



LABORATORIO DE DESARROLLO HARDWARE - MEMORIA DE ZPUINO



José Manuel Jarana Expósito

Objetivos.....	2
Introducción.....	2
ZPUIno	2
Desarrollo de la práctica	3
1. Nuestro primer objetivo para empezar a manejarnos bien con el diseñador de circuitos, haremos uno que simplemente consiste en crear un inversor y conectarle un led, para que se encienda con una entrada de 0.	3
2. Rediseñando el SoC. Añadiendo periféricos (Display 7-seg).....	5
3. Convirtiendo la placa papilio en un Analizador lógico	6
Conclusiones	7

Objetivos

Los objetivos con estas prácticas serán los mismo que con las prácticas pasadas, aprender esa vez sobre la placa ZPUIno y sobre las placas papilo, al igual de manejar bien el nuevo entorno de desarrollo que tendremos, aunque este no será muy distinto al anterior. Como podemos ver nada más abrir el Design Lab es prácticamente igual que la plataforma de Arduino, o por lo menos en lo que software se refiere, después podemos modificar de la parte nueva de este programa, que es la modificación en Hardware del SoC, diseñando circuitos a nivel de captura de esquemáticos e implementarlos en la FPGA. Además también utilizaremos WINGS que son las placas de expansión de ZPUIno, que se adaptan a los conectores de la placa.

1. Nuestro primer objetivo para empezar a manejanos bien con el diseñador de circuitos, haremos uno que simplemente consiste en crear un inversor y conectarle un led, para que se encienda con una entrada de 0.
2. Rediseñando el SoC. Añadiendo periféricos (Display 7-seg).
3. Convirtiendo la placa papilio en un Analizador lógico.

Introducción

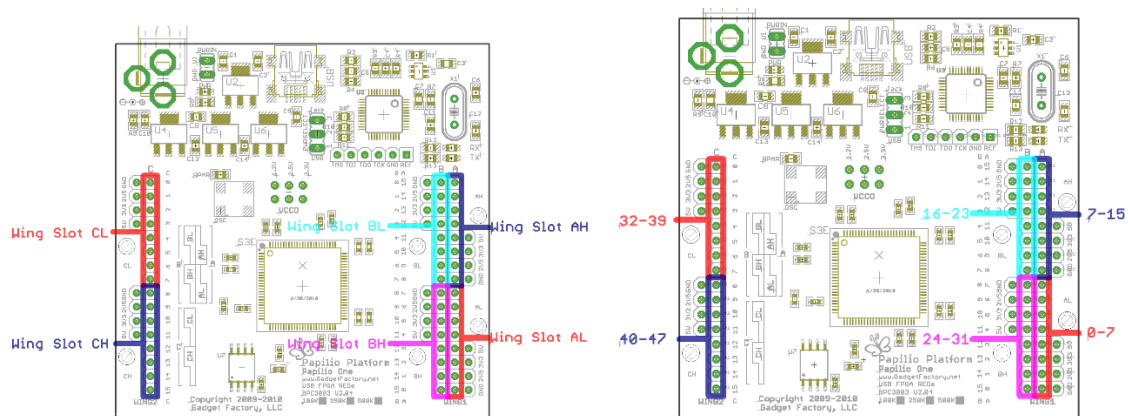
ZPUIno

Es un SoC implementado en VHDL (soft core) y adaptado para el uso de la IDE de Arduino basado en el microprocesador ZPU de Zylins. Es un procesador de 32 bits que trabaja a una frecuencia de 100MHz. Todo es controlado por Sketchs y por el sencillo uso de las bibliotecas de Arduino.

Como entorno de desarrollo utiliza Design Lab, que como ya dijimos antes, al estar basado prácticamente en Arduino, es muy parecido al entorno que utiliza este, pero añadiéndole la parte de diseño de Hardware para este dispositivo, permitiendo cargar este SoC e incluso modificarlo añadiéndole algún periférico nuevo. También incluye nuevas funciones como la captura de esquemáticos para implementarlos en la FPGA.

Nuestra placa de ZPUIno incluirá como placa de desarrollo de FPGA la Papilo one - Espartan 3E 500.

Además, podremos añadirle placas de expansión al conector de papilo. Estas placas se llaman Wings. En las siguientes prácticas por ejemplo nosotros utilizaremos una que se trata de un display de 7-seg, para realizar un código de contador muy parecido al utilizado en Arduino.



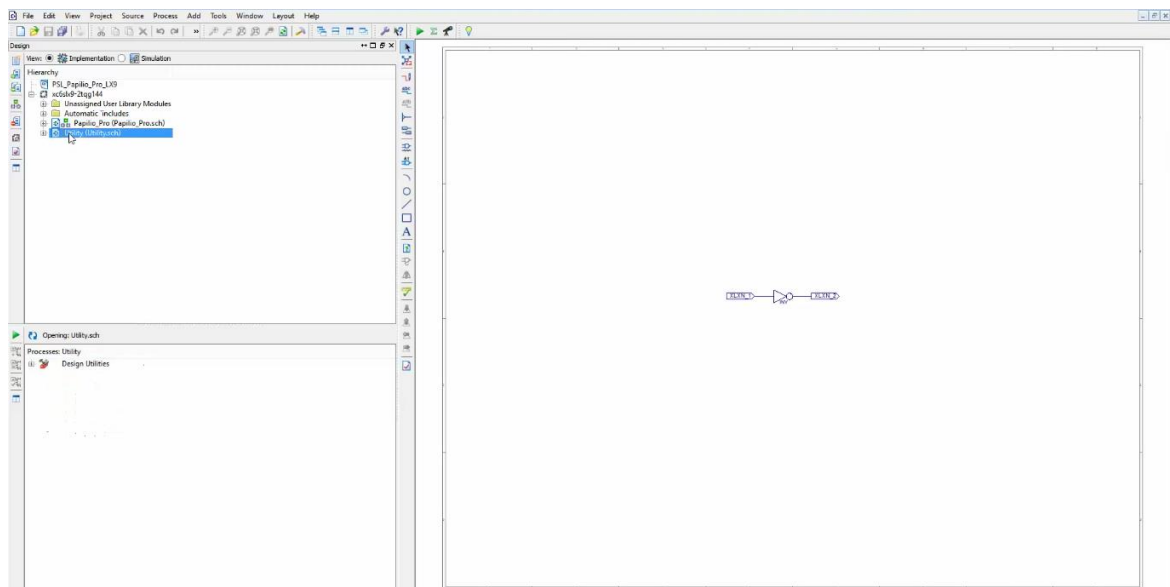
Disposición de los pines ZPUino

Desarrollo de la práctica

Aquí pondremos los diferentes códigos y pautas que hemos realizado para cumplir los objetivos que nos propusimos hacer para la plataforma de Arduino.

1. Nuestro primer objetivo para empezar a manejarnos bien con el diseñador de circuitos, haremos uno que simplemente consiste en crear un inversor y conectarle un led, para que se encienda con una entrada de 0.

En el diseñador de circuitos crearemos el inversor, conectándoles los pines para invertir la señal.



Tras crear el inversor en el diseñador de circuitos, los códigos son los mismo que en la plataforma de Arduino, cambiando simplemente el pin de la placa para ajustarla a ZPUino.

```
int led = 32;

void setup () {
```

```

pinMode(led, OUTPUT); //LED 13 como salida
Serial.begin(9600); //Inicializo el puerto serial a 9600 baudios
}

void loop () {
    if (Serial.available()) { //Si está disponible
        char c = Serial.read(); //Guardamos la lectura en una variable char
        if (c == 'H') { //Si es una 'H', enciendo el LED
            digitalWrite(led, HIGH);
        } else if (c == 'L') { //Si es una 'L', apago el LED
            digitalWrite(led, LOW);
        }
    }
}
}

```

El código del pc también será idéntico, cambiando de nuevo el puerto serie, y poniendo en el que tenemos conectado a nuestra placa.

```

import serial

arduino = serial.Serial('/dev/ttyUSB1', 9600)

print("Starting!")

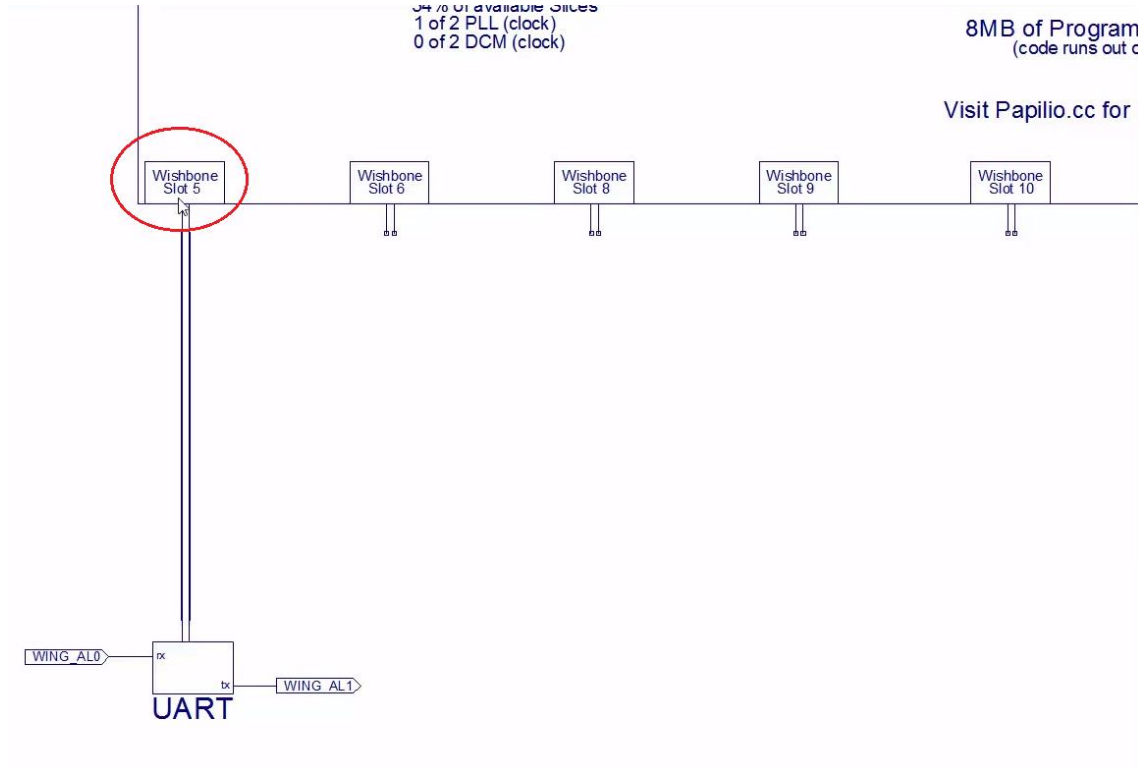
while True:
    comando = raw_input('Introduce un comando: ') #Input
    arduino.write(comando) #Mandar un comando hacia Arduino
    if comando == 'H':
        print('LED ENCENDIDO')
    elif comando == 'L':
        print('LED APAGADO')

arduino.close() #Finalizamos la comunicacion

```

2. Rediseñando el SoC. Añadiendo periféricos (Display 7-seg).

En este punto de la práctica lo que haremos será añadir un periférico al SoC de ZPUino, programar la FPGA y utilizarlo. Ahora añadiremos una UART(Un nuevo puerto serie) a la placa, a los pines WAL0 y WAL1.



Después de subir este diseño a nuestra placa, lo que haremos será conectarlo con el cable de L-RS232-USB los pines que habíamos marcado, pero teniendo en cuenta que el pin que esta en el diseño de circuito marcado como Tx(transmisión) debemos conectarlo con el cable el Rx(recepción), para que sea coherente, y viceversa.

Tras hacer la conexión correctamente, cogeremos exactamente el mismo programa que en el anterior punto para recibir información por el puerto serie (la nueva UART que creamos) y encender un led desde el PC. El código en Python también es bastante similar, pero debemos tener en cuenta que tenemos que cambiar el puerto serie para que sea el adecuado.

```
import serial

arduino = serial.Serial('/dev/ttyUSB2', 9600)

print("Starting!")

while True:
    comando = raw_input('Introduce un comando: ') #Input
```

```

    arduino.write(comando) #Mandar un comando hacia Arduino
    if comando == 'H':
        print('LED ENCENDIDO')
    elif comando == 'L':
        print('LED APAGADO')

arduino.close() #Finalizamos la comunicacion

```

3. *Convirtiendo la placa papilio en un Analizador lógico*

Vamos a comprobar cómo funciona nuestra placa Papilo en el modo de analizador lógico, que es una aplicación bastante útil que tienen este tipo de placas. Para hacer esto abriremos en el Design Lab el icono del analizador lógico y lo configuraremos correctamente para poner el puerto deseado y la frecuencia idónea para poder observar bien lo que nos muestra nuestro analizador. Comprobamos que la configuración está hecha de forma idónea poniendo una señal alta en un ping de la placa y le damos a “Capture”.

Tras comprobar que la configuración es correcta, ahora lo que hacemos es cargar el ejemplo “fade” para que vaya cambiando la intensidad en el led poco a poco. Ejecutamos el analizador lógico y vemos como va cambiando cada cierto tiempo a 1. Aunque lo realmente importante es hacer la modificación en este código de esta manera.

```

int led = 9;           // the pin that the LED is attached to
int brightness = 0;    // how bright the LED is
int fadeAmount = 5;    // how many points to fade the LED by

// the setup routine runs once when you press reset:
void setup() {
    // declare pin 9 to be an output:
    pinMode(led, OUTPUT);
}

// the loop routine runs over and over again forever:
void loop() {

```

```
// set the brightness of pin 9:
analogWrite(led, brightness);

// change the brightness for next time through the loop:
brightness = brightness + fadeAmount;

// reverse the direction of the fading at the ends of the fade:
if (brightness == 0 || brightness == 255) {
  fadeAmount = -fadeAmount ;
}
// wait for 30 milliseconds to see the dimming effect
delay(1);
}
```

En esta modificación cambiamos la última línea del código, y ponemos un delay de 1. Ejecutamos el analizador para capturar de nuevo. Ahora podemos ver, en vez de intervalo entre 1 o 0, vemos como el tiempo que va de 1 a 0 va variando y no es estable hasta que llega a un límite, y vuelve a partir de ahí a bajar el período que está a 0.

Conclusiones

Estás prácticas no han servido un poco de introducción a la hora de las próximas prácticas en las que tendremos que diseñar nuestras PCBs, ya que hemos manejado un poco el diseñador de circuitos para manejar las FPGA de nuestra placa Papilo. Sin embargo nos queda mucho por aprender, ya que lo que hemos dado ha sido sólo una pequeña introducción, aunque ya sabemos cómo funciona el entorno de desarrollo y sabemos cómo modificar algunos de los pines de nuestra placa, para que sea una UART por ejemplo.

También hemos visto otra aplicación que nos permite la placa y el entorno de desarrollo, el uso de la placa como analizador lógico, pudiendo darnos otro uso bastante amplio.