*PROYECTO HARDWARE*

******

**Realizado por:**

**Adrián Ortega Escalona**

*adrian.ortega@edu.upct.es*

**Javier Verdú Sánchez**

*javier.verdu2@edu.upct.es*

*ÍNDICE*

## INTRODUCCIÓN

Para la realización de esta práctica hemos partido del proyecto que se completó en las sesiones de prácticas de la asignatura, de tal forma que el nombre de la carpeta del proyecto es “P2c\_Picoblaze\_Helloworld\_RAM\_INT\_FLIP”.

En esta memoria vamos a explicar las modificaciones realizadas sobre el proyecto anterior. Este proyecto se compone de:

* Un nuevo Periférico llamado “**CRC**” conectado a PicoBlaze.
* Un nuevo Periférico llamado “**Sumador**” conectado al PicoBlaze.
* Modificación de la microarquitectura del PicoBlaze (introducción de una nueva instrucción, llamada “**CRC**”).

## Toplevel

Diagrama, Esquemático

Descripción generada automáticamente

Para la realización de esta parte Una vez hechos y comprobados los periféricos los hemos añadido como componentes a la arquitectura del Picoblaze además de añadir 3 memorias RAM para facilitar el proceso de Lectura de código ASCII para facilitar el código ensamblador y poder hacerlo mas eficiente   
Texto

Descripción generada automáticamente con confianza mediaTabla

Descripción generada automáticamente con confianza mediaTabla

Descripción generada automáticamenteTexto

Descripción generada automáticamenteInterfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamentePeriféricos:

### SUMADOR

Para comenzar, veremos que nuestro periférico sumador, va a ser de 8 bits que al introducir dos caracteres devuelve la suma de estos. Iría conectado a las siguientes salidas de PicoBlaze:

 - Out\_port [7:0]

- Port\_Id [7:0]

 - Write\_strobe

- data\_out

Primero de todo necesitamos hacer el vhdl que necesitaremos para la suma, tendremos que ponerle un port-id para coger el dato de cada valor. Los datos que vamos a importar del componente son el  “**0x44**” que será el primer valor, el segundo valor se leerá de la posición “**0x45**”. La suma de estos dos, saldra por data\_out que esta as.

Luego de hacer este componente, hay que instanciarlo y declararlo en el toplevel y por último, tendremos que añadir esa salida del periférico al multiplexor, sal\_sumador con su respectiva posición que sera la “**0x47**” .

Una vez terminado el vhdl, tendremos que implementar el asm. En el cual tendremos que hacer para que se ingresen dos números en dos registros y que te imprima la salida.

Llamamos a “recibe”, recibiendo el carácter que se le pasa al PC, después utilizamos “OUTPUT” para escribir en el periférico, sacando por el puerto la dirección de memoria “44” (R1) y por la salida de datos “rxreg” y poniendo “write\_strobe” a 1 en el segundo ciclo de reloj. Volvemos a llamar a recibe y repetimos el proceso, esta vez sacando la dirección de memoria “45” (R2). Ahora con un “INPUT” y por el “port\_id” de “43”, luego se meterá en txreg

Una vez con el asm realizado, para comprobarlo, tendremos que pasarlo a código máquina con nuestro archivo asm y el cpp. Una vez hecho esto tendremos el archivo TESTSUMA.vhd. Una vez hecho todo lo anterior debemos sintetizar, implementar y generar el archivo de programa de nuestro diseño. Simplemente ahora comprobaremos con el Hiperterminal el funcionamiento de nuestro periférico.

Texto

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

## CRC

Hemos creado un CRC de 8 bits que funciona de forma paralela***(x^8 + x^2 + x + 1),*** el cual se le introduce un valor y a ese valor se le calcula su CRC, el cual se trata de un circuito que implementa registros de desplazamiento con realimentación, esta será a través de puertas XOR. En este caso, estamos ante un código de redundancia ciclica en paralelo, lo cual nos permite realizar la operación en un ciclo de reloj. Se le introducirá un valor y el cual pasará por la secuencia de combinación pseudoaleatoria, y obtendremos el resultado.ría conectado a las siguientes salidas de PicoBlaze:

- Out\_port [7:0]

- Port\_Id [7:0]

- Write\_strobe

Las entradas del CRC Serian   
 -Data: Este esta conectado al outport por donde le entra la información

desde el picoblaze.

-crcIn: esto son puertos intermedios que conectan los biestables del

CRC

Las Salidas del CRC es:

-crcOut: esta apunta a Sal\_crc que esta conectado al multiplexor en la “43”

Texto

Descripción generada automáticamente

Interfaz de usuario gráfica, Tabla

Descripción generada automáticamente  
  
  
Se ha realizado un banco de pruebas para la comprobación de su correcto funcionamiento y se ha verificado que cumple su objetivo

## Instrucción CRC

Para la incorporación de una nueva instrucción, hemos ido fijándonos poco a poco, en cómo tenemos que ir cambiando los datos para la suma de esta instrucción. Es una instrucción de dos operandos.

El funcionamiento de esta instrucción es sencilla, simplemente hemos querido que haga la XOR de dos registros.

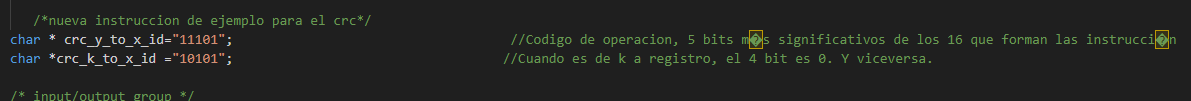
Para la incorporación de una nueva instrucción, hemos ido fijándonos poco a poco, en cómo tenemos que ir cambiando los datos para añadir esta instrucción. Desde la modificación del cpp. 

Imagen que contiene Interfaz de usuario gráfica

Descripción generada automáticamente  
Pantalla de computadora con letras

Descripción generada automáticamente con confianza media

Donde tenemos que aumentar este número máximo de instrucciones, incluimos el código de operación (dos que no estén en uso), añadir el nombre de la nueva instrucción (llamada “CRC”), ver a que grupo de instrucciones pertenece por sus datos. Es una instrucción de dos operandos.

Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

Una vez hecha todas estas modificaciones en el programa cpp, pasamos a hacer las modificaciones en el hardware.



Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

Imagen que contiene Interfaz de usuario gráfica

Descripción generada automáticamenteTexto, Carta

Descripción generada automáticamenteTexto

Descripción generada automáticamente

En la cual tendremos que hacer unas cuantas modificaciones en el procesador. Como algunas de, declarar el componente de la instrucción para que sea de dos operandos como nosotros queremos, declarar sus flags, instanciar el componente de la instrucción, etc.

Texto

Descripción generada automáticamente

Esta instrucción, diseñada para realizar la operación de exclusión lógica (XOR) entre dos registros, desempeña un papel crucial en nuestro trabajo final, por ello hemos implementado esto para a la hora del calculo del CRC no sea siempre los mismos valores.