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

Tarea #4

Diseño de una interfaz serial periférica (SPI)

1. Diseñar un transmisor y receptor de interfaz serial periférica (SPI) de acuerdo con las especificaciones estipuladas en el documento SPI Block Guide V4 de NXP disponible en el enlace.

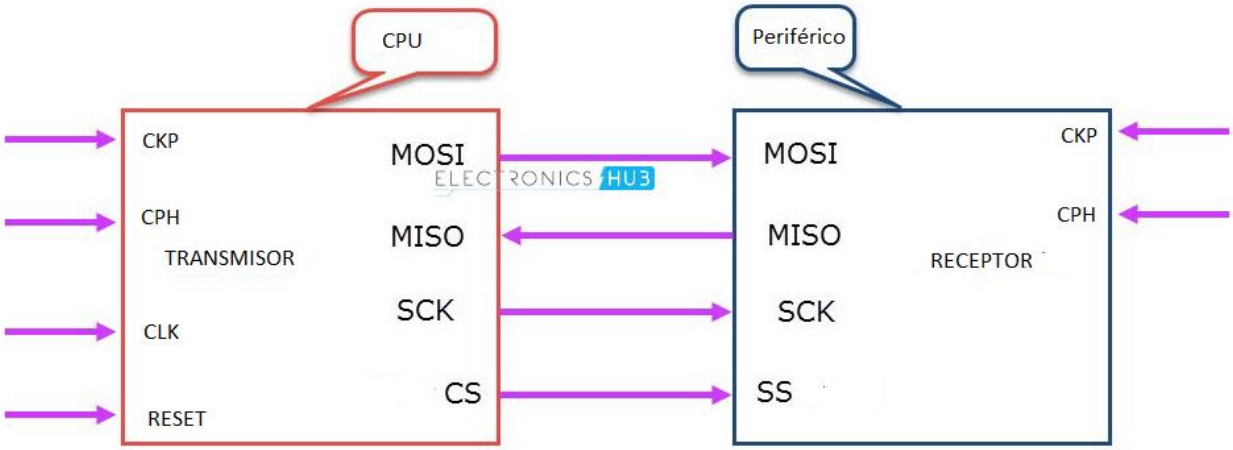

Figura #1: Interfaz serial periférica SPI

Interfaces del generador:

a) ***CLK –*** Es una entrada que llega al transmisor de SPI desde el CPU con una frecuencia determinada. El flanco activo de CLK es el flanco creciente.
b) ***RESET*** – Entrada de reinicio del transmisor. Si *RESET*=1 el transmisor funciona normalmente. En caso contrario, el transmisor vuelve a su estado inicial y todas las salidas toman el valor de cero.
c) ***CS*** – Indicación de "Chip Select". Es una salida del transmisor de SPI, para indicar que va a enviar una transacción hacia el receptor. La señal CS es activa en bajo.
d) ***SS*** – Indicación de "Slave Select". Es una entrada del receptor que le indica que el transmisor está enviándole una transacción. La señal SS es activa en bajo.
e) **SCK**– Salida de reloj para el transmisor de SPI. El flanco activo de la señal *SCK* es el flanco creciente. Observe que SCK es una salida del transmisor, que deberá tener una frecuencia del 25% de la frecuencia de la entrada CLK. El transmisor debe generar SCK con la frecuencia correcta para cualquier posible valor de la frecuencia de entrada CLK. La señal de SCK es, a su vez, una entrada para el receptor, que deberá entregar y recibir un bit nuevo en las interfaces MISO y MOSI en cada ciclo de SCK. Adicionalmente, el comportamiento de SCK deberá ser consistente con las indicaciones de CKP y CPH como se describe más adelante.

f) ***MOSI*** – Salida serial del transmisor, entrada serial para el receptor. En cada ciclo de SCK se espera que el transmisor envíe un nuevo bit de datos y que el receptor lo reciba a través de la entrada del mismo nombre.
g) ***MISO*** – Entrada serial del transmisor, salida serial del receptor. En cada ciclo de SCK se espera que el receptor envíe un nuevo bit de datos y que el transmisor lo reciba a través de esta señal.
h) ***CKP –*** La entrada de polaridad del reloj (clock polarity) determina el estado del reloj cuando el dispositivo está en reposo (idle), es decir, cuando no está realizando una transacción. Si es cero el reloj permanece en cero cuando no se está enviando una transacción y si es 1, deberá permanecer en 1.
i) ***CPH –*** La entrada de fase del reloj (clock phase) determina en cuál flanco de SCK se realiza la transición. Si es cero se transmite en el flanco creciente y si es 1 en el decreciente.

Descripción básica del protocolo:

El protocolo de interfaz serial periférica (SPI) fue desarrollado inicialmente por Motorola en la década de 1980 y se ha convertido en un estándar de facto en la industria, para conectar dispositivos periféricos (memorias, sensores, dispositivos de control, etc.) a un microcontrolador en tarjetas impresas (PCB, por sus siglas en inglés).

Dado que el intercambio de datos entre el transmisor y el receptor depende de cuatro señales (CS, MISO, MOSI, SCK), en la industria también se le conoce como “interfaz de cuatro cables” (4-wire interface).

El transmisor de SPI, que normalmente reside en el microcontrolador (CPU) es quien gobierna la comunicación. Es el encargado de iniciar las transacciones poniendo en bajo la señal de CS y determina el comportamiento de SCK de acuerdo a las entradas de polaridad del reloj (CKP o CPOL) y fase del reloj (CPH).

Por su parte, el receptor simplemente recibe las señales de CS y SCK, de manera que, según el modo configurado, envía y recibe los datos correspondientes en el flanco apropiado del reloj. La figura #2 (tomada de electronicshub.org) resume los diagramas de tiempo para el envío y recepción de datos según los cuatro modos de operación de SPI.

Un aspecto importante del protocolo SPI es que la transmisión de datos es *full-duplex*, es decir, que se envían datos en ambas direcciones de forma simultánea. El transmisor debe ser capaz de enviar bits hacia el receptor y al mismo tiempo almacenar los bits que va recibiendo desde el receptor. De la misma forma, el receptor debe poder recibir y enviar datos al mismo tiempo.

Su implementación debe añadir los dispositivos de almacenamiento necesarios para garantizar el correcto envío y recepción de los datos. Eso puede incluir la adición de variables internas o incluso entradas y salidas adicionales que no se muestran en el diagrama de la figura #1. Estas variables adicionales dependerán de la implementación particular de cada estudiante y deberán ser documentadas en el reporte correspondiente a la tarea #4.

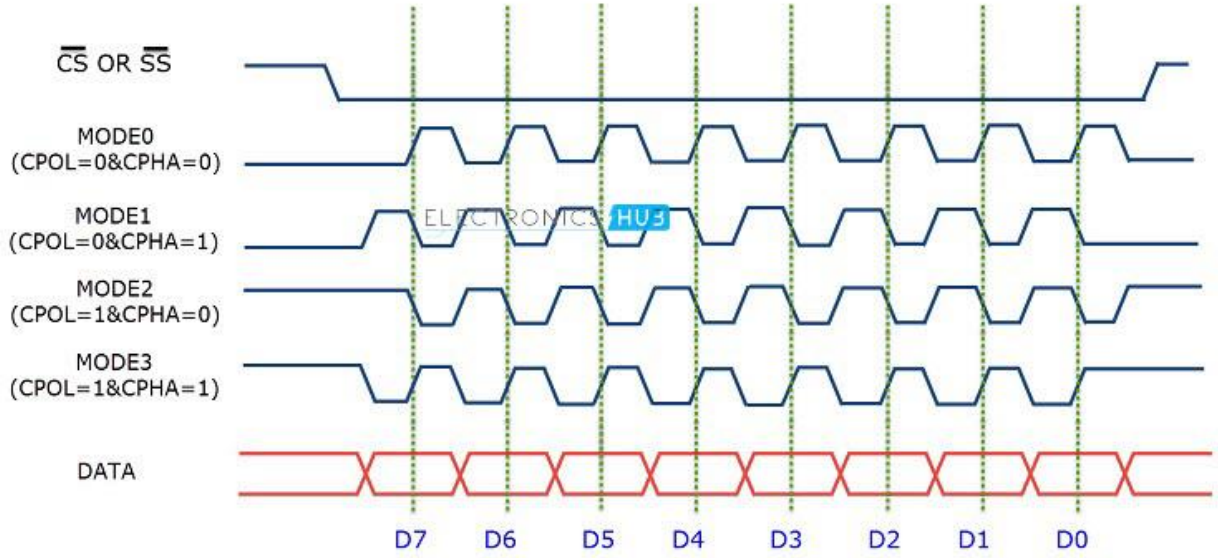

Figura #2: Diagramas de tiempo de transacciones SPI para los distintos modos de operación.

Su ambiente de pruebas debe verificar que los módulos periféricos del SPI pueden conectarse en una configuración de “Daisy Chain” (cadena de margaritas) como se muestra en la figura #3, por lo cual se debe instanciar al menos un módulo de CPU y dos módulos periféricos y conectarlos como se muestra en la figura.

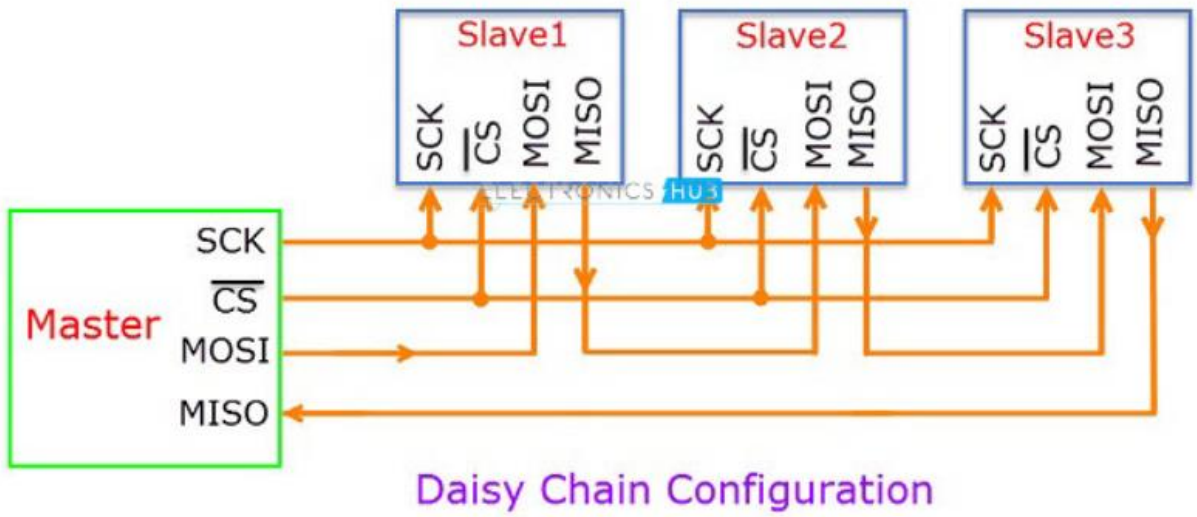

Figura #3: Módulos SPI conectados en configuración “Daisy Chain”.

Referencias adicionales sobre el protocolo SPI:

https://www.nxp.com/files-static/microcontrollers/doc/ref_manual/S12SPIV4.pdf

https://www.electronicshub.org/basics-serial-peripheral-interface-spi/

https://www.intel.com/content/www/us/en/docs/programmable/683126/21-2/motorola-spi-protocol.html

https://ww1.microchip.com/downloads/en/DeviceDoc/70067b.pdf

2. Escribir una descripción conductual del transmisor y el receptor de SPI usando Verilog. Esta descripción servirá como una especificación detallada y formal del funcionamiento del dispositivo diseñado.

3. La descripción en Verilog deberá tener al menos un módulo de banco de pruebas, un módulo con la descripción del transmisor y un módulo con la descripción del receptor. Alternativamente, puede construir un solo módulo de TX/RX que se instancie para funcionar en uno u otro rol. Use Icarus Verilog para su simulación.

4. 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, desde el punto de vista del transmisor. El transmisor debe enviar el tercer y cuarto dígito de su número de carné y el receptor debe enviar en la dirección opuesta el quinto y sexto dígitos, codificados como un byte binario. Por ejemplo, si su carné universitario fuera B41047, el transmisor deberá enviar los bytes 8'b00000001 y 8'b00000000, mientras que recibe del receptor 8'b00000100 y 8'b00000111. Adicionalmente, el plan de pruebas debe garantizar que el funcionamiento es correcto en los cuatro modos de operación definidos del protocolo SPI y que el comportamiento de la señal SCK en reposo es el correcto.

5. El banco de pruebas debe contener al menos tres instancias de módulos SPI, uno funcionando como CPU y dos funcionando como dispositivos periféricos. Las transacciones de escritura y lectura deben hacerse de modo que alcancen al último dispositivo periférico de la cadena.

Entregables:

Con el fin de facilitar el proceso de revisión, se le solicita 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:

- Reporte en formato PDF cuyo nombre debe seguir el patrón <# de carné>.pdf, por ejemplo, B41047.pdf
- Uno o varios archivos de Verilog con el formato <nombre de archivo>.v que construyan la solución que se solicita en la tarea.
- Un solo archivo probador llamado tester.v (opcional)
- Un solo archivo de banco de pruebas llamado testbench.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 #3: Ejemplo de Makefile

Rúbrica de Calificación

Tarea #5: Descripción conductual de una interfaz serial periférica	Categoría	% Categoría	% Rubro	% Total
Existe una descripción conductual en Verilog del generador y del receptor de transacciones de SPI. Esta descripción incluye al menos un testbench que instancia ambos módulos y los conecta entre sí de modo que se puedan intercambiar datos entre ellos.	Código	10%	20%	2%
Se incluye el código no sintetizable (ya sea en un probador o en el testbench) necesario para realizar el plan de pruebas mínimo definido en el enunciado de la tarea y cualquier prueba adicional descrita en el reporte.	Còdigo	10%	20%	2%
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.	Código	10%	10%	1%
Las descripciones en Verilog están comentadas adecuadamente para que otras personas entiendan la lógica de la descripción.	Código	10%	10%	1%
Las descripciones en Verilog compilan sin producir errores.	Código	10%	20%	2%
Las descripciones en Verilog ejecutan correctamente. Es decir, corren, entregan algunos resultados y finalizan.	Código	10%	20%	2%
El sistema completa una transacción full duplex de dos bytes tal y como se describe en el enunciado para el número de carné del estudiante y el dispositivo SPI operando en cada uno de los 4 modos de operación	Pruebas	80%	40%	32%
Todas las transacciones de prueba se realizan sobre un ambiente con al menos dos dispositivos periféricos conectados en configuración de "daisy chain" y se envían y reciben los datos hasta el último dispositivo de la cadena.	Pruebas	80%	40%	32%
Para todos los modos de operación, el valor de reposo del reloj SCK es el correcto.	Pruebas	80%	10%	8%
El reloj SCK muestra una frecuencia del 25% de CLK para cualquier frecuencia de CLK que se proponga.	Pruebas	80%	10%	8%
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	10%	40%	4%
El reporte explica con claridad los detalles relevantes del diseño particular 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	10%	40%	4%
La longitud del reporte no excede 10 páginas.	Reporte	10%	20%	2%

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.