Universidad de Costa Rica
Facultad de Ingeniería
Escuela de Ingeniería Eléctrica
IE-0523 Circuitos Digitales II

I ciclo 2024

Proyecto Final MDIO

Aurora Matamoros Cuadra B84707
Kenneth Rojas Rivera B96890
Jimena Gonzalez Jiménez B83443
Grupo 02

Profesora: Ana Eugenia Sanchez Villalobos

01/07/2024

Índice

1. Resumen

Este proyecto lo que consiste es en hacer un generador y un receptor de transacciones siguiendo el protocolo MDIO descrito en la clausula 22 de Ethernet según la IEEE, primero lo que se trabajo fue el generador. Como su nombre lo dice, su función principal es transmitir datos. Aunque por el protocolo también puede recibir datos del receptor.

El generador permite la comunicación entre el CPU y el receptor de transacciones transfiriendo *t_data* bit por bit a través de *mdio_out*. Este cuenta con un modo lectura, donde se reciben los datos del receptor por medio de *mdio_in* y el modo escritura que solo se encarga de transmitir la información de *t_data* al receptor.

El receptor de transacciones recibe datos tanto del generador como del PHY, logrando una correcta comunicación entre estos y asegurando la eficiencia de la transmisión de datos. Este dispone de tres modos principales: el primero recibe los datos enviados desde el generador por medio de *mdio_out* y transfiere los datos del PHY al receptor por medio de *mdio_in*; el modo escritura también transfiere datos del generador al receptor por medio de *mdio_out* y transfiere las instrucciones correspondientes al PHY por medio de*wr_data*.

2. Descripción arquitectónica

2.1. Generador

En esta ocasión, se programó un módulo que funciona con las siguientes señales: *mdio_start, t_data* y *mdio_in*; las cuales corresponden a las entradas. Cabe recalcar que la entrada *t_data* es de 32 bits y que se transmite del bit menos al más significativo. Luego, se tienen las salidas, que son: *rd_data, data_ready, mdc, mdio_oe* y *mdio_out*. Donde *rd_data* tiene 16 bits. A diferencia de los otros módulos, este funciona cuando la entrada *reset* está en uno. A grandes rasgos, en el modo escritura se pasan los datos cargados a *t_data* uno a uno por la salida *mdio_out*. Por otro lado, en el modo lectura se reciben uno a uno los bits de *mdio_in* en *rd_data*. Es importante recalcar que para ambos modos debe estar en uno *mdio_oe*. A pesar de que para el modo lectura este solo se pone en alto únicamente cuando se están transmitiendo los 16 bits relevantes para la transacción. Mientras que para la transacción esta se mantiene en alto durante la totalidad de la misma. Cabe recalcar que este módulo se programó con base en un código realizado por el profesor Enrique Coen.

Cuyo código consiste en una pequeña máquina de estados con tres estados (INICIO, LECTURA y ESCRITURA). En cada uno de los mismos, se define el comportamiento de cada una de las señales acorde a lo definido en el primer párrafo. De igual manera, se logró controlar exitosamente el *mdc* con la máquina. Originalmente este se generaba a parte pero ya cuando se utilizó este código como base se logró controlar dicho reloj con la misma máquina. Lo cual garantiza que en el modo escritura los datos se transfieren en el flanco positivo del mismo. De igual manera, se agregó un contador. El cual cuenta bits y permite irlos guardando o transmitiendo uno por uno por las respectivas señales. El otro contador que se agregó es el que funciona con el clk. El cual permite generar la señal *mdc*.

El diagrama de estados que permite la funcionalidad del generador es el siguiente:

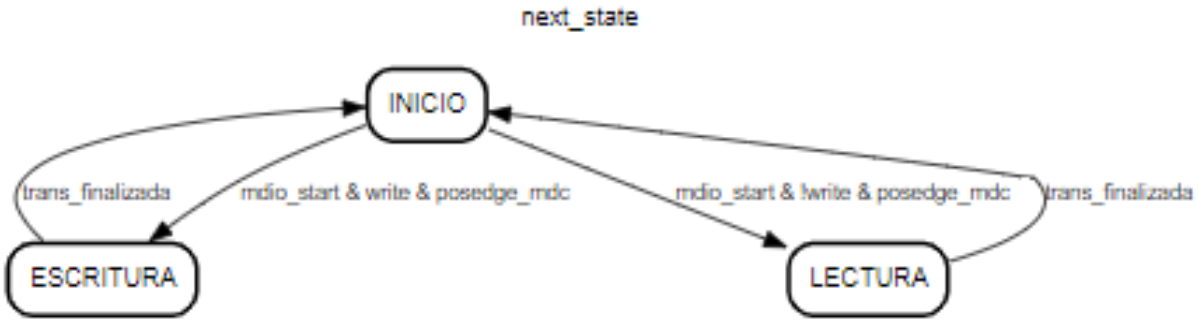

Figura 1: FSM del generador de transacciones MDIO.

2.2. Receptor

Para el receptor de transacciones el modo reset se aplicara cuando la señal *rst* esté en bajo, cuando *rst* vuelva a estar en alto se espera un funcionamiento normal del receptor.

Este recibe *t_data* secuencialmente por medio de la salida del generador *mdio_out*, estos datos son guardados en un registro interno para analizarse más adelante.A continuación, se evalúa si la operación a realizar es de escritura o lectura. Estos datos se reciben y se guardan desde el bit menos significativo hasta el bit más significativo.

Si el modo de operación indica escritura se transfiere al PHY la dirección del registro por medio de la señal de 5 bits *ADDR*, luego se transfieren los datos a escribir al mismo PHY por medio de la señal de 16 bits *wr_data*, esto coincide con los últimos 16 bits de la señal *t_data* que llegan al receptor. Seguidamente, *wr_stb* emite un pulso indicando que los datos de dirección y escritura son correctos y *mdio_done* emite un pulso indicando que se termina la transacción.

En el modo lectura se solicita la información dentro del registro indicado por medio de la señal de 5 bits *ADDR* transmitida en *t_data*, luego se transfiere secuencialmente por medio de *mdio_in* la información proporcionada por el PHY en la señal de 16 bits *rd_data*. Esto también se transmite desde el bit menos significativo hasta el más significativo dentro de la palabra. Al finalizar *mdio_done* emite un pulso indicando que se termina la transacción.

Para asegurar el correcto funcionamiento del receptor se recurrió a una maquina de estados finita, donde el estado 0 recibe *t_data* por medio de *mdio_out* y se almacenan en un registro interno para su análisis mas adelante. El estado 1 es el encargado del proceso de escritura descrito anteriormente. El estado 2 se encarga del proceso de lectura descrito con antelación. Por último el estado 3 es un estado de iddle, este se encarga de volver a asignar las señales en su estado inicial.

Para que la maquina cambie del estado 0 al 1 o al 2, *mdio_oe* debe estar en bajo, si esta señal estuvo en alto por 32 ciclos se pasa al estado de escritura (1); por otra parte si esta señal estuvo en alto solamente por 16 ciclos, la máquina de estados cambiará al estado de lectura (2). Al finalizar los procesos necesarios tanto para el modo escritura como el modo lectura, la máquina de estados cambiará al estado de iddle (3).

Un diagrama de estados de la maquina descrita sería el siguiente:

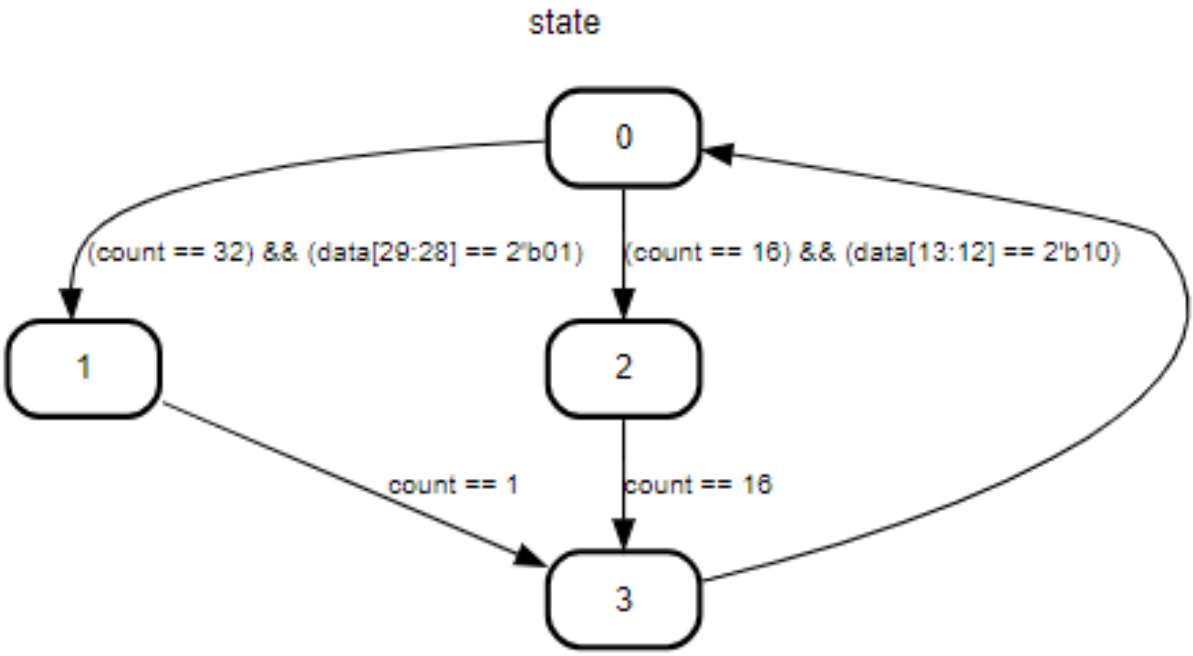

Figura 2: FSM del receptor de transacciones MDIO

3. Plan de pruebas

3.1. Generador

Para el generador de transacciones, fue necesario probar el modo escritura, el modo lectura y verificar que *reset* funcionara adecuadamente.

Para las pruebas de lectura y escritura se utilizó el mismo valor. Para escritura, $t\_data = 0101010101010101$ y para lectura, $rd\_data = 0110010101010101$.

3.1.1. Prueba de lectura 1

Durante el modo lectura deben pasar varias cosas en el generador. Primeramente, el dato por leer debe entrar bit por bit por *mdio_in*. Cabe recalcar que durante las transacciones de lectura la señal *mdio_oe* únicamente está en alto cuando se pasan los 16 bits que se van a leer. Hay que tener presente que los datos transferidos por MDIO tienen la siguiente estructura:

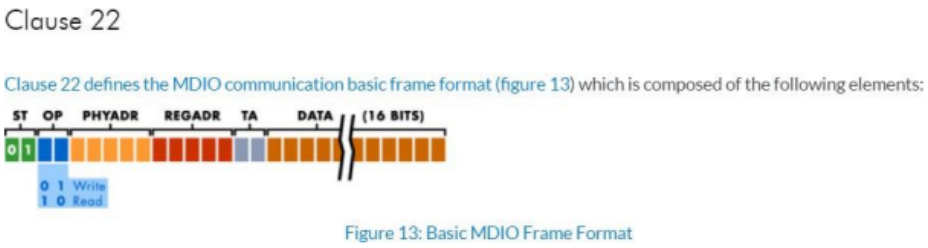

Figura 3: Transacción de MDIO.

Que como se ve en la figura 3 corresponden a los últimos 16. Otra cosa importante es que los bits 29 y 28 indican el tipo de transacción. Para que sea una de lectura, estos deben ser 10 respectivamente. Finalmente, con base en la imagen 4, se concluye que se pasó la prueba.

3.1.2. Prueba de escritura 2

Como se puede apreciar en la imagen 3, los bits 29 y 28 de los datos deben ser 01 respectivamente para llevar a cabo una transacción de escritura. En este caso, la señal *mdio_oe* debe estar en alto siempre que se esté ejecutando la transacción. Ya que se debe generar todo el dato, a diferencia de las transacciones de lectura. Por otro lado, el dato a transmitir se guardará en *t_data* y se irá transmitiendo bit por bit por la salida *mdio_out* en cada flanco positivo de *mdc*. Como se muestra en la siguiente figura. Finalmente, con base en la imagen 5 se concluye que se pasó la prueba de escritura.

3.2. Receptor

Para probar el receptor de transacciones se simula tanto el comportamiento del generador como el del PHY. Las pruebas se realizan una después de la otra en el orden descrito por este documento, permitiendo visualizar ambas pruebas en una sola compilación.

3.2.1. Prueba de lectura 1:

Durante la prueba de lectura se transmitirán los primeros 16 bits de *t_data* correspondientes a la palabra $t\_data = 0110111111100101$. Es importante recalcar que estos se transmiten de forma secuencial por medio de *mdio_out*.

Entrando al modo lectura, se solicita el acceso del registro a leer por medio de la señal de 5 bits *ADDR*, estos son transmitidos en *t_data* en la secuencia de bits correspondiente a *REGADR*.

Más adelante será necesaria retroalimentación por parte del PHY, este será simulado mediante la entrada de 16 bits *rd_data*, misma señal que será transmitida al generador de transacciones por medio de *mdio_in*. El valor de esta señal simulada será $rd\_data = 1001010001101111$.

Para finalizar la transacción, se emite un pulso en la señal *mdio_done*.

Esta prueba se evidencia con éxito en la sección de resultados6.

3.2.2. Prueba de escritura 2:

En la prueba de escritura se transmiten los 32 bits de *t_data* secuencialmente por medio de *mdio_out*, estos datos serán almacenados en un registro interno para su correcta manipulación mas adelante.

Una vez finalizada la transmisión de *t_data* se inicia el proceso de escritura informando al PHY la dirección del registro a escribir, esto se logra asignando a la señal de 5 bits*ADDR* la secuencia de bits en *t_data* correspondientes a la sección *REGADR*.

Seguidamente se transmiten los últimos 16 bits de *t_data* correspondientes a los datos que deben ser escritos en el registro especificado anteriormente. Esto se consigue transmitiendo estos 16 bits por medio de la señal de salida al PHY llamada *wr_data*.

Finalizando la prueba, la señal *wr_stb* emite un pulso indicando que los datos de *wr_data* son correctos. Inmediatamente después se emite un pulso en la señal *mdio_done* indicando que

se termina la transacción.

El correcto funcionamiento de esta prueba se evidencia en la sección de ejemplos y resultados 7.

4. Instrucciones de utilización de la simulación

Debido a que el proyecto se divide en dos partes, para simularlo también se debe hacer en dos partes. Para esto se debe asegurar de abrir la carpeta que contiene los archivos en la terminal y ejecutar las siguientes instrucciones:

- make generador
- make receptor

Con *make generador* se ejecutan las instrucciones necesarias para simular la parte del generador de transacciones, sus pruebas ya están incluidas dentro de estos archivos.

Y con *make receptor* se ejecutarán las instrucciones necesarias para simular las pruebas del receptor de transacciones.

5. Ejemplos de los resultados

5.1. Generador

5.1.1. Prueba lectura

El proceso de pruebas que llevó a estos resultados fue descrito en la sección de descripción de pruebas. Cabe recalcar que para esta prueba la figura pareciera que estuviera cortada ya que se hizo todo en un mismo tester. Además de que se usaron retrasos grandes. Entonces las pruebas están separadas por los mismos.

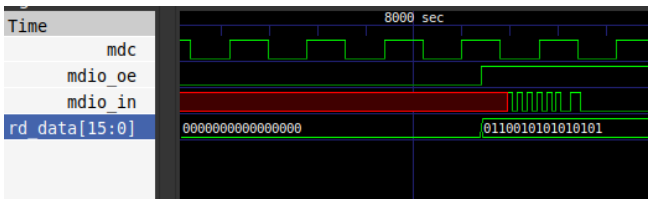

Figura 4: Resultado de la prueba del modo lectura del generador.

5.1.2. Prueba escritura

Lo mismo explicado para los resultados de la prueba de lectura aplican para las de escritura.

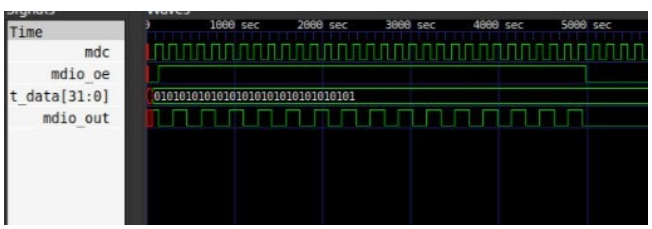

Figura 5: Resultado del la prueba de escritura del generador.

5.2. Receptor

5.2.1. Prueba de lectura

El proceso de funcionamiento de esta prueba fue descrito en la sección plan de pruebas. Cabe recalcar que el modo reset se aplica cuando la señal *rst* esta en bajo.

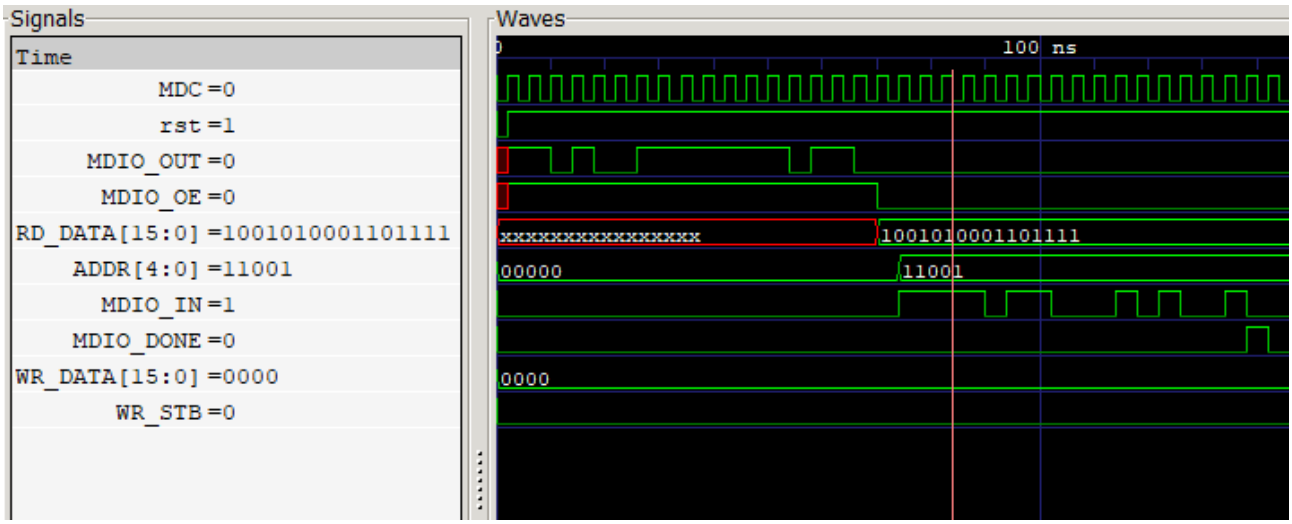

Figura 6: Prueba del modo lectura en el receptor de transacciones MDIO.

Los valores desconocidos (XX) obtenidos son generados debido a que dichos valores no durante el periodo donde se muestran, al ser inicializados estos valores dejan de ser desconocidos (XX). Se concluye que la prueba pasa con éxito.

5.2.2. Prueba de escritura

El funcionamiento de esta prueba fue descrito anteriormente en la sección plan de pruebas. Cabe destacar que, a diferencia de la prueba anterior, esta vez se transmiten 32 bits y no 16 por medio de la señal *mdio_out*.

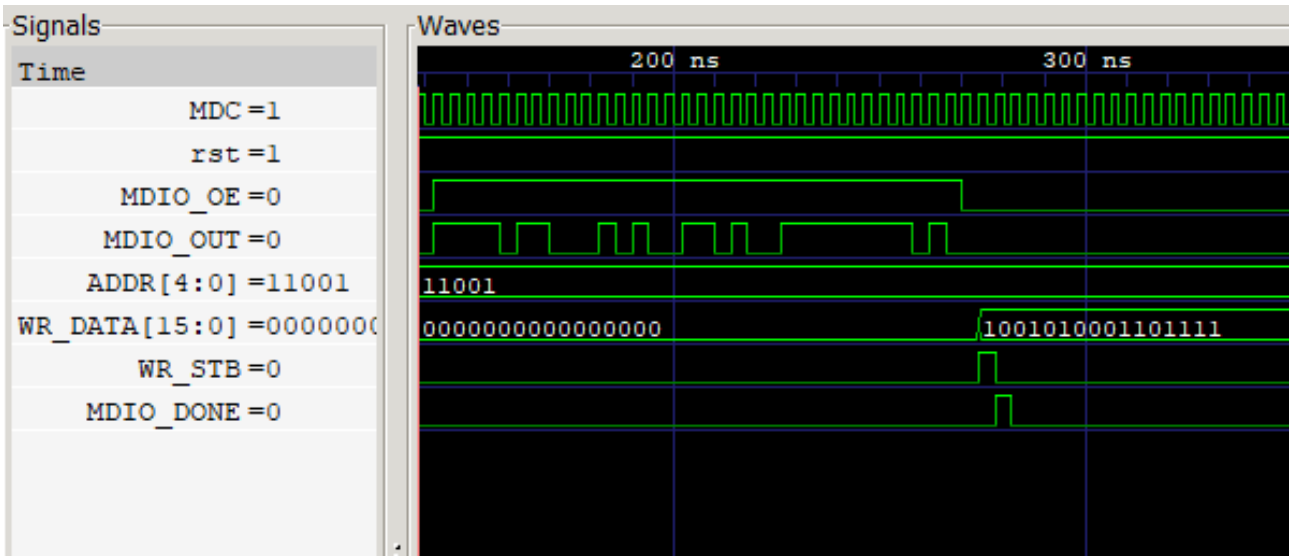

Figura 7: Prueba del modo escritura en el receptor de transacciones MDIO.

Se destaca que el waveform no inicia en el instante de tiempo $t = 0$ debido a que anterior a esta prueba se ubica la prueba de lectura. Se concluye que la prueba pasa con éxito.

6. Conclusiones y recomendaciones

Este proyecto consistió de hacer un un generador y un receptor usando el protocolo MDIO, se logro desarrollar con éxito tanto el generador como el receptor, también se le aplicaron pruebas para lectura y escritura para verificar su funcionamiento como anteriormente se explico.

Una forma de poder mejorar este diseño es el unir *MDI0_in* Y *MDIO_out* en una sola terminal llamada *MDIO* que sea bidireccional, es decir que transmita información de generador a receptor y de receptor a generador.

Una ventaja que se vio de este protocolo es que permite el uso de múltiples dispositivos físicos (PHY) para un solo controlador (CPU).

Uno de los problemas principales que salio a la hora de desarrollar el proyecto fue en el generador que el *MDIO_oe* oscilaba cuando tenia que mantenerse en alto.