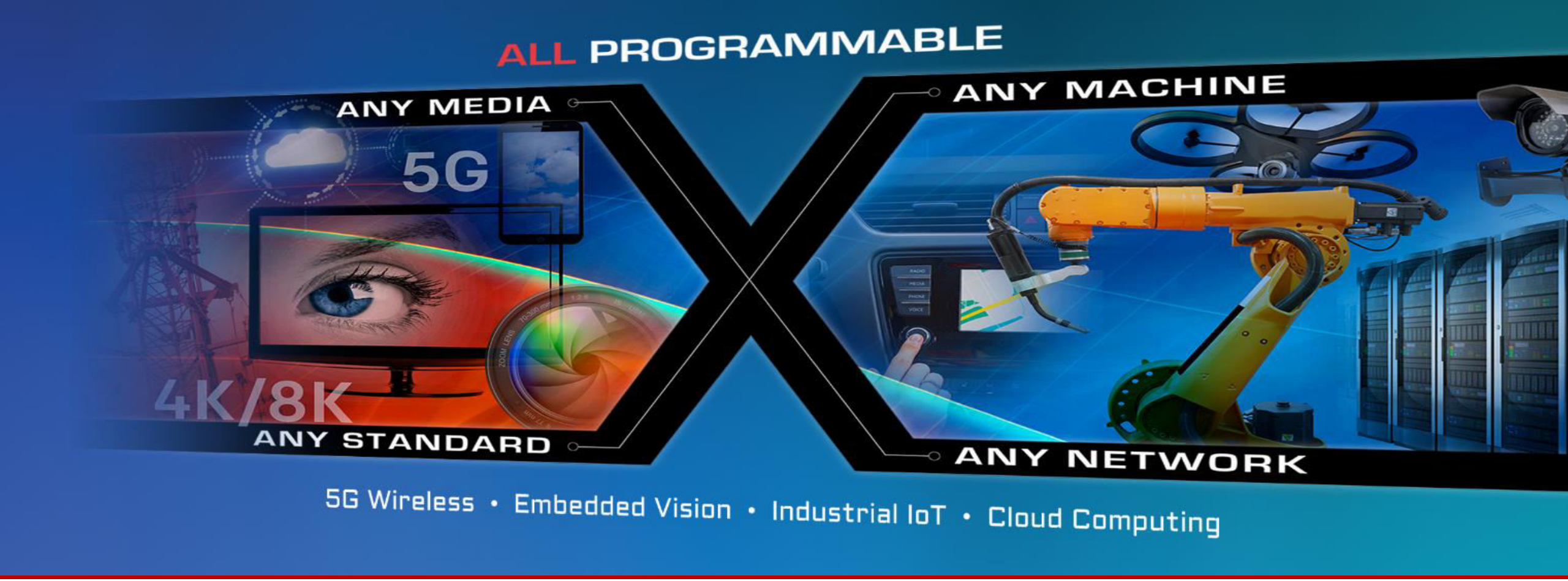


XILINX
ALL PROGRAMMABLE™

Depuración del Hardware

Contenido

- Conocer las opciones disponibles de depuración en el hardware
- Describir los distintos bloques para depuración y su funcionalidad
- Conocer cómo se incluyen los bloques de depuración en el sistema

Temario

- **Introducción**
- Bloques de Depuración
- Proceso de depuración
- Resumen

Depuración

- La depuración y verificación lleva alrededor del 40% del tiempo de desarrollo
- Es un proceso lento por la baja velocidad de las interfaces de comunicación
- Si la depuración no forma parte integral del desarrollo, genera retrasos innecesarios en el ciclo de desarrollo

Metodología de depuración

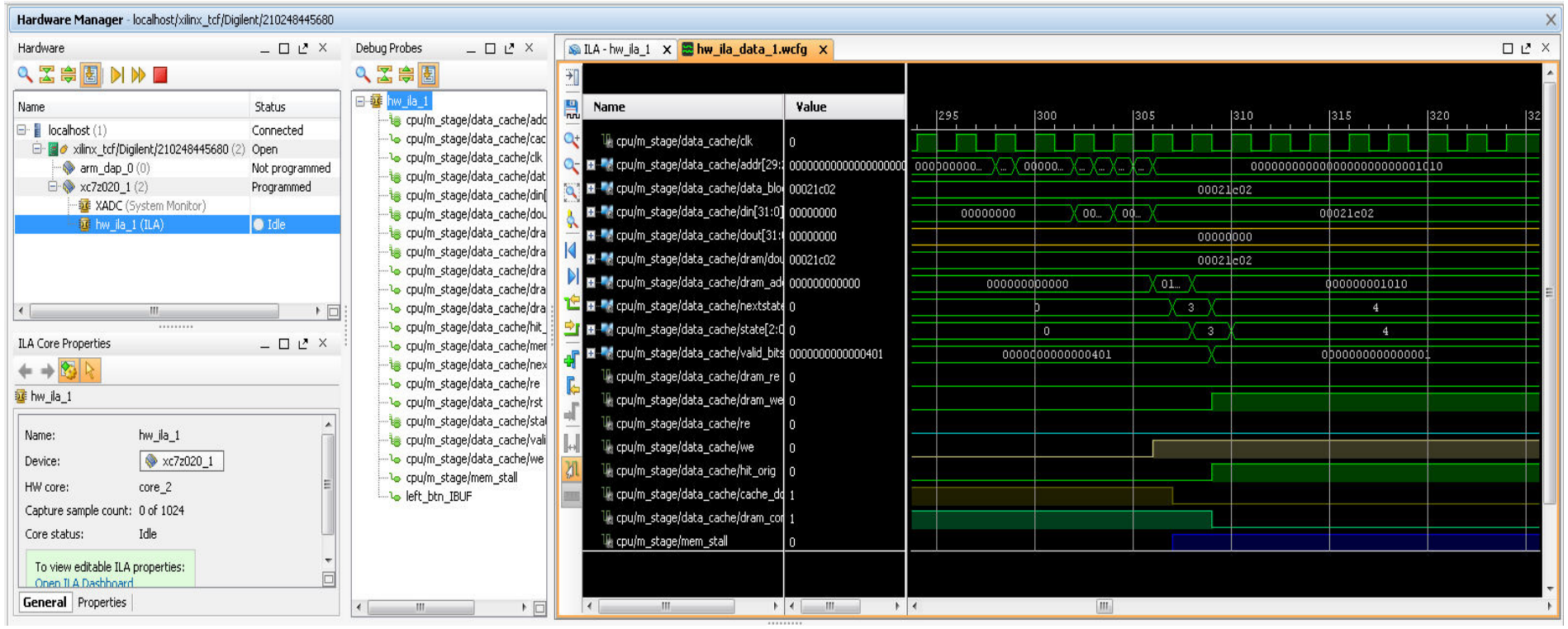
- Encarar el proceso de depuración de una manera sistemática
  - *Descomponer el problema en partes*
  - *Simplificar reduciendo variables y variabilidad (reducción de funciones)*
  - *Verificar los resultados contra salidas generadas por otros métodos*
  - *Planificar cómo y dónde aplicar depuración en el sistema*
- El diseño de sistemas con FPGA es un proceso iterativo
- Depurar un sistema basado en FPGA también es un proceso iterativo
  - ***1) Obtención de datos**: Agregar o modificar los bloques de depuración*
  - ***2) Implementación**: Compilar el sistema con los bloques de depuración configurados*
  - ***3) Analizar**: Analizar los resultados obtenidos y verificar si son correctos funcionalmente y en temporización*
  - ***4) Reparar**: Realizar las modificaciones necesarias y repetir el proceso*

Depuración

- **Herramienta *Hardware debugging***
  - Es un analizador lógico para las señales internas de la FPGA
  - Reemplaza el uso de un analizador lógico externo
- **Forma parte de las herramientas de desarrollo**
- **La misma conexión JTAG se utiliza en dos funciones:**
  - Configurar la lógica programable de la FPGA
  - Depuración del Hardware

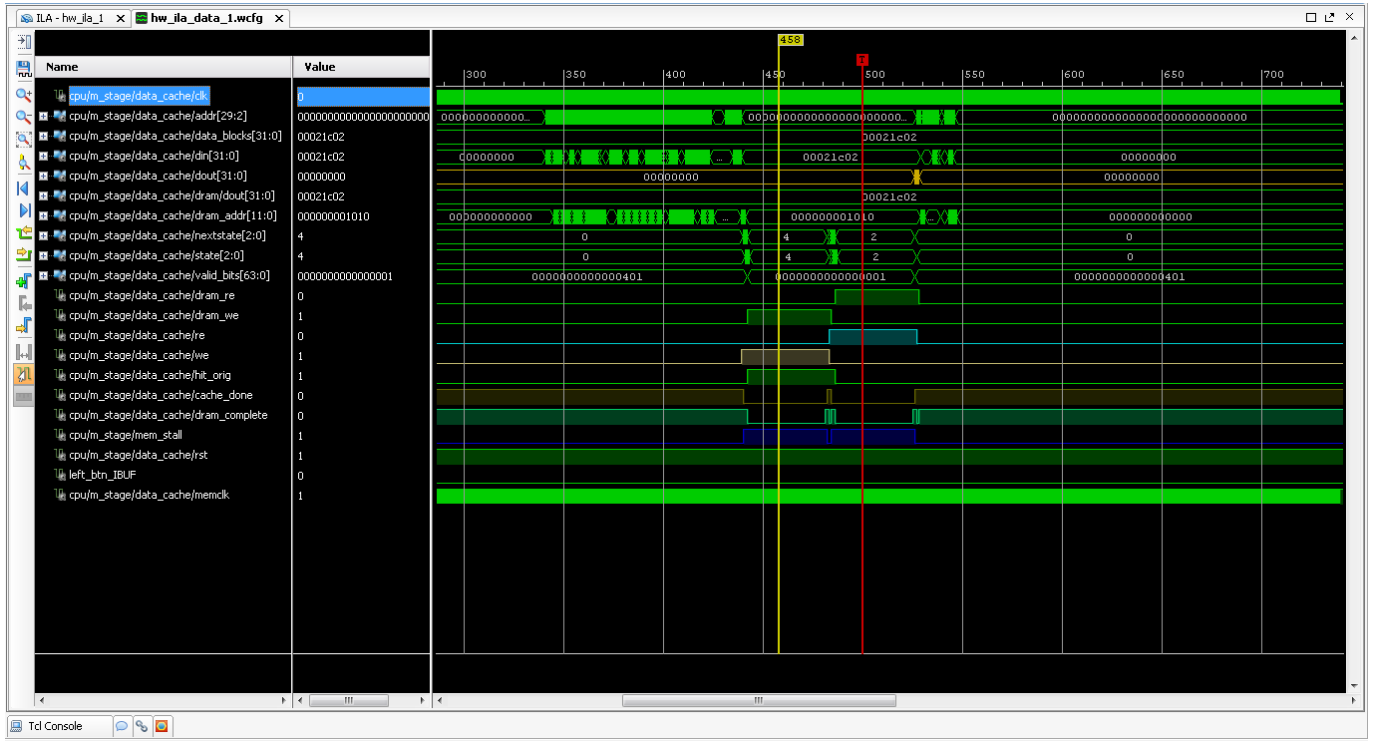
Analizador lógico integrado

- **Interfaz de software en tiempo de ejecución para interactuar con los bloques ILA y VIO**
- **Se pueden ver los resultados de varios bloques ILA simultaneamente**
- **Los resultados se pueden guardar y recuperar**



Analizador lógico integrado

- **Funciones**
  - Cursores y marcadores con medición de tiempos
  - Ampliación/Reducción
  - Herramientas para encontrar las transiciones de las señales
  - Busqueda de señales por el nombre
  - Representación de números en distintas bases



Analizador lógico integrado

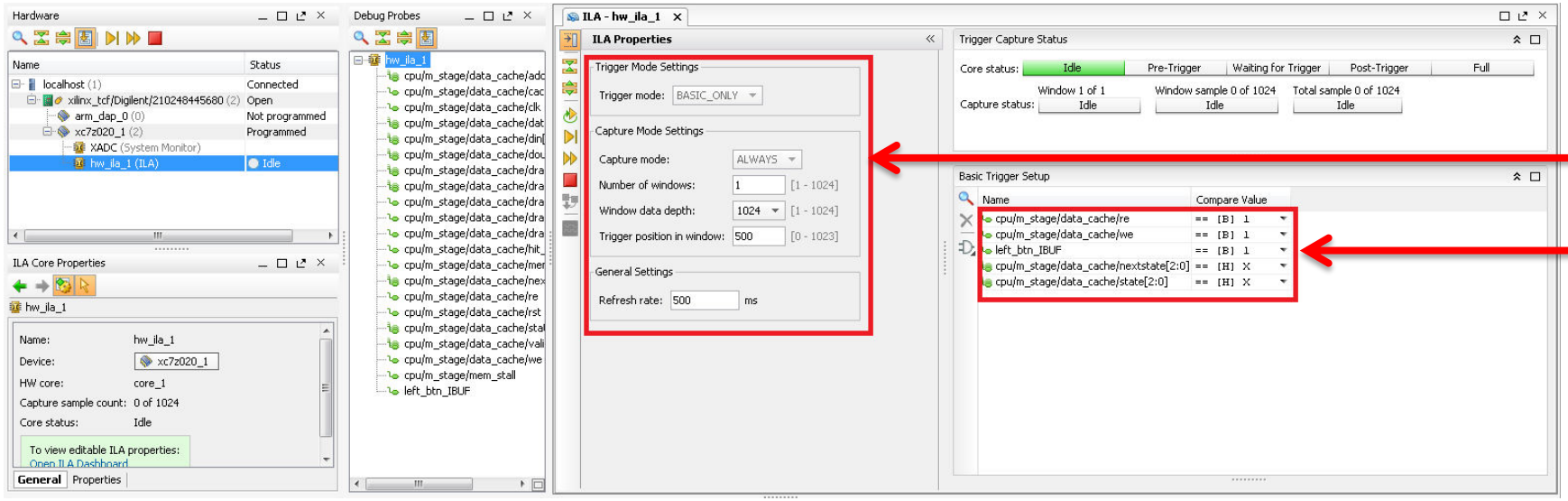
Ejecucion de comandos Tcl

- **Permite la automatización del proceso de depuración**
  - Se utiliza de la misma manera que para otros procesos
  - Las pruebas se pueden ejecutar en forma interactiva o automática
  - Se pueden guardar los resultados para posterior análisis
- **Se pueden crear funciones y pruebas personalizadas**
  - Permite realizar pruebas de regresión
  - Se pueden asociar secuencias de comandos Tcl a botones del IDE
  - La ejecución de scripts permite la integración con otros entornos de depuración

```
1  ##################################################################
2  # Connect to Digilent cable connected to my lab123
3  ##################################################################
4  connect_hw_server -host lab123:50001
5  current_hw_target [get_hw_targets */digilent_plugin/SN:12345]
6  open_hw_target
7
8  ##################################################################
9  # Program device with design.bit file and refresh device
10 ##################################################################
11 current_hw_device [lindex [get_hw_devices] 0]
12 set_property PROGRAM.FILE {./design.bit} [current_hw_device]
13 program_hw_devices [current_hw_device]
14 refresh_hw_device -update_hw_probes [current_hw_device]
15
16 ##################################################################
17 #Set up ILA core trigger position and probe compare value
18 ##################################################################
19 current_hw_ila [get_hw_ilas hw_ila_1]
20 set_property CONTROL.TRIGGER_POSITION 512 [current_hw_ila]
21 set_property COMPARE_VALUE.0 eq1'b0 [get_hw_probes PROBE_4]
22
23 ##################################################################
24 # Arm and wait for ILA core trigger, upload and save data
25 ##################################################################
26 run_hw_ila hw_ila_1
27 wait_on_hw_ila hw_ila_1
28 write_hw_ila_data [upload_hw_ila_data hw_ila_1] wavedata
29
```

Analizador lógico integrado

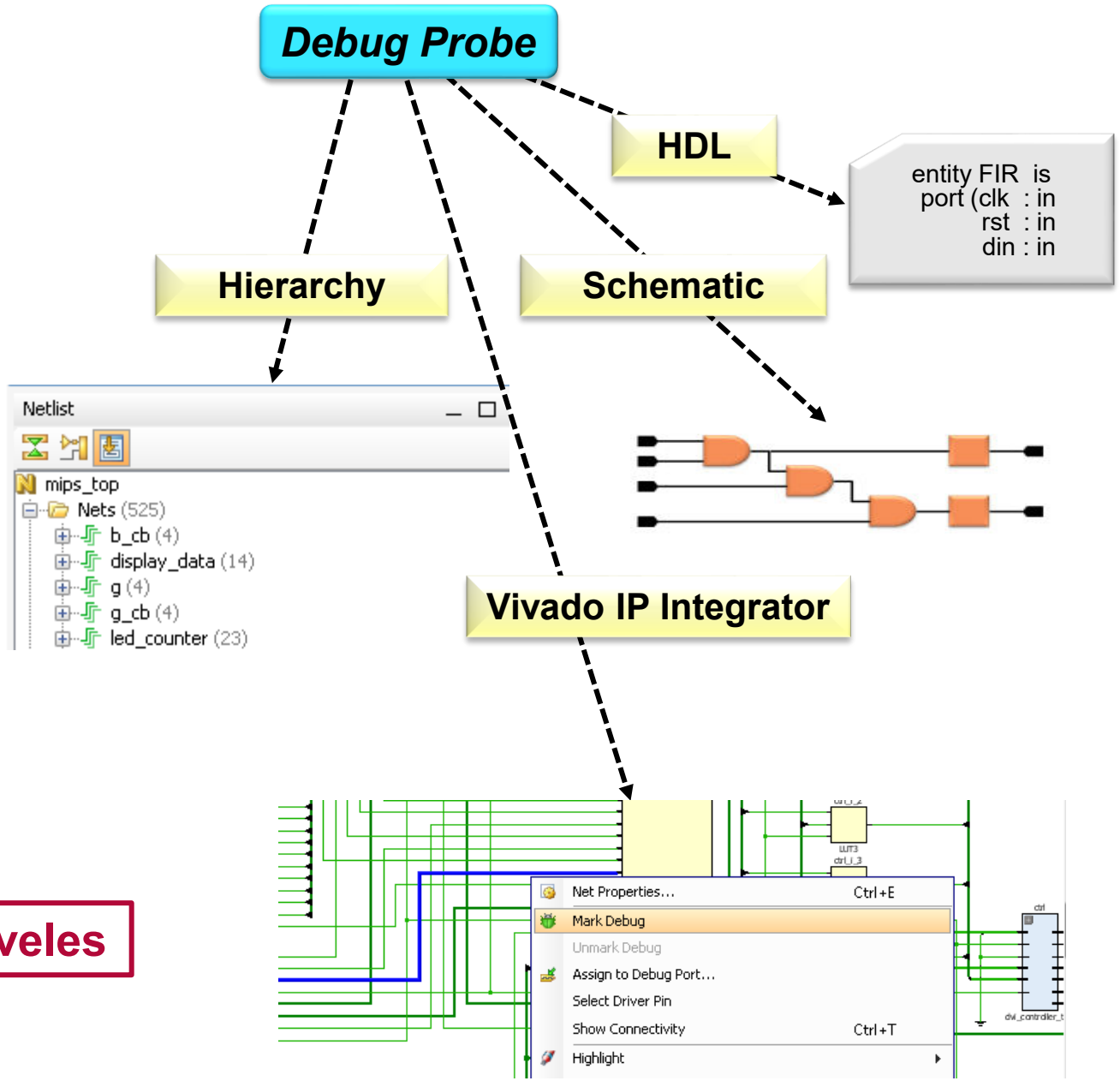
- **Una unidad de comparación (match unit) por cada señal a verificar**
  - Se pueden hacer todos los tipos de comparación (</>/=/!=/etc)
- **No es necesario instanciar un bloque de depuración específico para la conexión**
- **La mayoría de los parámetros de depuración se configuran en tiempo de ejecución**



Analizador lógico integrado

- Se pueden definir señales a depurar en el código HDL mediante la propiedad MARK_DEBUG
- Se puede depurar un sistema sintetizado
- Se puede depurar a nivel sistema, incluyendo IP agregada

Se puede depurar el sistema a distintos niveles

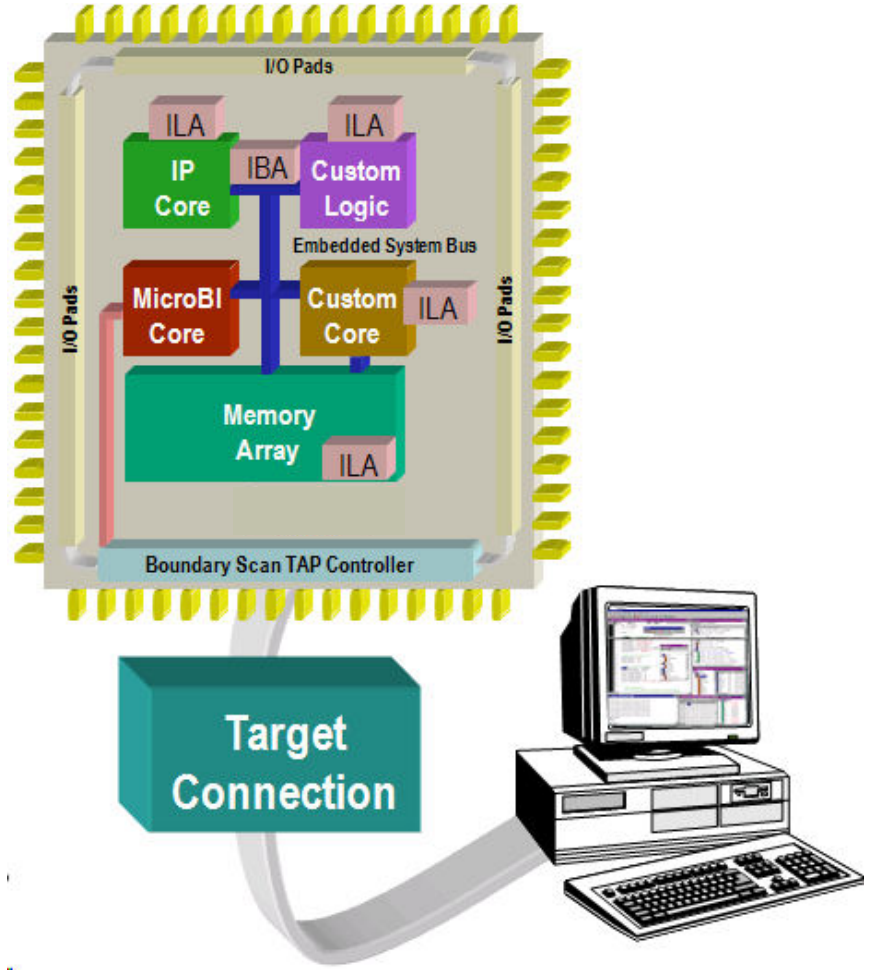


Temario

- Introducción
- **Bloques de Depuración**
- Proceso de depuración
- Resumen

Estructura de los bloques de depuración

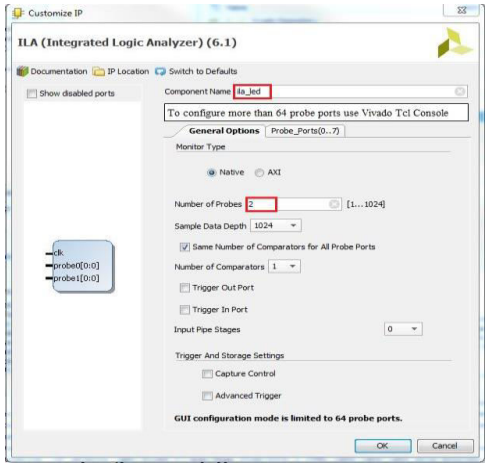
- **Los bloques de depuración permiten acceder a todas las señales internas de la FPGA**
  - Puertos de los bloques físicos
  - Señales intenas, puertos y nodos de la lógica
  - Se pueden simular señales externas mediante el bloque de depuración Virtual I/O core
- **La depuración se ejecuta a la velocidad del sistema**
  - Se utilizan los relojes definidos en el sistema
- **No se utilizan pines de I/O**
  - Se accede mediante la interfaz JTAG



Bloques de depuración

- **ILA**
  - Bloque para adquirir señales (Integrated Logic Analyzer)
  - Permite agregarse como Netlist o como código HDL
- **VIO**
  - Bloque para simular señales de entrada/salida (Virtual Input / Output)
  - Se agrega como código HDL

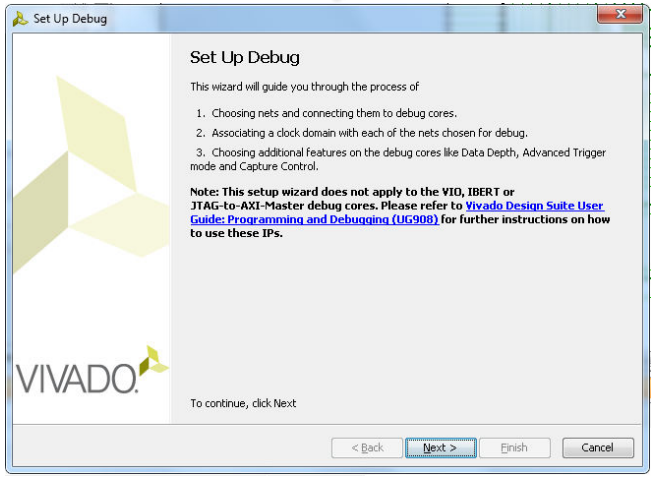
ILA 6.1 and VIO 3.0
HDL Instantiation




```
ila_v4_0_ila #(
.C_XLNX_HW_PROBE_INFO("NUM_OF_PROBES=1,DATA_DEPTH=1024,PROBE0_WIDTH=1,
.C_XDEVICEFAMILY("zynq"),
.C_CORE_TYPE(1),
.C_CORE_INFO1(0),
.C_CORE_INFO2(0),
.C_CAPTURE_TYPE(0),
.C_MU_TYPE(0),
.C_TC_TYPE(0),
.C_NUM_OF_PROBES(1),
.C_DATA_DEPTH(1024),
.C_MAJOR_VERSION(2013),
```

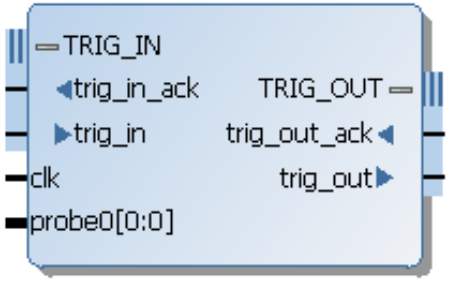
ILA 6.1
Netlist Insertion

Mark Debug Nets



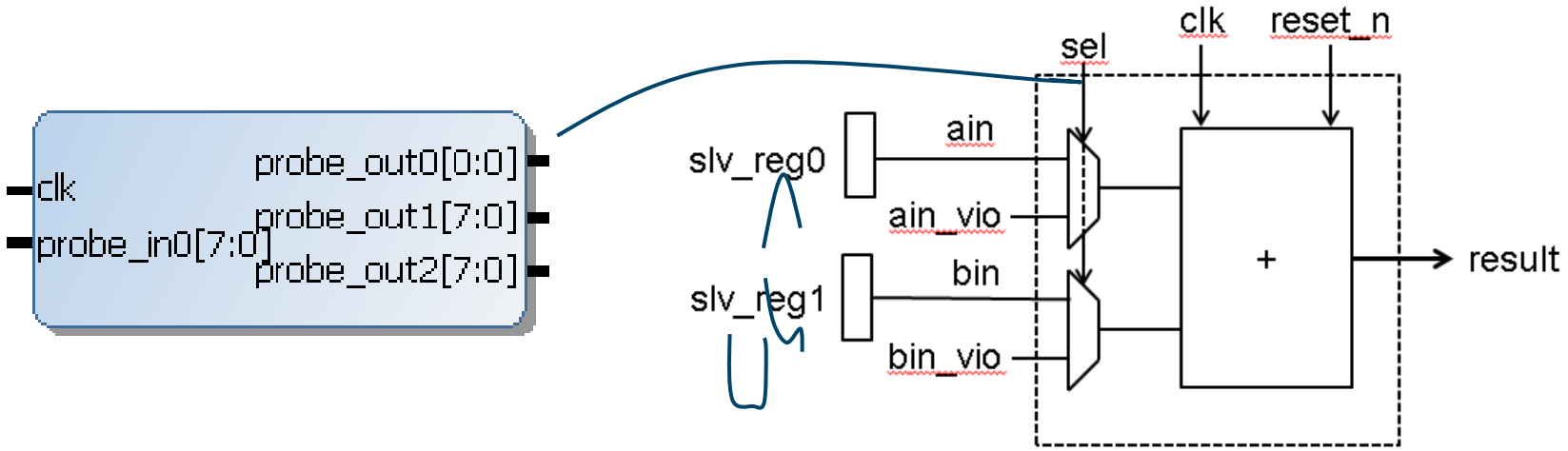
Bloque ILA

- Se utiliza para monitorear señales internas de la lógica programable
- Las unidades de disparo para adquisición son configurables
  - Se puede configurar el tamaño y el tipo de coincidencia para trabajar con distintos tipos y condiciones de las señales
- Entradas separadas de datos y disparo: una señal genera la condición de disparo y otras señales son adquiridas
- Se pueden capturar los datos antes, durante y después de la condición de disparo



Bloque VIO

- Permite simular señales de entrada/salida en tiempo real
- Tiene una unidad para generar las señales de entrada
- Tiene una unidad para almacenar las señales de salida



Propiedad *Mark_Debug*

- **Se puede usar esta propiedad para marcar señales para depuración en el código HDL**
- **En VHDL:**

```
attribute mark_debug : string;
attribute mark_debug of char_fifo_dout: signal is "true";
```

- **En Verilog:**

```
(* mark_debug = "true" *) wire [7:0] char_fifo_dout;
```

Temario

- Introducción
- Bloques de Depuración
- **Proceso de depuración**
- Resumen

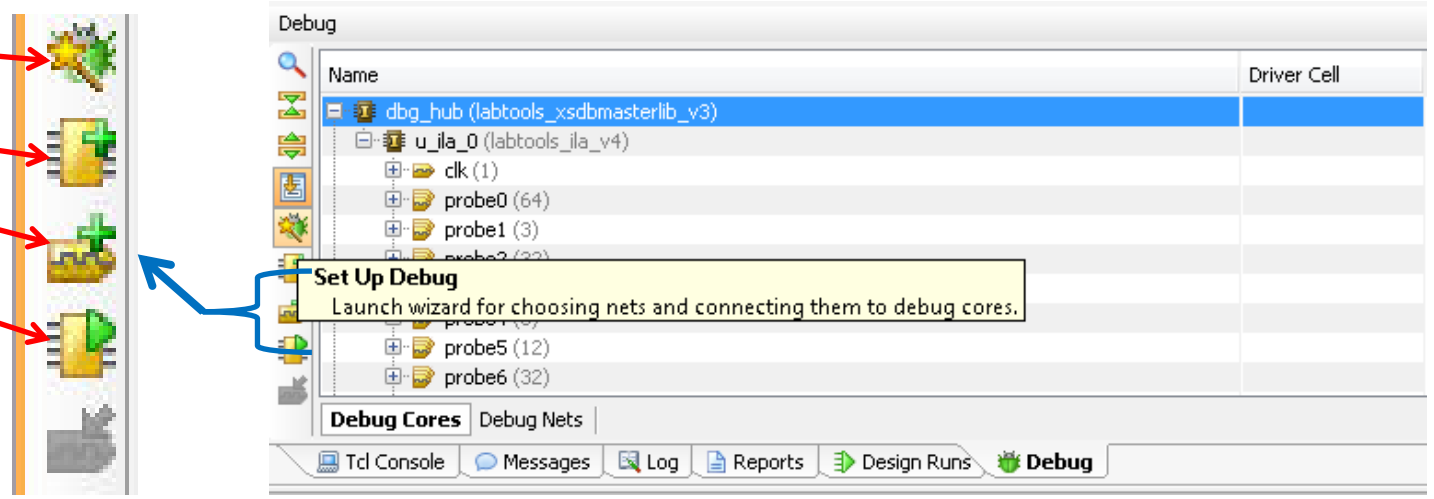
Proceso de depuración

- **Inserción como Netlist**
  - Es la forma más flexible
  - Se puede hacer en distintas etapas (HDL, sistema sintetizado, sistema implementado)
  - Se puede aplicar en desarrollos basados en proyectos o en desarrollos independientes
- **Instanciado en el código HDL**
  - Es la forma mas común
  - Se utiliza a nivel código HDL solamente
  - Se puede aplicar en desarrollos basados en proyectos o en desarrollos independientes
- **Se pueden aplicar ambos simultaneamente**

Herramientas de depuración en la interfaz gráfica

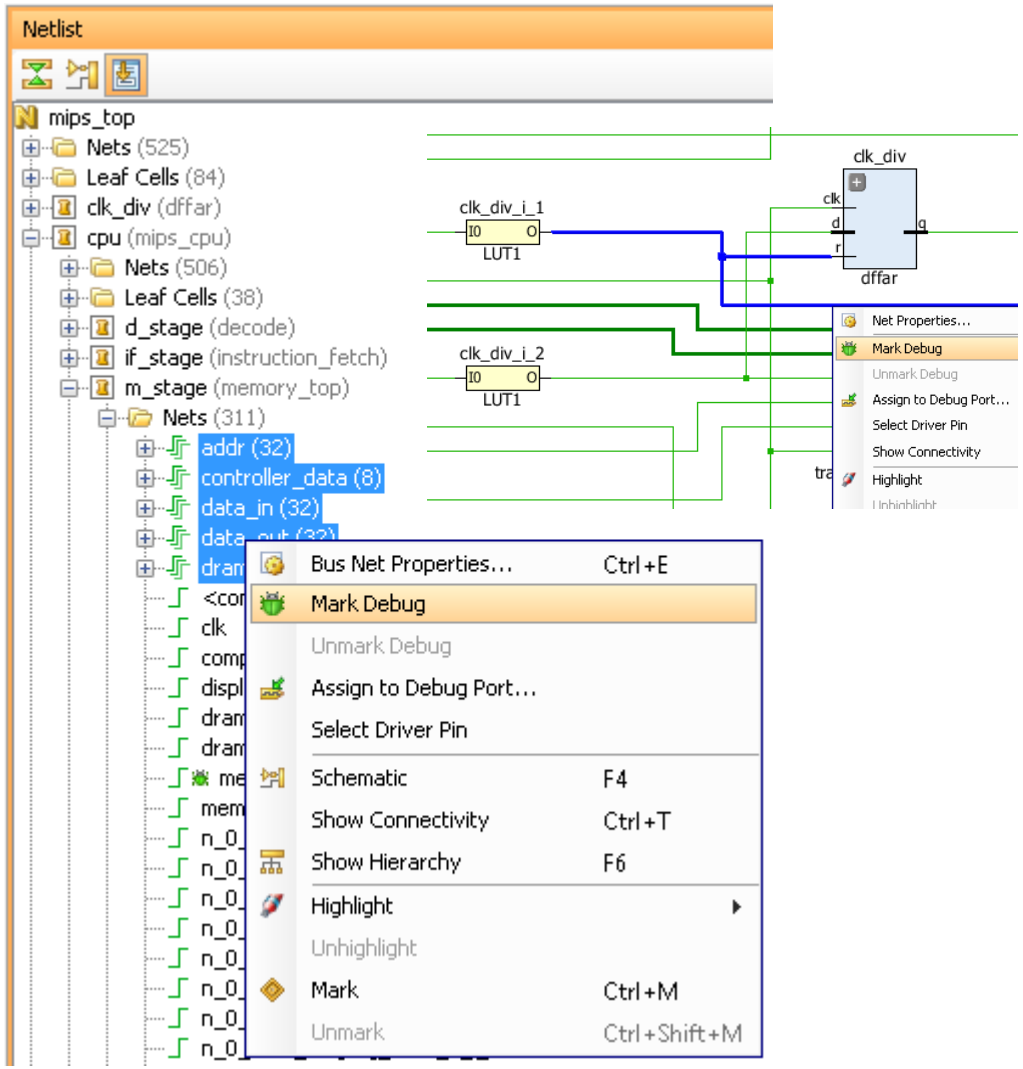
- **Seleccionar Tools ➔ Set up Debug para iniciar el asistente de depuración**
- **La pestaña de depuración (Debug) aparece en la vista de sistema sintetizado**

  - Agregar señales
  - Agregar bloques
  - Crear puertos o bloques
  - Implementar los bloques



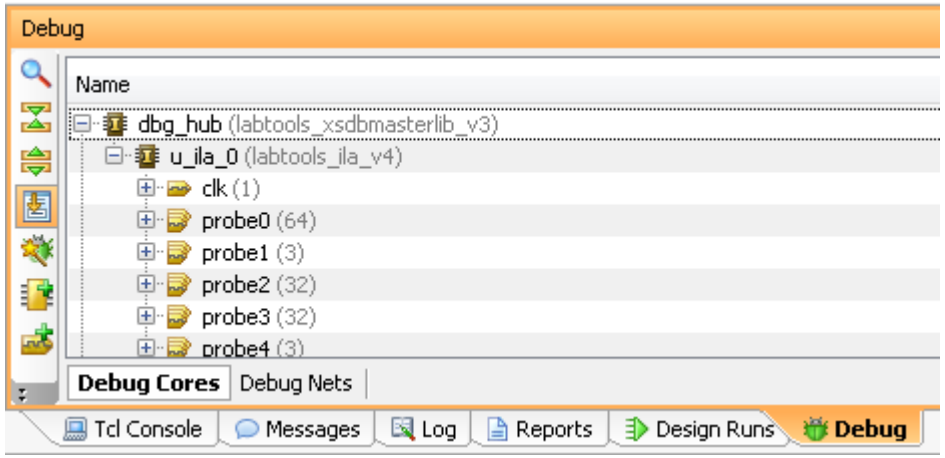
Selección de las señales para depuración

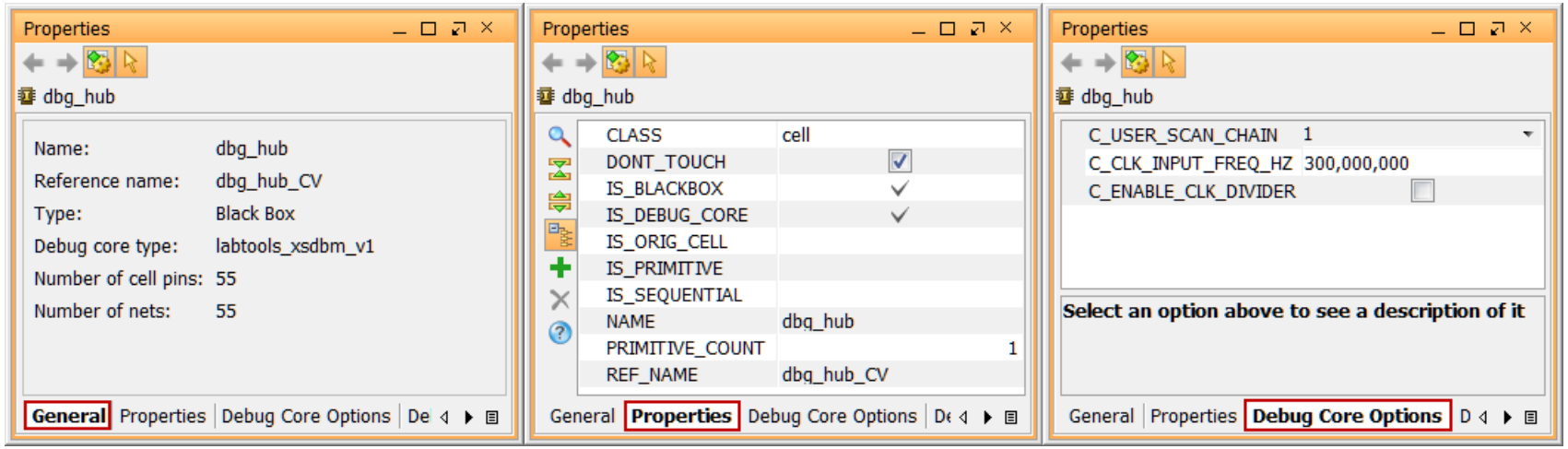
- **Como seleccionar las redes con las señales a verificar**
  - En la vista de redes (Netlist view) en la carpeta con la red (nets folders)
    - En esta vista está la estructura jerárquica de las señales
  - En el diagrama esquemático
  - Con la herramienta de búsqueda
- **Presionar con el botón derecho la red y seleccionar *Mark Debug***
- **En la vista de depuración, las redes sin asignar**
  - Aquí se ubican las redes de IP que no están configuradas
- **En el asistente de depuración también está disponible la herramienta de búsqueda**



Configuración de la herramienta de depuración

- **Se puede verificar el contenido y la configuración de las señales**
  - Señales de reloj (CLK)
  - Señales monitoreadas (PROBE)
  - Cantidad de señales en cada monitor
- **Se configuran las opciones en la pestaña de propiedades**





Temario

- Introducción
- Bloques de Depuración
- Proceso de depuración
- **Resumen**

Resumen

- **El analizador lógico integrado ILA tiene una interfaz similar a la del simulador y acepta comandos Tcl**
- **El tiempo y los mecanismos de depuración deben ser tenidos en cuenta como parte del ciclo de desarrollo de un sistema**
- **El analizador lógico integrado provee bloques para ver las señales internas del sistema y para simular las entradas y salidas externas a la velocidad del sistema**
  - Bloque ILA (señales internas)
  - Bloque VIO (señales de entrada/salida)
- **El asistente de depuración se utiliza para configurar los bloques lógicos y realizar las verificaciones utilizando una interfaz gráfica**
- **Hay 3 mecanismos para agregar los bloques de depuración**
  - Instanciado en el código fuente HDL
  - Inserción en la Netlist del sistema
  - Usando el atributo *Mark Debug*