Práctica 4: Sistema empotrado basado en codiseño para el procesamiento de imágenes
Tecnología de Microcontroladores
Curso 2004-2005

1.-Introducción

A lo largo de los últimos años han cobrado especial interés las técnicas basadas en codiseño en el entorno de los sistemas empotrados. Mediante estas técnicas es posible dividir las tareas a realizar (particionado software/hardware) de manera que parte de ellas son implementadas mediante un hardware de propósito específico y otras mediante la programación de un microprocesador de propósito general. De esta manera, las tareas más complejas y críticas se implementan en hardware mientras que aquellas que se adaptan mejor a la ejecución de un microprocesador y en las que se requiere mayor flexibilidad se programan. [1]

La existencia de tarjetas de prototipo basadas en FPGA (Field Programmable Gate Array) de bajo coste así como la disponibilidad de herramientas de desarrollo y de especificaciones en VHDL de numerosos recursos hardware (microprocesadores, controladores de E/S, etc.) permiten la realización de prácticas de laboratorio en las que el alumno puede desarrollar sistemas complejos sin demasiada dificultad.

2.- La aplicación a diseñar

Los alumnos deben desarrollar un pequeño coprocesador de imágenes capaz de rotar una imagen recibida por el puerto serie y enviarla de nuevo al PC. Para ello, disponen de la especificación VHDL de un 8051, de una UART y de un CORE (diseño previamente sintetizado) de un módulo CORDIC (COrdinate Rotation DIgital Computer). La tarea principal del alumno es desarrollar un programa C que previamente compilado será ejecutado por el 8051 empotrado. Para ello deberá actuar con la UART y el CORDIC. El origen de rotación será el centro de la imagen. Se recuerda que la matriz de rotación es:

$$R = \begin{pmatrix} \cos\phi & \sin\phi \\ -\sin\phi & \cos\phi \end{pmatrix}$$

2.1- El hardware

2.1.1- La tarjeta de prototipo Digilent Spartan-3 Starter Board

La tarjeta de prototipo [2] está basada en una FPGA Spartan-3 200-4ft256 [3]. Dispone además de 512Kb de SRAM, de un puerto serie, de un conector VGA, de un display 7-segmentos, de 7 LEDs, 4 botones y 8 switches, además de conectores de expansión. La placa dispone de un reloj de 50Mhz y de un zócalo para colocar uno externo. La programación de la FPGA se realiza mediante JTAG con conexión al puerto paralelo del PC. A continuación se muestran las señales de la placa que necesita conocer el alumno para realizar la práctica:

Nombre	Descripción	Tipo
AN1	Habilitación del digito más significativo del display	1bit de entrada
AN0	Habilitación del digito menos significativo del display	1bit de entrada
A B C D E F G	Display A ****** F * * B * G * ****** E * * C * * ****** D	1bit de entrada
BTN3	Botón de RESET	1bit de salida
BTN2	Botón disponible	1bit de salida
BTN1	Botón disponible	1bit de salida
BTN0	Botón disponible	1bit de salida
LD7	LED disponible	1bit de entrada

2.1.2.- Modelo VHDL de la UART

La especificación VHDL de la UART es open source [4]. Incluye 4 archivos. La entidad superior es llama mini_uart. Incluye las señales del bus paralelo de 8 bits y las de la interfaz serie:

Nombre	Descripción	Tipo
Bus paralelo		
WB_CLK_I	Reloj	1 bit de entrada
WB_RST_I	Reset	1 bit de entrada
WB_ADR_I	Dirección	2 bits de entrada
WB_DAT_I	Bus de datos de entrada	8 bits de entrada
WB_DAT_O	Bus de datos de salida	8 bits de salida
WB_WE_I	Escritura/Lectura	1 bit de entrada
WB_STB_I	Inicio de transferencia	1 bit de entrada
WB_ACK_O	Fin de transferencia	1 bit de salida
Interfaz serie		
IntTx_O	Interrupción Buffer de Transimisión Libre (no usada)	1 bit de salida
IntRx_O	Interrupción Dato recibido (no usada)	1 bit de salida
BR_Clk_I	Reloj	1 bit de entrada
TxD_PAD_O	Línea de transmisión RS232	1 bit de salida
RxD_PAD_I	Línea de recepción RS232	1 bit de entrada

El mapa registros de la UART es el siguiente:

Nombre	Dirección	Acceso
Buffer de recepción	0	Lectura
Buffer de transmisión	0	Escritura
Estado	1	Lectura
Reservado	2	
Reservado	3	

La UART transmite y recibe datos con la siguiente configuración: 9600 baudios, 8 bits de datos, sin bit de paridad y 1 bit de stop.

2.1.3.- Core del CORDIC

Este core diseñado por Xilinx permite realizar rotaciones, calcular senos y cosenos, arco tangentes, etc. Puede configurarse de diferentes maneras proporcionando diferentes funcionalidades y formatos de datos mediante la herramienta *CORE Generator*. El diseño que se proporciona está configurado para realizar rotaciones: requiere las coordenadas *x* e *y* y el ángulo de giro como entrada y genera las nuevas coordenadas *x* e *y* como salida. El ancho es de 8 bits para todos los datos y el formato usado para las coordenadas es 2QN (signo y 2 bits para la parte entera) y para el ángulo 1QN (signo y 1 bit para la parte entera). A continuación se muestran algunos ejemplos:

x: 00110000 (0.75)
θ: 11110000 (-0.5 radianes)

Las señales de entrada y salida del CORDIC se muestran a continuación:

Nombre	Descripción	Tipo
x_in	Coordenada *x* de entrada	8 bits de entrada
y_in	Coordenada *y* de entrada	8 bits de entrada
phase_in	Ángulo de rotación	8 bits de entrada
nd	Cuando está a 1 indica nuevo dato de entrada	1 bit de entrada
x_out	Coordenada *x* de salida	8 bits de salida
y_out	Coordenada *y* de salida	8 bits de salida
rdy	Cuando está a 1 indica que el dato de salida está listo	1 bit de salida
clk	Reloj	1 bit de entrada

2.1.4.- Modelo VHDL del microcontrolador 8051

La especificación VHDL del 8051 utilizado fue desarrollada en el marco del proyecto Dalton[4] y se distribuye libremente [5]. Incluye 13 archivos. La entidad superior se denomina: i8051_all. La entidad incluye: el reset, el reloj y 8 puertos (los puertos de entrada y salida utilizan líneas diferentes). Gracias a esta característica se va a poder hacer un uso más eficiente de los puertos. A continuación, se muestra el mapa de registros:

Puertos	Bits (7:0)							
Puerto 0 Entrada	WB_ACK_O	rdy	BTN2	BTN1	BTN0	-	-	-
Puerto 1 Entrada	x_out							
Puerto 2 Entrada	y_out							
Puerto 3 Entrada	WB_DAT_O							
Puerto 0 Salida	WB_WE_I	WB_ADR_I(0)	nd	WB_STB_I	nd	AN1	AN0	MOD0_C
Puerto 1 Salida (MODO_C)=0	A	B	C	D	E	F	G	-
Puerto 1 Salida (MODO_C)=1	x_in							
Puerto 2 Salida (MODO_C)=0	WB_DAT_I							
Puerto 2 Salida (MODO_C)=1	y_in							
Puerto 3 Salida	phase_in							

Para poder acceder a todos los dispositivos los puertos 1 y 2 de salida están multiplexados: cuando MODO_C (modo CORDIC) está a 1 se accede al CORDIC (entradas *x_in* e *y_in*); en caso contrario se accede al display (entradas A…G) y a la UART (entrada WB_DAT_I).

2.1.- Herramientas

2.1.1.- Xilinx ISE Foundation

El entorno Xilinx ISE Foundation [2] permite realizar todas las etapas del flujo de diseño en FPGAs: especificación, simulación, síntesis, implementación y programación.

El alumno debe crear un proyecto que incluya las especificaciones proporcionadas para el 8051(*.vhd), la UART (*.vhd), el CORDIC (.xco), la entidad superior que engloba todo el diseño (system.vhd) y el archivo de restricciones (system.ucf). En el archivo de restricciones se especifica el *pin-out*. También se debe incluir un divisor de reloj (.xaw) porque el 8051 implementado no soporta los 50Mhz del reloj que dispone la placa.

Tras el proceso de síntesis/implementación la herramienta genera un archivo *system.bit* que contiene la información para programar la FPGA. La programación se realiza mediante la herramienta IMPACT. El programa a ejecutar por el 8051 debe ir almacenado en la ROM. Una manera de hacerlo es añadir el contenido de la ROM al archivo de restricciones. Para evitar re-sintetizar el diseño cada vez de se modifique el programa puede utilizarse la herramienta DATA2MEM para actualizar los archivo *.bit*.

2.2.- El compilador SDCC

La herramienta SDCC es un compilador de C de libre distribución para diferentes microcontroladores entre los que se incluye el Intel 8051.

2.2.1.- Definiciones de variables asociadas a los puertos

El 8051 dispone de 4 puertos de E/S multiplexados: P0, P1, P2 y P3 asignados a las direcciones 0x80, 0x90, 0xA0 y 0xB0 respectivamente. Para definir una variable asociada a un bit de un puerto se utiliza la palabra clave *bit* y para el puerto completo *sfr*. Por ejemplo:

```
bit at 0x83 P0_3; // Bit 3 del puerto 0
sfr at 0x90 P1; // Puerto 1
```

2.2.2.- Definiciones de variables en memoria externa

El 8051 puede acceder una memoria RAM externa. Para definir una variable almacenada en la memoria externa se usa la palabra clave *xdata*. Por ejemplo:

```
xdata unsigned char dato;
```

2.4.- Archivo de comandos para automatizar el proceso

Con idea de automatizar el proceso de diseño se ha creado un archivo de comandos llamado *actualiza_bit* que permite compilar el programa C pasado como parámetro (nombre de fichero sin extensión .c) y actualizar el archivo *system.bit*. De esta manera cada vez que se modifique el programa hay que ejecutar dicho comando y después con el IMPACT programar la FPGA.

2.6.- Otros programas

Para enviar y recibir las imágenes por el puerto serie puede usarse la utilidad Hy*per terminal* de Windows. Además el alumno dispone de los siguientes programas:

- crea_imagen_bin: crea un archivo binario para ser enviado a la tarjeta. Recibe como parámetros el ángulo de giro (en grados) y la imagen (nombre del archivo sin extensión *bmp*) en ese orden. En el archivo binario aparece primero el ángulo y a continuación la imagen.
- crea_imagen_bmp: crea un archivo *bmp* del archivo binario recibido por el puerto serie. El programa recibe como parámetros un archivo binario (nombre del archivo si extensión *bin*)

3.- Desarrollo del programa

Para la programación de la aplicación se aconseja al alumno seguir el siguiente esquema:

1. Definir variables asociadas a todos los puertos (puede haber más de una variable asociada al mismo puerto: por ejemplo *BTN2* y *nd* para el bit 5 del puerto 0 o *WB_DATA_I* y *y_in* en el puerto 2)
2. Definir alguna macro para mostrar valores en el display para realizar la depuración
3. Para probar la comunicación serie puede enviar caracteres hacia la tarjeta mostrando su código ASCII (de dos dígitos) en el display y recibir un carácter de una secuencia predefinida (A,B,C, …) cada vez que pulse un botón (BTN0).
4. Para probar el funcionamiento del módulo CORDIC puede fijar unos valores de entrada y mostrar el resultado en el display.
5. Por restricciones de tamaño de la memoria interna del 8051 la imagen debe almacenarse en la memoria externa. Para evitar el uso de dos arrays del tamaño de la imagen hay que ir rotando las coordenadas conforme se van recibiendo los datos de la imagen.

4.- La memoria

El alumno debe entregar una memoria describiendo con detalle todo el proceso seguido hasta conseguir el correcto funcionamiento de la aplicación, incorporando el código C de todos los programas que haya utilizado (tanto el de la aplicación final como el de los programas usados para la depuración). Además debe contestar el siguiente cuestionario:

1. ¿Cuál es la frecuencia de reloj a la que funciona el 8051?
2. Indicar qué pasos deberíamos seguir si quisiéramos cambiar la velocidad a la que se transmiten y reciben datos por el puerto serie.
3. ¿Si quisiéramos añadir al diseño alguno de los conmutadores de los que dispone la placa (por ejemplo el SW7) que deberíamos hacer? ¿Y para otro LED (por ejemplo el LD6)?
4. ¿Por qué el archivo de restricciones incluye directivas de colocación para la memoria ROM?
5. ¿Cuál es el tamaño máximo de código que permite el 8051 implementado y qué valor tiene por defecto? Como puede modificarse y hasta que valor puede alcanzarse en la FPGA utilizada. Hacer los mismos cálculos para la memoria externa.
6. Explica las diferencias entre el código ensamblador generado para el acceso a variables internas y externas.
7. Describir el procedimiento a seguir si en vez de utilizar la matriz de rotación utilizáramos la descomposición (*R*) de Paeth [6] para evitar el efecto de la discretización (píxeles sin rellenar)

$$R = \begin{pmatrix} 1 & \tan(\frac{\phi}{2}) \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ -\sin\phi & 1 \end{pmatrix} \begin{pmatrix} 1 & \tan(\frac{\phi}{2}) \\ 0 & 1 \end{pmatrix}$$

5.- Material disponible para el alumno

- Documentación: Datasheet de la FPGA Spartan-3, descripción de la tarjeta de prototipo, documentación de la UART, datasheet del CORDIC, manual del compilador SDCC y tutorial del ISE Foundation
- Fuentes del todas las especificaciones hardware
- Archivos para programar la FPGA: diseño sin programa en la ROM (system.bit) y diseño con programa para girar la imagen (girar.bit)
- Compilador SDCC (es necesario instalarlo)
- Utilidades: programas *crea_imagen_bin* y *crea_imagen_bmp*, imagen de ejemplo y archivo de comandos *actualiza_bit.bat*

6.- Referencias

[1] Hardware/Software CO-Design: Principles and Practice, Wayne Wolf and Jorgen Staunstrup, Kluwer Academic Publishers
[2] http://www.digilentinc.com/
[3] http://www.xilinx.com
[4] http://www.cs.ucr.edu/~dalton/i8051/i8051syn
[5] www1.mmu.edu.my/~khkoay/8051core
[6] http://www.opencore.org
[7] A.W. Paeth, A fast algorithm for general raster rotation. In Graphics Interface'86, pp.77-81,196