ELO212: Laboratorio de Sistemas Digitales
Semestre I-2020
Guía de Actividades Sesión 5

16, 17, 19 de Junio de 2020

1. Objetivos

Los objetivos para esta sesión son:

- Revisar diseños previos y comparar con módulos de referencia.
- Familiarizarse con los procesos de especificación de constraints, síntesis lógica e implementación de un diseño para FPGA utilizando Vivado.
- Experimentar con conceptos básicos de circuitos secuenciales sincrónicos.
- Describir y experimentar con las estructuras básicas de circuitos retentores (bancos de registros y memorias).

2. Actividades

2.1. Revisión ALU de referencia y diseños previos.

En el repositorio Github del curso se encuentra disponible un archivo de SystemVerilog que describe una ALU como la solicitada para la sesión anterior.

Revise el código de referencia y compárelo con la descripción desarrollada y entregada por Ud. en la sesion anterior. Analice y compare el resultado del Elaborated Design de ambas implementaciones. Genere un testbench en donde instancie el módulo de referencia y el desarrollado por Ud., y compare los resultados.

Notar que el código de referencia es solo eso, una referencia desarrollada por alguien con mas experiencia, que ha aprendido los sucios detalles, y ya ha desarrollado un estilo de codificacion ordenado y eficiente. Se enfatiza que esto no representa la forma correcta de hacerlo, pero si una forma limpia, eficiente, y autodocumentada. Hay literalmente infinitas formas de describir el comportamiento para llegar a un resultado funcionalmente equivalente, y finalmente la herramienta de diseño es la que se encarga de inferir el hardware necesario para la implementacion.

La ALU que estaremos usando en el curso es una versión bastante simplificada de una ALU real. Para conocer más detalles, se recomienda revisar más detalles de operación, lo cual pueden realizar

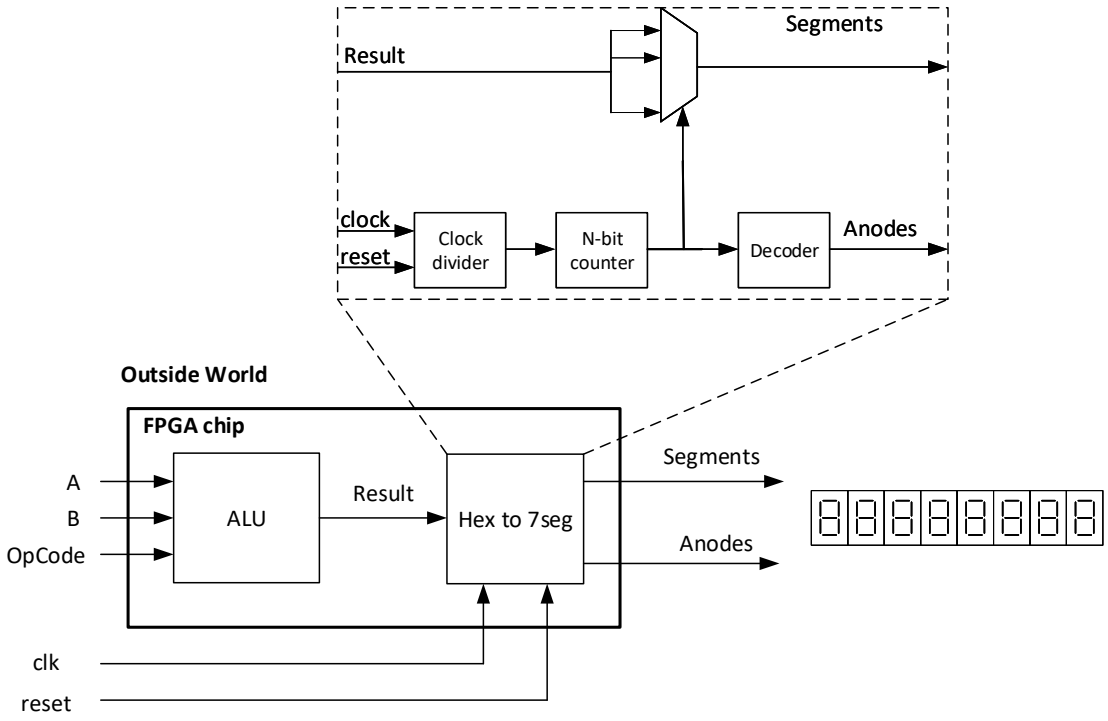

Figura 1: Diagrama de alto nivel para ALU con salida a display de 7-segmentos.

usando el libro de referencia de Harris, o hacer búsquedas en Internet. Por ejemplo, una posible implementación para CPU se muestra en el siguiente enlace: `https://zipcpu.com/zipcpu/2017/08/11/simple-alu.html`. En particular, la ALU seguirá siendo una pieza fundamental en asignaturas posteriores para la carrera de Ingeniería Civil Electrónica.

2.2. Procesos de síntesis lógica e implementación.

Utilizando como referencia el diagrama de alto nivel de la Figura 1, el profesor explicará como agregar archivos de constraints al proyecto, y los procesos de síntesis lógica e implementación para conectar los pines de sus módulos lógicos al mundo físico.

Posterior a la explicación, implemente un diseño similar al planteado en la Figura 1 utilizando sus módulos y considerando las especificaciones de la tarjeta de desarrollo Nexys4DDR o Nexys A7 para conectar las señales a periféricos de entrada y salida adecuados. Identifique cuáles serían las restricciones en la dimensión de los datos de entrada considerando los periféricos disponibles. El diagrama de alto nivel planteado es sólo una sugerencia, y Ud. puede tomar decisiones que le parezcan adecuadas, siempre utilizando los principios de jerarquía y modularidad, además de mantener los nombres de las señales de entrada y salida.

2.3. ALU con entradas registradas.

Utilizando los módulos desarrollados en etapas anteriores, extienda el diseño anterior para agregar la funcionalidad descrita en el diagrama de alto nivel de la Figura 2. En este caso, los operandos

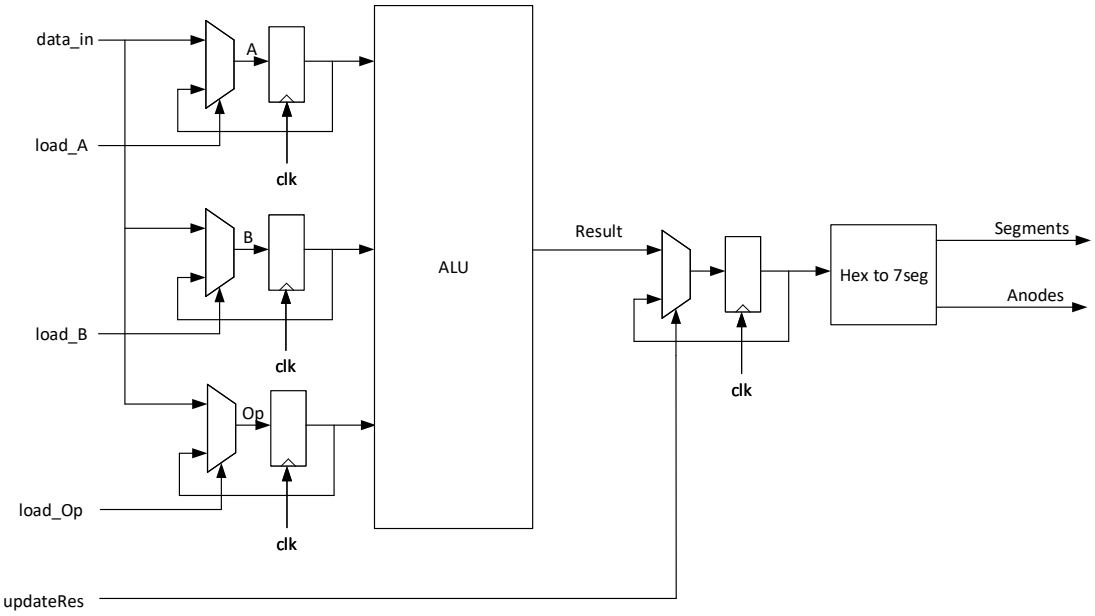

Figura 2: Diagrama de alto nivel para ALU con entradas registradas de carga condicional.

de entrada a la nueva ALU provienen de bancos de registros con carga condicional. Adicionalmente, se debe incorporar un banco de registros a la salida de la ALU para almacenar el valor de la última operación realizada.

El comportamiento esperado se especifica a continuación (de todas formas, este comportamiento debería inferirse directamente del diagrama):

- Existe un único bus de entrada de datos que está directamente conectado a las señales A, B, y OpCode.
- Las entradas de la ALU provienen de bancos de registros. El valor almacenado en el banco de registros se actualiza en base al valor actual del bus de datos sólo cuando se active la señal de carga correspondiente. Si las señales de carga no están activadas, entonces los cambios en el bus de datos no se ven reflejados en las entradas de la ALU.
- La ALU está activa en todo momento calculando la operación especificada en sus datos de entrada (es un circuito combinacional). Sin embargo, el resultado de la operación solo se propaga a los pines de salida cuando se activa la señal UpdateRes. De esta forma, solo se propaga el valor estable del resultado, evitando propagar cambios espúreos generados mientras se están actualizando los datos de entrada para calcular una nueva operación.
- Incluya un botón de reset que permita llevar la salida de los bancos de registros a un valor deseado en cualquier momento.

Puede revisar el material disponible en los links de Xilinx (propuestos en una sesión anterior) para técnicas de descripción de contadores y registros. Estos scripts están en Verilog clásico, pero deberían ser capaces de traducirlos a SystemVerilog.

Una vez descrito su diseño, observe, analice y compare los resultados obtenidos mediante el Elaborated Design y la síntesis lógica. Haga todas las consultas que considere necesarias para clarificar sus dudas. Revise cuidadosamente los mensajes entregados por el procesos de síntesis, y verifique que está todo en orden. Debe aprender a diferencias los Warnings que son informativos de los que son críticos para el diseño. Recuerde que sus descripciones nunca jamás deben inferir latches cuando está trabajando con FPGAs!

Mapee los pines a interfaces disponibles en la tarjeta. Analice y discuta las diferencias en términos de las restricciones en los datos de entrada para esta actividad, con el diseño de la actividad anterior.

2.4. Notación polaca inversa.

Investigue en que consiste la notación polaca inversa (`https://mathworld.wolfram.com/ReversePolishNotation.html`). Piense cómo integrar esta notación a su sistema, para implementar una calculadora básica que siga esta notación, similar a la que usan algunas calculadoras comerciales (`http://www.alcula.com/calculators/rpn/`). Considere que a parte del bus de datos, dispone de un único botón de ENTER que está compartido entre todos los registros para el ingreso de datos. Es decir, los operandos y operador deben ser ingresados en cierto orden. *Hint: diseñe una máquina de estados finitos para generar señales internas en base a una entrada ENTER.*