AMD
XILINX

System Debugging using Logic Debug Cores and Vitis IDE

Objectives

- After completing this module, you will be able to:
  - List various available debug cores and their functionality
  - Describe the process of including the Vivado Design Suite debug tool sampling cores
  - State how the Vivado integrated logic analyzer, Vitis Debug perspective, and TCF tools facilitate hardware and software debugging

Outline

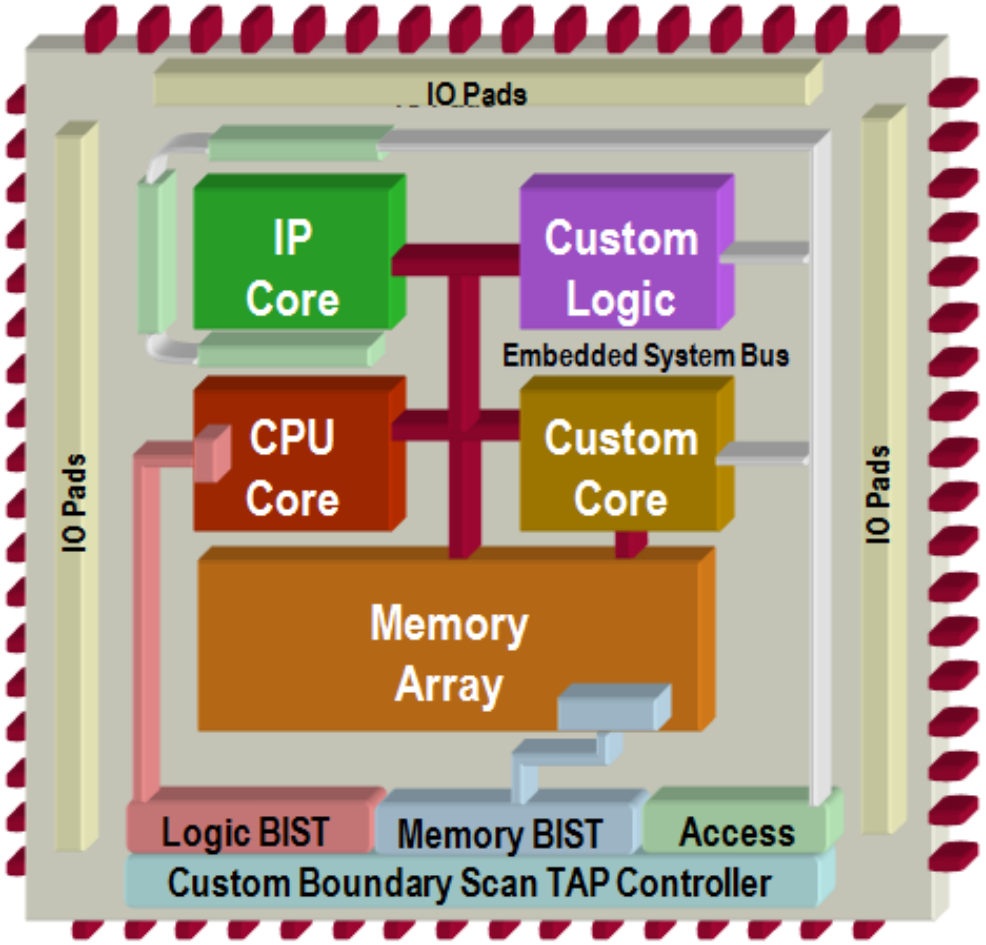
- *Introduction*
- Hardware Debugging
- Software Debugging
- System Debugging
- Summary

Introduction

- Debugging is an integral part of embedded systems development
- The debugging process is defined as testing, stabilizing, localizing, and correcting errors
- Two methods of debugging:
  - Hardware debugging via a logic probe, logic analyzer, in-circuit emulator, or background debugger
  - Software debugging via a debugging instrument
    - A software debugging instrument is dedicated hardware and part of the silicon that is accessible via JTAG or dedicated part pins
    - Controls the processor as an intrusive debug unit that is disabled during normal operation
    - Some "hard" processors have this feature permanently available while "soft" processors may have this physically removed from the delivered product
- Debugging types:
  - Functional debugging
  - Performance debugging

The Need for On-Chip Debug

- FPGAs/SoCs offer limited internal visibility
  - How to access the embedded system bus?
  - How to monitor memory and registers?
- Non-processor dedicated silicon IP cores
  - Cannot obtain internal access
- Full scan insertion
  - Increases overhead
- Changes late in the design cycle are ENORMOUSLY expensive
- Co-verification
  - Tools are cumbersome and slow
  - Modeling issues



AMD XILINX

Xilinx Solution for Debugging Embedded Designs

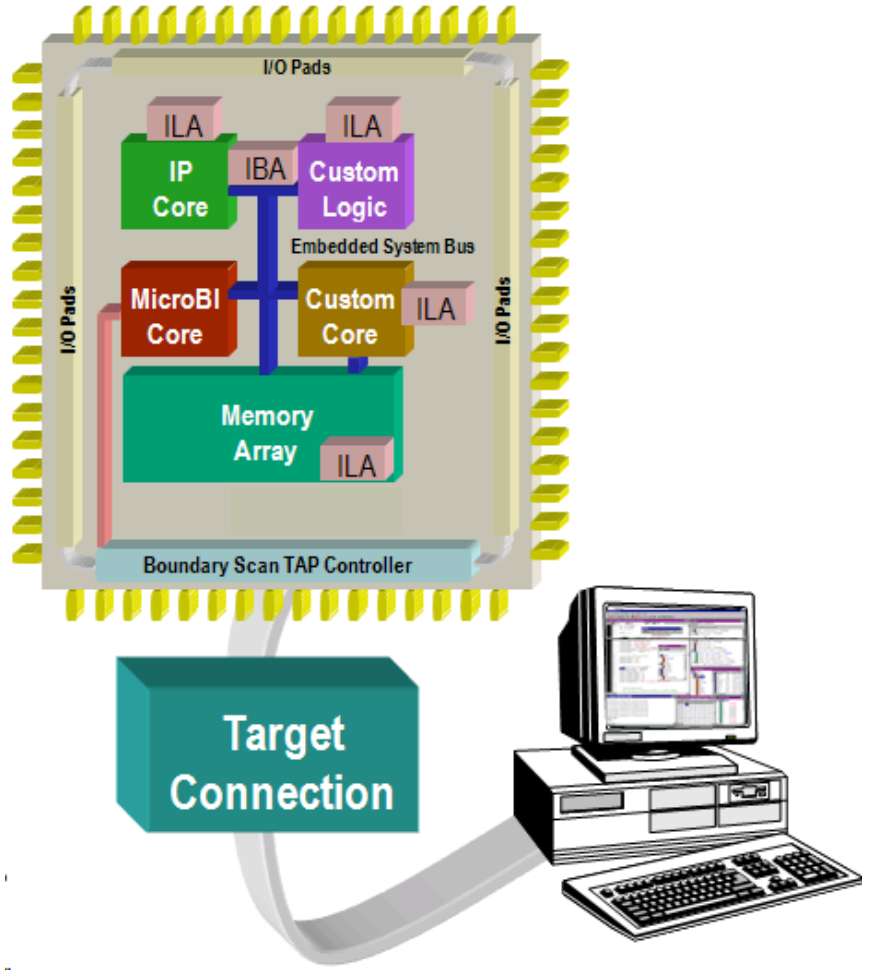
- Hardware debugging tool
  - Vivado logic analyzer for hardware; functionally replaces expensive external logic analyzer
- Software debugging tools
  - Vitis Debugging perspective and inexpensive JTAG cable replace ICE
- Built-in, cross-probing trigger capability between hardware and software
  - Vivado logic analyzer invoked through Vivado
  - Vitis Debugging perspective accessed from within Vitis
- A single JTAG connection can be used for
  - Programming the programmable logic
  - Downloading application
  - Hardware and software debugging

Hardware Debugging

AMD
XILINX

Vivado Logic Analyzer System

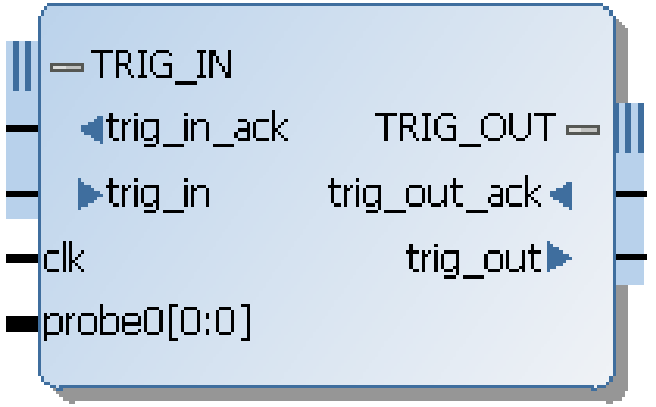
- Vivado logic analyzer tool cores provide full internal visibility to all soft IP
  - Access to hard IP ports
  - Accesses all the internal signals and nodes within the programmable logic (ILA)
  - Stimulus can be applied using the Virtual I/O core (VIO)
- Debugging occurs at, or near, system speeds
  - Debug on-chip using the system clock
- Minimize pins needed for debugging
  - Access via the JTAG interface



AMD XILINX

ILA Core

- Used for monitoring internal programmable logic signals for post-analysis
- Multiple configurable ILA trigger units
  - Configurable trigger input widths and match types for use with different input signals types
- Up to 64 probes through GUI
  - Up to 1024 probes through tcl command
- Sequential triggering
- Storage qualification
- Configurable cross triggering
  - Trigger in and Trigger out interfaces
- Pre- and post-trigger buffering (capture data before, during, and after trigger condition is met)



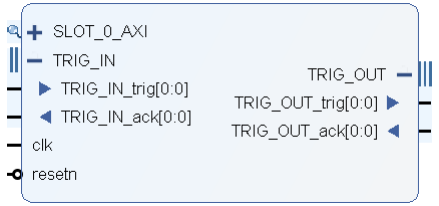
Core Resources Usage

- Debug tools will use block RAM to store samples
- More block RAM equates to larger (deeper and/or wider) trace buffers
- Use larger devices for debugging prototypes to provide extra block RAM
- If the application is utilizing too much block RAM, temporarily modify hardware applications to free up block RAM for debugging usage—small FIFOs and buffer memory, for example

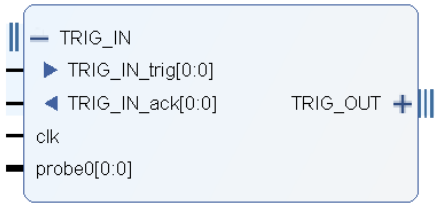
System ILA Core

- Multiple probe ports, which can be combined into a single trigger condition
- Debugging of any debuggable interface including AXI4-MM and Stream
- User-selectable AXI4-MM channel debug and AXI Data/Address width selection
- Data and Trigger probe and interface type selection
- BRAM estimation
- AXI4-MM and AXI4-Stream Protocol checking
- Allows mixing of native and interface modes

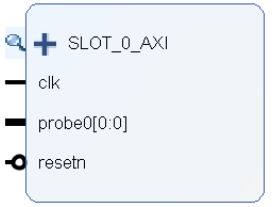
Interface with Cross-triggering



Native with Cross-triggering

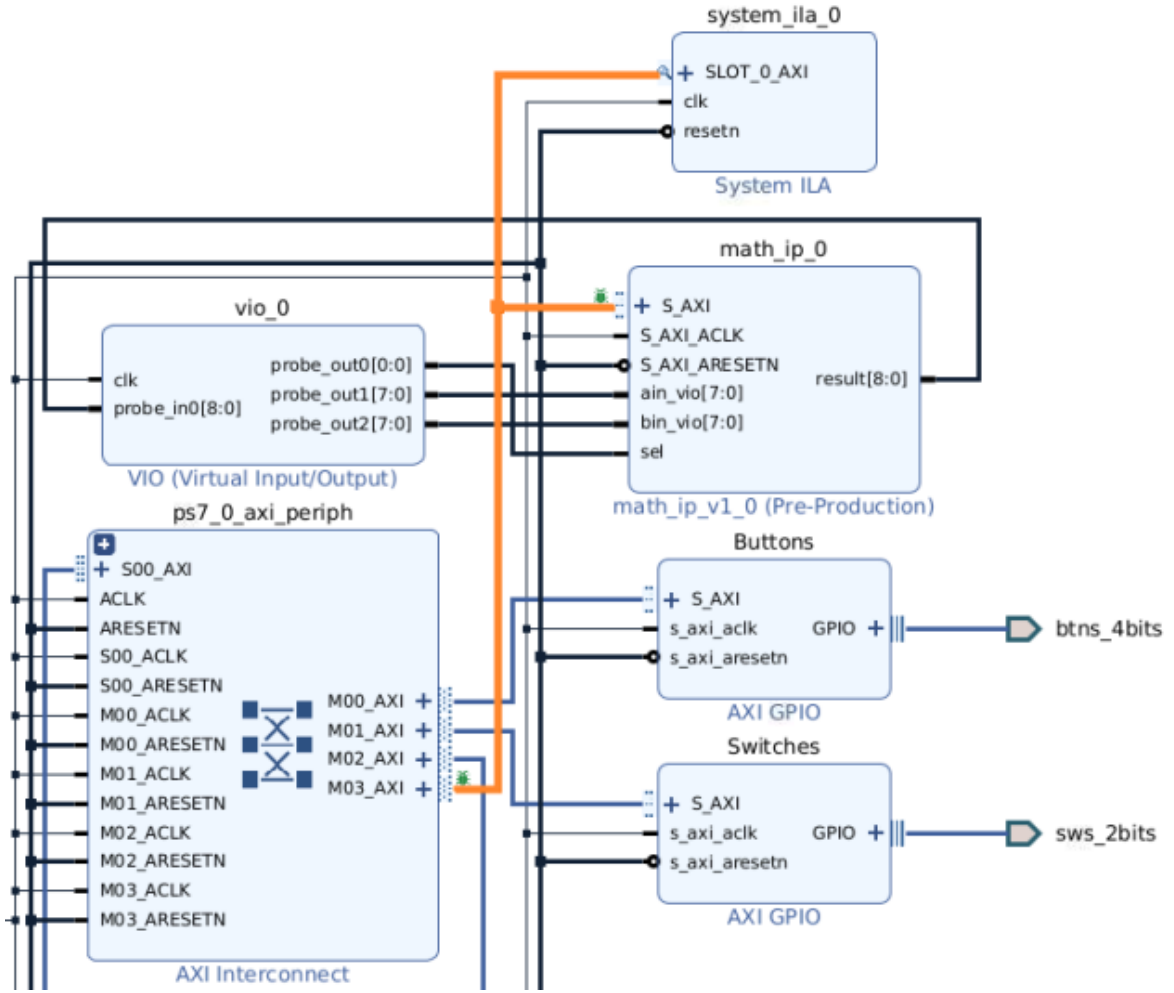


Native with no Cross-triggering



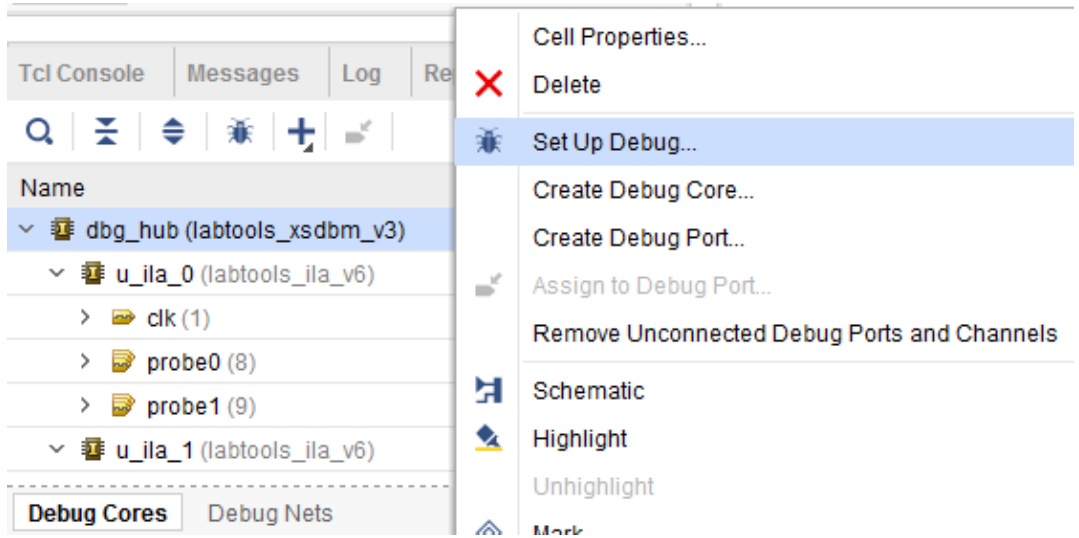
Adding Logic Analyzer Cores

- Instantiate in HDL
  - Cores added like any other peripheral
    - Found in the IP Catalog under Debug
  - User makes connections to ILA
    - Debug signals, clocks
- IP Integrator
  - Add core from IP Catalog
  - Interactively select signals and mark for debug
- mark_debug
  - Specify in HDL
  - Specify through tcl
  - Specify in xdc
  - Mark in IP Integrator



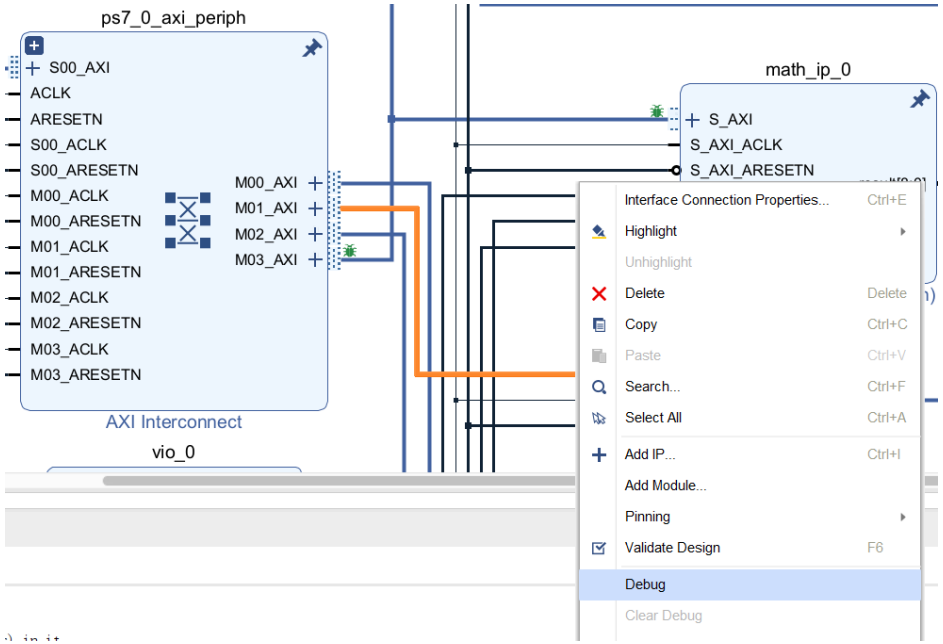
Vivado Debug Tool Access Points

- Select Tools > Set up Debug to launch the Vivado Debug Wizard
  - Synthesized design must be open
  - Debug tab is made visible
- Right-click the debug_hub icon in this window to launch/re-launch the wizard
  - Add signals
  - View resources
  - Create debug core or port
  - Implement the cores



Selecting Signals to Debug

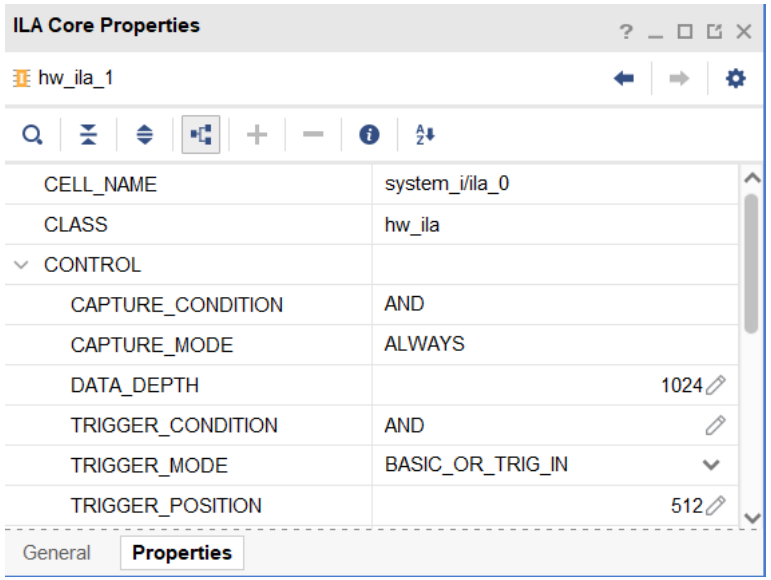
- Select nets in the Vivado tool by any means
  - Netlist view (nets folders)
    - Each level of logic hierarchy
  - Schematic
  - Find results
- Right-click the net and select Mark Debug
- Nets added to Unassigned Nets folder in the Debug tab view
  - Placeholder for probable nets prior to configuring cores
- Net name search also in the Set Up Debug Wizard

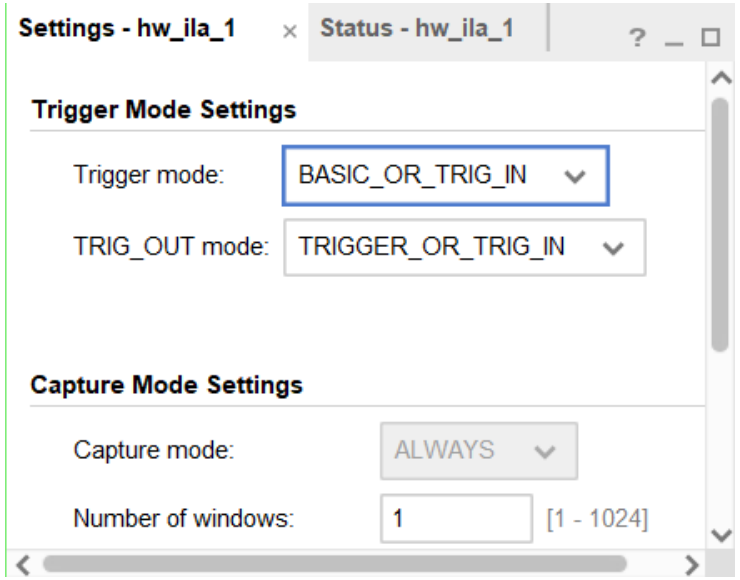


AMD XILINX

Debug Tool Configuration

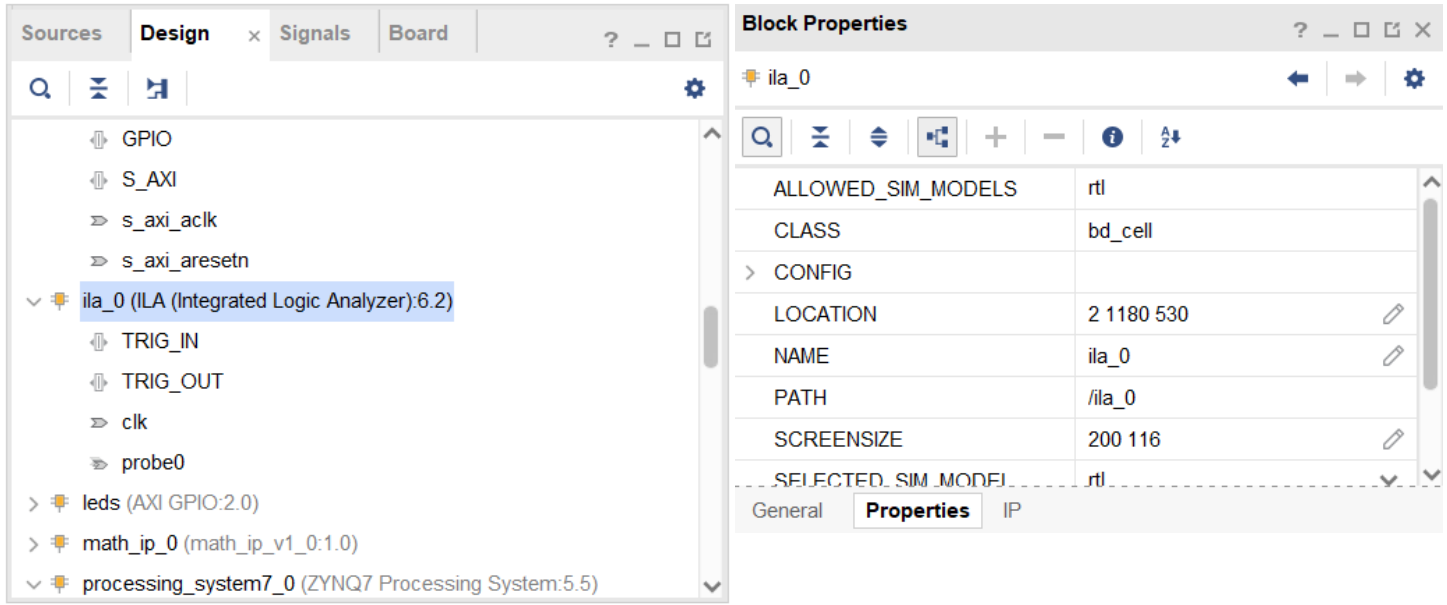
- The Vivado tool view displays core content and configuration
  - CLK, PROBE
  - Signal count
- Set options for cores and signals in the Properties view





Exploring Core Logic

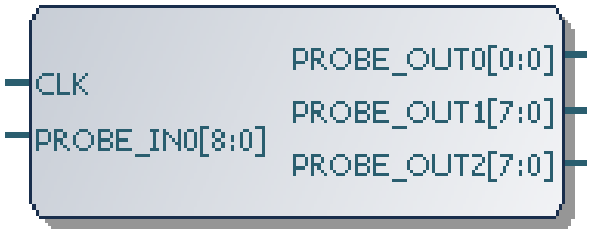
- Netlist window populated with implemented core logic
  - debug_core_hub displayed in Netlist view
  - Expandable logic
- Analyze internal core logic
  - Resource statistics
  - Schematic
  - Connectivity
- Floorplan the cores

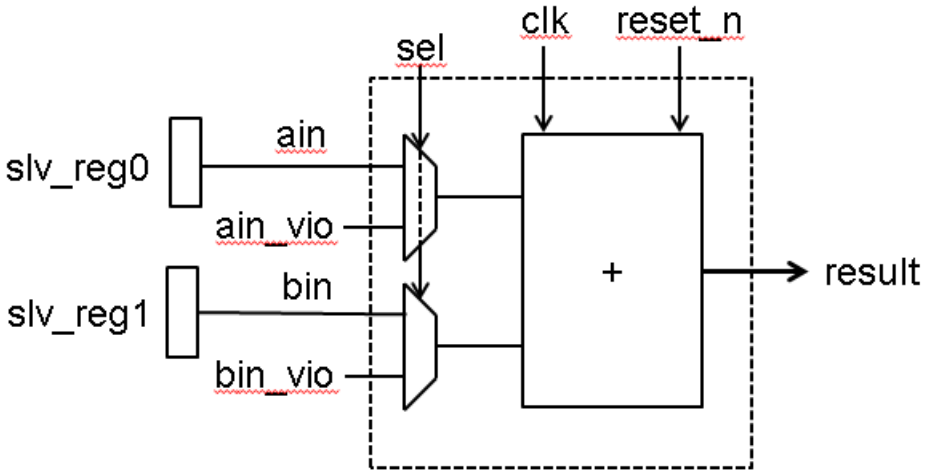


AMD XILINX

VIO Core

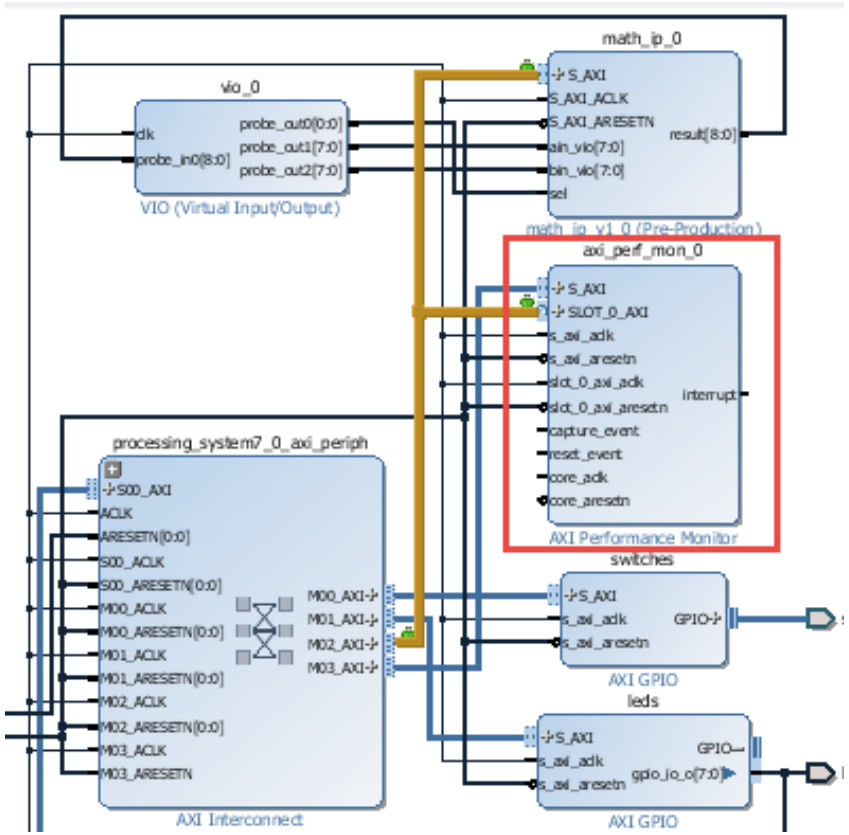
- Support for monitoring and driving internal programmable logic signals in real time
- Probe input unit
- Probe output unit





AXI Performance Monitor Core

- Enables AXI system performance measurement for multiple slots
  - AXI4 and AXI4-Stream
- AXI Performance Monitor supports analyzing system behavior on AXI interfaces
  - Event logging
    - Captures AXI events and external events
    - Time stamp between two successive events into streaming FIFO
  - Event counting
    - Measure events on AXI4/AXI4-Stream monitor slots or external event ports



AMD XILINX

Software Debugging

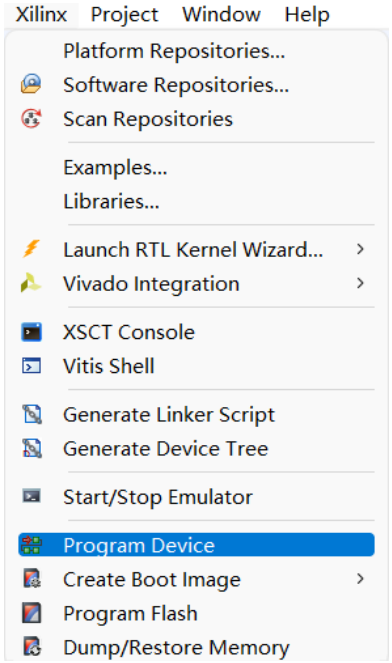
AMD
XILINX

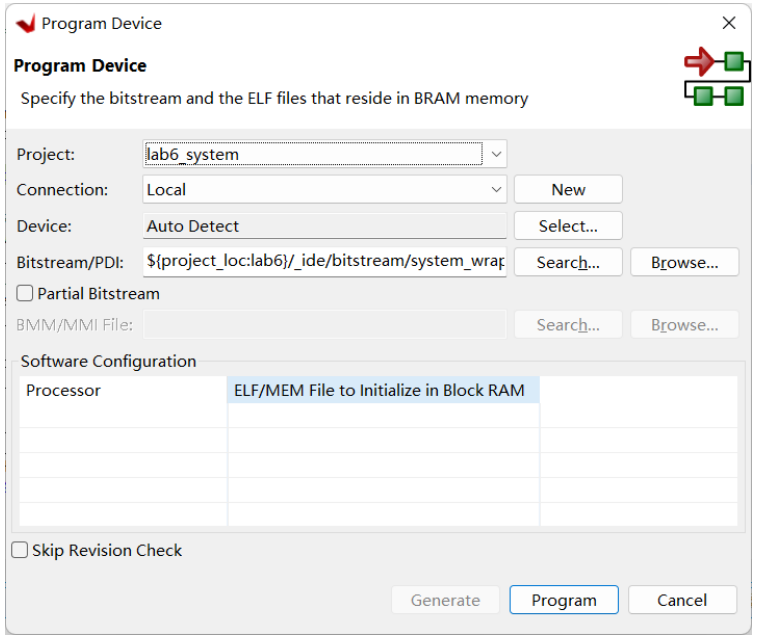
Software Debugging Support in Vitis IDE

- Vitis supports software debugging via:
  - Zynq Programmable SoC contains own internal trace port
    - Performs same general tasks as MDM for MicroBlaze
  - TCF(Target Communication Framework) debugger over digilent cable for ARM
    - Open source
    - Supports system level debugging
    - Improved performance

Configure the Programmable Logic

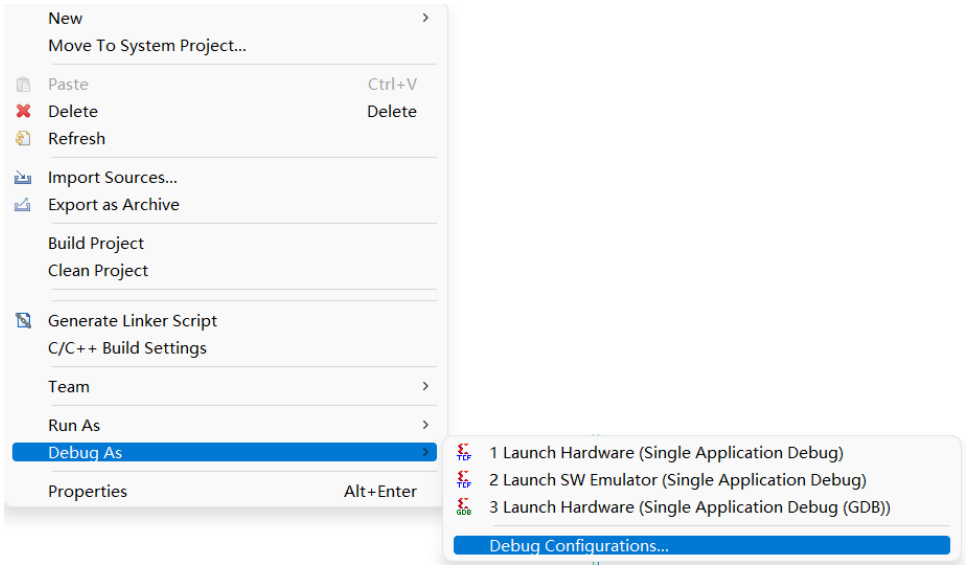
- Before debugging session can be launched, the target programmable logic must be configured
  - PL would have logic analyzer core(s) for hardware debugging
- Design must contain a hardware debug connection
  - CoreSight – built into the Zynq ARM Cortex-A9 processor
  - MDM – add in the MicroBlaze based system
- Verify in the Vitis Log tab that download succeeded
  - Done pin goes High

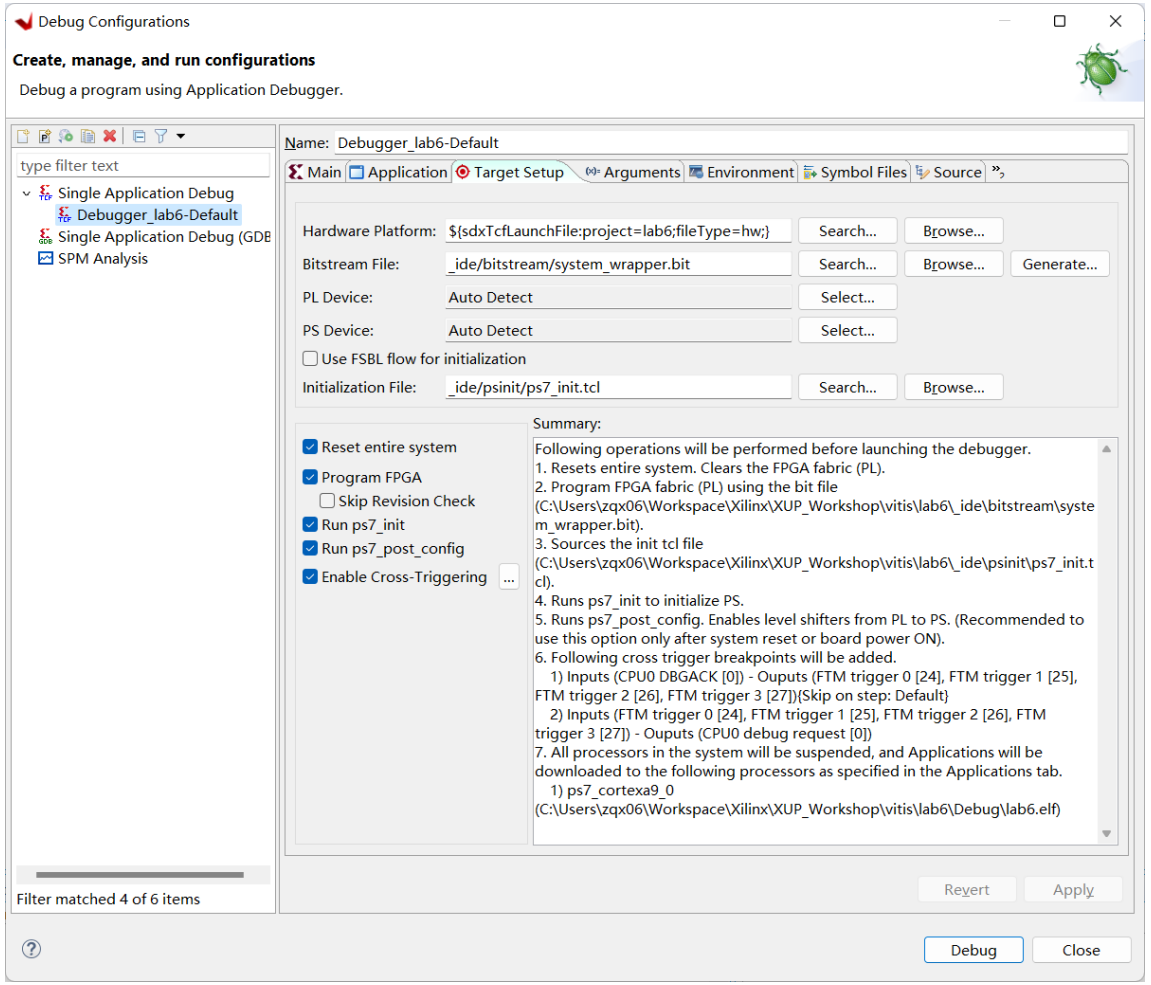




AMD XILINX

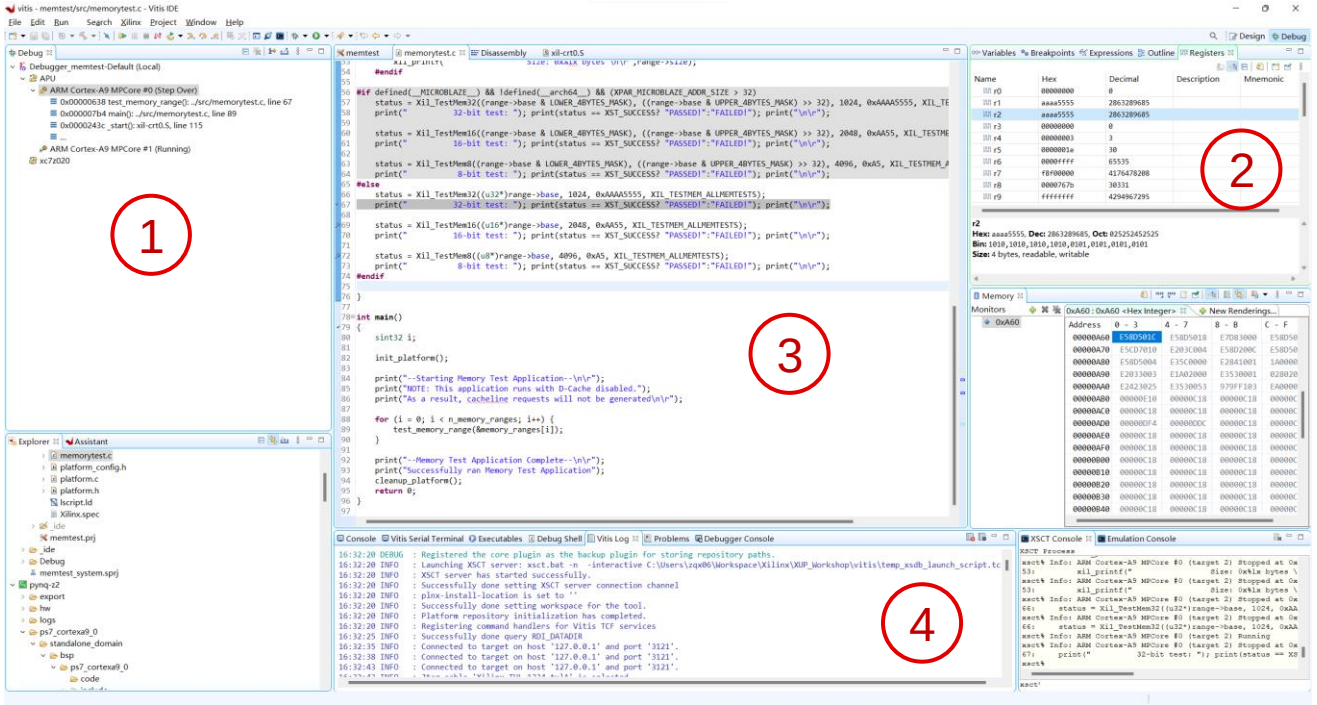
Create Debug Configuration





Vitis Debug Perspective

1. **Stack frame for target threads**
2. **Outline, Variables, breakpoints, and registers views**
   a. Disassembly view can be added using Window > Show View > Disassembly
3. **C/C++ editor**
4. **Console, Vitis Log, and Memory views**



AMD XILINX

System Debugging

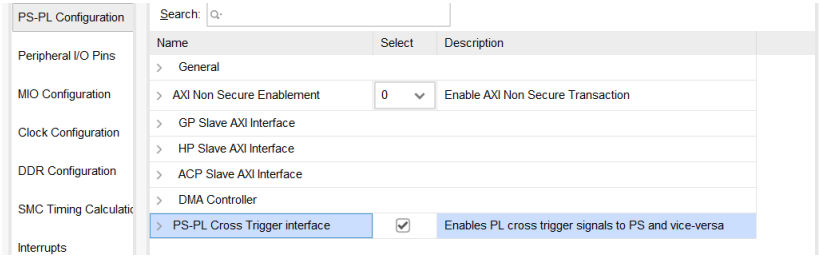
AMD
XILINX

What is System Debugging?

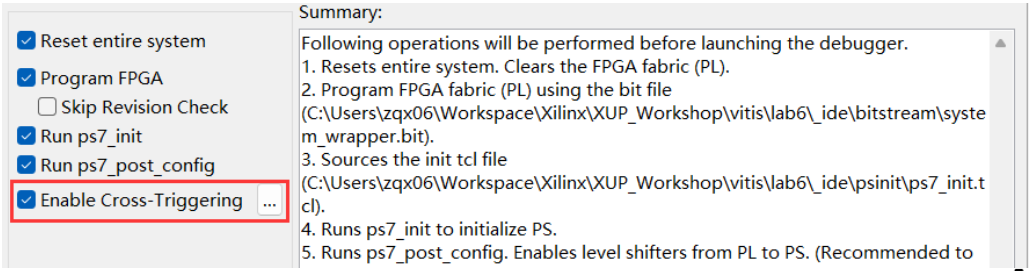
- System debugging is determining where the bug is:
  - In hardware?
  - In software?
- How is system debugging possible?
  - Set breakpoint: when hit → triggers the Vivado logic analyzer
  - Set trigger in Vivado logic analyzer: when hit → halts CPU and debugger stops

System Debug: Simultaneous Hardware/Software Debugging (1)

- TCF supports simultaneous access over Xilinx download cables
- ILA core in the target cores
- Software condition triggers hardware:
  - Software breakpoint asserts the P2F_trig signal and if the ILA is sensitive to this signal, it will capture
- Hardware condition triggers software:
  - Match in the ILA asserts the F2P_Trig signal, which halts the processor
- Vivado logic analyzer running on the host
  - Enable cross triggering in hardware

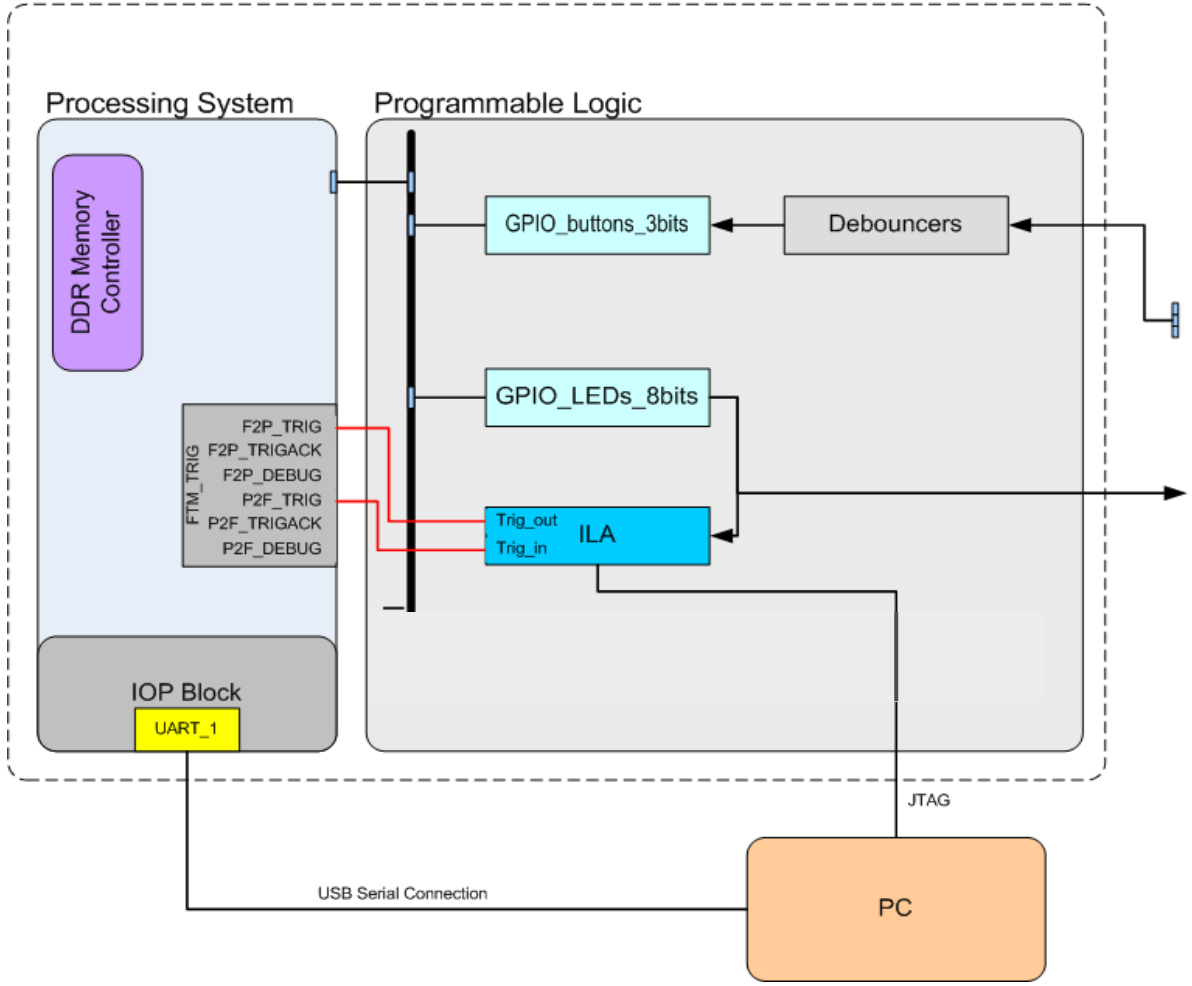


- Debugger on the host
  - Enable cross triggering in software



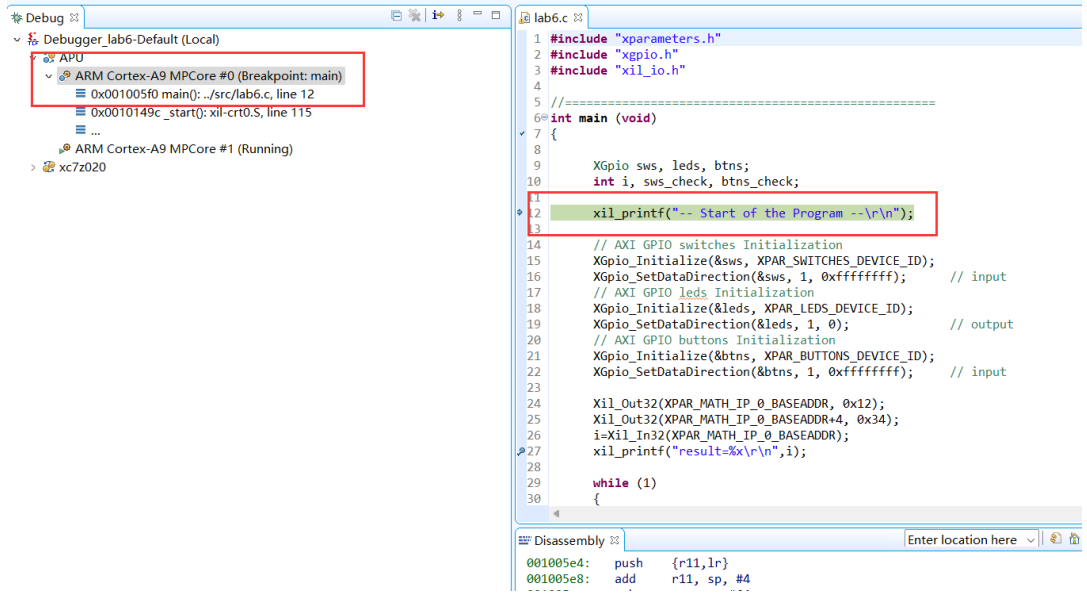
System Debug: Simultaneous Hardware/Software Debugging (2)

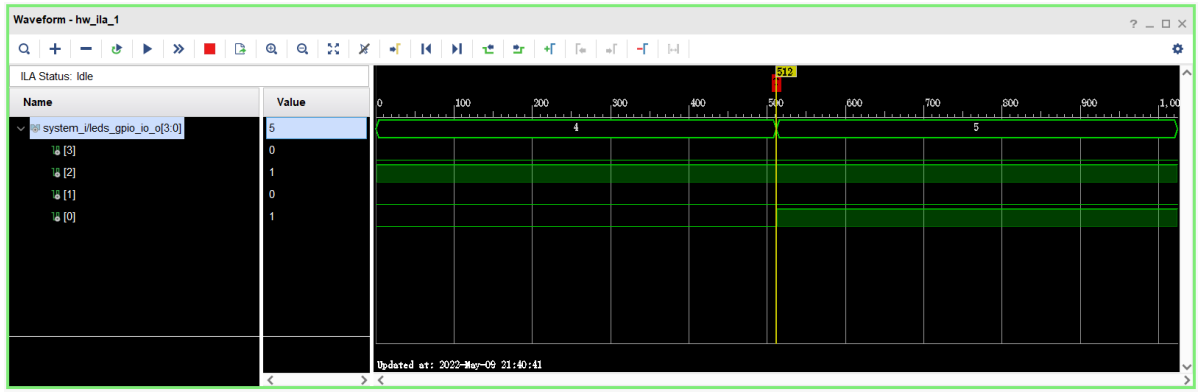
- Cross triggering connections between Zynq processor and ILA
- TCF support simultaneous access over Xilinx download cables
  - ILA instantiations
  - Treated like the peripheral cores
- Set breakpoint in the Vitis Debugging perspective: when hit, triggers the logic analyzer
- Set trigger in logic analyzer: when hit, halts CPU and debugger stops



AMD XILINX

System Debug: Simultaneous Hardware/Software Debugging (3)





AMD XILINX

Summary

AMD
XILINX

Summary

- Debugging is an integral part of embedded systems development
- Vitis provides tools to facilitate hardware and software debugging
  - Hardware debugging is done through using Vivado logic analyzer cores
  - Software debugging is performed using system debugger
- Vitis provides environment, perspective, and underlying tools to enable seamless software debugging
- A significant amount of hardware overhead may occur depending on the type and number of cores and sample depth
- The Debug Configuration wizard simplifies hardware connections and the cores inclusion
- With software debugging and the Vivado logic analyzer supporting cross-probing enables simultaneous hardware and software debugging
  - Use it to find and fix embedded system bugs faster

Thank You

Disclaimer and Attribution

AMD
XILINX