

Sistemas Digitales

Práctica 1

Creación de un Sistema Embebido

(Adaptado del curso “Embedded System Design Flow” de Xilinx)

Introducción

Esta práctica lo guiará a través del proceso de creación de un diseño simple basado en el procesador ARM Cortex-A9 a ser implementado en la placa Arty-z7-010 utilizando Vivado. Usará esta aplicación para crear el sistema de hardware y el SDK (Software Development Kit) para crear una aplicación de software de ejemplo para verificar la funcionalidad de hardware.

Objetivos

Después de completar esta práctica será capaz de:

- Crear un proyecto en Vivado para un sistema usando una Zynq
- Usar el IP Integrator para crear un sistema de hardware
- Usar el SDK para crear un proyecto para probar una memoria estándar
- Ejecutar la aplicación de prueba sobre la placa

Procedimiento

Esta práctica está separada en pasos que consisten en sentencias generales que proveen información sobre las instrucciones detalladas que le siguen. Siga estas instrucciones detalladas para avanzar dentro de esta práctica.

Esta práctica está compuesta por 5 pasos principales: crear un proyecto top-level usando Vivado, crear el sistema con el procesador usando IP Integrator de Vivado, generar el HDL top-level y exportar el diseño al SDK, crear una aplicación de prueba de una memoria en SDK, y finalmente, probar en hardware.

Descripción del Diseño

El propósito de las prácticas es recorrer todo el diseño del sistema que incluye el procesador, tanto en lo referente a hardware como a software. Cada práctica será armada en base a la anterior. El diagrama siguiente representa el diseño completo al que se llegará al final de la práctica 5 (Figura 1).

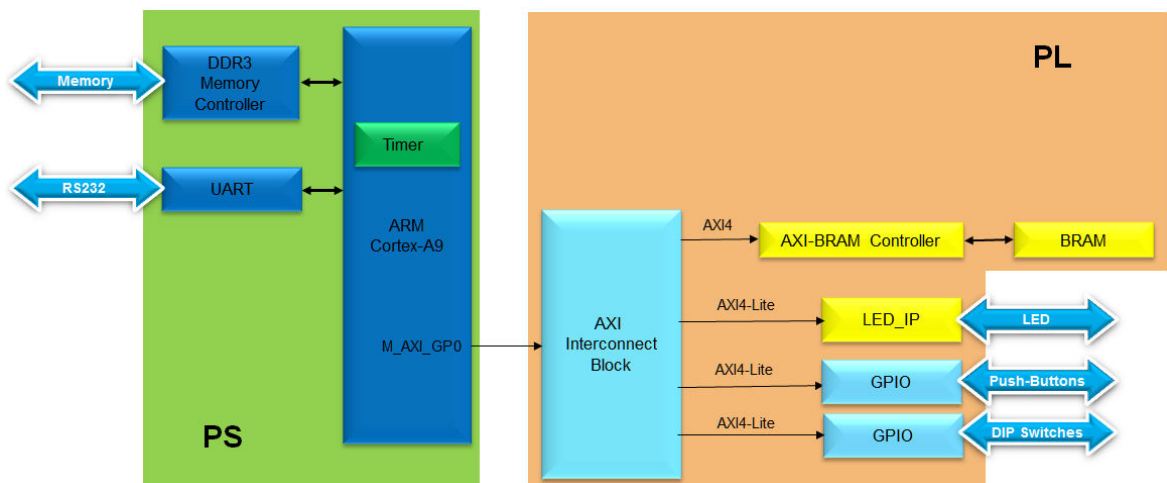


Figura 1. Diseño completo

En esta práctica usará el IP Integrator para crear un diseño basado en un sistema de procesamiento consistente de lo siguiente (Figura 2):

- Núcleo ARM Cortex A9 (PS)
- UART para comunicación serie
- Controlador DDR3 para una memoria externa DDR3_SDRAM

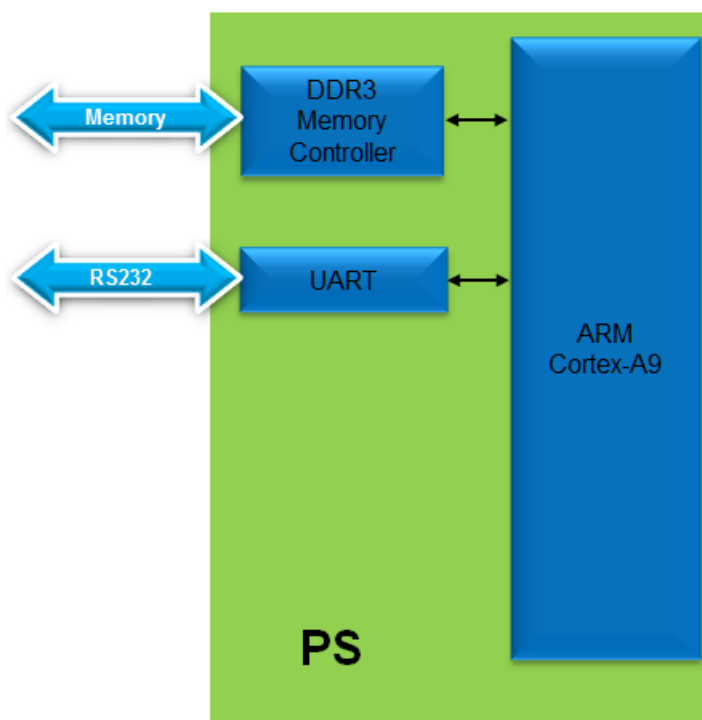


Figura 2. Diseño del procesador de esta práctica

Flujo General para esta práctica



Crear un Proyecto en Vivado

Paso 1

1-1. Ejecutar Vivado y crear un proyecto vacío para la placa Arty-Z7-010, usando el lenguaje VHDL.

1-1-1. Abrir Vivado seleccionando **Start ► Xilinx Design Tools ► Vivado 2018.1**

1-1-2. Hacer click en **Create New Project** para iniciar el asistente. Verá el cuadro de diálogo *Create a New Vivado Project*. Presionar **Next**.

1-1-3. Hacer click en el botón Browse del campo *Project Location* de la ventana **New Project**, navegar hasta el directorio donde se guardarán todos los proyectos, y presionar **Select**.

1-1-4. Introducir **lab1** en el campo *Project Name*. Asegurarse que el casillero *Create Project Subdirectory* esté marcado. Presionar **Next**.

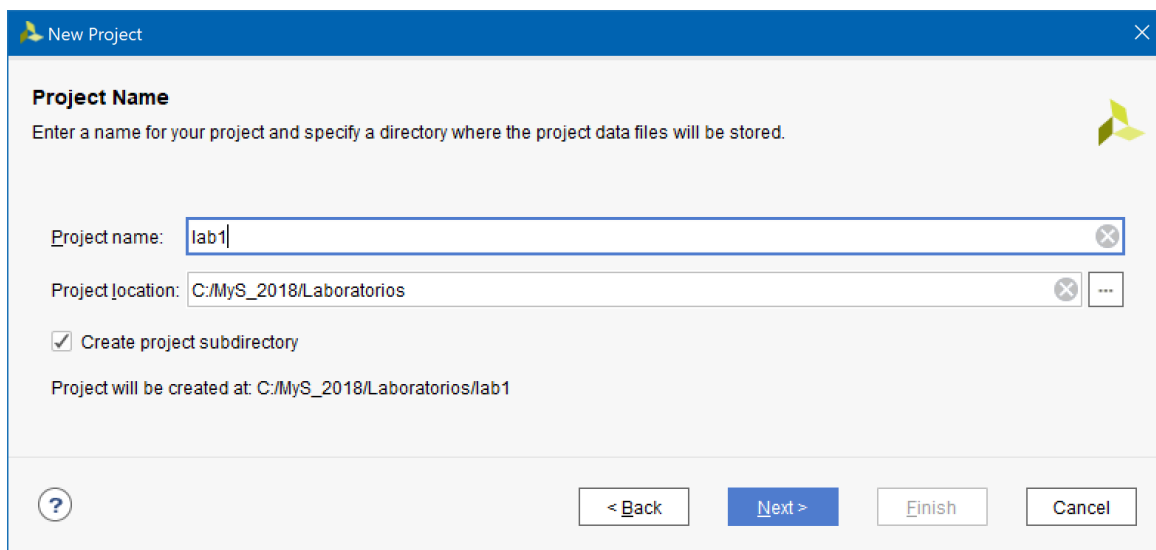


Figura 3. Ingreso del nombre del proyecto

1-1-5. En la ventana *Project Type* seleccionar **RTL Project**, y presionar **Next**.

1-1-6. En la ventana *Add Sources*, seleccionar **VHDL** como el *Target language* y **Mixed** como el lenguaje del simulador, y presione **Next**.

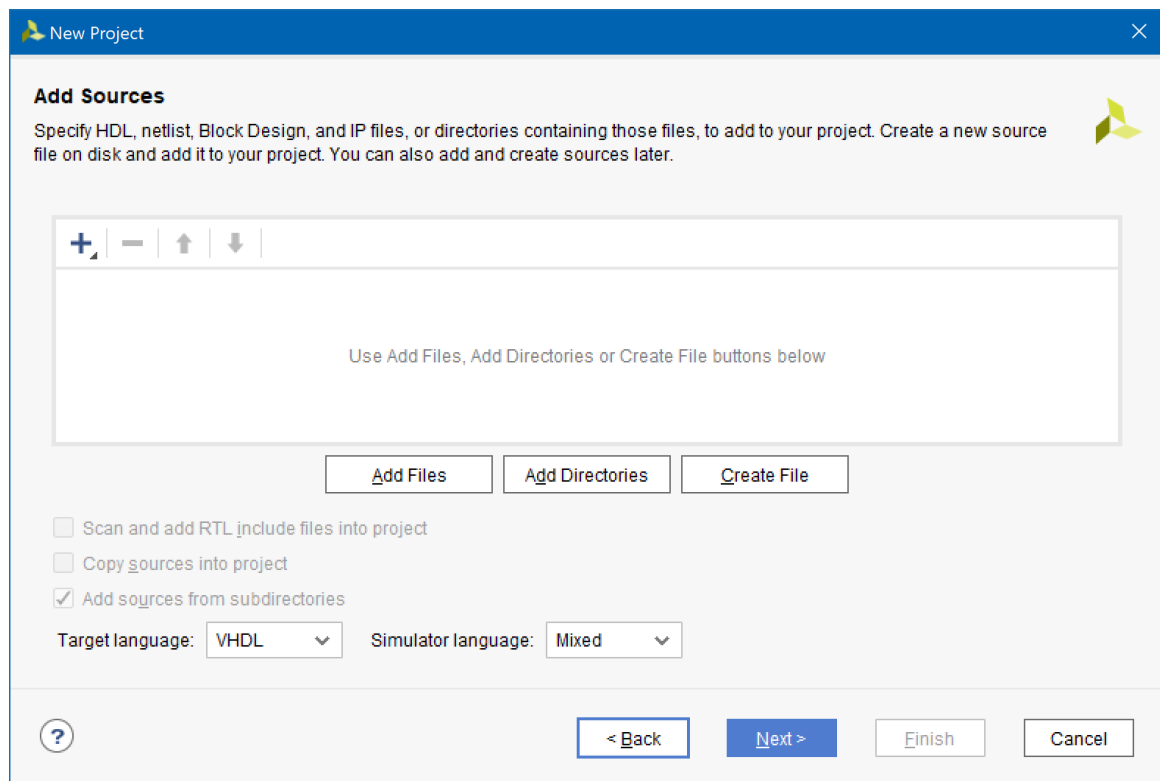


Figura 4. Agregar archivos fuente a un nuevo proyecto

- 1-1-7. Hacer click en **Next** dos veces más para saltar *Adding Existing IP* y *Add Constraints*
- 1-1-8. En la ventana *Default Part*, seleccionar *Boards*, y seleccionar la placa Arty Z7010. Presionar **Next**.

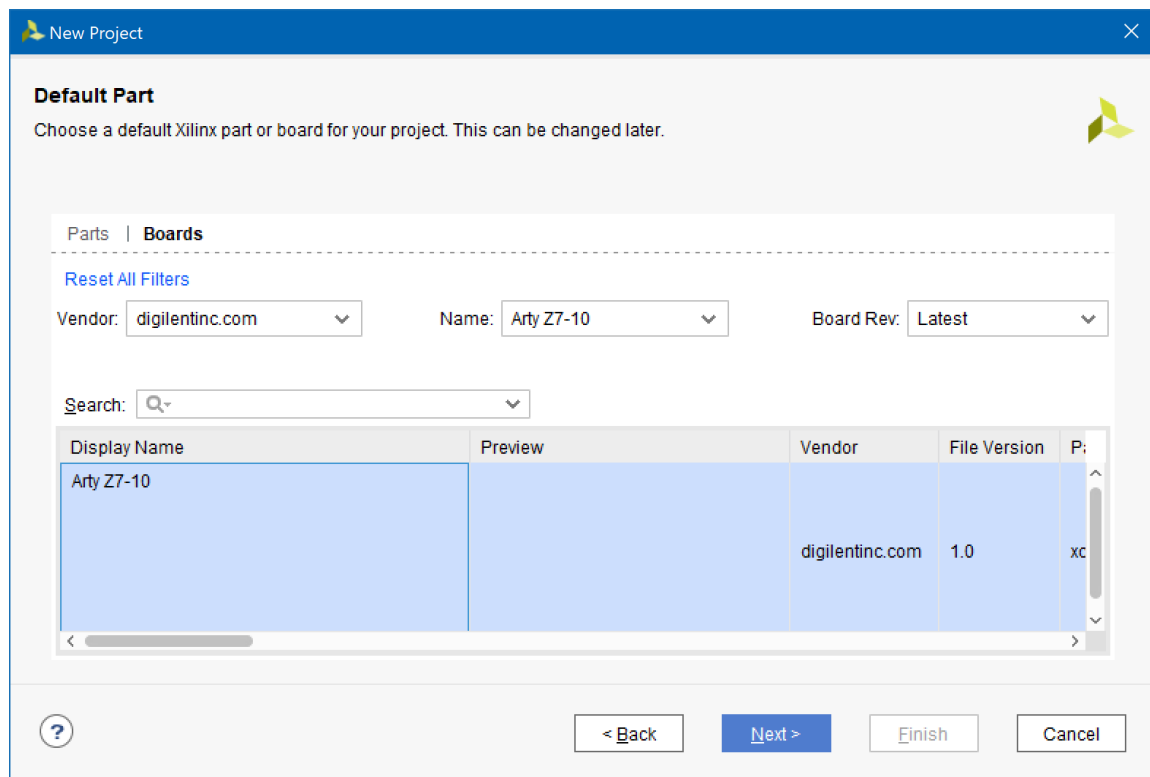


Figura 5 .Selección de placas (boards) y partes

1-1-9. Verificar el *Project Summary* y presionar **Finish** para crear un proyecto Vivado vacío.

Crear el Sistema Usando el IP Integrator

Paso 2

2-1. Usar el IP Integrator para crear un nuevo Bloque de Diseño, agregar el bloque de sistema de procesamiento ZYNQ, e importar el archivo xml provisto para la placa.

2-1-1. En el Flow Navigator, hacer click en **Create Block Design** debajo de IP Integrator

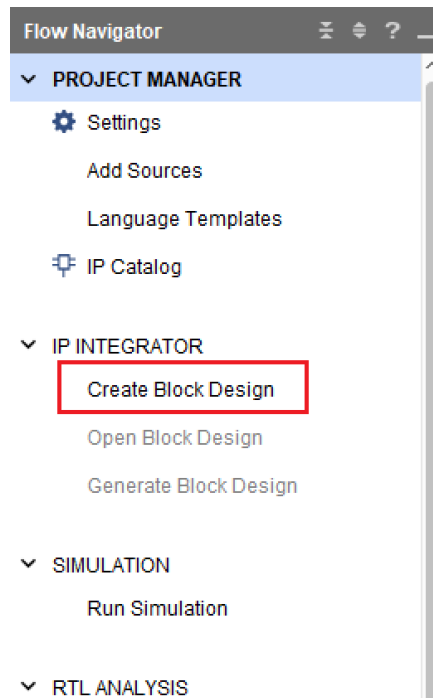


Figura 6. Crear IP Integrator Block Diagram

2-1-2. Ingresar **system** como nombre del diseño y presionar **OK**

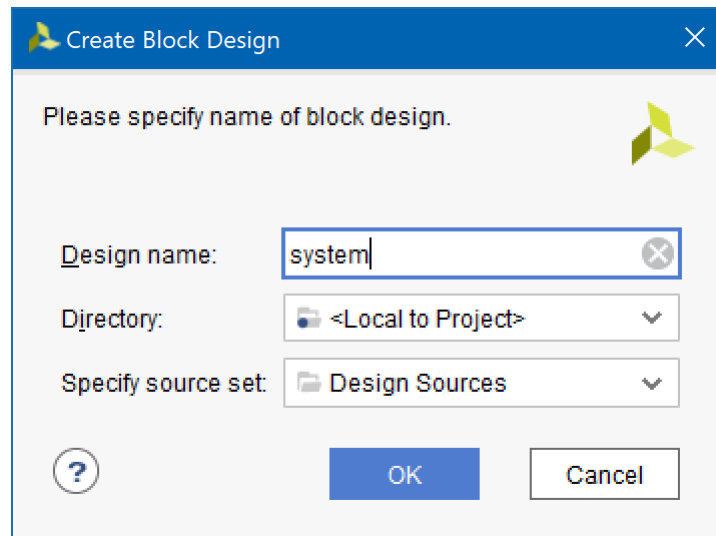


Figura 7. Crear un nuevo Block Diagram

2-1-3. La IP de catálogo puede ser agregada de diferentes maneras:

- Haciendo click en la barra de herramientas del panel *Diagram*
- Presionando Ctrl + I
- Haciendo click en el operador + en el centro del panel *Diagram*
- Click-derecho en cualquier lugar del espacio de trabajo del panel *Diagram* y seleccionar *Add IP*.

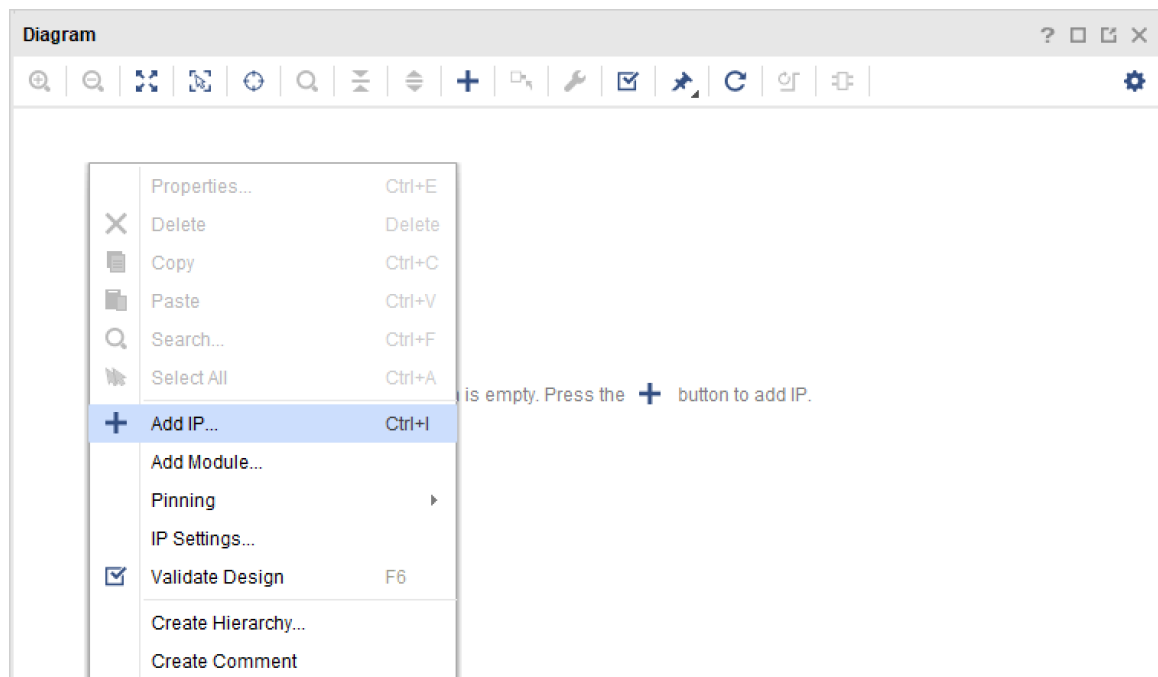


Figura 8. Agregar IP al Block Diagram

- 2-1-4. Una vez que el catálogo de IP está abierto, tipear “z” dentro de la barra de búsqueda, encontrar y hacer doble click sobre **ZYNQ7 Processing System**, o hacer click en la entrada y presionar la tecla enter para agregarlo al diseño.

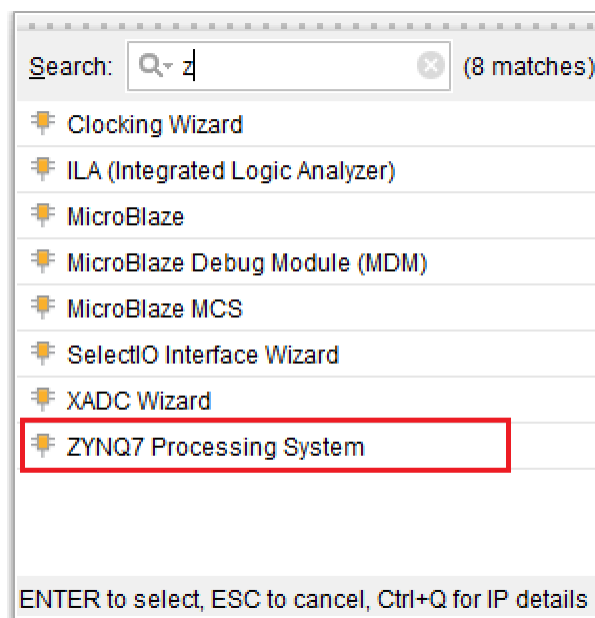


Figura 9. Agregar el bloque Zynq al diseño

- 2-1-5. Notar el mensaje en la parte superior de la ventana *Diagram* diciendo que está disponible asistencia para el diseñador. Hacer click en *Run Block Automation* y seleccionar **/processing_system7_0**

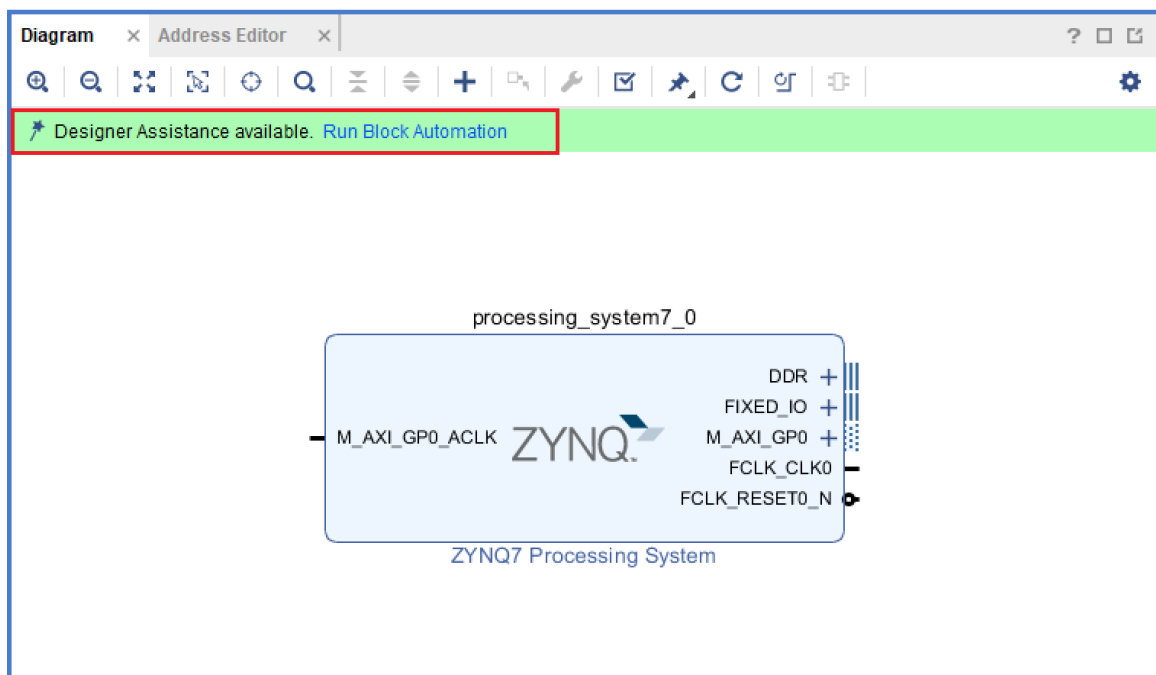


Figura 10. Ejecutar el block automation

- 2-1-6. En la ventana *Run Block Automation*, dejar seleccionadas las opciones por defecto, incluyendo *Apply Board Preset* marcada, y presionar **OK**

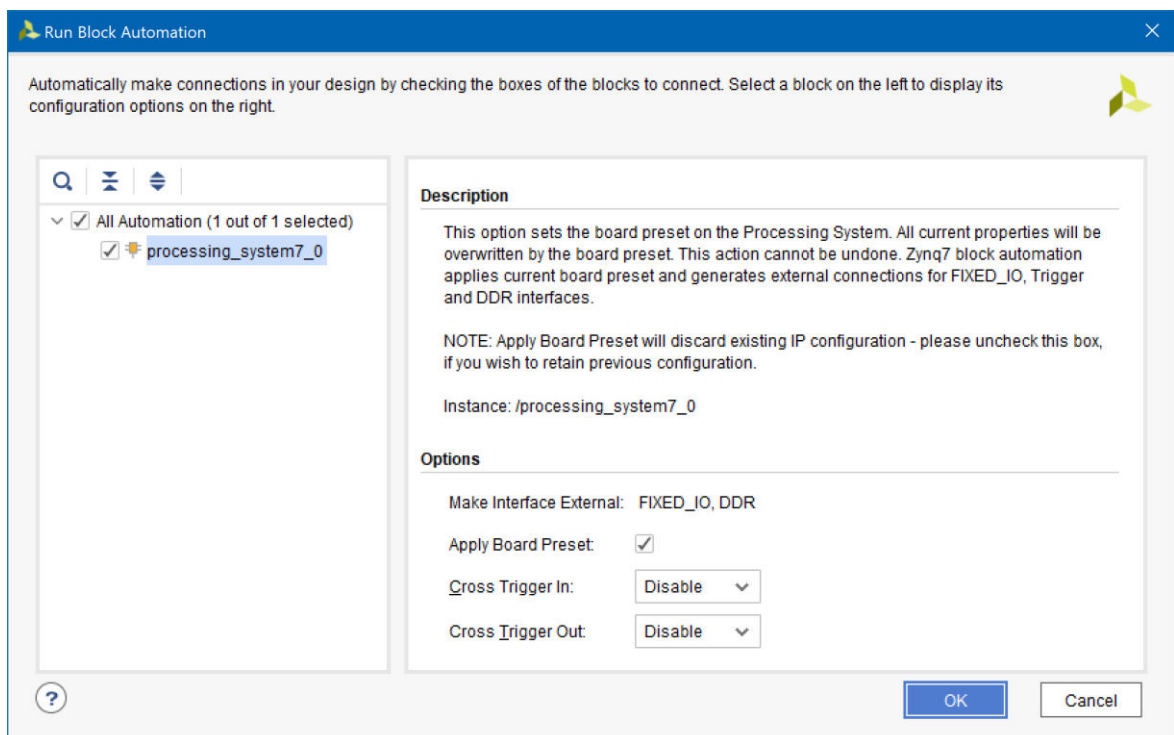


Figura 11. Configuración del Block Automation

Una vez que ha sido completado el *Block Automation*, notar que un puerto ha sido automáticamente agregado para la DDR y Fixed IO, y algunos puertos adicionales ahora son visibles. La configuración importada para el Zynq asociada con la placa Arty Z7-010 ha sido aplicada y ahora será modificada.

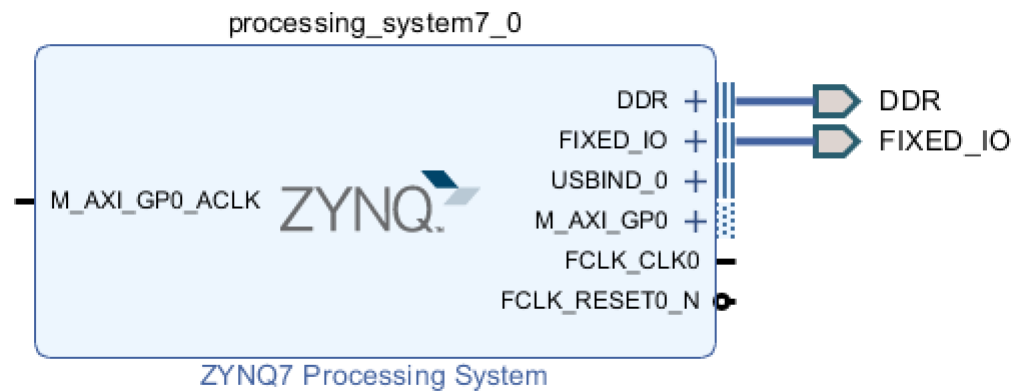


Figura 12. Bloque Zynq con DDR y puertos Fixed IO

2-1-7. Hacer doble click sobre el bloque que se ha agregado para abrir su ventana de personalización.

Notar que ahora la ventana *Customization* muestra los periféricos seleccionados (con marcas de selección). Esta es la configuración por defecto para la placa aplicada por el bloque de automatización.

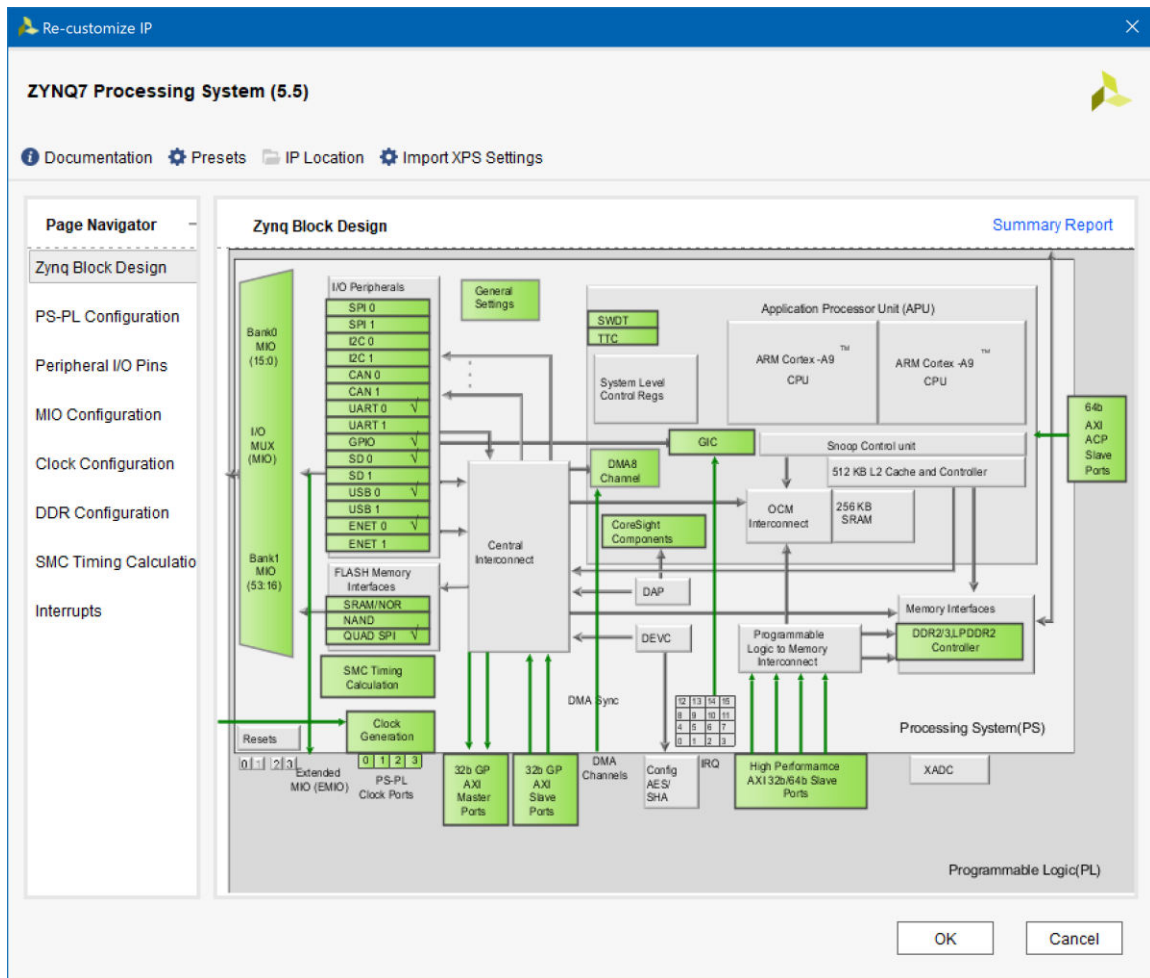


Figura 13. Configuración de los periféricos importados

2-2. Configurar el bloque de procesamiento con el periférico UART 0 solamente habilitado.

2-2-1. En este momento debería aparecer abierto nuevamente un diagrama en bloques del Zynq, mostrando varios bloques configurables del Sistema de Procesamiento.

En esta etapa, el diseñador puede hacer click sobre varios bloques configurables (resaltado en verde) y cambiar la configuración del sistema.

Para esta práctica sólo se requiere la UART, por lo tanto los otros periféricos serán descartados.

2-2-2. Hacer click sobre uno de los periféricos (en verde) en el bloque *IOP Peripherals* block, o seleccionar la pestaña *MIO Configuration* a la izquierda para abrir la ventana de configuración.

2-2-3. Expandir I/O peripherals si es necesario, y asegurarse de que todos los periféricos estén desmarcados excepto la UART 0. Es decir, remover:

ENET 0
USB 0
SD 0

Expandir **GPIO** to deselect *GPIO MIO*

Expandir **MemoryInterfaces** to deselect *Quad SPI Flash*

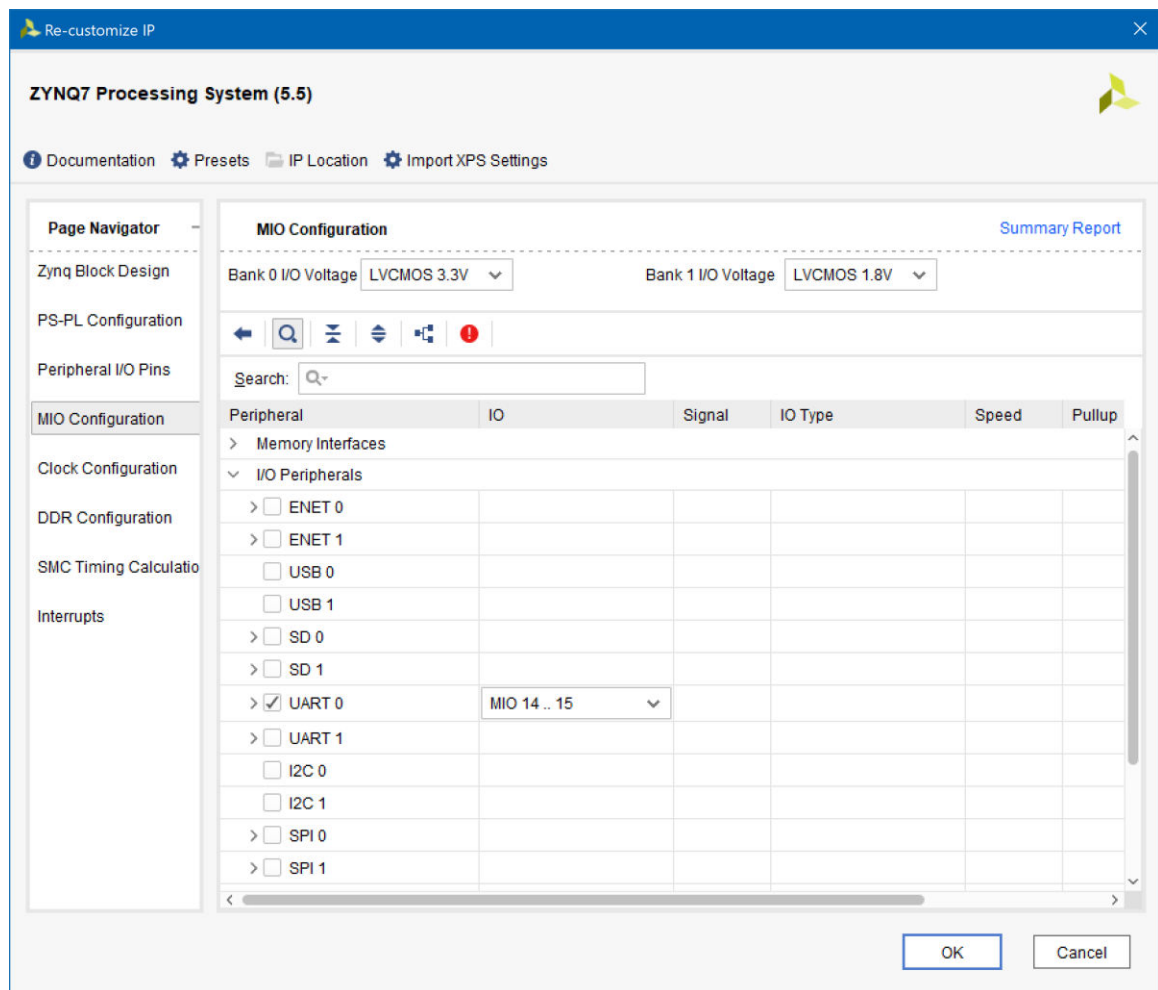


Figura 14. Selección de la UART 0

- 2-2-4. Seleccionar la pestaña **PS-PL Configuration** que se encuentra a la izquierda.
- 2-2-5. Expandir *AXI Non Secure Enablement* ► *GP Master AXI interface* y des-seleccionar la interfaz **M AXI GP0**.
- 2-2-6. Expand **General** ► **Enable Clock Resets** y des-seleccionar la opción **FCLK_RESET0_N**.
- 2-2-7. Seleccionar la pestaña **Clock Configuration** que se encuentra a la izquierda. Expandir **PL Fabric Clocks** y des-seleccionar la opción **FCLK_CLK0** y hacer click en **OK**.

Presionar sobre el icono indicado en la figura 15 (*Regenerate Layout*) y observar el siguiente diagrama en bloques.

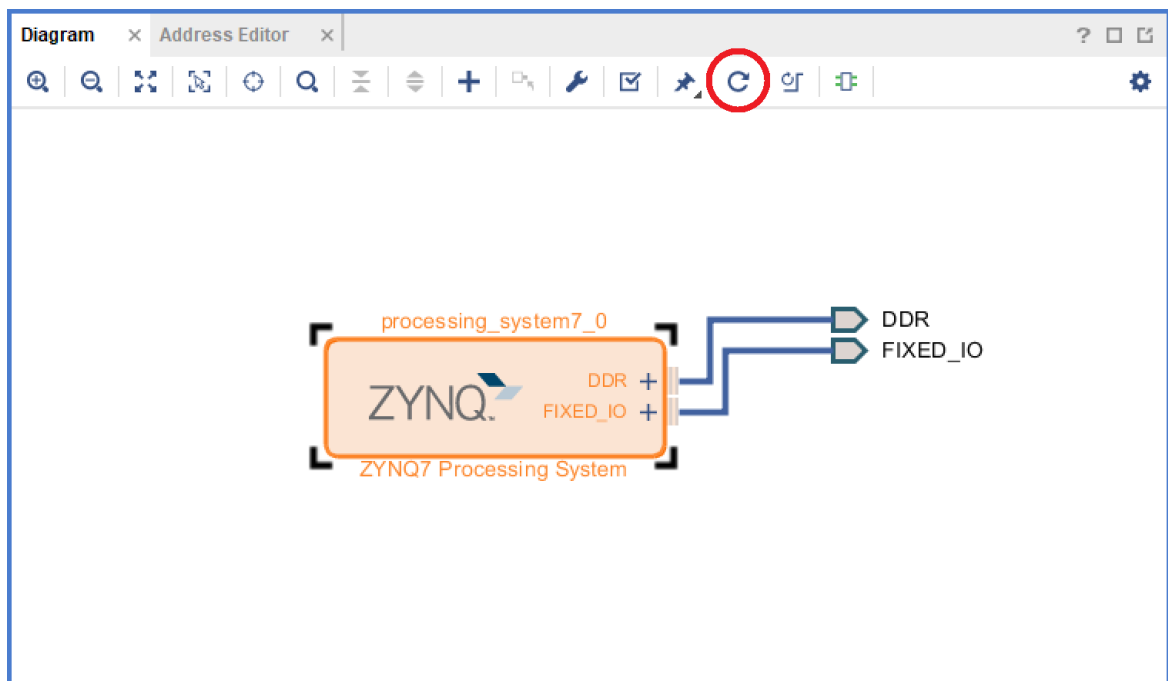


Figura 15. Regenerar el Layout

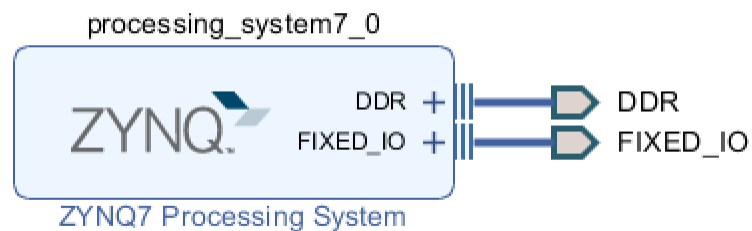


Figura 16. Bloque Zynq Actualizado

2-2-8. Presionar el botón  (Validate Design) y asegurarse de que no haya errores.

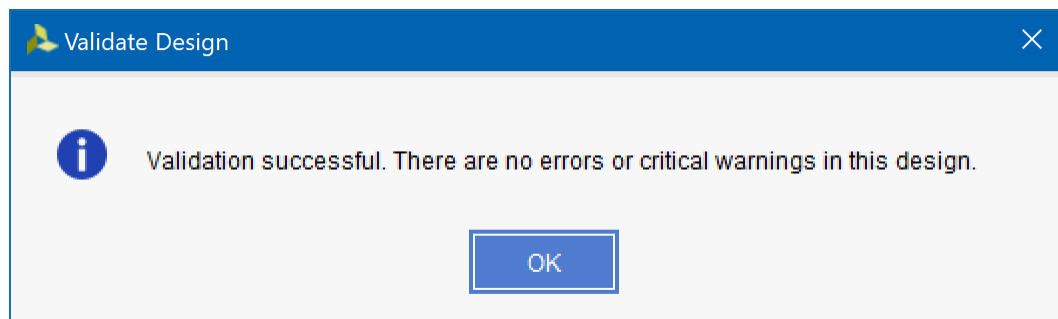


Figura 17. Validación

Generate Top-Level and Export to SDK

Paso 3

3-1. Generar las salidas de IP Integrator, el HDL top-level, y ejecutar el SDK exportando el hardware.

3-1-1. En el panel *Sources*, hacer click-derecho sobre *system.bd*, y seleccionar **Generate Output Products ...** y presionar **Generate** para generar los archivos de implementación, simulación y síntesis para el diseño (también puede hacer click sobre **Generate Block Design** en el panel Flow Navigator para hacer lo mismo). Presionar **OK**.

3-1-2. Hacer click-derecho nuevamente sobre *system.bd*, y seleccionar **Create HDL Wrapper...** para generar el modelo top-level de VHDL. Dejar seleccionada la opción *Let Vivado manager wrapper and auto-update*, y presionar **OK**.

Se creará el archivo *system_wrapper.vhd* y se agregará al proyecto. Hacer doble-click sobre el archivo para ver el contenido en el panel auxiliar.

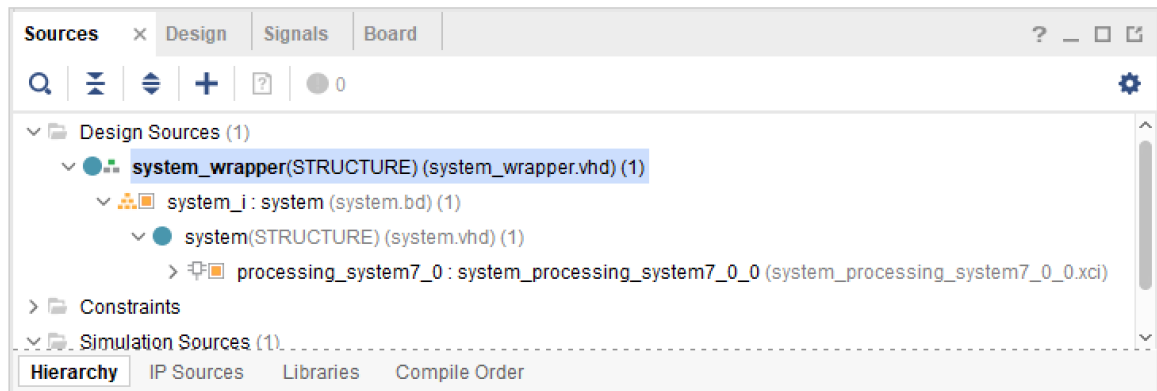


Figura 18. Archivo Wrapper HDL generado y agregado al proyecto

3-1-3. Notar que el archivo está configurado como el módulo principal (*Top module*) en el diseño, indicado por el icono

3-1-4. Seleccionar **File ► Export ► Export hardware** y presionar **OK**. (Guardar el proyecto si es requerido). En este momento se crea la carpeta *Nombre_del_proyecto.sdk*.

Nota: Ya que no se tiene ningún hardware en la lógica programable (PL) no existe ningún bitstream para generar, por lo tanto la opción de incluir bitstream no es necesaria en este momento.

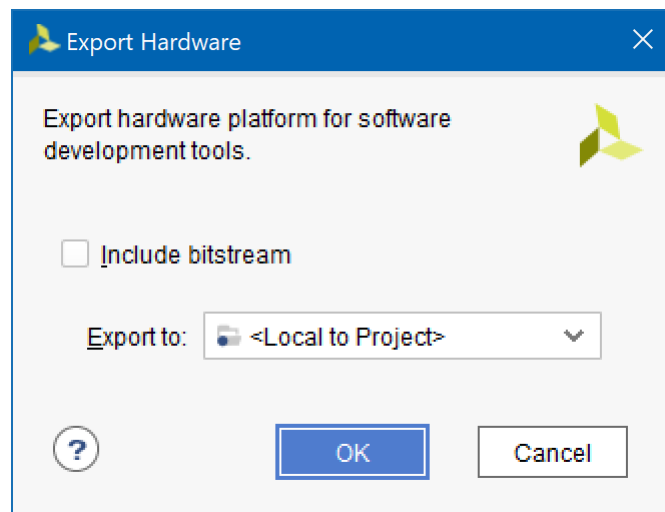


Figura 19. Exportar hardware

3-1-5. Seleccionar **File ► Launch SDK** dejando la configuración por defecto, y presionar **OK**

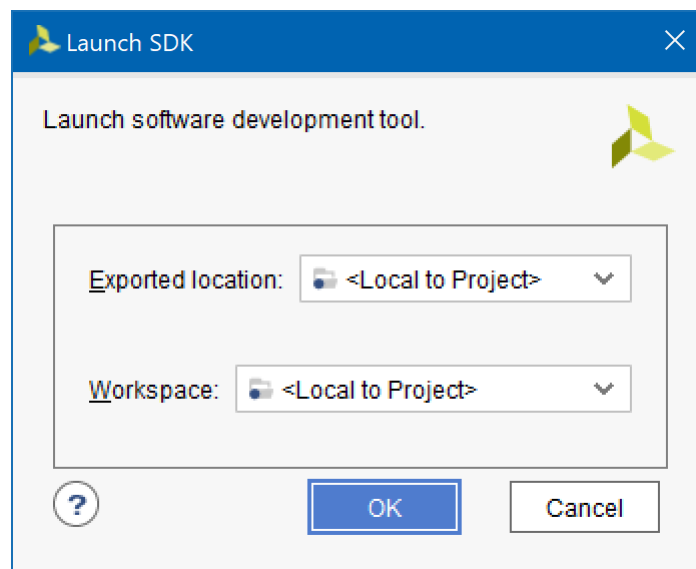


Figura 20. Ejecutar SDK

El SDK debería estar abierto en este momento. Si sólo el panel de Bienvenida es visible, cierre o minimice este panel para ver el *Project Explorer* y el panel *Preview*. Debería haberse creado un proyecto de plataforma de hardware automáticamente, y el directorio `system_wrapper_hw_platform_0` debería existir en el panel *Project Explorer*.

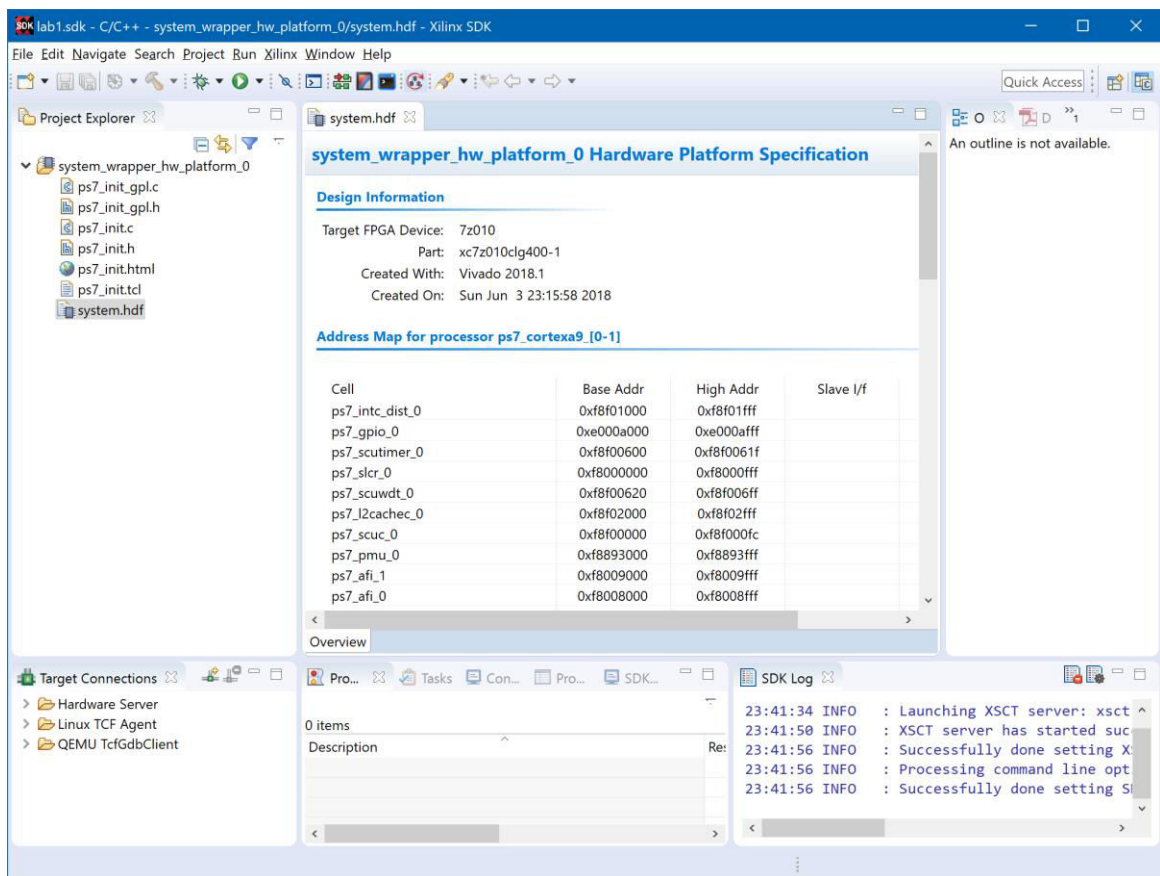


Figura 21. Vista de desarrollo de SDK C/C++

El archivo system.hdf (Hardware Description File) para la plataforma de Hardware debería abrirse en el panel de preview. Efectuar un doble click sobre system.hdf para abrirlo si es que no lo hizo.

En el archivo .hdf se encuentra la información básica sobre la configuración del hardware del proyecto, junto con los mapas de direcciones para los sistemas de PS, e información de driver. El archivo .hdf es usado en el entorno de software para determinar los periféricos disponibles en el sistema, y su ubicación en el mapa de direcciones.

Generate Memory TestApp in SDK

Paso 4

4-1. Generar la aplicación de prueba de la memoria usando uno de los modelos de proyectos estándar.

4-1-1. En SDK, seleccionar **File ► New ► Application Project**

4-1-2. Nombrar el proyecto como **mem_test**, y en la sección *Board Support Package*, dejar seleccionado *Create New* y dejar el nombre por defecto *mem_test_bsp* y presionar **Next**. (Notar que esta aplicación se ejecutará sobre ps7_cortexa9_0, es decir sobre el core 0 de los dos cores disponibles del procesador)

The screenshot shows the 'New Project' dialog in the SDK. The dialog is titled 'Application Project' and contains the following fields and options:

- Project name:** mem_test
- ☒ Use default location
- Location:** C:\MyS_2018\Laboratorios\lab1\lab1.sdk\mem_test (with a 'Browse...' button)
- Choose file system:** default
- OS Platform:** standalone
- Target Hardware:**
 - Hardware Platform:** system_wrapper_hw_platform_0 (with a 'New...' button)
 - Processor:** ps7_cortexa9_0
- Target Software:**
 - Language:** C (selected) or C++
 - Compiler:** 32-bit
 - Hypervisor Guest:** N/A
 - Board Support Package:** Create New (selected) or Use existing. The 'Create New' option has a text field containing 'mem_test_bsp'.

At the bottom of the dialog, there are four buttons: '?', '< Back', 'Next >', and 'Finish' (which is highlighted with a blue border). There is also a 'Cancel' button.

Figura 22. Crear un nuevo proyecto de aplicación en SDK

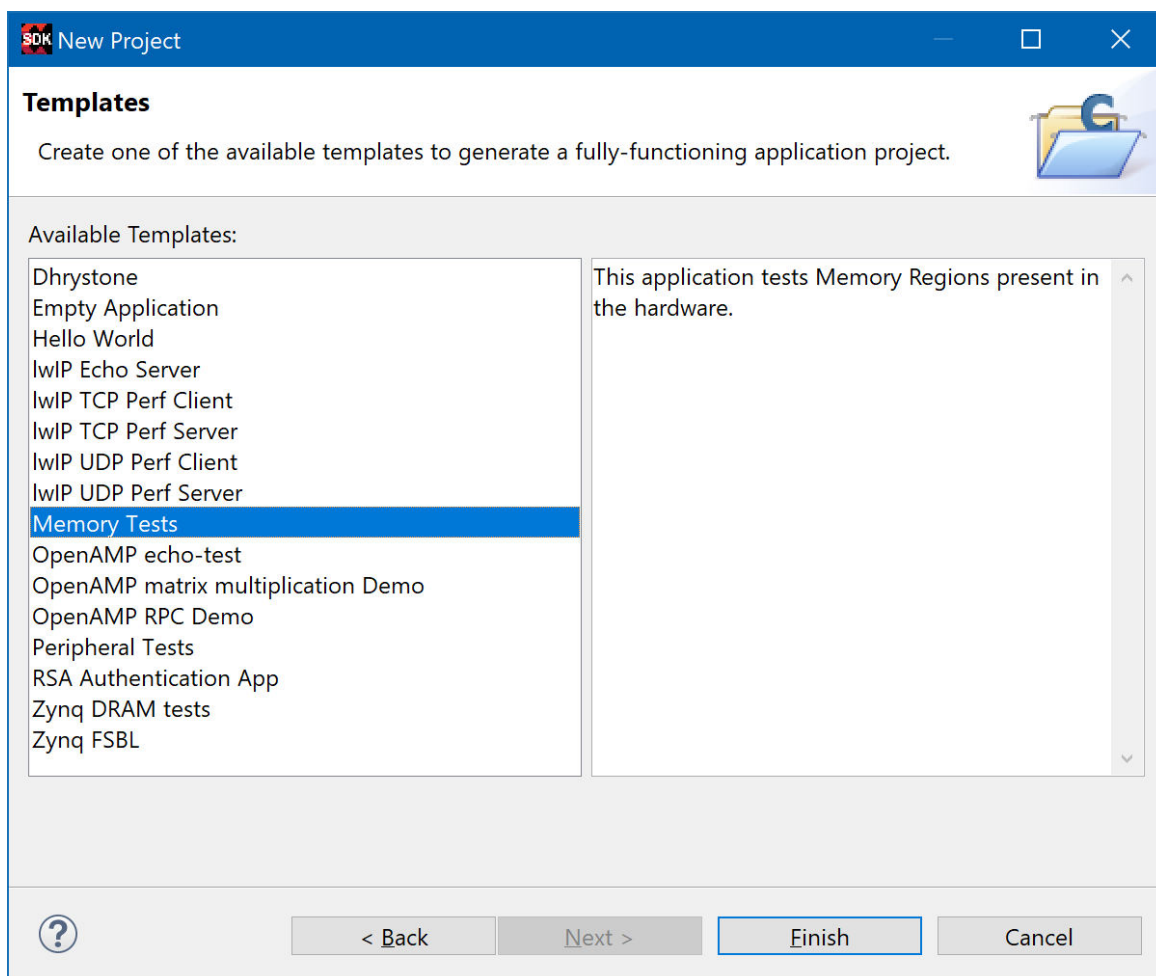
4-1-3. Seleccionar **Memory Tests** desde la ventana *Available Templates*, y presionar **Finish**.

Figura 23. Creando un proyecto en C de prueba de memoria

El proyecto **mem_test** y el proyecto board support **mem_test_bsp** serán creados y serán visibles en la ventana del Project Explorer del SDK, y los dos proyectos serán implementados automáticamente (build). Puede monitorear el progreso en el panel *Console*.

4-1-4. Expandir las carpetas en la vista Project Explorer, y observar que hay tres proyectos – *system_wrapper_hw_platform_0*, *mem_test_bsp*, y *mem_test*. El proyecto *mem_test* es la aplicación que usaremos para verificar la funcionalidad del diseño. La *hw_platform* incluye la función *ps7_init* que inicializa el PS como parte de la primera etapa del bootloader, y *mem_test_bsp* es el board support package.

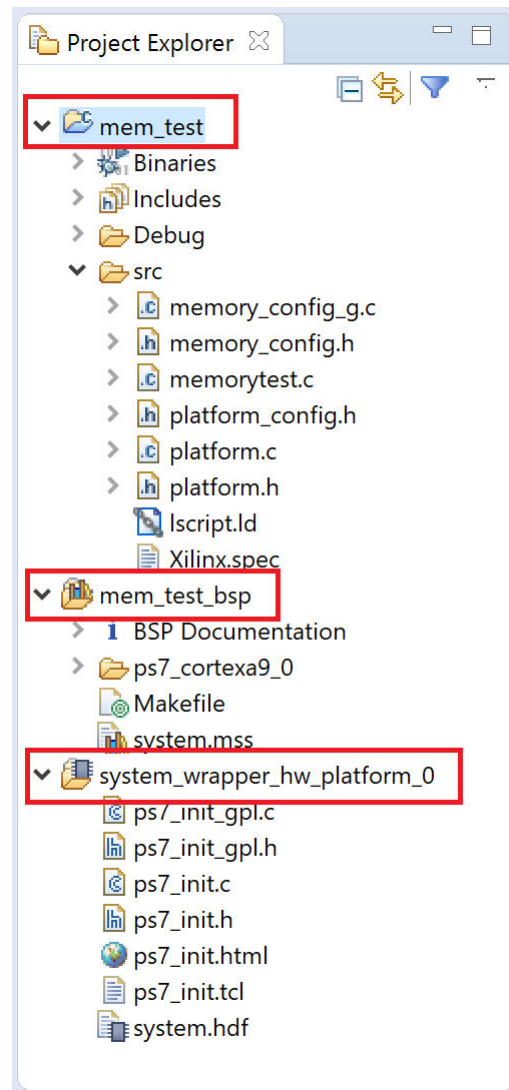


Figura 24. Vista Project Explore

- 4-1-5. Abrir el archivo **memorytest.c** en el proyecto **mem_test** (debajo de **src**), y examinar los contenidos. Este archivo llama a las funciones para probar la memoria.

Prueba en Hardware

Paso 5

- 5-1. **Asegúrese de que el jumper JP5 esté configurado para seleccionar alimentación por USB y el J4 como modo JTAG. Conectar la placa con un cable micro-usb.**

Establecer la comunicación serie usando la pestaña *Terminal* del SDK.

- 5-1-1. Asegúrese que el jumper JP5 seleccione USB, y el JP4 JTAG. Asegúrese que el cable micro-USB esté conectado al conector JTAG PROG (al lado del conector USB Host).

- 5-1-2. Seleccionar la pestaña .  **Terminal**

Si no es visible entonces seleccione **Window ► Show view ► Other ► Terminal ► Terminal**.

Se puede utilizar cualquier terminal disponible.

- 5-1-3.** Hacer click sobre el icono connect  y si es requerido, seleccionar el puerto COM apropiado (depende de la computadora),y configurarlo con los parámetros como se muestra a continuación.

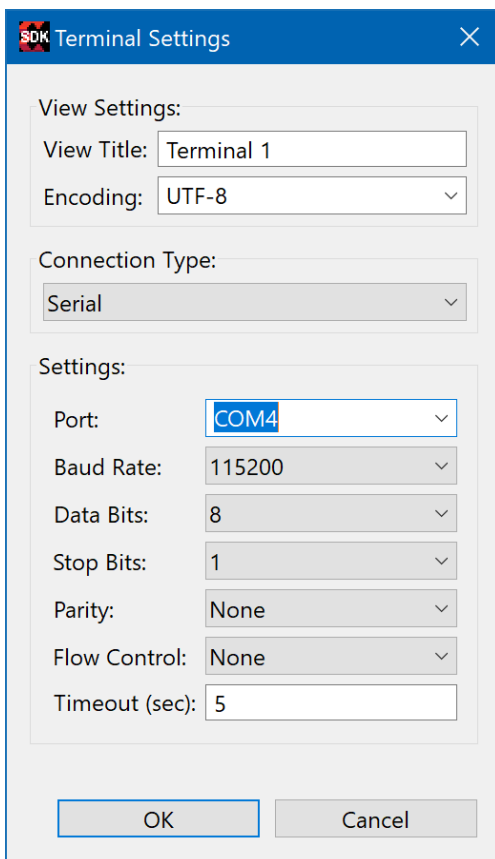


Figura 25. Configuración del terminal SDK

Se puede encontrar el Puerto COM en el Manejador de Dispositivos de Windows, en este caso, COM4:

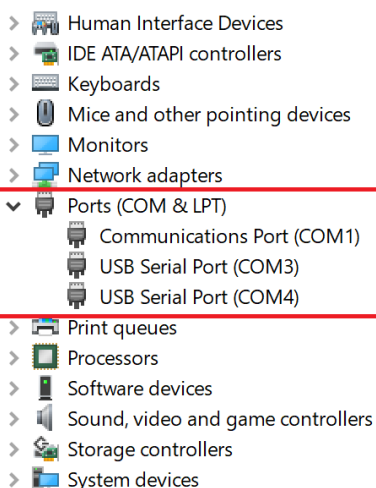


Figura 26. Listado de puertos COM en el manejador de dispositivos de Windows

5-2. Ejecutar la aplicación mem_test y verificar la funcionalidad.

- 5-2-1.** En el SDK, seleccionar el proyecto **mem_test** en el *Project Explorer*, click-derecho y seleccionar **Run As ► Launch on Hardware (GDB)** para descargar la aplicación, y ejecutar ps7_init, y entonces ejecutar mem_test.elf (user application).

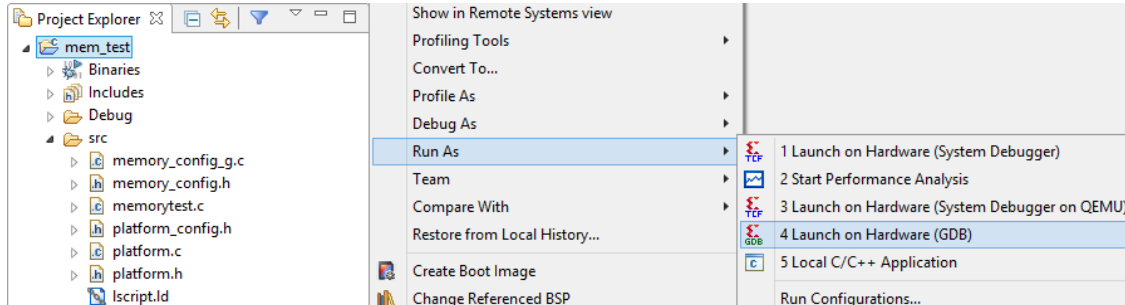


Figura 25. Ejecutar la aplicación

- 5-2-2.** En la pestaña *Terminal* deberá ver la siguiente salida.

```
--Starting Memory Test Application--
NOTE: This application runs with D-Cache disabled.As a result, cacheline request
s will not be generated0}
Testing memory region: ps7_dds_0
    Memory Controller: ps7_dds_0
        Base Address: 0x100000
            Size: 0x1FF00000 bytes
            32-bit test: PASSED!
            16-bit test: PASSED!
            8-bit test: PASSED!
Testing memory region: ps7_ram_1
    Memory Controller: ps7_ram_1
        Base Address: 0xFFFF0000
            Size: 0xFE00 bytes
            32-bit test: PASSED!
            16-bit test: PASSED!
            8-bit test: PASSED!
--Memory Test Application Complete--
```

Figura 26. Salida de la terminal SDK

- 5-2-3.** Cerrar SDK y Vivado seleccionando **File ► Exit** en cada programa.

Conclusión

Vivado e IP Integrator permiten generar muy rápidamente sistemas base con procesador integrado y aplicaciones. Después de que el sistema ha sido definido el hardware puede ser exportado y el SDK puede ser llamado desde Vivado. El desarrollo del software se realiza en el SDK, el cual provee varias plantillas de aplicaciones, incluyendo pruebas de memoria. Se puede verificar la operación del hardware descargando una aplicación de prueba, ejecutándola en el procesador, y observando la salida en la ventana de la terminal serie.