

# Procedimiento para implementar un procesador NIOS II en un FPGA Cyclone II

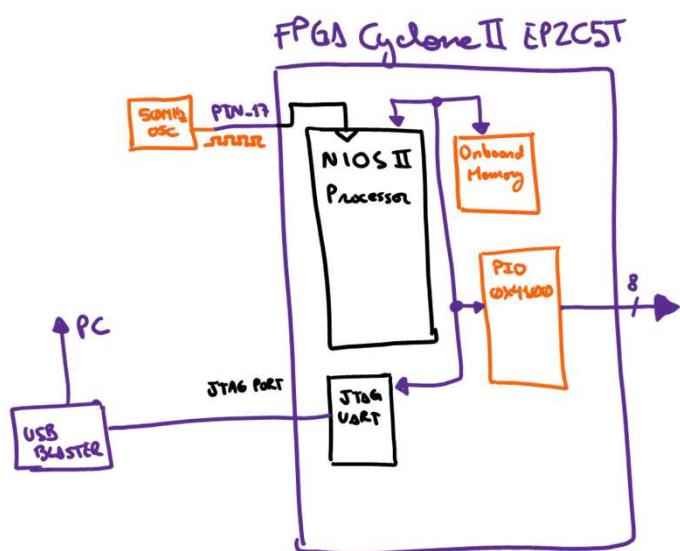
Por: Kalun José Lau Gan

2021

1

## Aspectos iniciales:

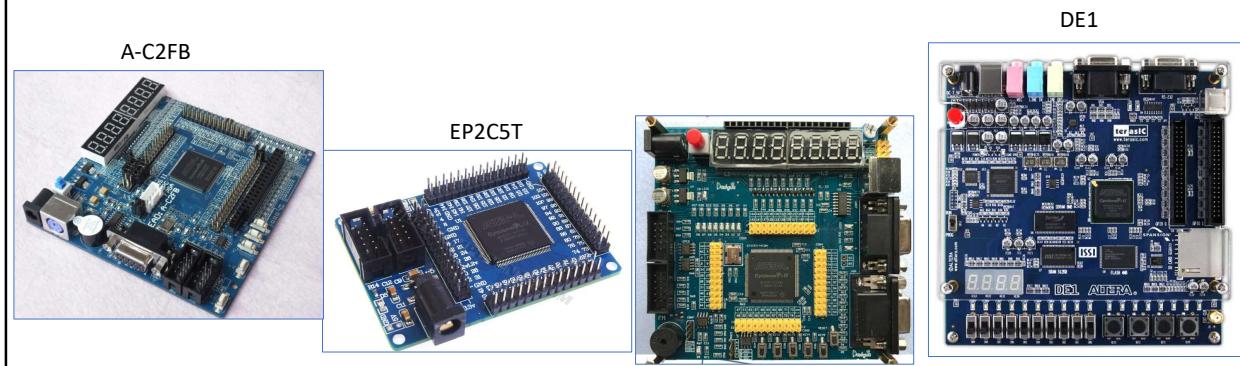
- Tener en cuenta que la implementación del NIOS II en el presente documento será en una FPGA Cyclone II EP2C5T.
- Se empleará el Quartus II versión 13.0 SP1 que es el último que soporta los FPGA Cyclone II.
- Se utilizará recursos del mismo FPGA para implementar una memoria para el NIOS II.
- Como ejemplo se implementará el NIOS II con un puerto PIO con dirección 0x4100, de 8 bits y como salida.
- Se empleará el lenguaje C para la aplicación que correrá el NIOS II



2

## Aspectos iniciales

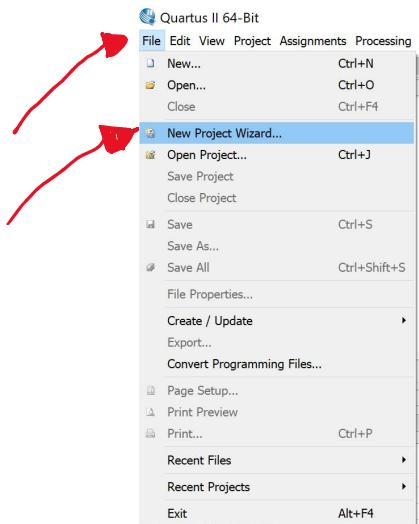
- Existen diversas tarjetas de desarrollo empleando el FPGA Cyclone II EP2C5T.
- Cada tarjeta tendrá (o carecerá de) periféricos externos conectados al FPGA de manera particular por lo que se tendrá que consultar la hoja técnica de cada una de dichas tarjetas para ver sus conexiones.



3

## Procedimiento para implementar el NIOS II

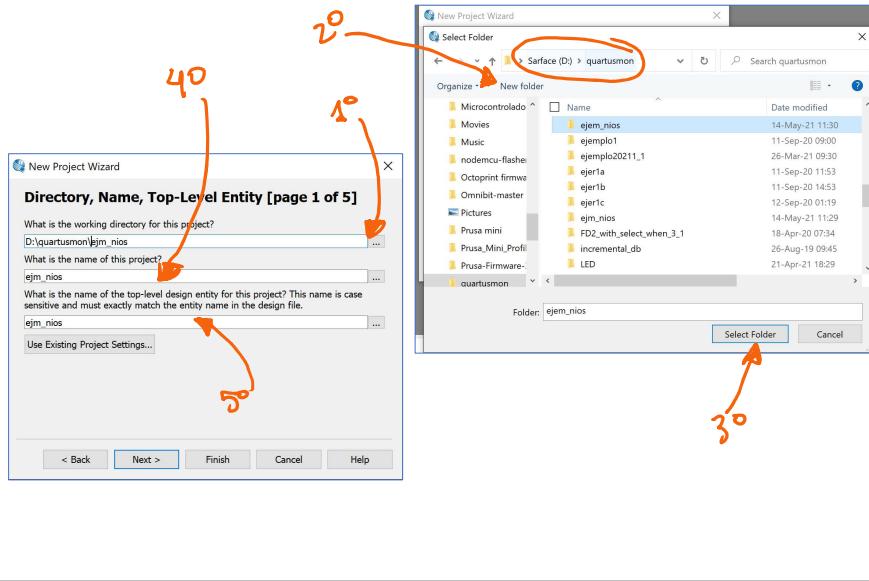
- Usar el Proyect Wizard para crear un proyecto en el Quartus II



4

## Procedimiento para implementar el NIOS II

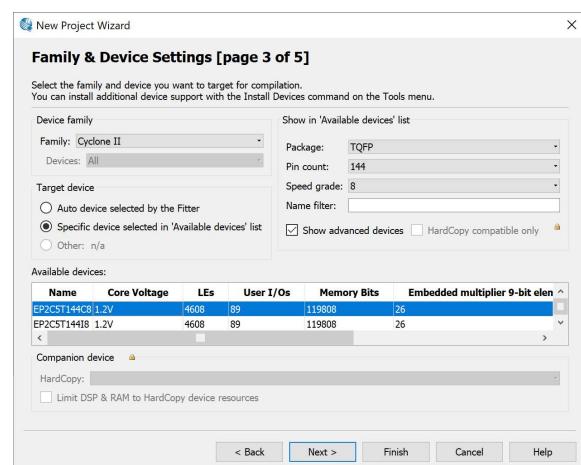
- De preferencia crear una carpeta de manera manual para alojar todo el proyecto.
- Tener en cuenta que la ruta de la carpeta del proyecto sea fácil de encontrar y no debe de haber espacios tanto en la ruta como el nombre del proyecto.
- Otra consideración a tener en cuenta es que el nombre de la carpeta coincide con el nombre del proyecto y también del top-level.



5

## Procedimiento para implementar el NIOS II

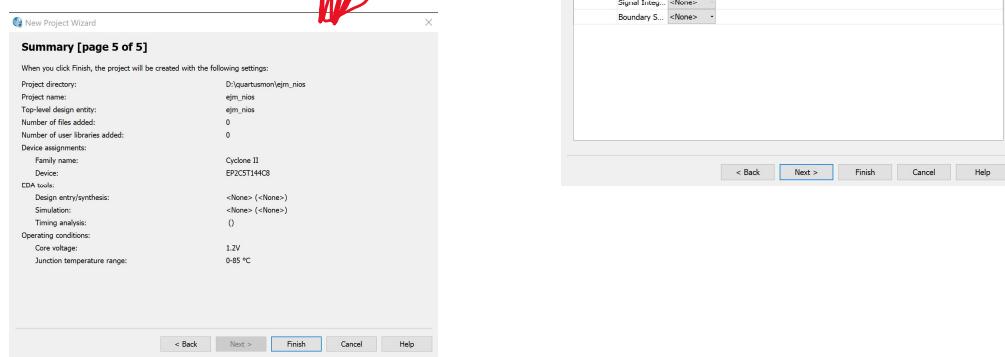
- Escoger el dispositivo FPGA correcto, en este ejemplo usaremos el Cyclone II EP2C5T144C8N



6

## Procedimiento para implementar el NIOS II

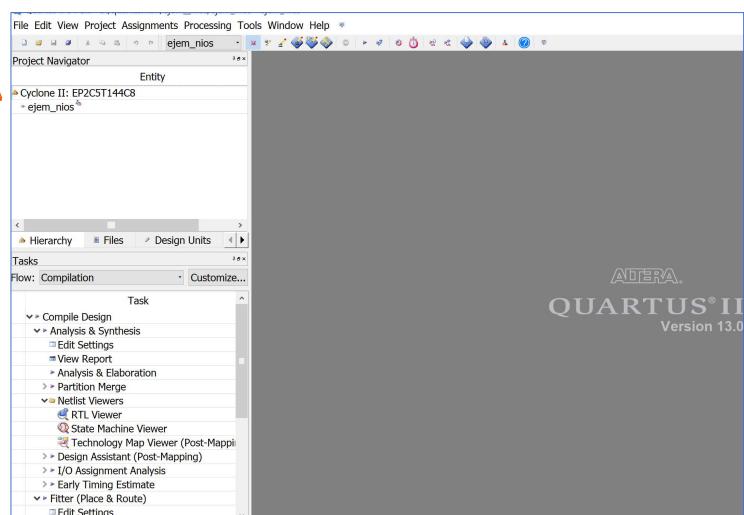
- No escoger nada y darle siguiente.
- Aparecerá el resumen de las configuraciones realizadas



7

## Procedimiento para implementar el NIOS II

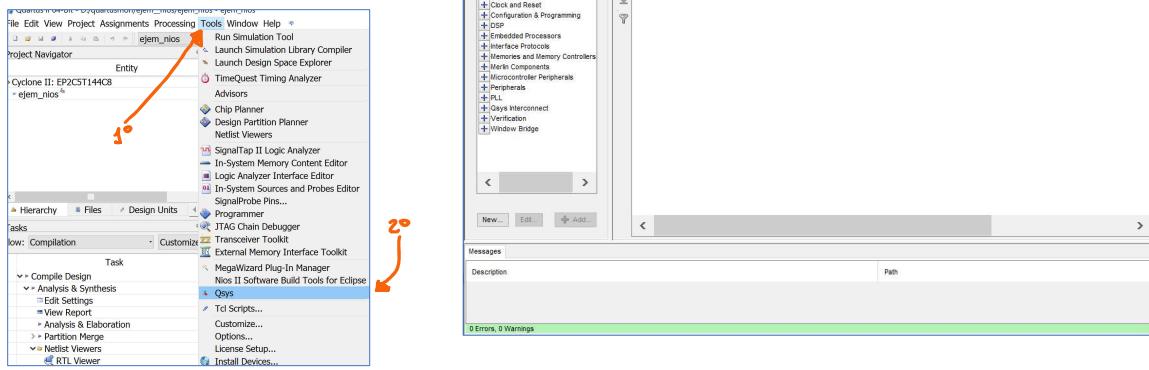
- Al retornar a la ventana principal del Quartus II se deberá verificar en la ventana "Project Navigator" si se escogió el dispositivo correcto y el nombre del top-level



8

# Procedimiento para implementar el NIOS II

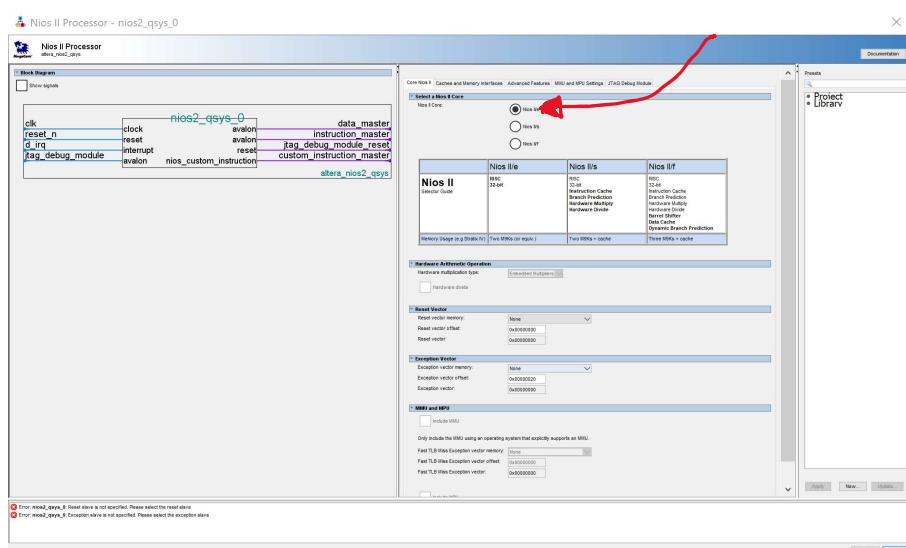
- Abrimos el Qsys para construir la plataforma de procesador NIOS II



9

# Procedimiento para implementar el NIOS II

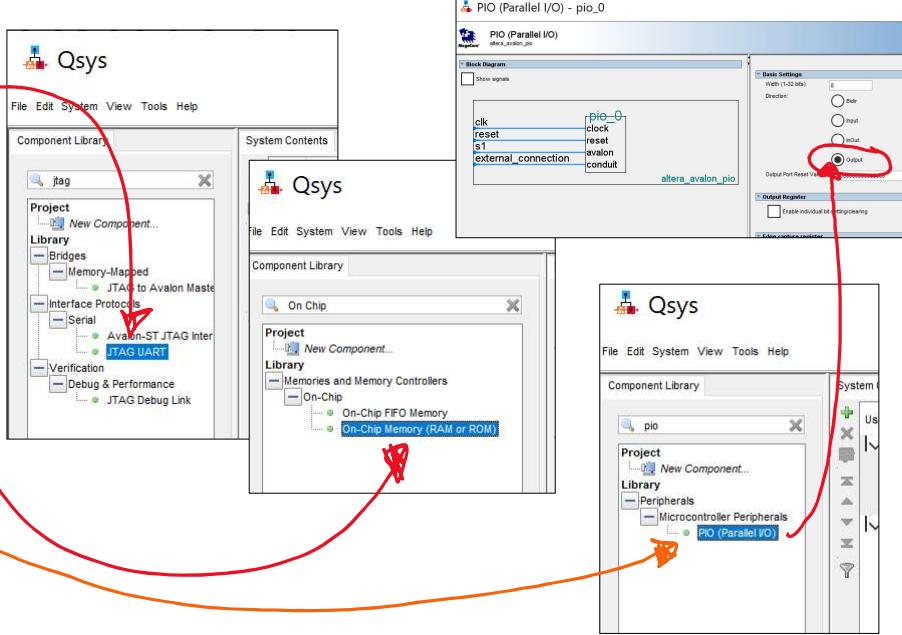
- Agregar el procesador NIOS II desde la ventana de la izquierda.
- En las opciones de la ventana escoger el NIOS II/e



10

## Procedimiento para implementar el NIOS II

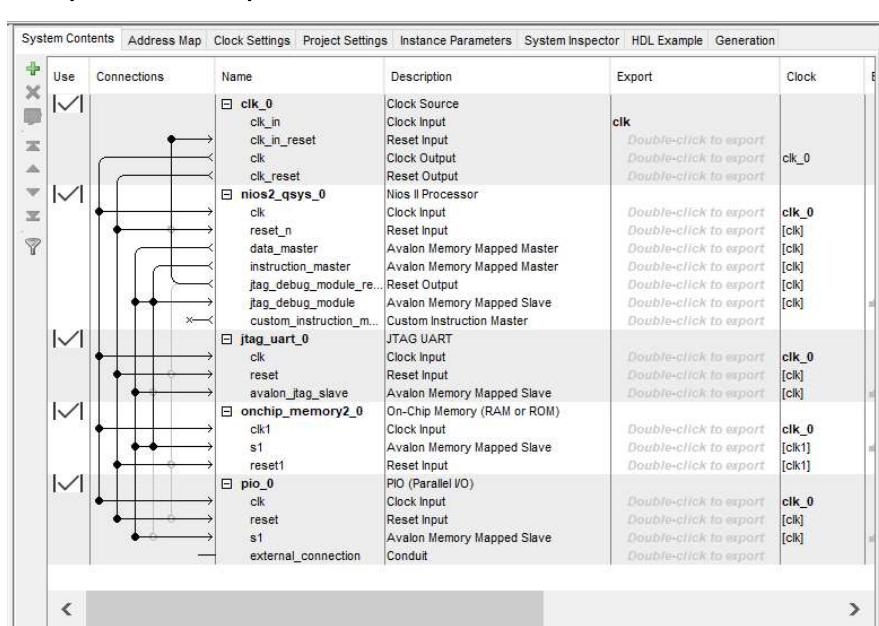
- Agregar el JTAG UART (configuración por defecto)
- Agregar On-Chip Memory (configuración por defecto)
- Agregar PIO, verificar que sea de ancho de 8 bits y colocarlo como salida



11

## Procedimiento para implementar el NIOS II

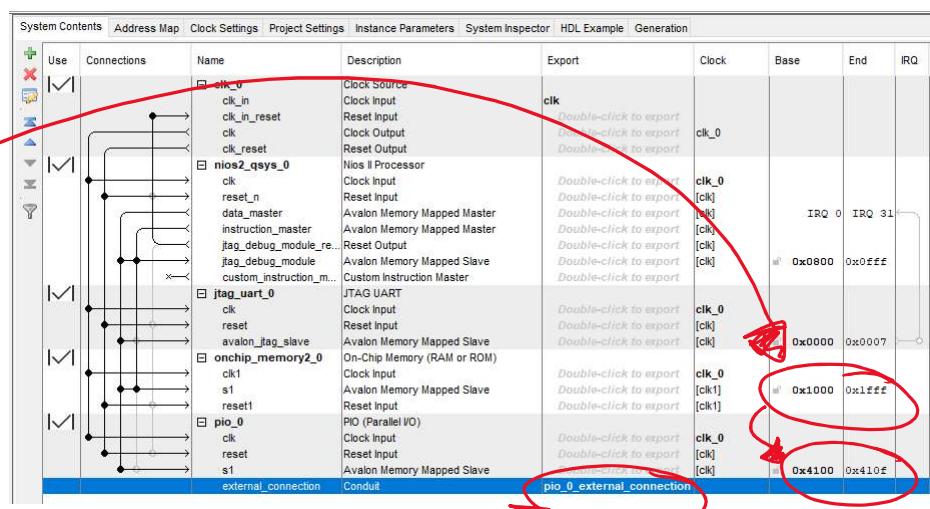
- Hacer las conexiones de los componentes de la plataforma NIOS (reloj principal, reset, buses Avalon: datos e instrucciones)



12

## Procedimiento para implementar el NIOS II

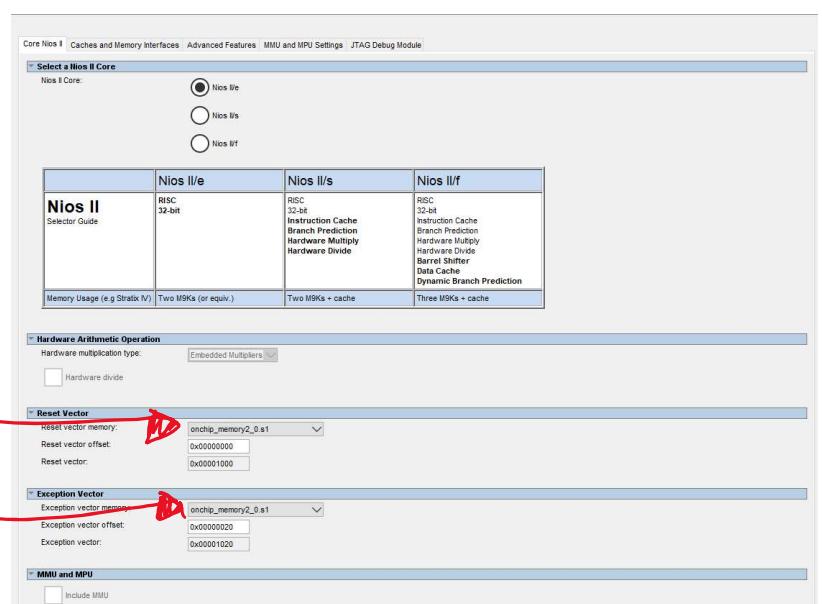
- Hacer el mapeo de direcciones para que no haya conflicto de recursos
- Definir nombre de exportación del puerto de salida PIO



13

## Procedimiento para implementar el NIOS II

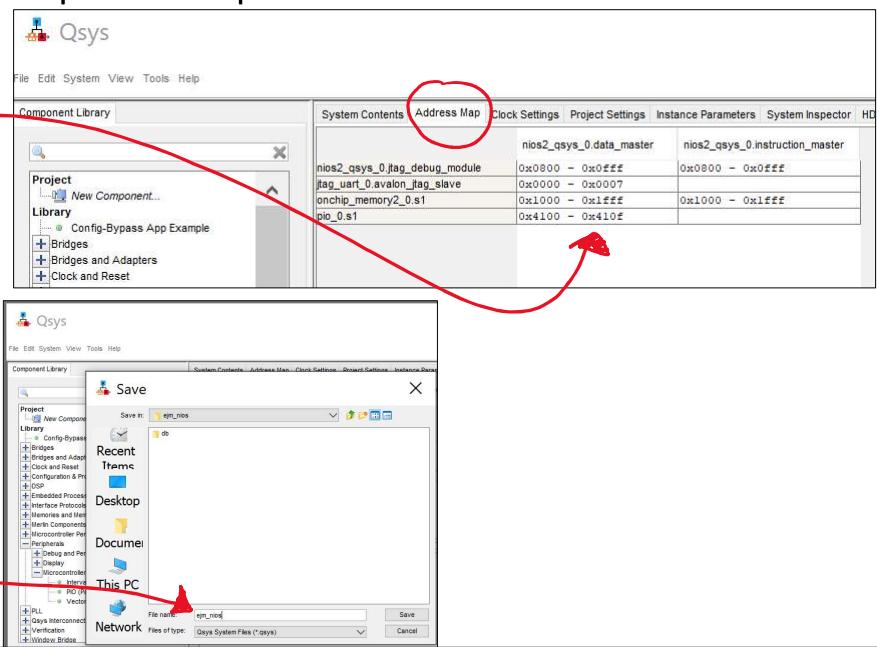
- Seleccionar onchip\_memory tanto en Reset Vector como en Exception Vector



14

## Procedimiento para implementar el NIOS II

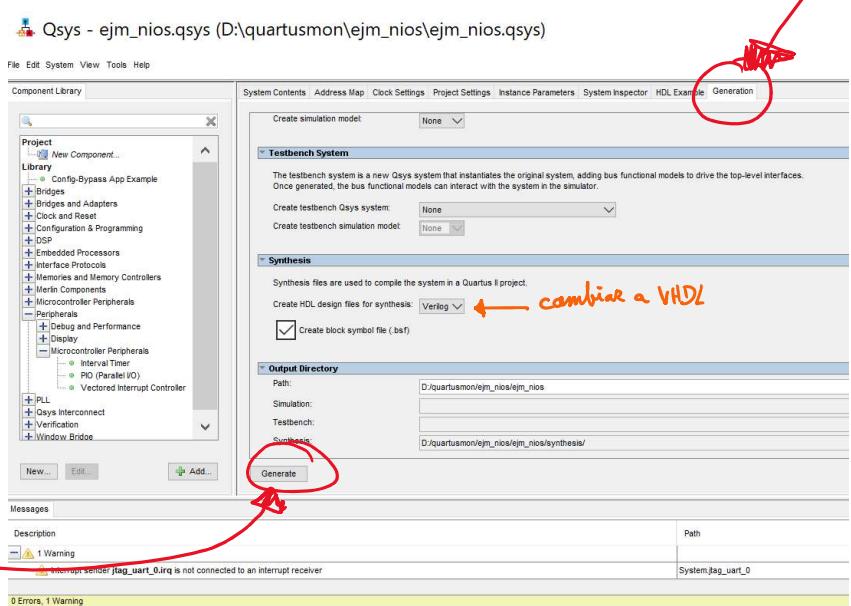
- Revisar mapeo de memoria
- Grabar lo realizado usando el mismo nombre del top-level



15

## Procedimiento para implementar el NIOS II

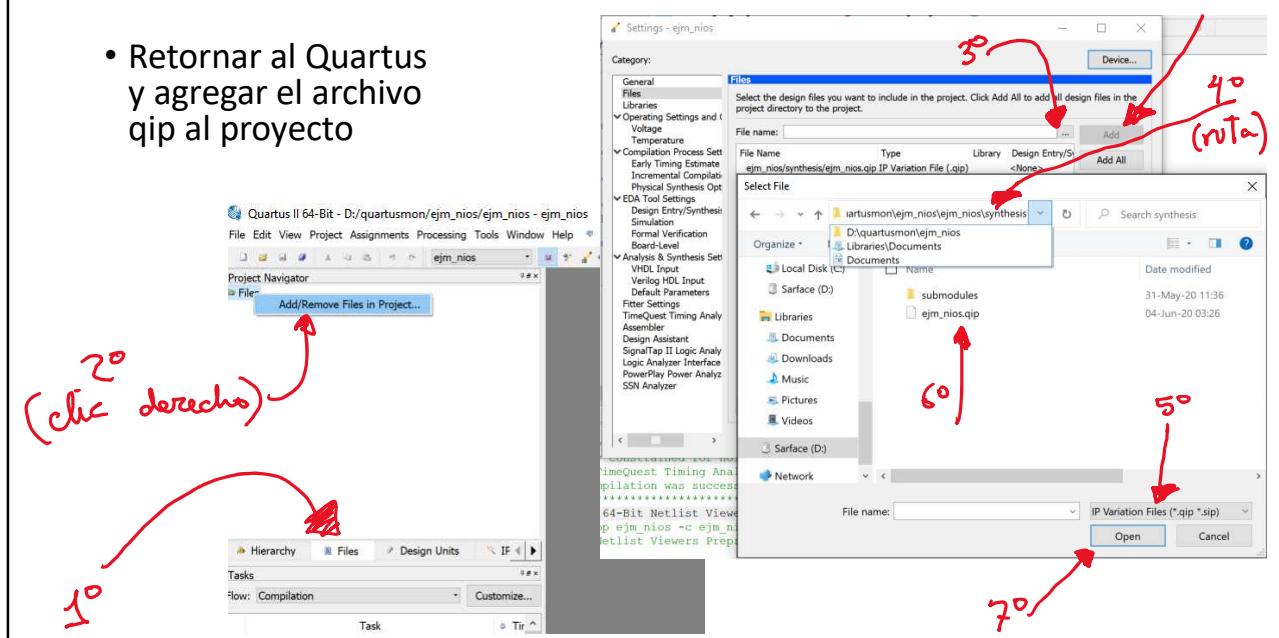
- Dirigirse a la última pestaña (Generation) y pulsar el botón "Generate", esto construirá el sopcinfo necesario para el Eclipse.
- Nótese que no debe de salir error alguno al término de la generación



16

## Procedimiento para implementar el NIOS II

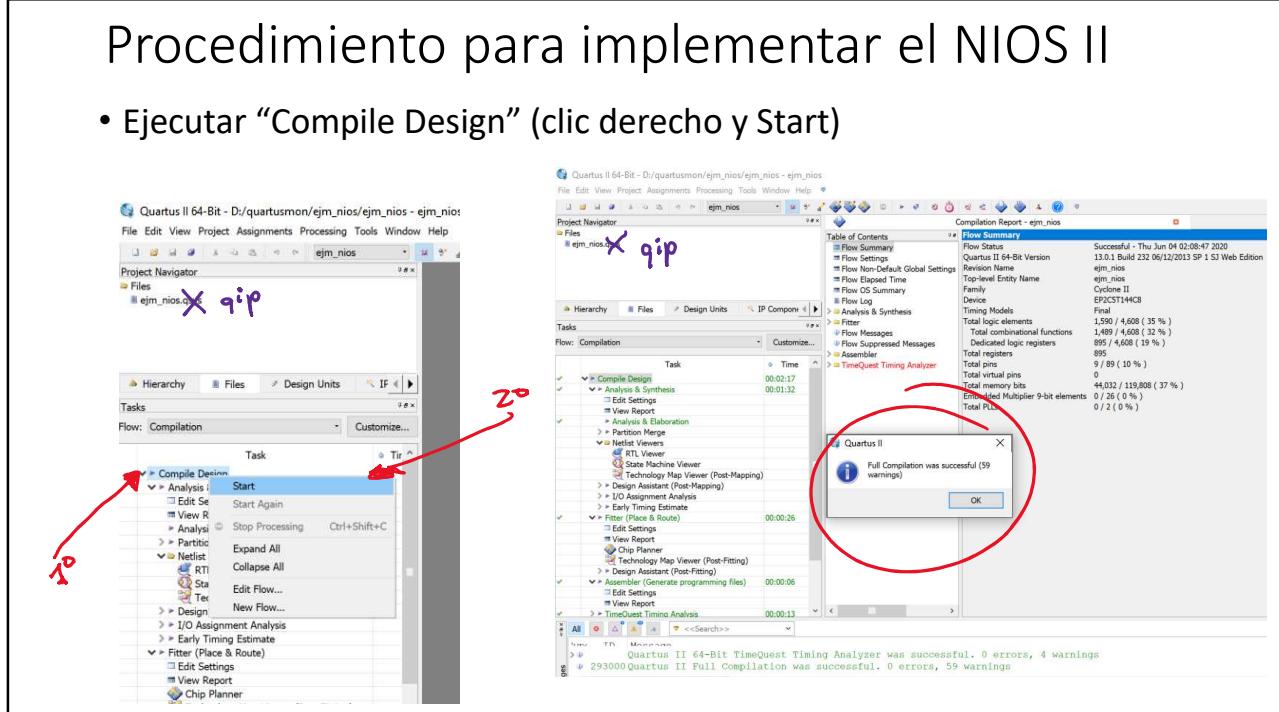
- Retornar al Quartus y agregar el archivo qip al proyecto



17

## Procedimiento para implementar el NIOS II

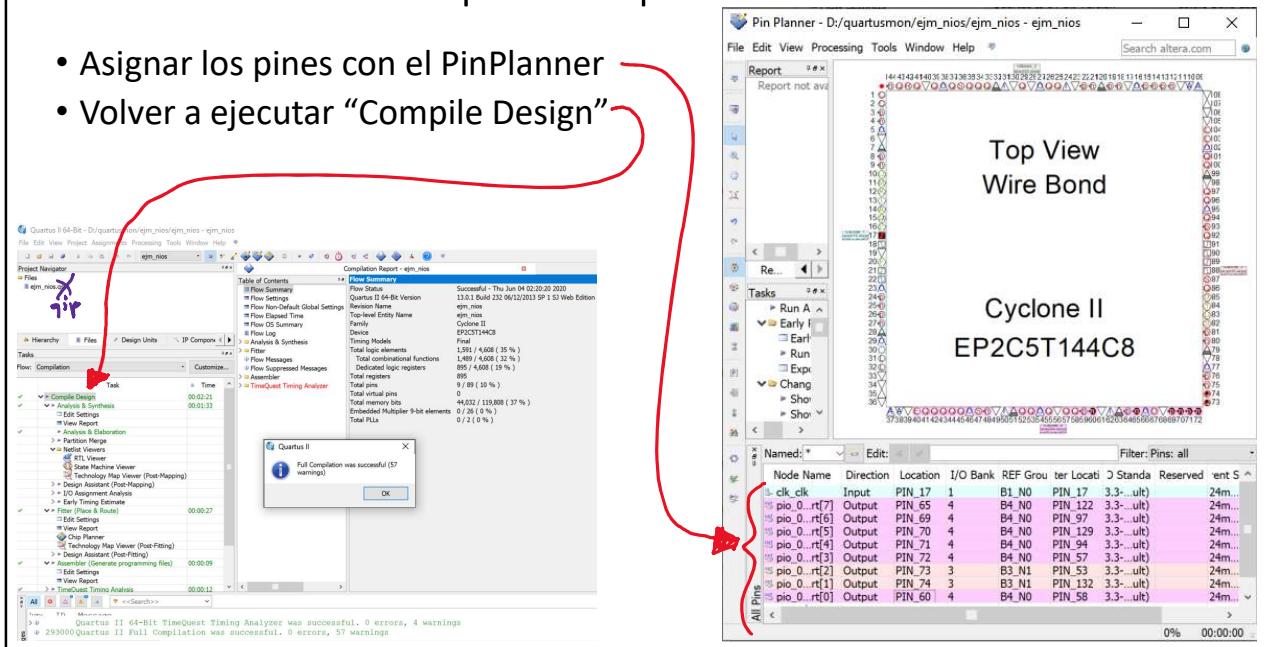
- Ejecutar “Compile Design” (clic derecho y Start)



18

## Procedimiento para implementar el NIOS II

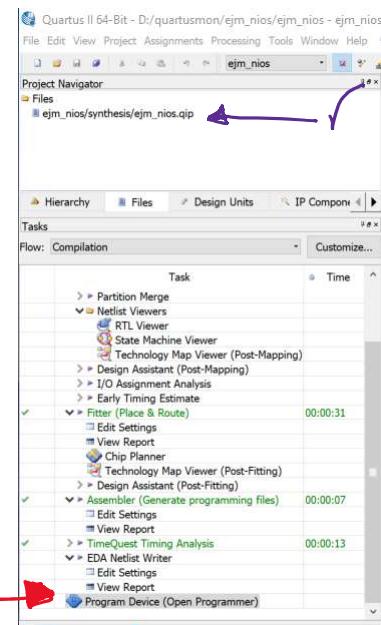
- Asignar los pines con el PinPlanner
- Volver a ejecutar “Compile Design”



19

## Procedimiento para implementar el NIOS II

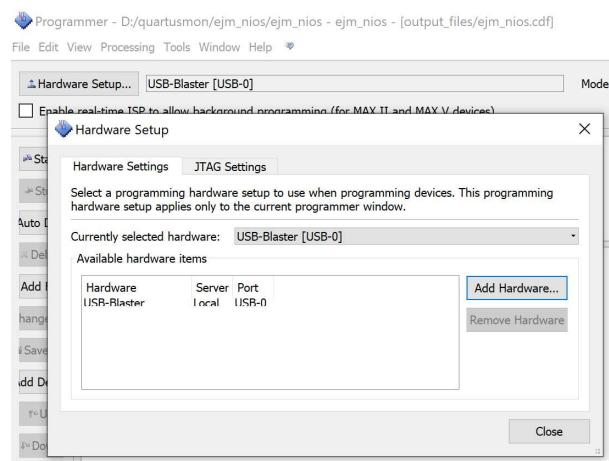
- Conectar el USB-Blaster al computador
- Conectar el USB-Blaster a la tarjeta de FPGA
- Conectar la fuente de energía a la tarjeta FPGA y encender la tarjeta
- Programar el dispositivo
  - Program Device (Open Programmer)



20

## Procedimiento para implementar el NIOS II

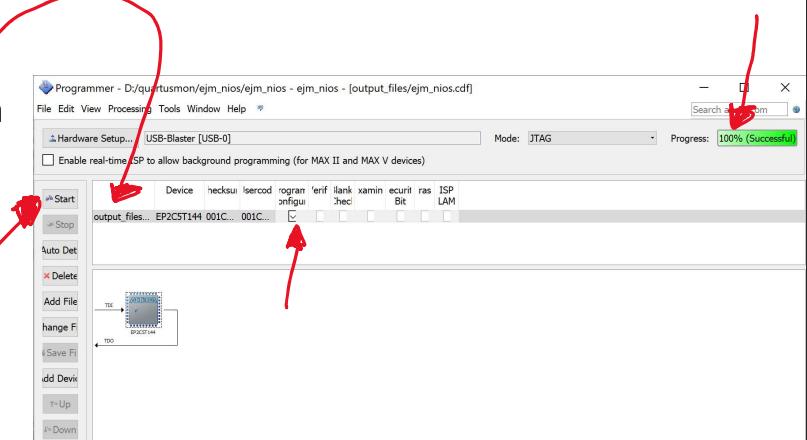
- En la ventana de Programmer, si no aparece el USB Blaster: entrar a Hardware Setup y seleccionar USB-Blaster (previamente se debió de instalar manualmente el driver y el S.O. Windows debe de haber reconocido correctamente el USB-Blaster)



21

## Procedimiento para implementar el NIOS II

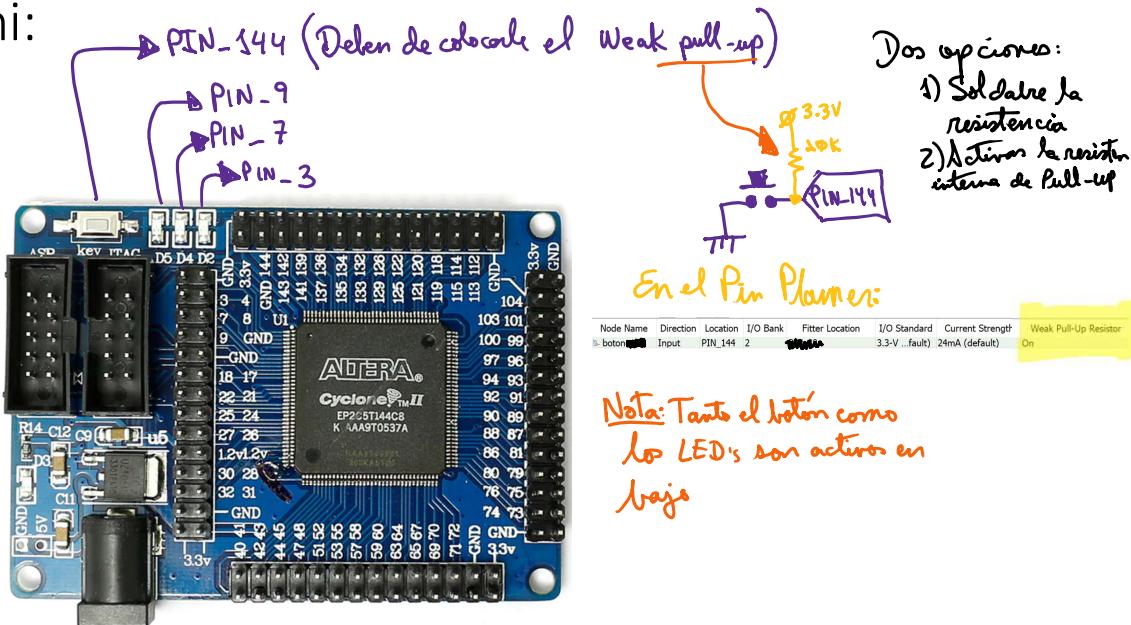
- Verificar que el archivo .sof sea el correcto
- Clic en Start y se iniciará la grabación del FPGA, esperar al 100%



22

## Usando los periféricos integrados del EP2C5T

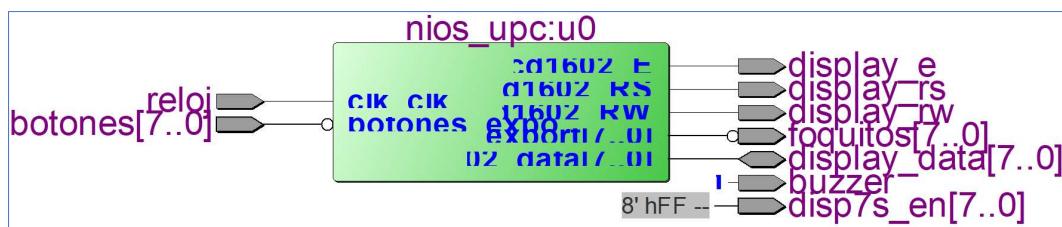
Mini:



23

## Vista RTL del NIOS II implementado:

- Nota: Orientado a la tarjeta A-C2FB:

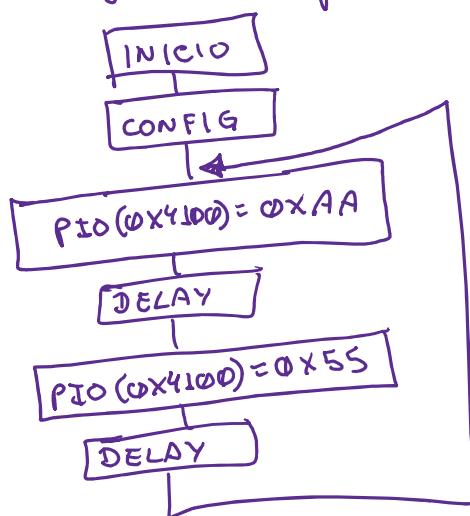


24

## Procedimiento para implementar el NIOS II

- Aplicación ejemplo a desarrollar en C para el procesador NIOS II:
  - Efecto de iluminación con los LEDs de la tarjeta que están conectados al PIO con dirección 0x4100

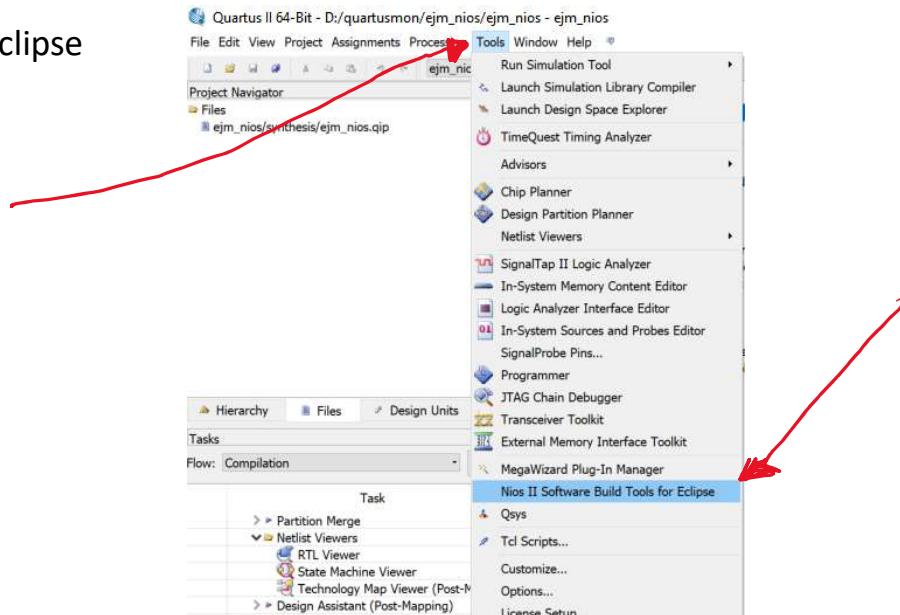
Diagrama de flujo:



25

## Procedimiento para implementar el NIOS II

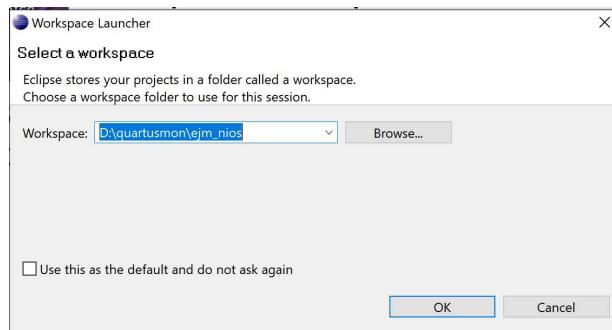
- Abrir el Eclipse



26

## Procedimiento para implementar el NIOS II

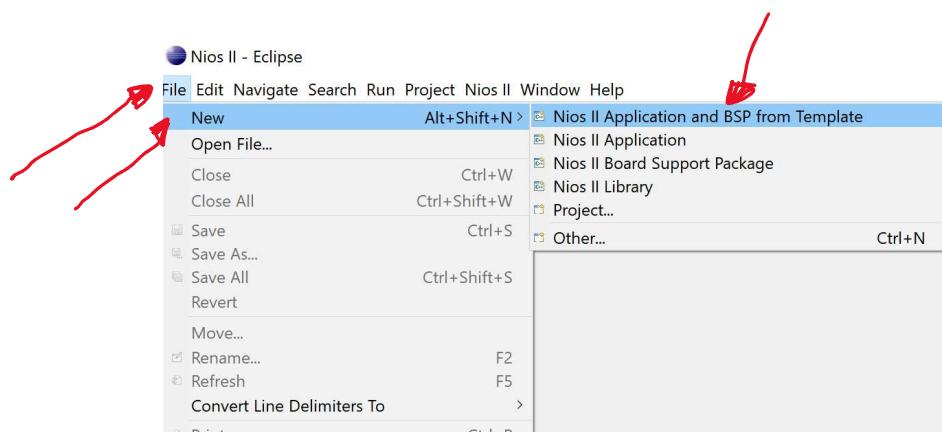
- Usar la misma carpeta donde esta creado el proyecto ejemplo



27

## Procedimiento para implementar el NIOS II

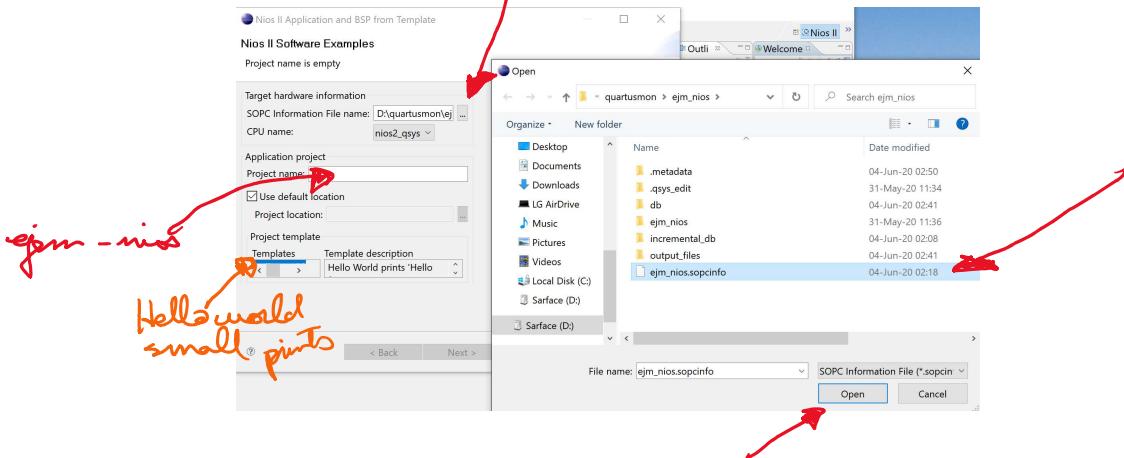
- Para crear el archivo fuente, escoger Nios II Application and BSP from Template



28

## Procedimiento para implementar el NIOS II

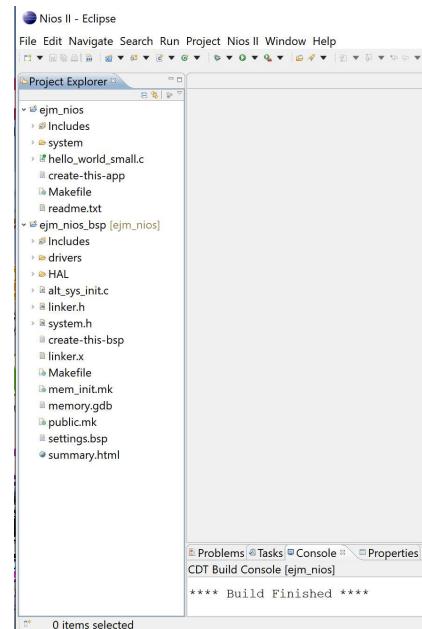
- Seleccionar el sopinfo creado en Qsys:
- El nombre del proyecto en Eclipse deberá ser el mismo
- Escoger el Template Hello World small prints



29

## Procedimiento para implementar el NIOS II

- Se habrá creado un árbol de archivos



30

## Procedimiento para implementar el NIOS II

- Abrir el archivo fuente  
hello\_world\_small.c

The screenshot shows the Eclipse IDE interface with the title "Nios II - parpadea/hello\_world\_small.c - Eclipse". The Project Explorer view on the left shows a project named "ejm\_nios" containing files like "Binaries", "Includes", "obj", and "system", along with the source file "hello\_world\_small.c". The right-hand editor pane displays the code for "hello\_world\_small.c". The code includes function prototypes for "alt\_printf", "alt\_putstr", "alt\_putchar", and "alt\_getchar", followed by an include directive for "stdio.h". A purple arrow points from the list item "Abrir el archivo fuente" to the "hello\_world\_small.c" file in the Project Explorer.

```

Function
=====
alt_printf
alt_putstr
alt_putchar
alt_getchar
*/
#include <stdio.h>

```

31

## Procedimiento para implementar el NIOS II

- Escribir el código de programa siguiente:

The screenshot shows the Eclipse IDE interface with the code for "hello\_world\_small.c". The code defines a "delay" function and a "main" function. Handwritten annotations explain the code: curly braces group the "delay" function body as "función para generar un retardo" (function to generate a delay), the "main" function body as "función main" (main function), and the infinite loop in "main" as "repetición infinita" (infinite loop). Red arrows point from these annotations to specific parts of the code. Handwritten binary values "101011010" and "01010105" are shown above the assembly output, with arrows pointing from them to the assembly code. Red annotations also describe the assembly output as "librería para usar los PIO" (library to use the PIO) and "escribe 0XAA en el puerto con dirección 0X4100" (writes 0XAA to port at address 0X4100) and "escribe 0X55 en el puerto con dirección 0X4100" (writes 0X55 to port at address 0X4100).

```

#include <stdio.h>
#include "system.h"
#include "altera_avalon_pio_regs.h" // librería para usar los PIO

void delay(int a){
    volatile int i = 0;
    while (i < a*10000){
        i++;
    }
}

int main(){
    while(1){
        IOWR_ALTERA_AVALON_PIO_DATA(0x4100, 0xAA); // 101011010
        delay(30);
        IOWR_ALTERA_AVALON_PIO_DATA(0x4100, 0x55); // 01010105
        delay(30);
    }
    return 0;
}

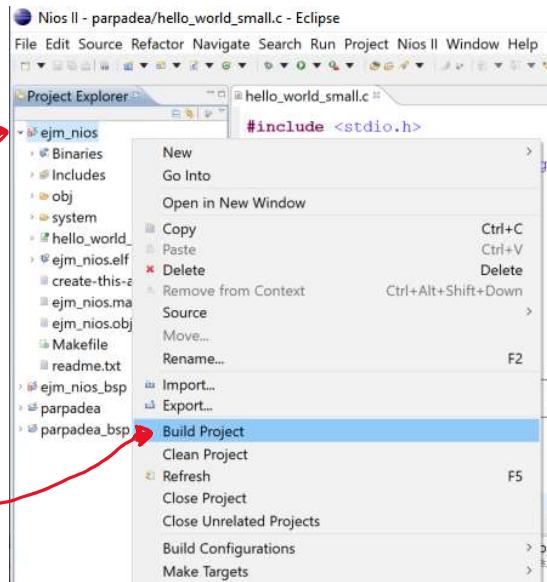
```

32

## Procedimiento para implementar el NIOS II

- Armar el proyecto (Build Project)

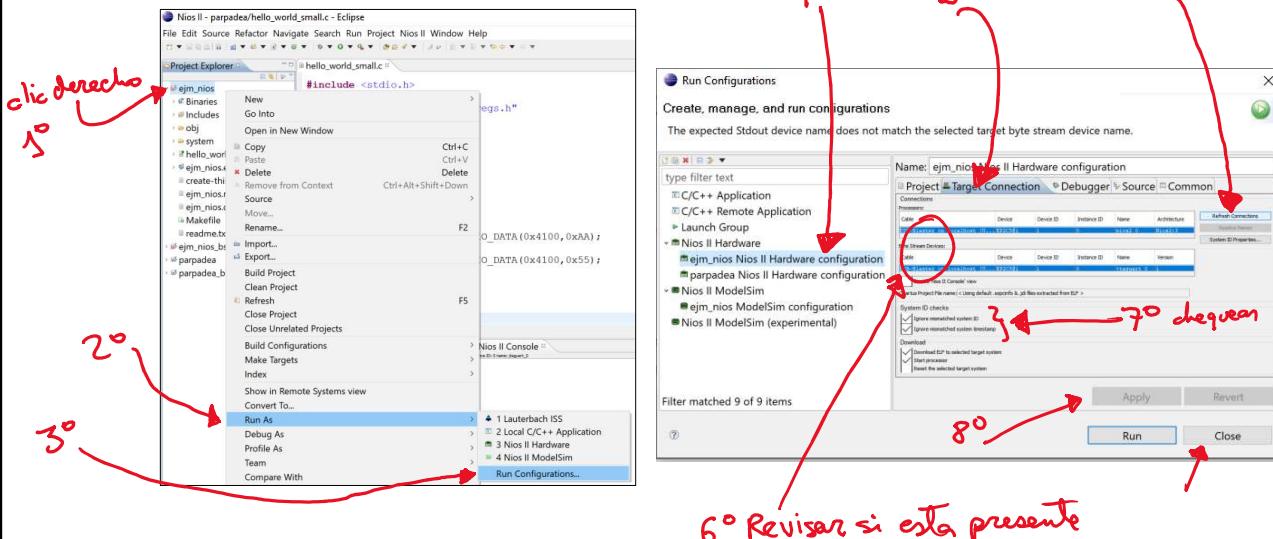
1º clic derecho



33

## Procedimiento para implementar el NIOS II

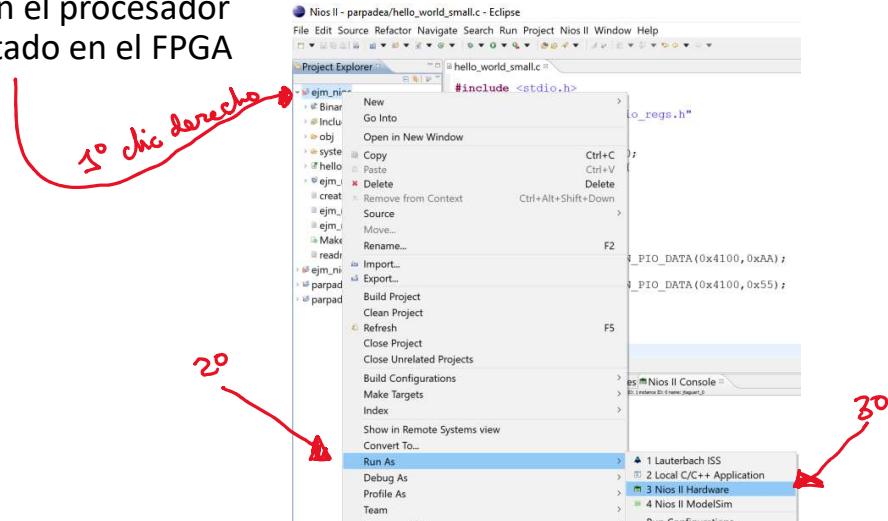
- Configurar conexión del NIOS II previamente grabado con el Eclipse



34

## Procedimiento para implementar el NIOS II

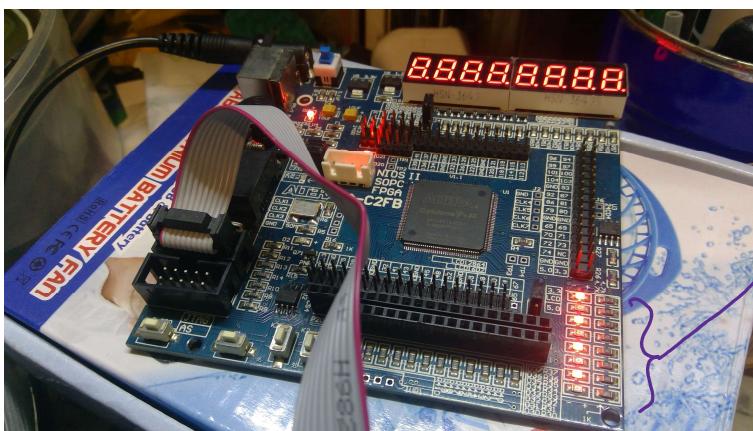
- Correr el código en el procesador NIOS II implementado en el FPGA



35

## Procedimiento para implementar el NIOS II

- Verificar el funcionamiento en la tarjeta A-C2FB:



Leds conectados al  
PIO con dirección  
0x4100 del NIOSII

36

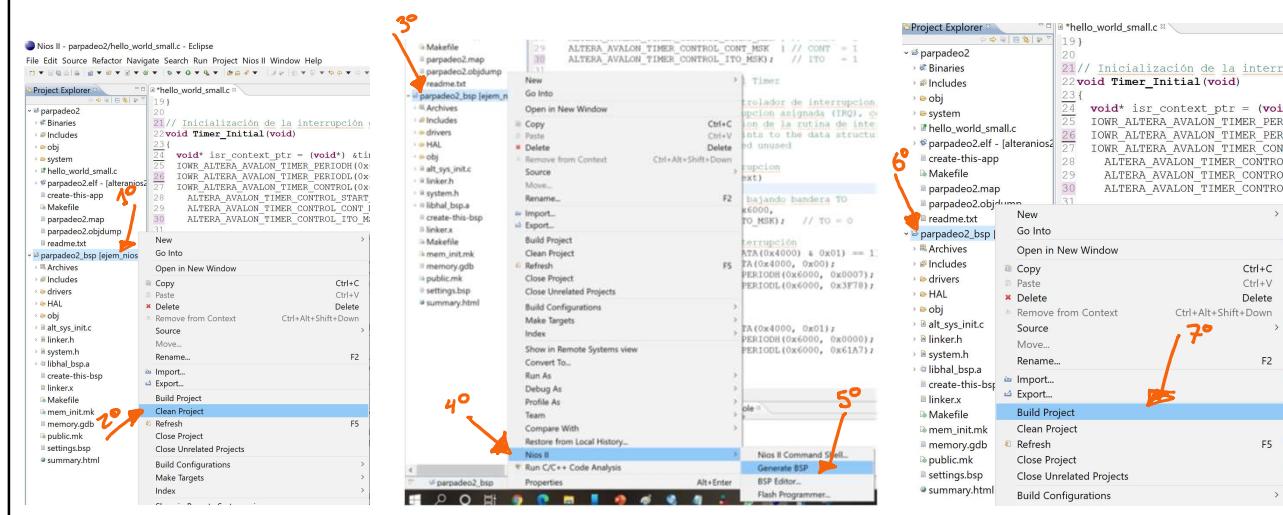
# Consideraciones al modificar la estructura en QSys o en el proyecto en Quartus II

- Estando en la etapa de desarrollo del programa de usuario del NIOS II en Eclipse requieres modificar la estructura del hardware o parámetros tanto en el QSys como en el Quartus II, será necesario hacer lo siguiente:
  - Si se modificó algo en QSys (estructura o parámetros), se tendrá que volver a generar el sopcinfo, luego compilar el proyecto en el Quartus y grabar el FPGA.
  - Si se modificó algo en el Quartus (tanto en estructura como en asignación de pines) se compilará el proyecto para luego grabarlo en el FPGA.

37

# Consideraciones al modificar la estructura en QSys o en el proyecto en Quartus II

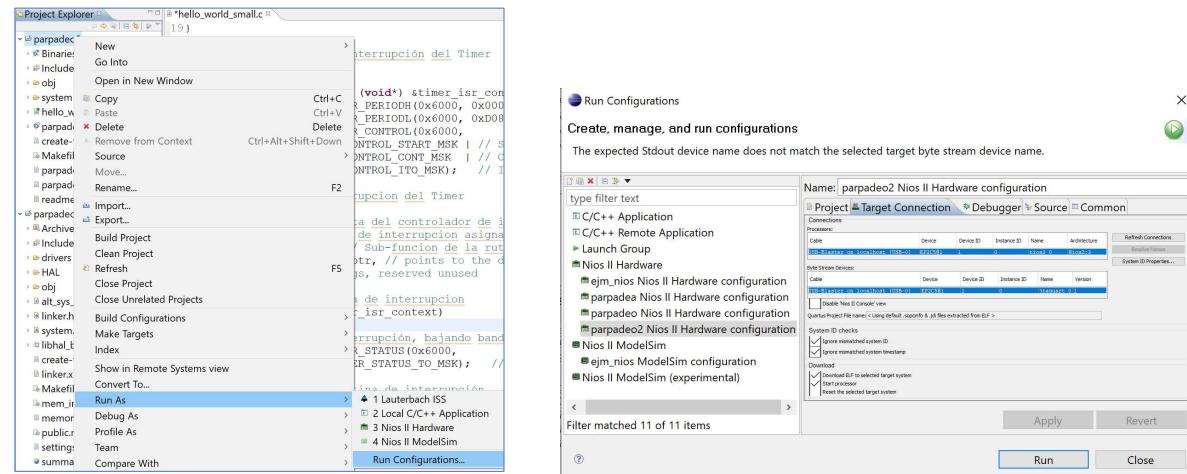
- En Eclipse hace lo siguiente:



38

# Consideraciones al modificar la estructura en QSys o en el proyecto en Quartus II

- Revisar en Run As / Run Configurations si aparece el NIOS II implementado ingresando al menú Target Connection



39

## Consideraciones finales

- Será necesario revisar la hoja técnica de la tarjeta de FPGA Cyclone II que se está utilizando para ver que dispositivos externos posee.
- Para el caso de la tarjeta A-C2FB se tendrá que instanciar el NIOS II en un código VHDL para poder declarar señales adicionales y desactivar algunos de esos dispositivos externos conectados al FPGA.
  - El puerto del FPGA conectado al buzzer deberá de declararse como salida y activado en alto para que deje de sonar.
  - Tanto los LEDs, displays como los pulsadores que están en la tarjeta son activos en bajo.
- Si se graba el NIOS II en la memoria de configuración externa EPROM, solo se grabará la implementación del procesador mas no el código fuente. Para ello se requerirá de una tarjeta que posea memoria flash externa al FPGA para el programa.

40