Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ingeniería Eléctrica
IEE2463 Sistema electrónicos programables

Ayudantia 08 2S23

ZYBOZ7 - BOOT y Debug con Vitis

Ayudante: Catalina Sierra catalina.sierra@uc.cl
Prof. Dr.-Ing. Félix Rojas - felix.rojas@uc.cl

---

1. Objetivos de la Ayudantía

- Aprender a cargar un código de BOOT a la Zybo programando la memoria *flash*.
- Comprender cómo usar el modo Debug en Vitis.
- Ejercitar el uso de funciones, macros y arreglos.

2. Actividades Previas a la Ayudantía

En el video de la ayudantía se explica y desarrolla el *hardware* y código C utilizados, además del uso del modo Debug en Vitis y cómo cargarle un código de BOOT a la Zybo programando su memoria *flash*. Para el desarrollo de esta ayudantía se debe:

- Crear un nuevo proyecto en Vivado asociado a la ZyboZ7-10 y en el mismo un *Block Design* donde se conectarán los módulos Zynq7 Processing System y AXI Gpio de acuerdo a lo visto en el video de la ayudantía.
- Incluir archivo *Constraints* asociado a la tarjeta y editarlo de acuerdo al *Block Design* desarrollado.
- Generar el *HDL Wrapper* del *Block Design* y luego el *bitstream*, para exportar el *hardware* y luego abrir Vitis desde Vivado.
- Una vez en Vitis crear un *Platform Project* con el archivo xsa generado y buildear. Tras esto, crear un *Application Project* vacío, luego en el mismo en la carpeta `src` haciendo click derecho añadimos un archivo C, el cual se corresponderá con el archivo boot_sw.c disponible en el repositorio. Luego buildear nuevamente.

- **Debuggeo**: En Vitis vaya a Debug (esquina superior derecha, al lado de Design) y clickear. Una vez en el modo Debug, vaya a la ventana Explorer en la esquina inferior izquierda y clickeando sobre `Boot_SW_System` → `Debug As` → `1 Launch Hardware`. Tras lo anterior ya debería estar andando y debería visualizar algo como la imagen que se muestra a continuación.

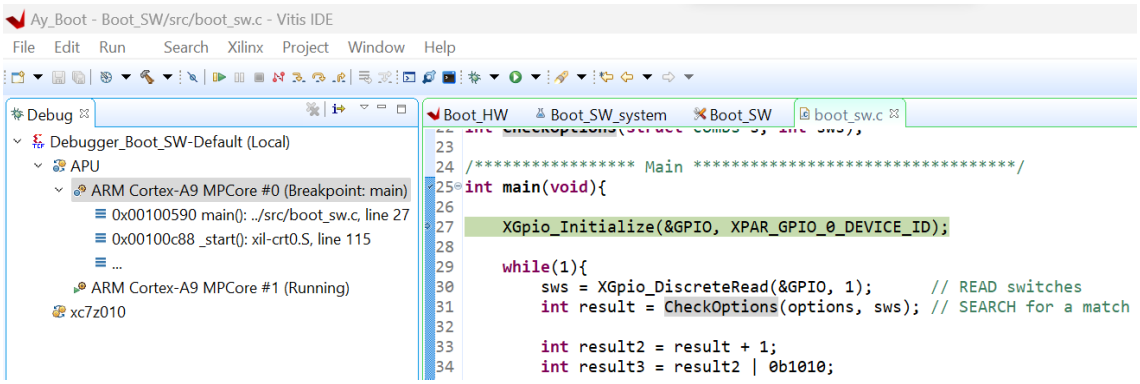

Figura 1: Vista al debugear correctamente en Vitis

El debuggeo y la adición de expresiones para su visualización se explican en el video de la ayudantía, pero tenga en consideración que con la flecha `Step Over (F6)` debería poder ir avanzando línea por línea en el código. Finalmente, cabe mencionar que generalmente hay varias opciones al seleccionar `Debug As`. En esta ayudantía se usa una de forma arbitraria, pero pueden probar con otras formas y no debería haber problema.

- **Booteo**: Para generar el archivo `BOOT.bin` necesario para programar la memoria flash de la Zybo, considerando el código desarrollado en la ayudantía, debe seguir los siguientes pasos:

  1. Build Project
  2. Click derecho sobre `Boot_SW` → `Create Boot Image`
  3. Una vez en el *wizard* respectivo, añadir los archivos y settear los parámetros de acuerdo a lo visto en la ayudantía.
  4. `Xilinx` → `Program Flash` → `Program`, cuidando que Flash Type se encuentre en la opción qspi-x4-single.
  5. Una vez programada la memoria *flash* de la Zybo, desconéctela del PC (desenergícela) y cambie el tope azul en JP5 de JTAG a QSPI. Tras ello vuelva a alimentar la Zybo. Debería ver en funcionamiento la secuencia de BOOT que se trabajó en la ayudantía y cargó a la memoria *flash*, pudiendo mover los *switches* y ver cómo cambian los *leds* en base a ello.

3. Actividades Durante la Ayudantía

El ejercicio propuesto para esta ayudantía consiste en modificar el código C desarrollado creando una nueva función. A continuación se presentará una serie de pasos para orientar el desarrollo de esta actividad.

- Crear una nueva función `CheckParity` que revise si el resultado final (`result3`) es par o no, y que en base a ello el despliegue del resultado final en los leds ya no sea fijo, si no que sea parpadeante, con un parpadeo lento para indicar imparidad y uno rápido para indicar paridad (el criterio para esta diferenciación queda a su parecer). Para esto pueden serle de utilidad las funciones `XGpio_DiscreteWrite` y `usleep`.
- Tras la edición → `Build Project`.
- Cárguele el código a la Zybo por **JTAG** como normalmente lo hace, verifique el funcionamiento de su implementación y, de ser necesario, emplee el modo de Debug para solucionar incongruencias que encuentre.

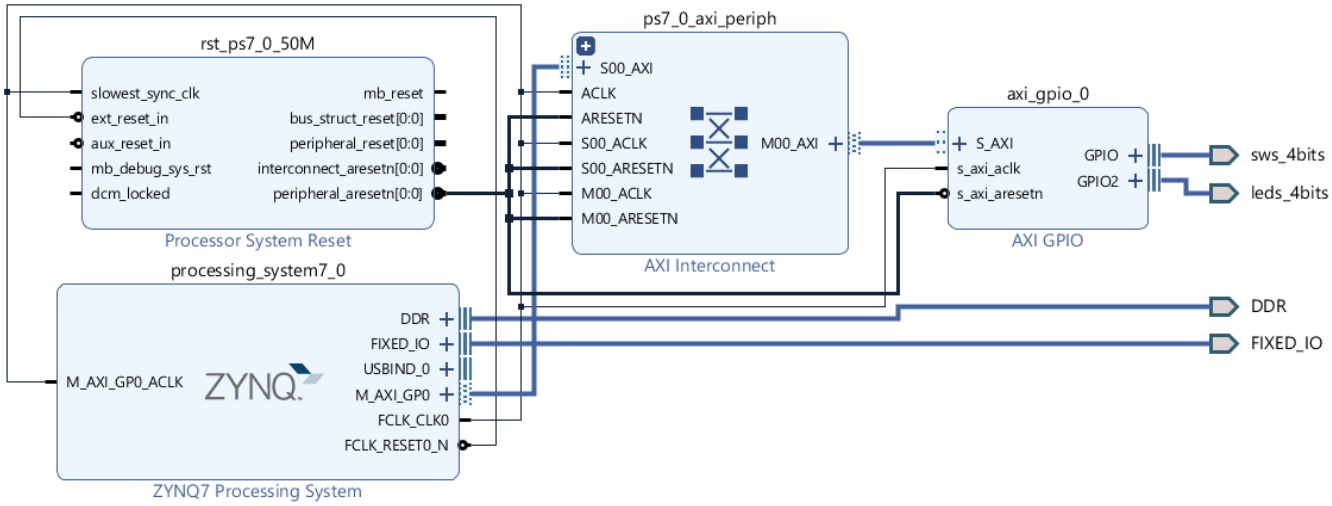

Figura 2: *Block Design* en Vivado del proyecto