Escuela de Ingeniería Electrónica
Licenciatura en Ingeniería Electrónica
EL3313 Taller de Diseño Digital
II Semestre 2024

TEC | Tecnológico de Costa Rica

Instructivo de Laboratorio

Lab. 2: Lógica secuencial

Profesor

Kaleb Alfaro Bonilla, M.Sc.

Basado en instructivos con autoría de

Dr.-Ing. Pablo Alvarado Moya
Jeferson González Gómez, M.Sc.
Roberto Molina Robles, M.Sc.

1. Introducción

Un circuito secuencial es aquel cuyas salidas dependen tanto de las combinaciones de las entradas **actuales** como de las salidas en instantes de tiempo **anteriores**. Esta característica hace que los sistemas secuenciales sean poderosos en tareas que involucran tiempo, a la vez que presentan mayor complejidad que los circuitos combinacionales.

En sistemas digitales complejos, como los microcontroladores y controladores de dispositivos externos, los circuitos secuenciales juegan un papel fundamental para el almacenamiento de los datos a procesar así como para el desarrollo de unidades de control central, las cuales son requeridas para orquestar el procesamiento de los datos.

En este laboratorio el estudiante aplicará los conceptos de lógica secuencial en el diseño de circuitos digitales para almacenar datos, los cuales serán esenciales para la construcción de un microcontrolador (laboratorio 3).

Recuerde que los diseños desarrollados en este laboratorio **deben** ser completamente **sintetizables**. Las simulaciones finales de cada problema deben ser presentadas a nivel de **implementación temporizada** aplicando técnicas de **auto-verificación** (*self-checking*). Además, **todos** los ejercicios de este laboratorio **deberán ser implementados en la tarjeta con FPGA**. Una consecuencia de lo anterior es que usted debe de incluir a su diseño la lógica necesaria para realizar pruebas físicas (e.g., conexiones a switches o botones, o bloques como el *Integrated Logic Analyzer* o los *Virtual Input/Output*). Finalmente, es un requisito que todos los diseños trabajen

con **único dominio de reloj**, por lo que de requerir diferentes tasas de procesamiento debe de hacer uso de técnicas apropiadas de manejo de reloj y *clock gating*.

1. **Lea y trate de comprender todo el trabajo solicitado antes de iniciarlo.**
2. Incluya en su documentación todas las tablas de verdad, circuitos, figuras, diagramas, etc., que haya usado en la solución de cada ejercicio.
3. Para la presentación funcional, se le pedirá que muestre algunos de los diseños propuestos, ya sea a nivel de implementación o simulación. Dicha selección se hará el día de la presentación.
4. Asegúrese de que el avance del proyecto y los aportes de cada estudiante queden debidamente protocolados con suficientes *commit* en el repositorio y entradas a la bitácora a lo largo de todo el periodo disponible para trabajar en este laboratorio.
5. Se recomienda seguir la metodología de *pull-request* con revisión de pares para realizar los *commits*. [Tutorial]

2. Objetivos

1. Desarrollar un sistema digital complejo e incluir en el proceso de desarrollo etapas de validación.
2. Profundizar en los conceptos asociados al diseño digital secuencial.
3. Describir el funcionamiento de una máquina de estados finitos sencilla.
4. Emplear un controlador maestro SPI para programar una pantalla LCD.
5. Emplear comunicación serie con un computador externo mediante protocolo UART.
6. Evaluar el funcionamiento correcto de un sistema combinacional/secuencial, mediante simulaciones e implementaciones en FPGA.

3. Cuestionario previo

1. Investigue sobre el funcionamiento de máquinas de estado finitos. Explique la diferencia entre una máquina de Moore y una de Mealy, y muestre la diferencia por medio de diagramas de estados y señales.
2. Explique los conceptos de *setup time* y *hold time*. ¿Qué importancia tienen en el diseño de sistemas digitales?
3. Explique los conceptos de tiempos de propagación y tiempos de contaminación en circuitos combinacionales. Investigue sobre la ruta crítica y cómo esta afecta en el diseño de sistemas digitales complejos; por ejemplo, un procesador con *pipeline*. Investigue su relación con la frecuencia máxima de operación de un circuito.

4. Investigue sobre las mejores prácticas para la asignación de relojes y división de frecuencia en FPGAs. En este apartado haga énfasis en el uso de las entradas habilitadoras de reloj (*clock enables*) presentes en las celdas de la FPGA, para lograr tener tiempos de ejecución diferentes a lo largo del sistema mientras se utiliza un solo reloj.

5. Investigue sobre el fenómeno de *rebotes* y ruido en pulsadores e interruptores. Defina qué técnicas digitales (circuitos) se utilizan para cancelar este fenómeno. Además, investigue sobre los problemas de metastabilidad cuando se tienen entradas asíncronas en circuitos digitales. Finalmente, presente circuitos que permitan la sincronización de entradas como pulsadores e interruptores.

6. Investigue sobre la especificación de la interfaz SPI. Preste atención a los aspectos necesarios para poder diseñar un controlador maestro de SPI, además de los diferentes *modos* de SPI.

7. Investigue sobre la comunicación serie UART. Preste atención a las diferentes características de configuración necesarias para la comunicación serie mediante UART (por ejemplo, *baud rate*, paridad, etc). Además, investigue cómo puede utilizar puertos seriales en su computadora, considerando el sistema operativo que utilice.

8. Investigue el funcionamiento básico del controlador ST7789V de la pantalla LCD RGB de la tang nano 9k. La hoja de datos será entregado por el profesor del curso.

4. Procedimiento

4.1. Ejercicio 1 - Diseño antirebotes y sincronizador

1. Escriba los bloques digitales necesarios, en SystemVerilog, para eliminar rebotes y sincronizar entradas provenientes de pulsadores e interruptores. Elabore un diagrama de bloques.

2. Incluya en su diseño y diagrama de bloques el siguiente `contador de pruebas`, el cual le será dado. El bloque está sincronizado con el reloj por medio de la entrada `clk`. Además, hace uso de un reset `rst` **activo en bajo**. Una señal habilitadora **activa en alto**, `EN`, permite realizar un incremento cada vez que se da un flanco positivo en esta señal. Una señal con rebotes puede causar conteos indeseados.

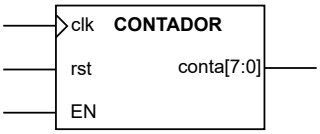

Figura 1: Contador de pruebas para sistema antirrebote.

3. Genere pruebas en FPGA con un bloque de pruebas que reciba como entrada uno de los pulsadores presentes en la tarjeta y lo use como `EN` de su circuito contador. La señal proveniente del botón debe ser procesada por su bloque **sincronizador y anti-rebote**.

4. En su bloque (módulo) superior (`top`) haga uso de los LED incluidos en la tarjeta para mostrar la salida de su contador.

5. Escriba un *testbench* para su sistema. Tiene que elaborar una prueba con el escenario de un pulso contaminado por rebotes y repetirse varias veces. Asegúrese de realizar las mismas simulaciones en *post-implementación*.

6. Descargue el diseño a la tarjeta FPGA y verifique su diseño. Asegúrese de asignar apropiadamente las señales.

4.2. Ejercicio 2 - Interfaz para teclado hexadecimal

La Figura 2 muestra un diagrama básico de una interfaz para un teclado hexadecimal. En el laboratorio 1 se desarrollaron los bloques combinacionales presentes en el sub-cuadro *protoboard*. En este laboratorio se completará la interfaz por medio de bloques implementados en la FPGA. El objetivo de esta interfaz es determinar que presionó una tecla y que la salida muestre el valor de *la etiqueta* en el teclado (e.g., si presionó la tecla con la letra A, la salida debe ser el valor hexadecimal `Ah`).

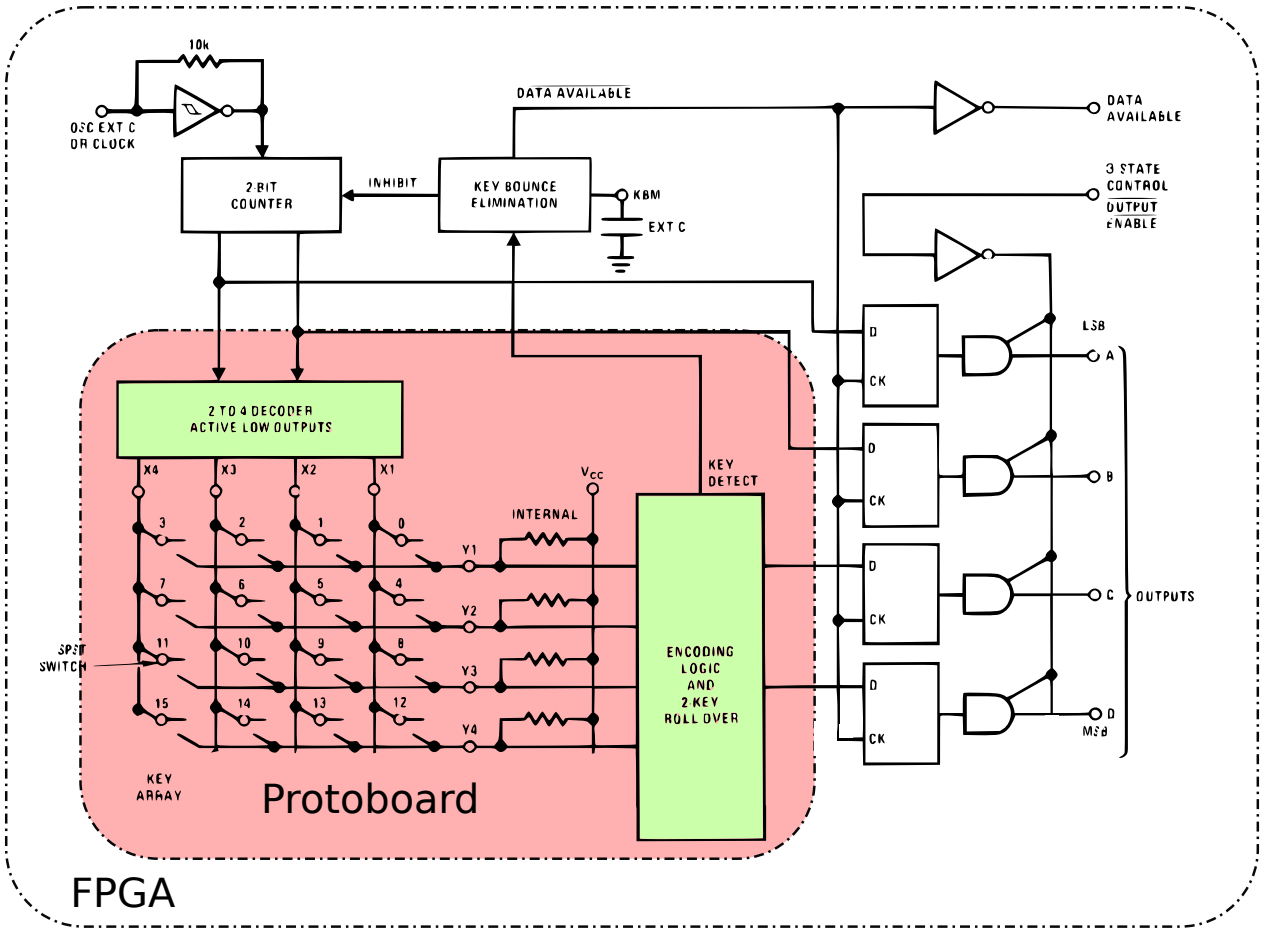

Figura 2: Diagrama conceptual de la interfaz de teclado hexadecimal (no se muestran líneas de reset o reloj)

1. Analice la función que cumple cada uno de los bloques presentes en la interfaz (contador de 2 bits, divisor de reloj, codificador de tecla, y sincronizador y antirebote). Preste especial

atención a la temporización de las señales y el flujo de estas desde la entrada hasta la salida del módulo.

2. A partir del análisis anterior, defina las tablas de verdad, diagramas de estado y diagramas temporales requeridos para resolver cada bloque. **Sugerencia:** usar esta herramienta para los diagramas de tiempo

3. Desarrolle de forma individual cada uno de los bloques necesarios para la interfaz en SystemVerilog. Se recomienda usar un nivel alto de abstracción posible (no usar compuertas lógicas explícitamente). Valide la funcionalidad de cada sub-bloque.

4. Defina un módulo, de mayor nivel de jerárquía, que combine de forma estructural cada subloque para formar la interfaz dentro de la FPGA. Seleccione uno de los puertos del pin header en su tarjeta FPGA para conectar las señales hacia y desde la *protoboard* que contiene el teclado y otros bloques combinacionales.

5. Desarrolle y ejecute una prueba de validación a nivel de simulación de la interfaz completa. Usted debe implementar un escenario dónde se simule el rebote de una de las teclas.

6. Implemente el sistema en la tarjeta FPGA. Recuerde que debe agregar hardware adicional para demostrar el funcionamiento (LEDs y/o *display* de 7 segmentos).

4.3. Ejercicio 3 - Interfaz Serial Asíncrona

Este protocolo se ha utilizado por décadas para la comunicación entre sistemas. En este laboratorio estaremos desarrollando una interfaz UART (Universal Asynchronous Receiver/Transmitter). Con el bloque aquí desarrollado, la FPGA podrá comunicarse de forma bidireccional con otros sistemas de procesamiento (por ejemplo, un computador personal).

1. Para implementar la interfaz UART puede hacer uso de un código encontrado en algún repositorio cuya licencia sea libre. La integración y funcionamiento de dicho módulo corre por su cuenta.

2. Debe elaborar un set de pruebas para asegurar que dicho módulo UART funciona correctamente.

3. La interfaz UART a desarrollar debe ser capaz de manejar el protocolo serial bidireccional a una velocidad de 9600 baudios.

4. Desarrolle un bloque de pruebas apropiado que permita enviar a una computadora personal valores introducidos por el teclado desarrollado en el ejercicio 2. Adicionalmente, la prueba debe mostrar en los LEDs datos recibidos desde dicha computadora. En la figura 3 se ilustra el diagrama de bloques ideal para la prueba en físico.

5. Debe al menos elaborar una simulación de integración total de todos los bloques que haya elaborado: teclado, LEDs y UART.

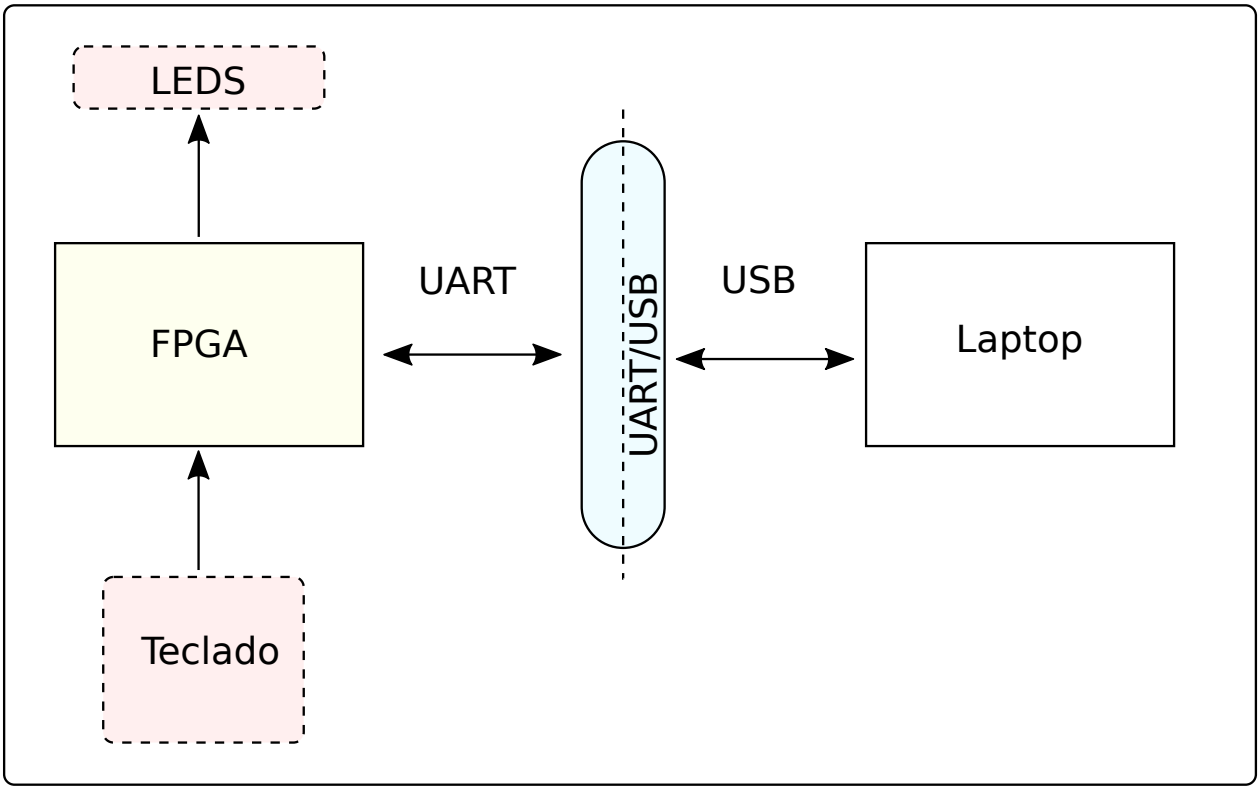

Figura 3: Diagrama conceptual de la interfaz UART conectado al computador y FPGA. LEDS muestran datos recibidos por el computador.

4.4. Ejercicio 4 - Interfaz Serial Síncrona

El *Serial Peripheral Interface* (SPI) es un protocolo de comunicación serie, síncrono, basado en una arquitectura maestro-esclavo o principal-secundario. Este protocolo se utiliza comúnmente para comunicar un juego de dispositivos periféricos (esclavos) con uno central (maestro). En esta parte del laboratorio, usted desarrollará una interfaz SPI maestro la cual permitirá comunicar el sistema en el FPGA con dispositivos externos, en este caso la pantalla LCD.

1. Para implementar la interfaz SPI puede hacer uso de un código de un tercero en donde se cumplan los mismos requisitos de licencia del ejercicio anterior.

2. Se debe prestar atención como utilizar el SPI para programar una secuencia de transferencia de datos. Elabore una simulación en dónde se compruebe que el módulo SPI escogido cumpla el protocolo para un único dispositivo.

3. Con base a la hoja de datos del controlador ST7789V, elabore un procedimiento de configuración para programar la LCD. Puede basarse en alguna biblioteca de Arduino o del ejemplo encontrado en `https://github.com/sipeed/TangNano-9K-example/tree/main/spi_lcd`. Se debe documentar cada instrucción del LCD enviada por SPI.

4. Con base a la ilustración de la figura 4, elabore un diseño donde desde el computador se escoge dos posibles configuraciones de patrón de color. Cada configuración de color va a tomar únicamente dos posibles colores; P1:Rojo y P2:Azul como configuración 1, y P1:Verde

y P2:Azul como configuración 2 por ejemplo. A partir de estos dos colores se pintará una grilla de colores en pantalla ante el accionar del teclado de la laptop. Debe ser posible alternar entre cada configuración una vez inicializada el LCD.

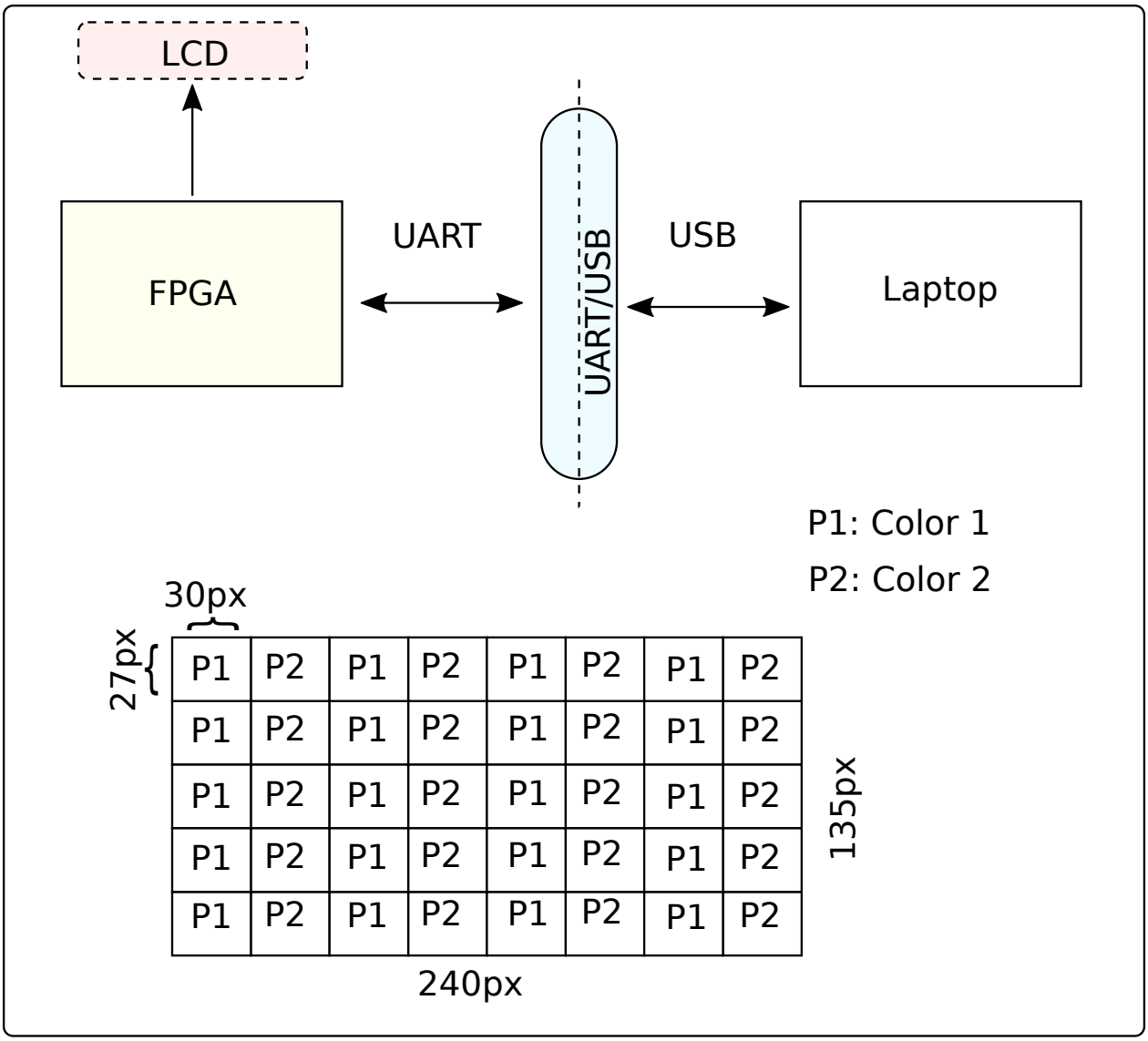

Figura 4: Diagrama conceptual de la interfaz SPI de la LCD conectado a la FPGA y Computador. El computador elige entre dos posibles configuraciones de colores.

5. Evaluación

Las fechas oficiales de inicio, fin y entregas parciales de este laboratorio son:

- **Inicio:** viernes 30.08.2024
- **Fin:** viernes 20.09.2024 (demostración funcional - **Simulaciones y en FPGA**)
- **Cuestionario previo:** 06.09.2024
- **Revisión del repositorio:** 18.09.2024

El presente laboratorio tiene un valor del **20 %** de la nota final del curso y se evaluará de la siguiente manera:

1. Funcionalidad final 40 %
   - *a*) Antirebotes [15 %]
   - *b*) Teclado [25 %]
   - *c*) UART [30 %]
   - *d*) SPI [30 %]
2. Cuestionario previo 10 %
3. Documentación técnica del sistema desarrollado 25 %
4. Manejo de repositorio 25 %

Recuerde citar adecuadamente en su documentación cualquier información proveniente de fuentes bibliográficas.