IE-0523 Circuitos Digitales II
II Ciclo 2024
Prof. E.Coen

Tarea #4

Diseño de un generador y un receptor de transacciones I^{2}C

1. Diseñar un generador de transacciones I^{2}C de acuerdo con las especificaciones estipuladas en el documento I2C-bus Specification and User Manual revisión 7.0 disponible en el enlace.

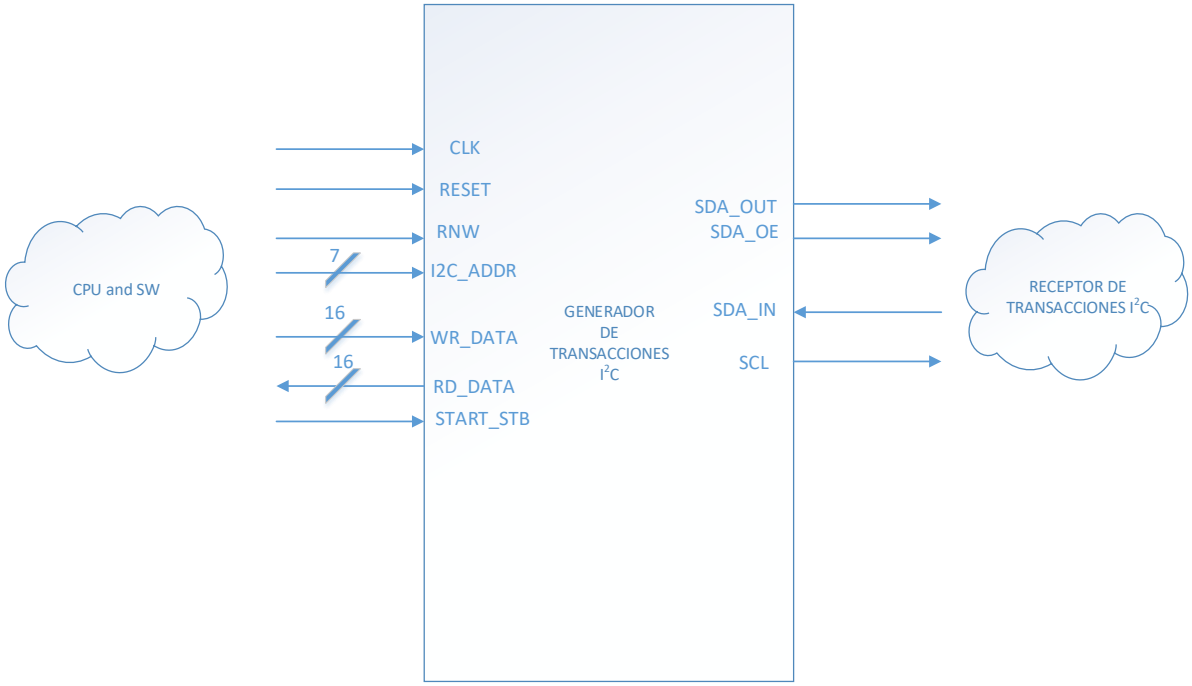

Figura #1: Generador de transacciones I^{2}C

Interfaces del generador:

a) ***CLK*** **–** Es una entrada que llega al generador de transacciones desde el CPU con una frecuencia determinada. El flanco activo de CLK es el flanco creciente.
b) ***RESET*** – Entrada de reinicio del generador. Si *RESET*=1 el generador funciona normalmente. En caso contrario, el generador vuelve a su estado inicial y todas las salidas toman el valor de cero.
c) ***START_STB*** – Strobe (pulso de un ciclo de reloj). Indica al generador que el CPU quiere iniciar una transacción de I^{2}C.
d) ***RNW*** – Read not write. Esta señal indica la dirección de la transacción que desea ejecutar el CPU. Cuando RNW=1 se trata de una transacción de lectura y cuando RNW=0 se trata de una escritura.
e) ***I2C_ADDR[6:0]*** **–** Esta entrada proviene de un registro del CPU y contiene la dirección del receptor de transacciones con quien el generador se quiere comunicar.
f) **SCL**– Salida de reloj para el I^{2}C. El flanco activo de la señal *SCL* es el flanco creciente. Observe que SCL es una salida del generador, que deberá tener una frecuencia del 25% de la frecuencia de la entrada CLK. El generador debe generar SCL con la frecuencia

correcta para cualquier posible valor de la frecuencia de entrada CLK.

g) ***SDA_OUT*** – Salida serial. Debe tener el comportamiento especificado por el protocolo I^2C para las condiciones de START, STOP y transacciones de escritura y lectura.
h) ***SDA_OE*** – Habilitación de SDA_OUT. Se debe poner en 1 para aquellas porciones de la transacción donde el generador tiene control del bus I^2C y ponerse en 0 para las porciones de la transacción donde el receptor tiene control del bus, de acuerdo con las especificaciones del protocolo.
i) ***SDA_IN*** – Entrada serie proveniente del probador. Debe tener el comportamiento especificado por el protocolo I^2C para producir la indicación de ACK, así como proporcionar los datos de entrada en transacciones de lectura.
j) ***WR_DATA[15:0]*** – Entrada paralela. Contiene los 16 bits que se deben enviar por la interfaz I^2C durante una transacción de escritura de acuerdo a este enunciado.
k) ***RD_DATA[15:0]*** – Esta salida debe producir los 16 bits que se reciben desde el receptor de transacciones I^2C durante una transacción de lectura recibida en SDA_IN. Una vez que se ha recibido los 16 bits completos, el generador de transacciones debe enviar la condición de STOP de acuerdo al protocolo I^2C.

2. Diseñar un receptor de transacciones MDIO de acuerdo con las especificaciones estipuladas en el documento I2C-bus Specification and User Manual revisión 7.0 disponible en el enlace.

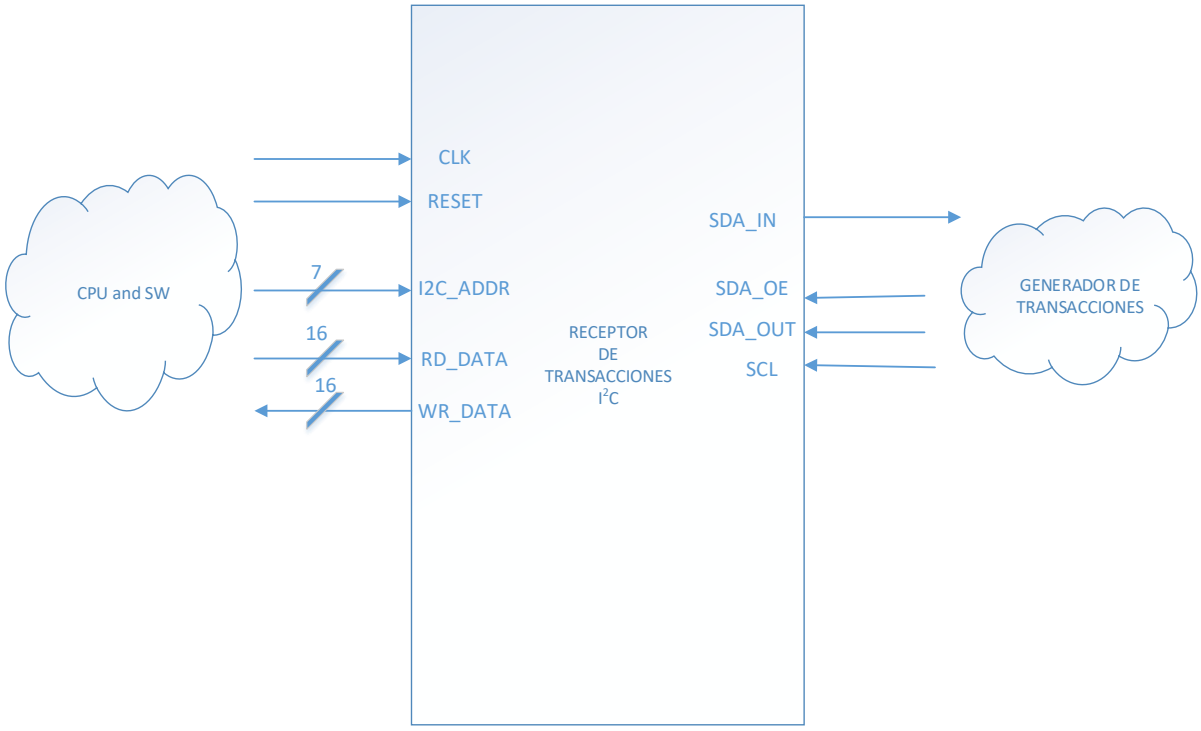

Figura #2: Receptor de transacciones I^2C

Interfaces del receptor:

l) ***CLK –*** Es una entrada que llega al receptor de transacciones desde el CPU con una frecuencia determinada. El flanco activo de CLK es el flanco creciente.

m) ***RESET*** – Entrada de reinicio del generador. Si *RESET*=1 el generador funciona normalmente. En caso contrario, el generador vuelve a su estado inicial y todas las salidas toman el valor de cero.

n) ***I2C_ADDR[6:0] –*** Esta entrada proviene de un registro del CPU y contiene la dirección del receptor de transacciones. Cuando el receptor recibe una transacción, debe cerciorarse de que es una transacción para sí mismo, comparando el valor recibido en SDA_IN con el valor almacenado en este registro.

o) **SCL**– Entrada de reloj para el I^2C. El flanco activo de la señal *SCL* es el flanco creciente. Observe que SCL es una entrada del receptor, que deberá provenir del generador de transacciones o de un probador que emule su comportamiento.

p) ***SDA_OUT*** – Entrada serial. Note que la dirección de esta señal es inversa a la del generador de transacciones. Debe tener el comportamiento especificado por el protocolo I^2C para las condiciones de START, STOP y transacciones de escritura y lectura.

q) ***SDA_OE*** – Habilitación de SDA_OUT. Entrada serial. Note que la dirección de esta señal es inversa a la del generador de transacciones. Se debe poner en 1 para aquellas porciones de la transacción donde el generador tiene control del bus I^2C y ponerse en 0 para las porciones de la transacción donde el receptor tiene control del bus, de acuerdo con las especificaciones del protocolo.

r) ***SDA_IN*** – Salida serial enviada desde el receptor hacia el generador de transacciones. Debe tener el comportamiento especificado por el protocolo I^2C para producir la indicación de ACK, así como proporcionar los datos de entrada en transacciones de lectura.

s) ***WR_DATA[15:0]*** – Salida paralela. Note que la dirección de esta señal es inversa a la del generador de transacciones. Contiene los 16 bits que se reciben por la interfaz I^2C durante una transacción de escritura de acuerdo a este enunciado.

t) ***RD_DATA[15:0]*** – Note que la dirección de esta señal es inversa a la del generador de transacciones. Esta entrada debe producir los 16 bits que se envían desde el receptor de transacciones I^2C durante una transacción de lectura recibida en SDA_OUT. Los bits de RD_DATA se deben enviar a través de SDA_IN de acuerdo con el protocolo.

Descripción básica del protocolo:

- Para iniciar una transacción de I^2C, el generador de transacciones debe producir la condición de inicio. Es decir, debe bajar la señal SDA mientras el SCL permanece en alto. Note que esta transición NO ocurre en un flanco del SCL.
- Una vez que se da la condición de inicio, en el siguiente flanco positivo de SCL se debe transmitir el primer bit de la transacción.
- Al completarse el envío de un byte (8 bits), el generador debe esperar a que el receptor le envíe una señal de confirmación (ACK) de que el byte se recibió correctamente, poniendo en 1 la señal de SDA.
- Una vez que se completa toda la transacción, el generador de transacciones debe enviar una condición de parada (P), que consiste en poner en alto la señal de SDA mientras el SCL también se encuentra en alto.
- La figura #3 resume una transferencia de datos genérica usando el protocolo I^2C.

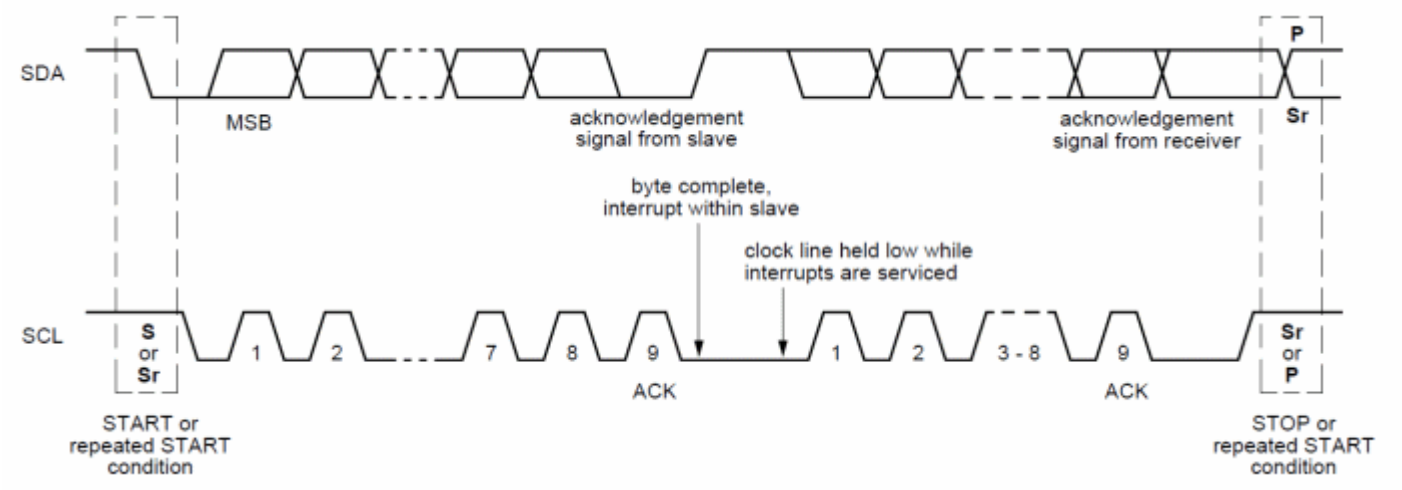

Figura #3: Transferencia de datos de I^2C

- En una transacción de I^2C, el primer byte se usa para identificar el dispositivo con el que se quiere comunicar, y el último bit del primer byte indica si se trata de una transacción de lectura o escritura, como se muestra en la figura #4:

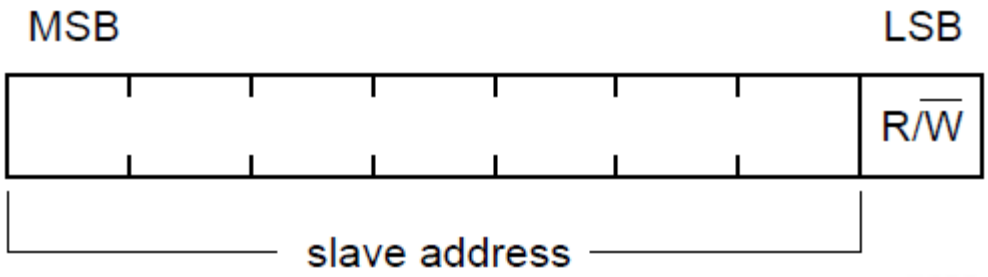

Figura #4: Primer byte de una transacción de I^2C

- Cuando se trata de una transacción de escritura, el generador de transacciones envía el primer byte, espera la confirmación del receptor, y luego repite el proceso para enviar bytes adicionales hasta completar la escritura. Una vez enviado el último byte de la transacción y recibida la confirmación correspondiente, el generador debe enviar la condición de parada para señalar que la transacción ha terminado. La figura #5 ilustra este proceso.

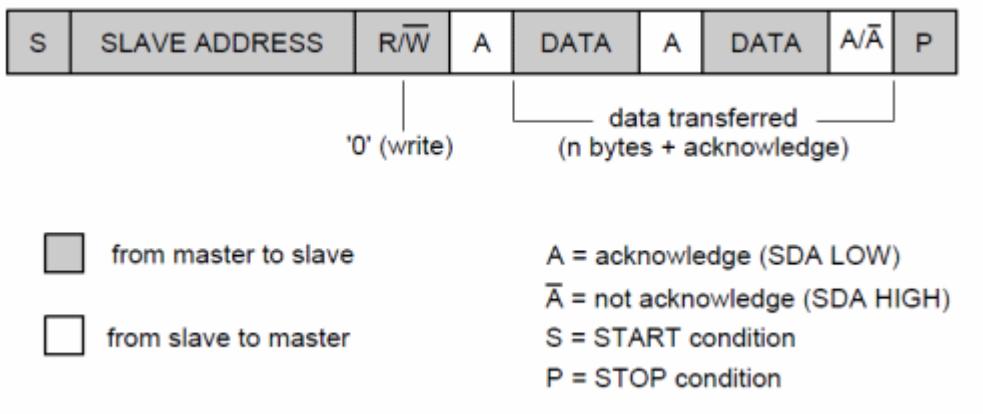

Figura #5: Transacción de escritura I^2C

- Cuando se trata de una transacción de lectura, el generador de transacciones envía el primer byte, espera la confirmación del receptor, y luego espera la recepción del primer byte de la transacción. Una vez recibido el byte, el generador envía una confirmación al receptor y espera el siguiente byte de la transacción. Cuando el generador ha recibido todos los datos que esperaba, envía la condición de parada, para indicar al receptor que debe detener el envío de datos. La figura #6 ilustra este proceso.

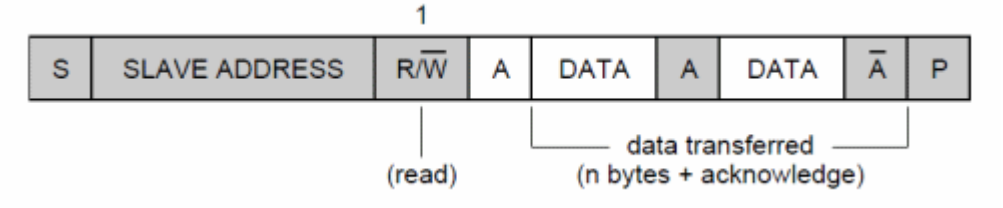

Figura #6: Transacción de lectura I^2C

Referencias adicionales sobre el protocolo I^2C:

https://i2c.info/i2c-bus-specification#:~:text=Standard%20mode%20of%20I2C%20bus,and%20larger%20address%20space%20emerged.

https://www.i2c-bus.org/specification/

https://www.nxp.com/docs/en/application-note/AN10216.pdf

3. Escribir una descripción conductual del generador y del receptor de transacciones de I^2C usando Verilog. Esta descripción servirá como una especificación detallada y formal del funcionamiento del dispositivo diseñado.

4. La descripción en Verilog deberá tener al menos un módulo de banco de pruebas, un módulo probador, y un módulo con la descripción del generador.

5. Definir un plan de pruebas para garantizar el funcionamiento del diseño. El plan de pruebas debe incluir al menos una transacción de escritura de dos bytes de datos y una transacción de lectura de dos bytes de datos. En ambos casos, la dirección del dispositivo receptor deberá corresponder al número decimal representado por los dos últimos dígitos de su carné universitario, codificados en binario. Por ejemplo, si su carné universitario fuera B41047, la dirección de su receptor deberá ser 47 en decimal que corresponde a 7'b0101111

6. Debe incluir en su reporte las formas de onda correspondientes a una transacción de lectura y una transacción de escritura.

7. Con el fin de facilitar el proceso de revisión, se le solicita entregar los archivos como un proyecto en EDA playground o bien organizar los entregables de la tarea de la siguiente manera:

    a. Entregar un solo archivo comprimido, y nombrado según el patrón <# de carné>.<formato de compresión>, por ejemplo B41047.zip

    b. El archivo comprimido descrito en el rubro a) deberá contener específicamente los siguientes archivos:

        i. Reporte en formato PDF cuyo nombre debe seguir el patrón <# de carné>.pdf, por ejemplo, B41047.pdf
        ii. Uno o varios archivos de Verilog con el formato <nombre de archivo>.v que construyan la solución que se solicita en la tarea.
        iii. Un solo archivo probador llamado tester.v
        iv. Un solo archivo de banco de pruebas llamado testbench.v
        v. Un archivo Makefile que permita correr todos los pasos de simulación con una sola línea de comando, tal como se muestra en el ejemplo de la figura #4.

c. El banco de pruebas, testbench.v, deberá incluir a todos los demás archivos *.v, de modo que la compilación y simulación del testbench implique la compilación y simulación de todos los archivos internos.

d. El probador, tester.v, debe escribirse de forma tal que una sola simulación contenga los resultados de una escritura, una lectura y un proceso de RESET, que ejemplifiquen el funcionamiento esperado del módulo.

e. El Makefile debe contener, como mínimo, los comandos necesarios para correr la compilación y simulación de los módulos de la tarea. La figura #3 muestra un ejemplo general como guía. Note que este ejemplo incluye también la síntesis, que NO se solicita en esta tarea.

```
tarea: testbench.v mdio.ys #Archivos requeridos
       yosys -s mdio.ys    #Corre sìntesis
       iverilog -o salida testbench.v #Corre Icarus
       vvp salida #Corre la simulación
       gtkwave resultados.vcd #Abre las formas de onda
```

Figura #7: Ejemplo de Makefile

Rúbrica de Calificación

Tarea #5: Generador y receptor de transacciones I2C	Categoría	% Categoría	% Rubro	% Total
Existe una descripción conductual en Verilog del generador de transacciones de I2C. Esta descripción incluye al menos un módulo de banco de pruebas (testbench.v), un módulo probador (tester.v) y un módulo para el dispositivo bajo prueba (DUT).	Código	20%	20%	4%
Existe una descripción conductual en Verilog del receptor de transacciones de I2C. Esta descripción incluye al menos un módulo de banco de pruebas (testbench.v), un módulo probador (tester.v) y un módulo para el dispositivo bajo prueba (DUT).	Código	20%	20%	4%
Las descripciones de Verilog se entregan en archivos distintos al reporte, listos para ser simulados, e incluyen un archivo de Makefile, de modo que la simulación se corre con una sola línea de comando. Alternativamente se entrega un proyecto en EDA playground.	Código	20%	10%	2%
Las descripciones en Verilog están comentadas adecuadamente para que otras personas entiendan la lógica de la descripción.	Código	20%	10%	2%
Las descripciones en Verilog compilan sin producir errores.	Código	20%	20%	4%
Las descripciones en Verilog ejecutan correctamente. Es decir, corren, entregan algunos resultados y finalizan.	Código	20%	20%	4%
El dispositivo completa una operación de escritura de dos bytes, de forma correcta de acuerdo con la especificación dada.	Pruebas	60%	25%	15%
El dispositivo completa una operación de lectura de dos bytes, de forma correcta de acuerdo con la especificación dada.	Pruebas	60%	25%	15%
El dispositivo reacciona ante la condición de inicio (START) y de parada (STOP) de forma correcta de acuerdo con la especificación dada.	Pruebas	60%	25%	15%
Las transacciones válidas se realizan hacia la dirección correspondiente a los dígitos de su carné. El receptor ignora las transacciones cuando van dirigidas hacia otros dispositivos.	Pruebas	60%	25%	15%
El reporte contiene las seguientes secciones: Resumen, descripción arquitectónica, plan de pruebas, instrucciones de utilización de la simulación para quien califica, ejemplos de resultados, conclusiones y recomendaciones.	Reporte	20%	40%	8%
que se hizo, las partes del diseño que dieron más trabajo para completar y por qué, una explicación de los problemas que se presentaron y cómo se solucionaron.	Reporte	20%	40%	8%
La longitud del reporte no excede 10 páginas.	Reporte	20%	20%	4%

Guía para el reporte (Sigue los mismo lineamientos del reporte de los proyectos)

Se debe entregar en forma electrónica un documento, a lo sumo de 10 páginas de longitud, que incluya los siguientes puntos:

1. **Resumen**: Breve (Media página máximo) descripción de **todo** el proyecto. Esta sección es fundamental pues puede determinar si el lector se interesa o no en leer los detalles del proyecto. Un resumen mal hecho puede esconder un excelente proyecto. El resumen debería incluir:

    a) Descripción breve del sistema, es decir, qué hace. Incluya alguna característica que considere que distingue este diseño en particular.
    b) Las pruebas que se realizaron y qué resultados se obtuvieron. Indique problemas que se tuvieron que considere importante resaltar.
    c) Conclusiones más importantes y recomendaciones para un diseño posterior.

2. **Descripción Arquitectónica**: Incluye un diagrama de bloques con las señales más importantes que sirve como base para describir el funcionamiento del sistema. La descripción va en términos de lo que se espera que el sistema haga. Es decir, se debe detallar la funcionalidad del sistema, el protocolo de las señales que se usan para que funcionen cada una de las partes y las secuencias de eventos que se deben dar. Esta descripción podría ir acompañada de tablas de verdad, tablas de transición de estados, diagramas de estados, diagramas temporales, etc.

3. **Plan de Pruebas**: Aquí se deben enumerar, esto es, se debe presentar una **lista detallada** de las pruebas que se le van a hacer al diseño para verificar que está funcionando de acuerdo a las especificaciones dadas. La lista debe contener por lo menos los siguientes elementos i) Nombre/número de prueba, ii) Descripción de la prueba, y iii) Una indicación de si el diseño la falló o la pasó. Estas pruebas podrían incluir la generación de vectores de entrada para probar en forma exhaustiva todas las líneas de una tabla de verdad o tabla de estados, patrones aleatorios de entradas para tratar de causar errores en la respuesta del diseño, o patrones específicos que ejerciten un cierto modo de funcionamiento. Cada prueba debería ser claramente enumerada en el plan para que también se pueda hacer referencia a ella en el código del banco de pruebas del diseño.

4. **Instrucciones de utilización de la simulación**: Esta sección debe mostrar los comandos necesarios para hacer funcionar la simulación en todos los casos que especifica el plan de pruebas. Hay que suponer que el diseño de un grupo puede ser utilizado por otro grupo o el profesor. Si los resultados no se pueden repetir porque no se conocen los comandos para hacer funcionar la simulación entonces es como si el diseño no funcionara del todo. Se recomienda crear un Makefile de modo que se pueda correr todas las pruebas del caso con un solo comando en Icarus Verilog y GTKwave.

5. **Ejemplos de los resultados**: Una descripción de los resultados más importantes acompañados de los diagramas temporales de la simulación (GTkWave) o cualquier otra salida que demuestre claramente el comportamiento descrito. No es necesario incluir una muestra exhaustiva de resultados, sino que los más representativos del diseño. El punto es mostrarle al lector los comportamientos más sobresalientes para formarle una idea clara

del funcionamiento del diseño. Ya verá el lector si desea más detalles, entonces podrá correr una simulación.

6. **Conclusiones y recomendaciones**: Basado en los resultados obtenidos se indica aquí qué se logró con el proyecto. Puede ser que se concluya que con el diseño propuesto se tiene una limitación en la velocidad de respuesta de… etc. O que con ciertas combinaciones de entradas el diseño se vuelve inestable o los resultados no son los esperados. También se puede concluir qué ventajas o problemas encontraron al seguir el plan de trabajo. A raíz de las conclusiones se puede también recomendar cómo se podría mejorar el diseño o qué otras pruebas se le podrían hacer para garantizar su funcionamiento en otras condiciones que al principio no se consideraron, o también cómo se debería planear el siguiente proyecto para poder cumplirlo a tiempo.