Miguel Ángel Lara Guarino

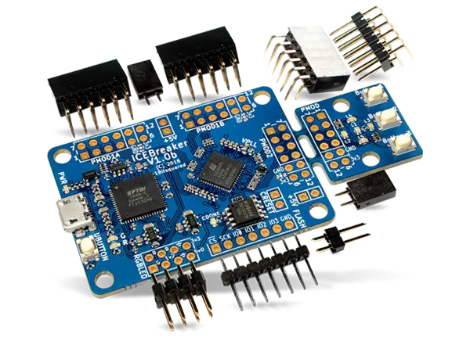
Alfonso Manchado Pérez

**Icono

Descripción generada automáticamente**

**SEPA 2023:**

**Memoria de Prácticas y Proyecto**



SoftCore

En esta memoria, nos adentramos en el emocionante mundo de la FPGA Lattice IceBreaker, una placa de desarrollo poderosa y compacta, y exploramos en detalle su integración con el procesador Neorv32 como SoftCore. Este procesador de 32 bits, diseñado específicamente para su implementación en FPGA, proporciona una sólida base de cómputo y control, ampliando las capacidades de la IceBreaker y ofreciendo una plataforma adaptable para numerosas aplicaciones.

A lo largo de este informe, comentaremos las diferentes prácticas realizadas durante la asignatura y acabando con la explicación de nuestro proyecto final, donde la combinación de la FPGA IceBreaker y el procesador Nerov32 ha demostrado ser excepcionalmente efectiva.

Pero, antes de nada, ¿por qué el uso de un procesador dentro de una FPGA? La inclusión de un procesador en una FPGA (Field-Programmable Gate Array) ofrece una versatilidad única al permitir la personalización del hardware para aplicaciones específicas, combinando la lógica personalizada con el procesamiento de propósito general. Es decir, tenemos por un lado el poder de paralelizar múltiples tareas con la FPGA y por el otro, el desarrollo secuencial por parte del procesador que nos permite tener más control sobre el sistema.

**Índice**

[1. Práctica 1: Toma de contacto y uso de los leds 3](#_Toc153653511)

[**Hardware** 3](#_Toc153653512)

[**Software** 3](#_Toc153653513)

[2. Práctica 2: Uso del Teclado numérico mediante el GPIO 4](#_Toc153653514)

[**Hardware** 4](#_Toc153653515)

[**Software** 4](#_Toc153653516)

[3. Práctica 3: Uso de un periférico Wishbone 4](#_Toc153653517)

[**Hardware** 4](#_Toc153653518)

[**Software** 4](#_Toc153653519)

[4. Proyecto Final 4](#_Toc153653520)

[**Hardware** 4](#_Toc153653521)

[**Software** 4](#_Toc153653522)

# Práctica 1: Toma de contacto y uso de los leds

Para esta primera práctica, entraremos en contacto con el manejo de las herramientas que se nos proporcionan para realizar la síntesis, implementación y generación del bitstream por parte de los códigos VHDL y la compilación para la parte del procesador Neorv32.

Los programas que nos permitirán efectuar estos pasos son:

* Síntesis 🡪 Yosys + Ghdlsynth
* Place & Route 🡪 Nextpnr
* Bitstream 🡪 Icestorm
* Build 🡪 GNU Make

## **Hardware**

Para la parte del hardware, se hará uso de los pines de comunicación GPIO para crear una conexión directa entre los Leds y el soft processor Neorv32. Además, se agregará una señal de reset controlada por el soft processor y, finalmente, se comunicará también mediante el GPIO, el valor del botón pulsado.



Gpio\_o

Gpio\_i

Leds (4 downto 0)



Reset signal

**ICE40UP5k**

**FPGA**

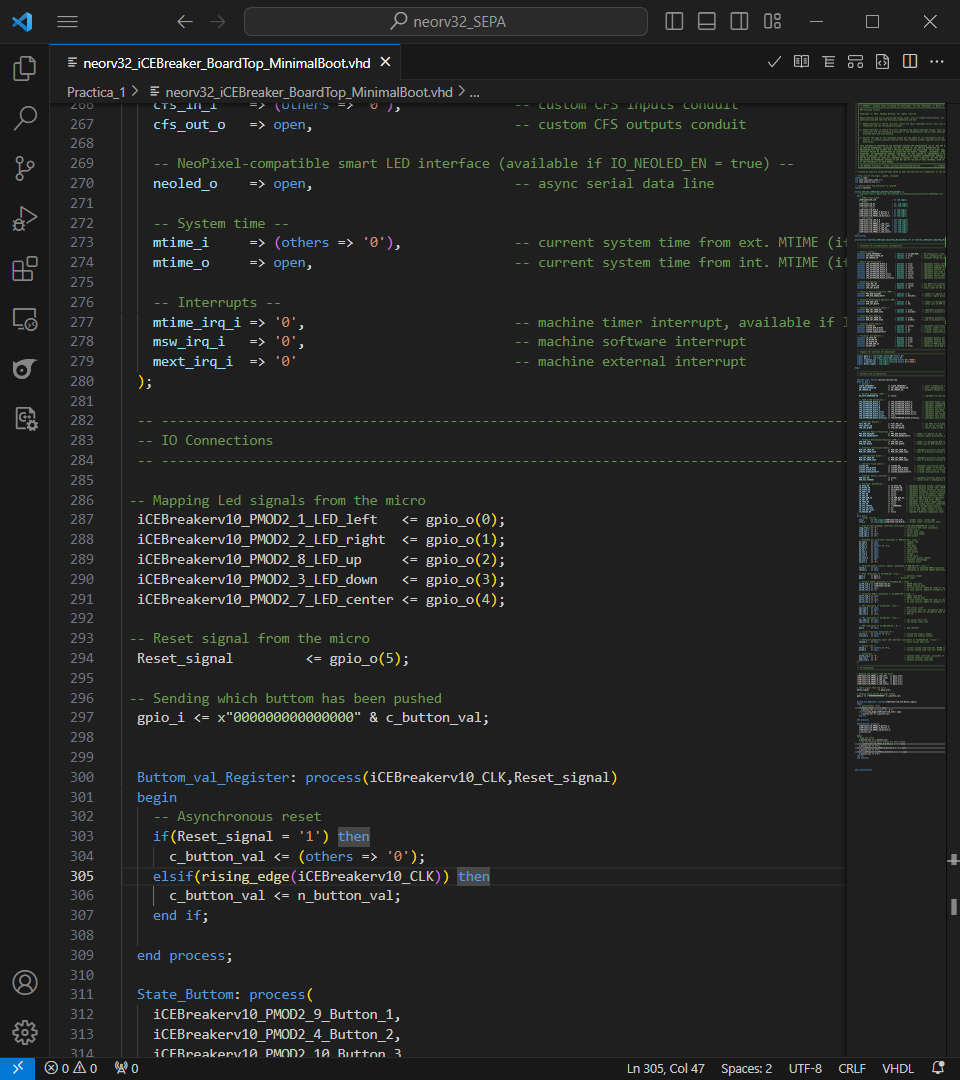
Button (2 downto 0)

1. Texto

   Descripción generada automáticamenteAñadimos las señales necesarias:
2. Asignamos a la señal ‘c\_button\_val’ el valor del botón que haya sido pulsado:

Texto

Descripción generada automáticamente

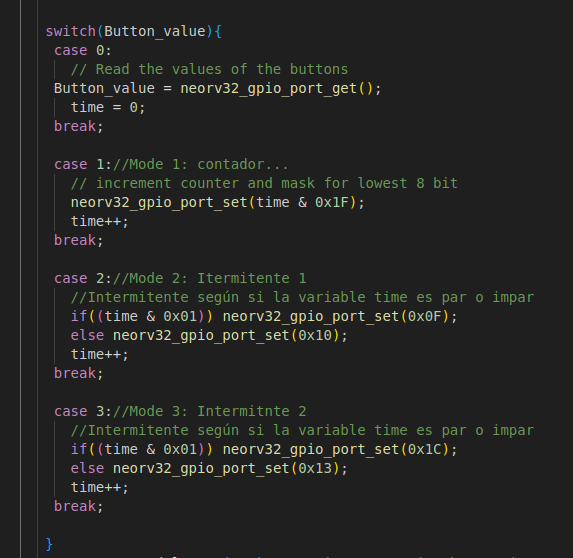
1. Asignamos valor a las salidas:

## **Software**

En la parte software habrá un único cometido, recibir los datos enviados a través de la GPIO y traducirlo en una secuencia de leds.

Para esto hay 3 secuencias de leds a elegir con los 3 botones de nuestra FPGA mediante una máquina de estados, y tras esto, aprovechamos la señal de reset diseñada en hardware para volver al inicio pasado unos segundos.

1. Máquina de estados en función de la pulsación de los botones



1. Timer para resetear la GPIO

Texto

Descripción generada automáticamente

# Práctica 2: Uso del Teclado numérico mediante el GPIO

Una vez adaptados a este nuevo entorno de trabajo, en esta segunda práctica tocaremos el uso de periféricos externos instanciados en nuestro hardware, en concreto, usaremos el teclado de 16 valores PMOD KYPD. Nuestro objetivo personal para esta práctica es conseguir el funcionamiento correcto de este, y aprovecharlo para realizar una calculadora a nivel software.

## **Hardware**

Para la parte del hardware, se hará uso de los pines de comunicación GPIO para crear una conexión directa entre el teclado y el soft processor Neorv32. Además, se instanciará una entidad llamada “peripheral\_teclado.vhd” la cual realizará el proceso de lectura del teclado.

s\_leds [4:0]

Gpio\_o

Gpio\_i



**ICE40UP5k**

**FPGA**

c\_button\_val [2:0]

s\_reset

Key\_value [15:0]



Peripheral\_teclado.vhd

Interfaz de usuario gráfica

Descripción generada automáticamente con confianza bajaPara realizar la lectura del del teclado en hardware, se ha diseñado la siguiente entidad en VHDL:

Texto

Descripción generada automáticamenteCaptura de pantalla de computadora

Descripción generada automáticamentePara poder realizar la lectura sobre el teclado, utilizaremos un contador de 2 bits para realizar la secuencia: ‘1110’, ‘1101’, ‘1011’ y ‘0111’ según el contador sea ‘00’, ‘01’, ‘10’ y ‘11’, respectivamente. Por cada secuencia se realizará una lectura de las señales col\_1\_o, col\_2\_o, col\_3\_o y col\_4\_o para comprobar si existe algún cero, en cuyo caso, invertiremos la señal y la guardaremos en el registro c\_key como codificación One Hot, el cual volcará sus datos sobre el registro c\_key\_value cada 4 ciclos (Tras un barrido de lectura sobre el teclado) y limpia el registro c\_key. Finalmente, el valor leído del teclado es enviado directamente al top, en el cual se envía al Gpio\_i para que el soft processor pueda leer la tecla pulsada.

|  |  |  |  |
| --- | --- | --- | --- |
| Tecla | Codificación | Tecla | Codificación |
| Ninguna | 0x0000 | Ninguna | 0x0000 |
| “1” | 0x0008 | “2” | 0x0800 |
| “4” | 0x0004 | “5” | 0x0400 |
| “7” | 0x0002 | “8” | 0x0200 |
| “0” | 0x0001 | “F” | 0x0100 |
| “A” | 0x0080 | “3” | 0x8000 |
| “B” | 0x0040 | “6” | 0x4000 |
| “C” | 0x0020 | “9” | 0x2000 |
| “D” | 0x0010 | “E” | 0x1000 |

## **Software**

A nivel software se diseña una función para decodificar lo recibido a través de “Key\_value”, y con una máquina de estados, en función de lo pulsado, realizamos las cuentas de la calculadora.

Para esto hemos tratado cada letra del teclado como una operación:

1. 🡪 Ans(guardamos el resultado anterior)
2. 🡪Suma
3. 🡪Resta
4. 🡪Multiplicación
5. 🡪Ac(Reseteo)
6. 🡪Resultado
7. Actualización del valor pulsado.

Texto

Descripción generada automáticamente

1. Máquina de estados para la pulsación de las letras(operaciones)

Texto

Descripción generada automáticamenteTexto

Descripción generada automáticamente

1. Función auxiliar para evitar repeticiones en las pulsaciones.

Texto

Descripción generada automáticamente

# Práctica 3: Uso de un periférico Wishbone

((Alfon explica la intro))

## **Hardware**

Para la parte del hardware, se hará uso de un periférico WishBone. De este modo, creamos una nueva entidad llamada “wb\_peripheral\_teclado.vhd” el permitirá establecer una comunicación segura entre el soft processor y el periférico sobre los registros 0, 1, 2, 3 y 4 instanciados en la dirección 0x90000000. Por tanto, nuestro periférico estará situado entre 0x90000000 y 0x9000001F.

**ICE40UP5k**

**FPGA**



Diagrama

Descripción generada automáticamente

s\_leds [4:0]

Gpio\_o

Gpio\_i

s\_reset

c\_button\_val [2:0]

**wb\_tag\_m2s [2:0]**

**wb\_dat\_s2m [31:0]**

**wb\_adr\_m2s [31:0]**

**wb\_dat\_m2s [31:0]**

**wb\_sel\_m2s [3:0]**

**wb\_err\_s2m**

**Wb\_Peripheral\_teclado.vhd**

**wb\_ack\_s2m**

**wb\_lock\_m2s**

**wb\_cyc\_m2s**

**wb\_stb\_m2s**

**wb\_we\_m2s**

## **Software**

# Proyecto Final

## **Hardware**

## **Software**

# Anotaciones adicionales