Universidad de Costa Rica
Facultad de Ingeniería
Escuela de Ingeniería Eléctrica
IE-0523 Circuitos Digitales II
II ciclo 2022

Tarea 4

Diseño de un generador de transacciones I^2C

Javier Solera Bolaños B66963
Grupo 1

Profesor: Enrique Coen Alfaro

25/10/2022

Índice

1. Resumen

En este trabajo se realizo una maquina de estados para representar el protocolo de I^2C, el cual es un protocolo recibe un bus de datos y dependiendo del dato de RNW, el va a escribir o a leer datos. En esta tarea se implanto con éxito un convertidor de datos paralelo a datos seriales. Entonces el primer dato que recibe el protocolo es el dato de START STB, el cual es un indicador de que puede iniciar una transacción. Después los siguiente bits son Slave address, estos bits por instrucciones del trabajo seria los últimos 2 bits de mi carnet. Luego sigue el bit que me define si es una transacción de escritura o una de lectura, es decir, si RNW = 1, se refiere a una transacción de lectura y si RNW = 0, se refiere a una transacción de escritura. Luego sigue el bit del acknowlege. Luego siguen 8 bits del dato (el dato es de 16 bits, pero el protocolo lo divide en 2), y en medio de los dos datos hay un acknowlege, luego esta el acknowlege negado y por ultimo esta una condición de parada.

También se logro implementar el reloj SCL el cual tiene una frecuencia del 25 % del reloj clk, de manera que todas las transacciones están sincronizadas con el reloj SCL. También se logro implementar la conversión le paralelo a serie en la parte de escritura que seria el RD DATA, es una palabra que llega en paralelo y se convierte en serie y se ve reflejado en el SDA OUT.

2. Descripción Arquitectónica

EL Generador de transacciones que se diseñó, es el siguiente:

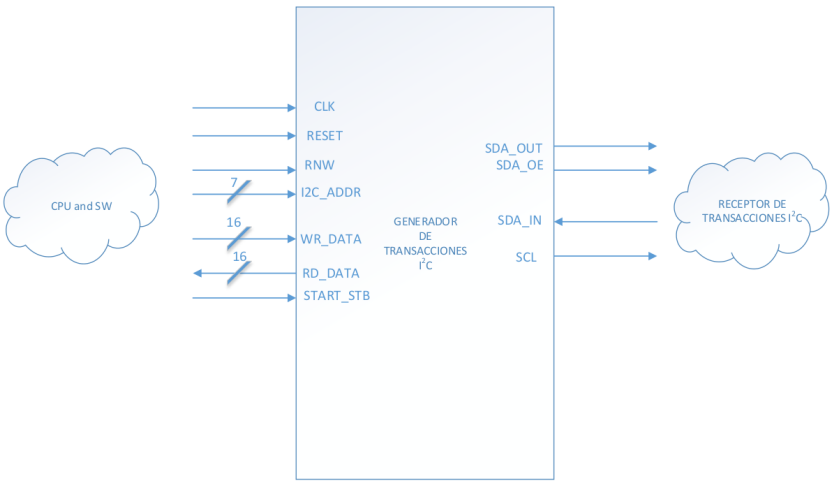

Figura 1: Generador de Transacciones I^2

La clausula en la que se basó la maquina de estados es la siguiente:

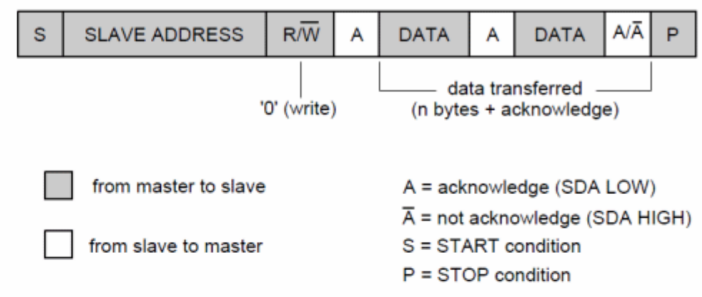

Figura 2: Transacción de Escritura

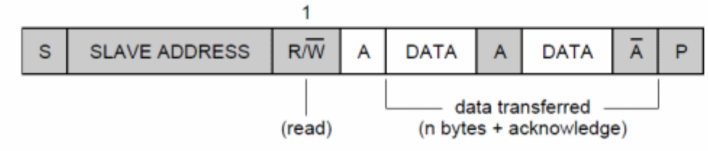

Figura 3: Transacción de Lectura

Para esta tarea, se realizo la siguiente maquina de estados para representar el protocolo de I^2C:

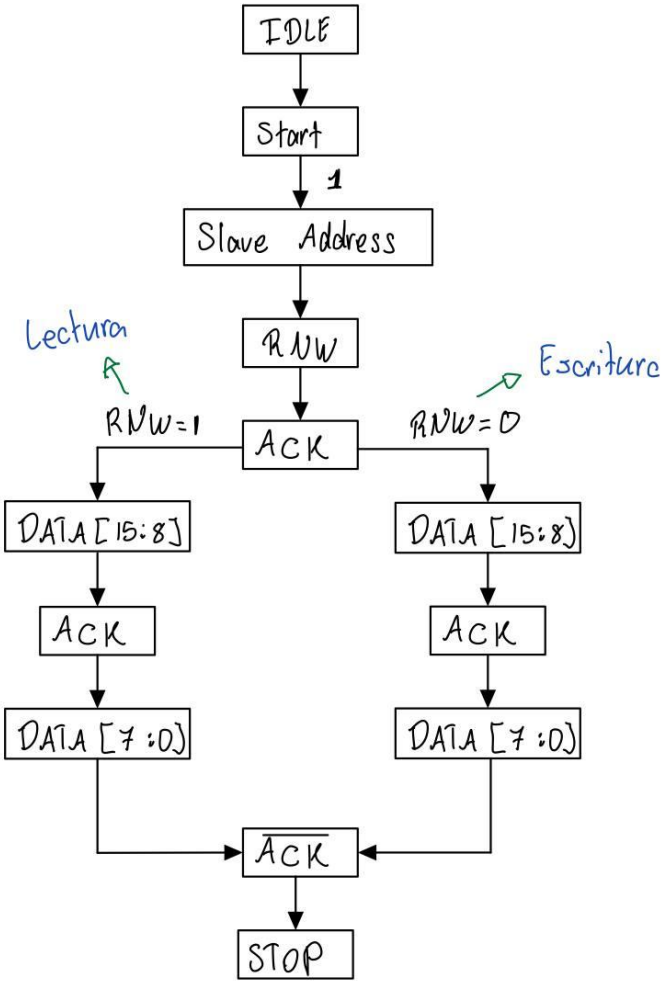

Figura 4: Maquina de Estados para el I^2C

En este diseño, se espera que la maquina de estados, al recibir una palabra y dependiendo de la transaccion que se quiere realizar, escribiria en WR DATA o leeria los datos del SDA IN. Entonces el primer estado es el estaddo IDLE, el cual sirve para resetear todo los valores, luego pasa al estado START, este estado se encarga de indicar cuando se quiere hacer una transaccion, luego esta el estado de Slave address, este estado es el que transmite el numero de carnet, despues esta el estado RNW, este estado es que define si es escritura o lectura, luego esta el estado acknowlege, en este estado se hace la difurcacion de la parte de lectura y de escritura, luego esta el estado de escritura para los primeros 8 bits de la palabra, luego esta el acknowlege, despues llega a otro estado de escritura, pero para los otros 8 bits de la palabra, pasa lo mismo con los estados de lectura y por ultimo pasa al estado de acknowlege negado, el cual tiene un valor contrario al acknowledge, y por ultimo la condicion de paro.

3. Plan de Pruebas

Para esta tarea, se realizaron 3 pruebas importantes:

1. **Prueba para el RESET**: En esta prueba se trata se implementar que mientras el reset esta en bajo, las señales RD_DATA, SDA_OE, SDA_OUT, SCL, RNW y WR_DATA se

ponen en bajo, entonces la maquina de estados no funcionaria.

2. **Prueba para la Transacción de Lectura**: Para esta prueba, se recibe una unos bits que contiene la condición de inicio (START STB), luego el slave address (el cual seria mi carnet, 63), la condición de RNW = 1 y un acknowlege. Luego se reciben 8 bits del SDA IN y se cargan en RD DATA y hace una conversión de paralelo a serie en SDA OUT, luego se recibe el acknowlege, luego los otros 8 bits del SDA IN, luego se recibe el acknowlege negado y por ultimo la condición de parada

3. **Prueba para la Transacción de Escritura**: Esta prueba se recibe una unos bits que contiene la condición de inicio (START STB), luego el slave address (el cual seria mi carnet, 63), la condición de RNW = 0 y un acknowlege. Luego se reciben los 16 bits de WR DATA, los cuales se reciben primero 8, luego un acknowlege. luego otros 8, un acknowlege y por ultimo la condición de parada, estos bit se ven reflejados en el SDA OUT

4. Instrucciones de utilización de la simulación

Es importante que los archivos Maquina_Estado.v, testbench.v, teste.v y el Makefile se encuentren en la misma carpeta.

Entonces puede abrir la terminal y ubicarse en donde están los archivos mencionados anteriormente y correr el comando make, este ejecutará el archivo Makefile y complilará y correrá los archivos.

También puede seguir las instrucciones en la terminal son: iverilog -o tb.vvp testbench.v, esta linea lo que hace es compilar los archivos. Luego tienen que escribir en la terminal: vvp tb.vvp, esto lo que hace es correr la simulación. Y por ultimo, se escribe en la terminal: gtkwave tb.vcd, esta lo que hace es abrir el contador en la aplicación gtkwave.

5. Ejemplos de los resultados

Prueba para el reset en bajo:

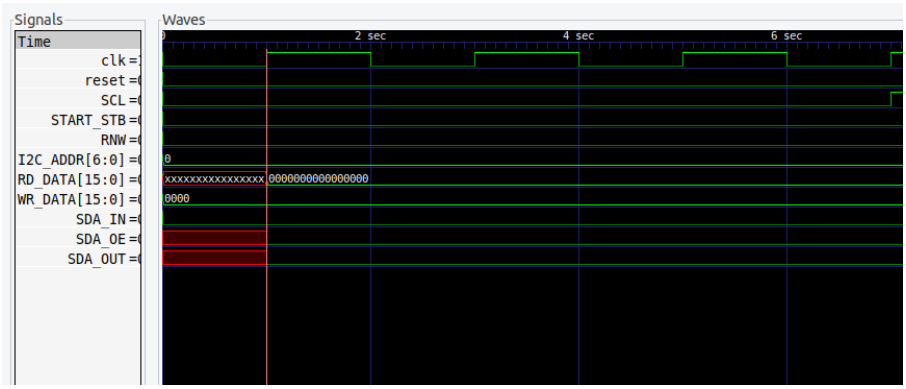

Figura 5: Transacción de escritura

Prueba para la transacción de escritura:

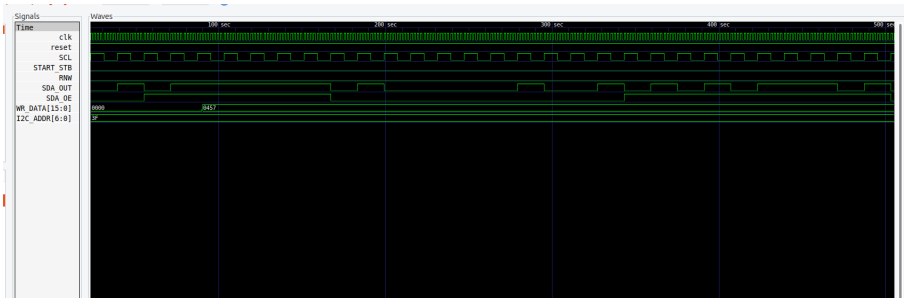

Figura 6: Transacción de escritura

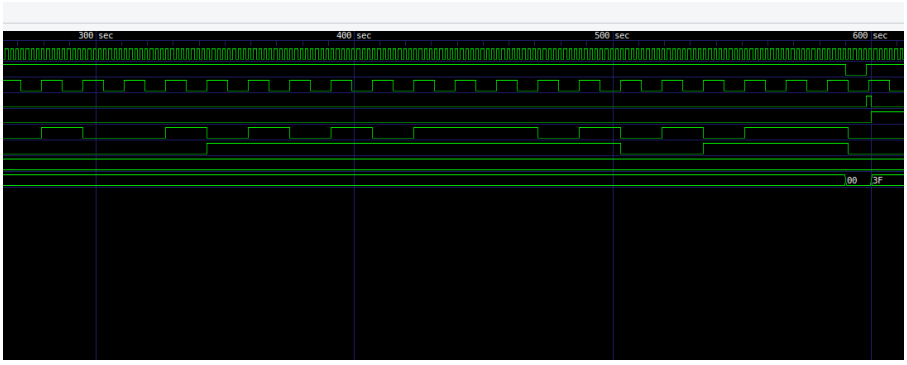

Figura 7: Transacción de escritura

Prueba para la transacción de Lectura:

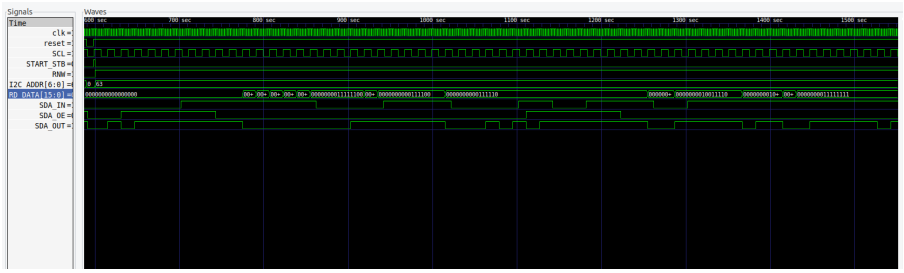

Figura 8: Prueba de Lectura

6. Conclusiones y recomendaciones

Se logro implementar correctamente el funcionamiento del protocolo I^2C, por medio de una maquina de estados y así comprender este protocolo, para lectura y escritura de datos, y además poder implementar correctamente una maquina de estados, la cual dependiendo del estado en que se encuentre me va a realizar la lectura o escritura.

También se logro implementar el convertidor de de paralelo a serie, para una palabra de 16 bits, y que las señales de SDA OUT reciba la palabra en paralelo y la convierta en serie y el SDA OE se ponga en alto cada vez que se necesite que este en alto. Y para la lectura, se logro cargar la palabra proveniente del SDA IN, y que tambien se cargara en SDA OUT.

También se logro implementar que la maquina de estados funcione siempre que la entrada en reset se encuentre en alto, y si esta entrada esta en bajo, todas las señales se pone en bajo.

Como recomendación, es bueno estudiar y comprender el protocolo I^2C IEEE 802.3, de manera que se tenga mas claro el como implementar la maquina de estados para lograr realizar el protocolo I^2C.