Estructura de Computadores

Apellidos y Nombre: _____

Calculadora RPN Simple

Departamento de Tecnología Electrónica
Universidad de Sevilla
Germán Cano Quiveu, Jorge Juan Chico,
David Guerrero Martos, Isabel Mª Gómez González
Adaptación Marzo 2024

1 Material

- Ordenador con Xilinx ISE instalado.
- Basys2 development board and documentation. Incluye la FPGA Xilinx Spartan-3E.
- Ficheros con las plantillas para el diseño (lab kit).

2 Objetivos

- Describir Sistemas digitales en Verilog utilizando una metodología por fases.
- Simular el comportamiento de cada componente del sistema así como del sistema completo.
- Implementar sistemas digitales en una FPGA.
- Dotar al sistema de entrada/salida para permitir la introducción de datos y la visualización del resultado.

3 Especificación del sistema

El objetivo de esta práctica de laboratorio es diseñar una calculadora muy simple basada en la Notación Polaca Inversa (*Reverse Polish Notation* –RPN–). Se tendrá que realizar una fase de especificación y simulación y otra de implementación. Este apartado está dedicado a la especificación. Para realizarlo se utilizarán las técnicas de sistemas digitales vistas en la teoría de la asignatura.

El sistema podrá realizar cuatro operaciones seleccionables mediante 4 entradas: *enter*, *add*, *sub* y *neg*. Esta conformado por una ALU y un registro (yreg) conectado a su salida que permite almacenar el resultado de la operación efectuada. Los operandos son: 1) un dato externo (x_in) conectado a la entrada de la ALU y, 2) el dato almacenado en yreg. En la Tabla 1 se muestran las macrooperaciones del sistema.

Tabla 1: Macrooperaciones.

enter	add	sub	neg	OPERACIÓN
1	0	0	0	yreg ← x_in
0	1	0	0	yreg ← x_in + yreg
0	0	1	0	yreg ← yreg - x_in
0	0	0	1	yreg ← -yreg
Otras combinaciones				yreg ←yreg

En la Figura 1 se muestran la unidad de datos propuesta y su diagrama de bloque. En la Figura 2 se muestra la descripción funcional de los componentes de la unidad de datos propuesta.

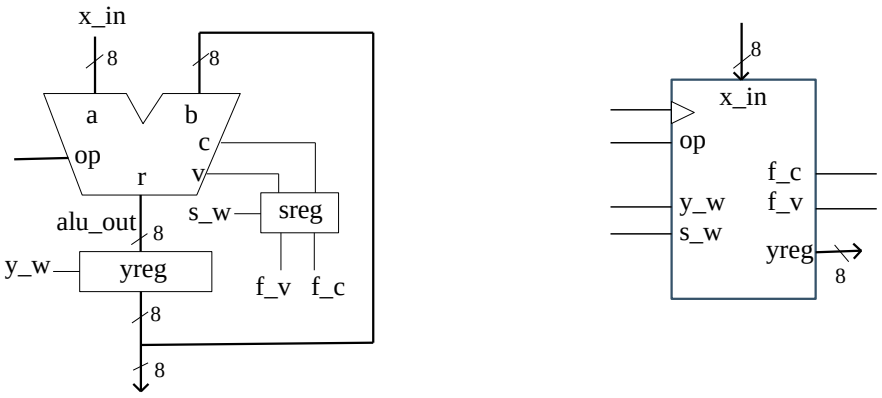

Figura 1: Unidad de datos del sistema digital.

ALU		yreg		sreg	
op	r	y_w	yreg ←	s_w	sreg ←
0 0	a + b	0	yreg	0	sreg
0 1	b - a	1	in	1	{v,c}
1 0	a				
1 1	-b				

Figura 2: Descripción funcional de los componentes de la unidad de datos.

En la Figura 3 se muestra la carta ASM de la unidad de control y su diagrama de bloque.

Se puede observar que tiene dos estados. En el estado READY se espera la activación de alguna de las señales de selección de operación. Posteriormente estas señales se asociarán con los los 4 pulsadores que tiene la placa BASYS2 sobre la que se realizará la implementación. Se debe

considerar que la pulsación de un botón durará muchos ciclos de reloj interno de la placa ya que éste es de 50 Mhz, es por ello que se ha tenido que poner el estado WAIT que es un estado donde se espera que se suelte el botón pulsado para luego volver a comenzar la operación. Si no se pusiera el estado WAIT, se estaría continuamente modificando los registros, lo cual no es correcto.

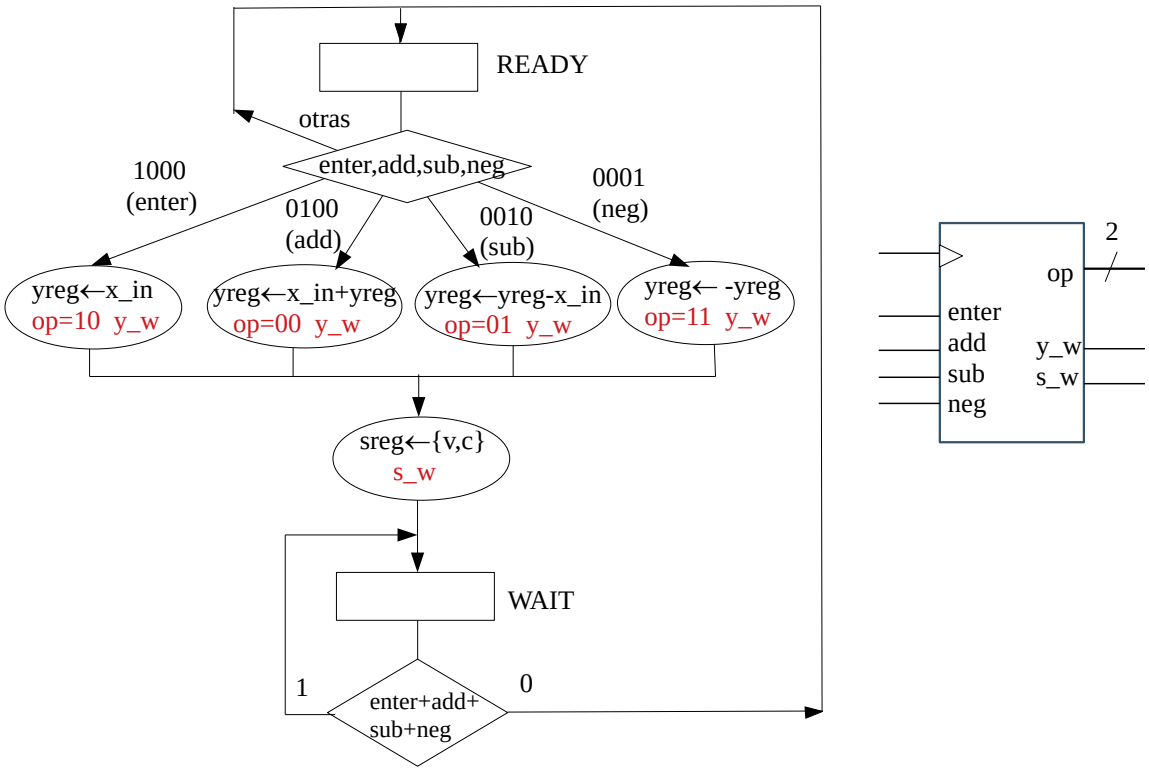

Figura 3: Carta ASM y diagrama de bloque de la unidad de control.

Como se ha comentado anteriormente, las señales *enter, add, sub* y *neg,* se asociarán con los los 4 pulsadores que tiene la placa BASYS2 sobre la que se realizará la implementación. En cuanto al dato de entrada (x_in) se introducirá en el sistema mediante los 8 interruptores (*switches*) que tiene la placa. Por último, es necesario que el usuario tenga información del resultado de las operaciones que realiza la calculadora. Para ello se utilizarán los visualizadores 7 segmentos que posee la placa BASYS2. Todo esto habrá de configurarse adecuadamente en el archivo *system.ucf* que se proporciona.

En la Figura 4 se muestra la unión entre unidad de datos y unidad de control que da lugar al módulo calculator.

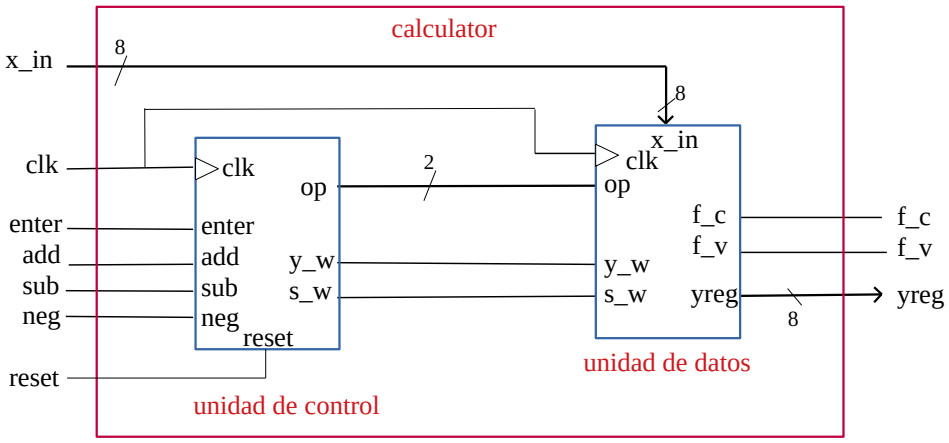

Figura 4: Calculator: interconexión entre las unidades de datos y de control.

En la Figura 5 se muestra un esquema del sistema completo donde se establece la interconexión con los elementos de la placa mencionados. Se puede observar que existen 4 visualizadores pero solo una entrada de 7 bits *seg* por lo que es necesario ir transmitiendo la información a visualizar en cada display de forma periódica mediante una frecuencia de refresco que no sea perceptible al ojo humano. Se incluye un módulo de control del display ya diseñado para tal efecto (*display_ctrl.v*). No es necesario modificar este módulo sino solo usarlo en la última fase del diseño antes de la implementación en la placa.

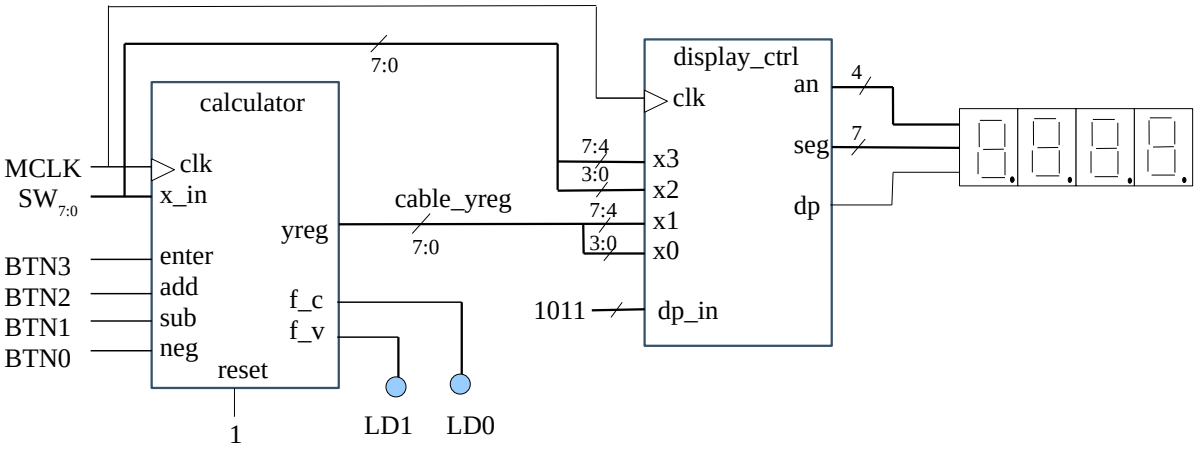

Figura 5: Esquema del sistema.

4 Trabajo de laboratorio

Antes de realizar este apartado debe haber leído y comprendido el apartado 3 donde se dan las especificaciones del sistema que se va a implementar.

Debe crear un proyecto como habitualmente utilizando el entorno Xilinx ISE y añadir los archivos

que se le proporcionan cuando se vaya indicando.

El diseño del sistema se va a abordar siguiendo una serie de etapas o fases. En cada una se incluye un módulo hardware (que habrá que completar) y un *testbench* que permitirá chequear su funcionamiento mediante simulación. Fase a fase iremos consiguiendo el sistema completo.

Fase 1

Se va a trabajar con los siguientes archivos del lab kit:

- stage1/
  - alu.v
  - alu_tb.v

Complete el diseño de la ALU y verifique su comportamiento utilizando el fichero de *testbench*. En la Tabla 2 (también en la Figura 2) se describen sus operaciones. La ALU tiene además dos salidas de estado, *carry* (c) y *overflow* (v).

Tabla 2: Descripción funcional de la ALU.

op[1:0]	**alu_out[7:0]=**
00	a + b
01	b – a
10	a
11	– b

Fase 2

Se va a trabajar con los siguientes archivos del lab kit:

- stage2/
  - data_unit.v
  - data_unit_tb.v

Implemente la unidad de datos de la Figura 1 con una instancia de la ALU (los registros ya están incluidos). La salida de datos de la ALU se almacena en un registro llamado ***yreg.*** Las salidas de estado de la ALU se almacenan en un registro llamado ***sreg.*** Verifique su comportamiento utilizando el fichero de testbench.

Fase 3

Se va a trabajar con los siguientes archivos del lab kit:

- stage3/
  - control_unit.v
  - control_unit_tb.v

Implemente la unidad de control a partir de su carta ASM (Figura 3). Verifique su comportamiento utilizando el fichero de *testbench*.

Fase 4

Se va a trabajar con los siguientes archivos del lab kit:

- stage4/
  - calculator.v
  - calculator_tb.v

Implemente la calculadora mediante la interconexión de la unidad de datos y la unidad de control (Figura 4). Verifique su comportamiento utilizando el fichero de *testbench*.

Fase 5

Se va a trabajar con los siguientes archivos del lab kit:

- stage5/
  - system.v
  - display_ctrl.v (no hay que modificarlo)
  - Basys2/system.ucf

Edite el archivo system.v para implementar el sistema completo conectando la calculadora con el controlador del display 7 segmentos como se muestra en la Figura 5.

Edite el archivo system.ucf para conectar el sistema completo a los dispositivos de entrada/salida de la placa de desarrollo, es decir, las conexiones a los switches, los botones, los leds, y los visualizadores 7 segmentos. En la figura Figura 5 se muestra esa información.

5 Implementación en BASYS 2

- Implemente el diseño básico en la placa de desarrollo Basys2.
- Realice el siguiente ejemplo de uso (((3 + 4) – 5) + 8)
- La calculadora puede ser optimizada añadiendo operaciones nuevas. El alumno deberá realizar estas mejoras durante la sesión de laboratorio según indique el profesor.