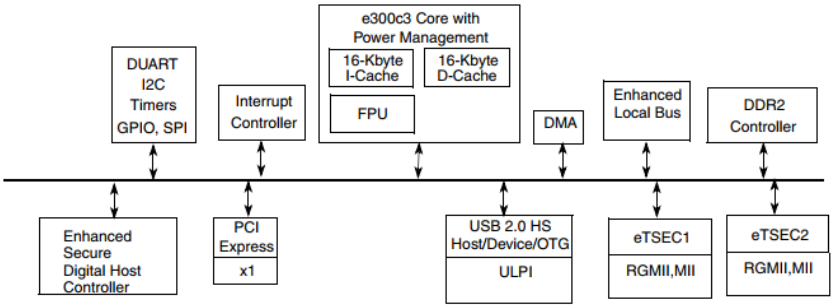

Figura 3: Diagramas de MPC8308 de NXP

2.3. Precio, Frecuencia y consumo de energía para ser competitivos en el mercado

En esta sección compararemos algunas características de los circuitos, para así poder determinar el rango en el que debería andar nuestro circuito para ser competitivo en el mercado actual. Los dos primeros son los PCIe, los dos siguientes los USB y el último es el que trae varias interfaces.

Modelo	Precio $	Frecuencia de Operación	Consumo de Energía (V)
xio1100	5.25	100 MHz-125 MHz	3.3/1.8/1.5
PX1011B	5	100 MHz	3/3.6
PTN5100D	1.96	100kHz-1MHz	1.8/3.3
TUSB1210	0.92	60 MHz	1.8
MPC8308EC	12	400 MHz	1.2

3. Bitácora

Etapa	Dueño	T. Estimado	T. Dedicado	Avance	Entrega	Comentarios
Mux/Demux	Fernando	2	4	100 %	16/05	
Byte Striping	Alejandro	4	6	100 %	25/05	
Serial/Paralelo	Andrés			100 %		
Interconexión	Fernando y Andrés	4	10	100 %	26/05	
Pruebas	Andrés y Alejandro	3	5	100 %	27/05	
Síntesis	Fernando	1	5	100 %	29/05	Formato .ps
Documentación	Todos	1	3	100 %	29/05	

4. Resultados

Como se dijo anteriormente, nuestro proyecto está dividido en tres grandes bloques y uno pequeño, de los cuáles veremos sus resultados a contunuación, para luego explicar el funcionamiento de los cuatro trabajando al mismo tiempo.

4.1. Instrucciones de utilización de la simulación

- Para simular el Byte Striping hacer "make.en las carpetas Tx y Rx.
- Para el resto de simulaciones hacer "makez luego "make gtk".

4.2. Muxes de Control

Para esta sección del circuito, se utilizan dos módulos distintos: uno para la parte de TX y uno para la parte de RX. Ambos utilizan la misma codificación para los bloques de comunicación para así evitar que el usuario se vea en la tarea de colocarlos en sus entradas.

4.2.1. Mux

El primer bloque es un mux de la etapa de transmisión. Este mux posee una línea de selección de 5 bits para poder seleccionar alguna de las 9 opciones de envío en su salida de dos caracteres hexadecimales. El usuario solo deberá definir cuál línea de control desea utilizar (cuál bloque desea enviar) y colocar además los datos en su entrada que desea enviar por la estructura de PCIe. La entrada puede ser de cualquier tamaño pero deberá mantenerse de un tamaño de 8 bits o múltiplos del mismo.

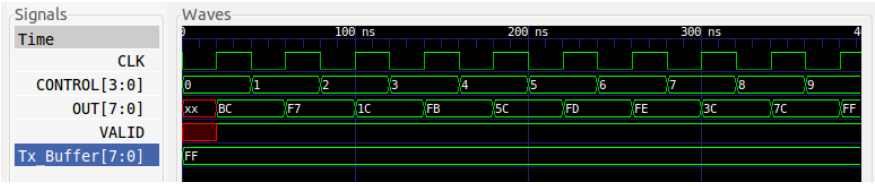

Figura 4: Mux simulación GtkWave

Las pruebas realizadas son:

1. Prueba de cada salida: Se cambia la línea de control para que pase por todas las opciones del mux enviando así cada uno de los bloques. Esta prueba fue exitosa.
2. Prueba de bit de válido: Se revisa que el bit de válido se encienda cuando comienza la transmisión de una palabra válida. Esta prueba fue exitosa.
3. Diferentes valores de entrada: Se utilizan valores distintos de datos del usuario para verificar que no hayan errores de transmisión. Esta prueba fue exitosa.

4.2.2. De-mux

El de-mux de la estructura PHY PCIe es un módulo que recibe cada palabra proveniente de la etapa de transmisión y elimina del bus de salida las palabras que no son necesarias, como SKP, END, PAD, entre otros. Esto entrega a su salida un bus de datos limpios en donde solo encontramos lo que el usuario envió desde la etapa anterior. Además levanta un bit de válido para que las etapas siguientes sepan cuándo está entrando un bloque válido para lectura.

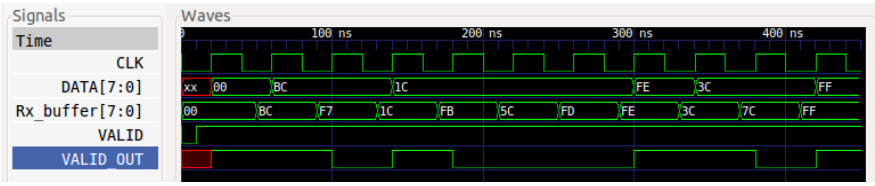

Figura 5: Demux simulación GtkWave

Las pruebas realizadas son:

1. Prueba de cada bloque: Se cambia el bloque de entrada porque cada uno de los disponibles para verificar que pasan correctamente. Esta prueba fue exitosa.

2. Prueba de bit de válido: Se revisa que el bit de válido se encienda cuando llega un bloque correcto y no de los que se deben eliminar. Esta prueba fue exitosa.

3. Diferentes valores de entrada: Se utilizan valores distintos de datos del usuario para verificar que no hayan errores al recibir. Esta prueba fue exitosa.

4.3. Byte Striping

Este bloque lo que pretende es una máquina de estados que acomode bytes que vienen de un arreglo a cuatro lanes o viceversa, por eso es que tenemos un bloque transmisor y otro receptor, ya que ocupamos acomodar los datos tanto cuando viene paquetes como cuando salen.

4.3.1. Tx

El bloque Tx (Transmisor), acomoda los bytes de un arreglo de datos en cuatro lanes o canales, conforme estos van llegando. Como podemos ver en la figura para esta prueba se tiene un arreglo de bytes (dato) que podemos ver en código hexadecimal del 01 al 12, el cuál va pasando a cada lane, de dataOut0 a dataOut3.

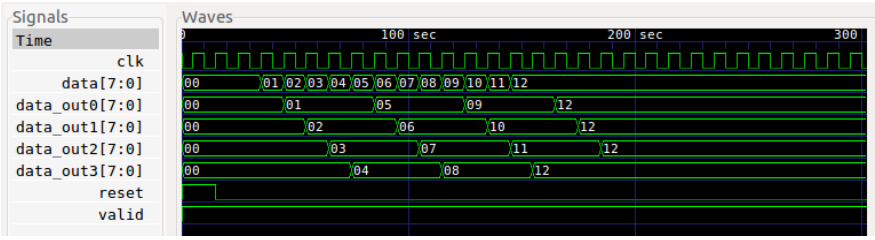

Figura 6: Byte Striping Tx simulación GtkWave

4.3.2. Rx

El bloque Rx (Receptor), acomoda los bytes que están en los lanes a un arreglo de datos, donde se van acomodadando un byte por lane conforme van recorriendo estos, como podemos ver en la figura la prueba acá es contraria, tenemos bytes en dataIn0 hasta dataIn3, y estos se van acomodando el Data recorriendo los 4 lanes varias veces.

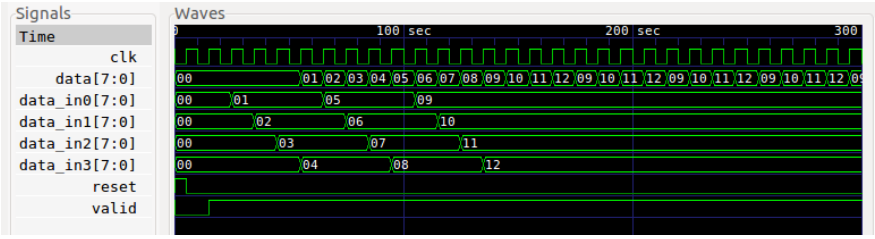

Figura 7: Byte Striping Rx simulación GtkWave

4.4. Parallel to Serial // Serial to Parallel

Este bloque se encarga de poder pasar la información por un cable. Como se utilizan palabras de 1byte más bits extra al inicio y al final de la palabra. Estos para poderse poder pasar por solo un cable se tienen que serializar y despues volver a paralelisar para dar la lectura.

4.4.1. Parallel to Serial

Este modulo se encuentra en la parte del pcietx. Se encarga de recibir la informacion del byte Striping la cual viene paralela y se encarga de serializarla. Este sirve con una maquina de estados que depende en que estado esté este leera un cable distinto del byte de entrada. Por lo que tardará 8 veces más en enviar un dato ya que este tendrá que leer cada bit a la vez por 8 ciclos de reloj.

Como se puede ver en la figura 8, este espera al primer byte que tenga un bit de valido encendido y lo convierte data_in el cual es un byte a data_out el cual es un bit.

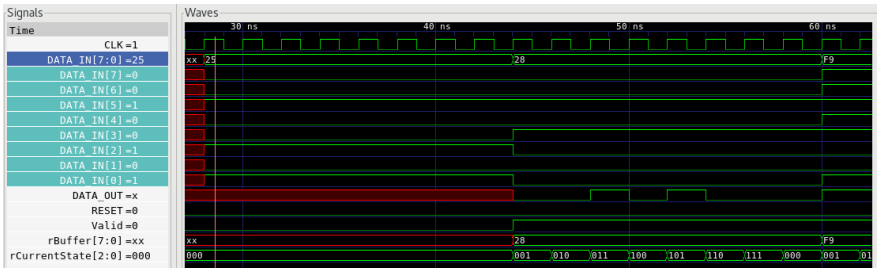

Figura 8: Parallel to Serial simulación GtkWave

4.4.2. Serial to Parallel

Este modulo es la contraparte del paralelo a serial. Este se encuentra en la parte de pcierx. Se encarga de recibir la cadena de bits y transformarlo en un solo byte, para que despues pueda ser leido por el byte stripping Rx.

En la figura 9 se puede observar como este modulo se queda esperando la palabra BC de data_in para levantar el bit de valido y empezar a mandar información en paralelo por data_out.

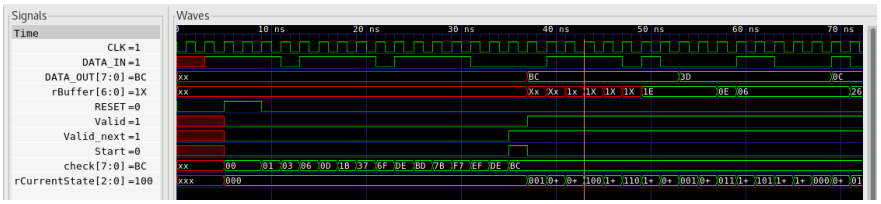

Figura 9: Serial to Parallel simulación GtkWave

4.4.3. Clk modifier

Este modulo se encarga de bajarle la frecuencia al reloj por un factor de 8. Ya que el bloque de paralelo a serial y serial a paralelo es 8 veces más lento, se necesita que su reloj funcione 8 veces más rapido.

Por lo tanto se pasa el reloj de 2MHz a 250kHz. Para que cada modulo este pasando 1MB/s.

4.5. Resultados Generales del PHY

4.5.1. Parallel to Serial to Parallel

Esta es una de las pruebas que se tuvieron que dar para asegurar el comportamiento de los modulos que convierten un byte a una secuencia de un bit a un byte de nuevo.

Como se ve en la figura 10 hasta que se manda el byte BC, la información llega hasta el otro lado.

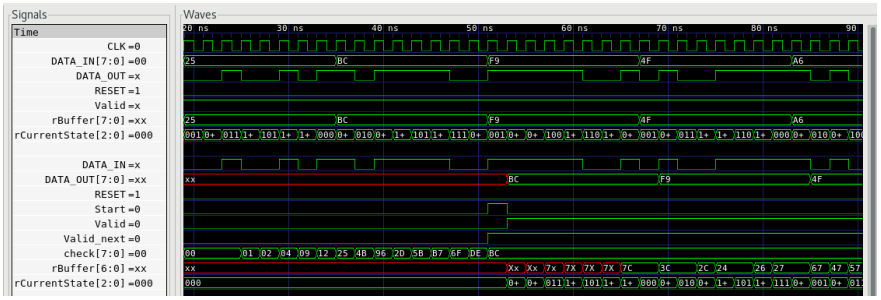

Figura 10: Parallel to Parallel simulación GtkWave

4.5.2. ByteTx to ByteRx

Posteriormente cuando se tuvieron resultados exitosos de la prueba anterior, se paso a añadir los modulos del byte stripping.
Primero se pasa el clk por el modificador para volverlo más lento y que los modulos del byte stripping utilicen este. Mientras que los otros dos modulos usen el clk original. Luego se le añade Datos al ByteTx el cual se los manda al modulo de Parallel a Serial. Este lo manda al bloque de RX el cual lo recibe el modulo de Serial to Parallel y se lo da a ByteRx.

Como se puede ver en la figura 11, la información recibida es dividida por 4 caminos y llega por cuatro caminos distintos para ser unida de nuevo y volver a tener la infomación original.

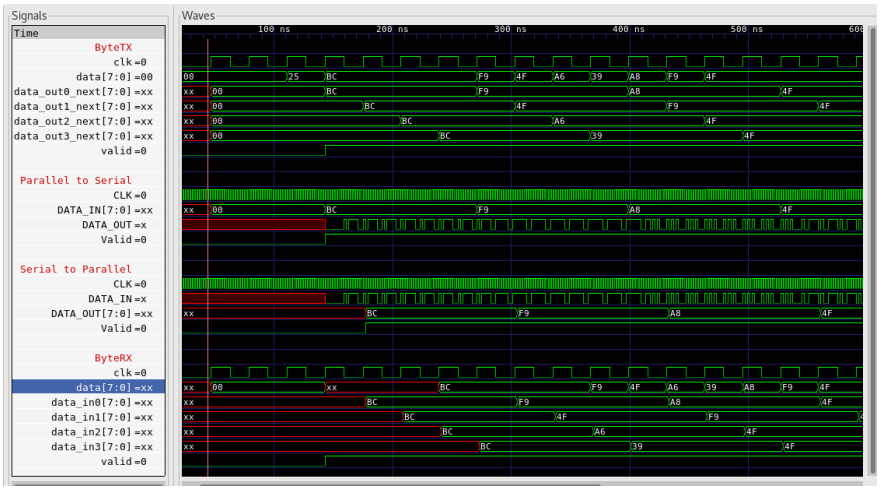

Figura 11: Byte Tx to Byte Rx simulación GtkWave

4.5.3. Pcie

Esta es la prueba donde conectamos todos los módulos, como podemos ver en la figura en la primera parte tenemos el Mux, para el cual la señal Çontrol"nos da los modos de trabajo como vemos el 0 es el COM, que es donde se inicializa todo y hasta que llega el 9, que es donde se mandan los datos, el TxBuffer son los datos que se mandan y como vemos pasan al Out a partir de este momento.

Estos datos pasan por los módulos Byte Striping, paralelo- serial, serial-paralelo y de nuevo por el Byte Striping, para llegar al próximo bloque, donde tenemos el RxBuffer donde van llegando todos los comandos, los cuales pasan la información relevante al Data.

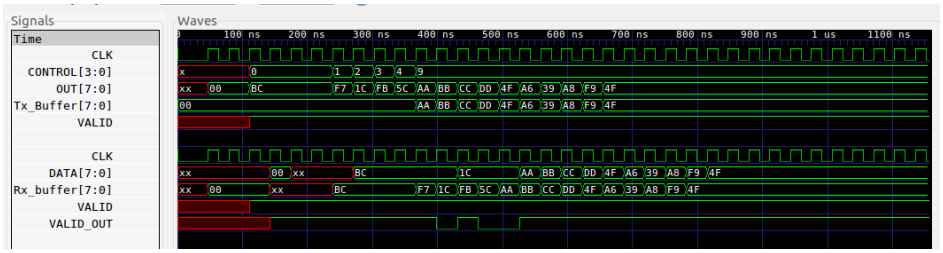

Figura 12: Simulación del PCIe en GtkWave

5. Conclusiones

El modelo diseñado de la parte PHY de una interfaz de tipo PCIe se puede concluir como exitoso, debido a que de manera funcional cumple con las especificaciones de diseño. No utiliza relojes extras, sino que utiliza módulos para retrasar señales de reloj y así ahorrarse problemas de sincronización futuros. Esto es muy importante ya que los relojes son partes muy caras en un diseño y que generan muchos problemas.

Subdividir el diseño en pequeños módulos funcionales permitió solucionar problemas que tal vez en el modelo general no se hubieran encontrado con facilidad. Así mismo se pudo avanzar de manera más rápida dividiendo tareas y observando cómo las mismas se interconectaban y dependían de las etapas contiguas para realizar un diseño ideal que no presentara problemas de comunicación a futuro.

Referencias

[1] Data Sheet xio1100. *Textas Instruments*. Extraído el 28 de Mayo del 2018 de: http://www.ti.com/lit/ds/symlink/xio1100.pdf

[2] Data Sheet tusb1210. *Textas Instruments*. Extraído el 28 de Mayo del 2018 de: http://www.ti.com/lit/ds/symlink/tusb1210.pdf

[3] Data Sheet PX1011B. *NXP*. Extraído el 28 de Mayo del 2018 de: https://www.nxp.com/products/analog/interfaces/pci-express/pci-express-stand-alone-x1-phy:PX1011B?fsrch=1sr=1pageNum=1

[4] Data Sheet PTN5100D. *NXP*. Extraído el 28 de Mayo del 2018 de: https://www.nxp.com/docs/en/data-sheet/PTN5100D.pdf

[5] Data Sheet MPC8308EC. *NXP*. Extraído el 28 de Mayo del 2018 de: https://www.nxp.com/docs/en/data-sheet/MPC8308EC.pdf