Debugging the Zynq All Programmable SoC
Objectives

After completing this module, you will be able to

➤ Explain the various JTAG chain configurations for the Zynq All Programmable SoC

➤ List the features of CoreSight technology

➤ Describe the configuration for hardware/software co-simulation

➤ Describe the difference between the system creator (SC) and software developer (SD) versions of the Zynq All Programmable SoC simulation virtual platforms
Introduction to CoreSight Technology

- Introduction to CoreSight Technology
- Zynq Device JTAG – One Cable or Two
- Vivado Logic Analyzer Integration
- Hardware and Software Co-Simulation
- Zynq All Programmable SoC Virtual Platform
- Summary
SoC Debug Overview – Xilinx Style

Based on ARM® CoreSight architecture
- Embedded trace buffer (ETB)
- Program trace module (PTM)
- Instrument trace module (ITM)

Xilinx adds two additional CoreSight technology components
- AXI monitor (AXIM)
- Fabric trace module (FTM)
Debug Tools Enabled by ARM CoreSight Technology (1)

- **Invasive software debug**
  - Six hardware break points
  - Four watch points

- **Non-invasive software debug**
  - Traces instruction flow
    - Captures off-chip, real-time trace by using trace port pins (configurable from 2-32)
    - Captures real-time trace by using on-chip 4KB buffer and accessed via JTAG

- **Software debug with SDK**
  - Eclipse IDE and GNU gdb tools
Debug Tools Enabled by ARM CoreSight Technology (2)

- Hardware debug and cross trigger with Vivado logic analyzer and AXI monitor
  - Cross trigger Cortex-A9 processor with CoreSight technology in the PS
  - Cross trigger Cortex-A9 processor with Vivado logic analyzer in the PL
  - Simple debugging of transactions between AXI masters and slaves
  - Complex debugging of malformed transactions and protocol violations
  - Performance analysis and tuning of AXI-based systems

- Third-party tools via their cables
Difference Between Debug and Development Boot Stages

- Debugging is performed via JTAG or CoreSight technology cable

- In debug mode, boot ROM makes JTAG master of the system
  - Processor is placed in idle state
  - Non-secure boot and configuration only

- Ability to update PL bitstream without effecting the PS
  - PL has to be powered on and in cascaded mode
  - PL JTAG TAP controller cascaded with the PS ARM DAP controller
CoreSight Technology Cortex-A9 Processor Debugging

► **Debug state**
  - Core is halted and isolated from rest of system
  - Processor and system state can be viewed/modified
    - By providing instructions to the core and executing them
  - Clock now controlled by debug logic
  - No interrupts will be handled until execution restarted by debugger

► **Entry into debug state is controlled by registers in the debug logic block**
  - Debug control register (DCR)
    - Enables debugger to request entry into debug state (via DBGRQ bit)
  - Debug status register (DSR)
    - Allows core to acknowledge entry to debug state (via DBGACK bit)
  - Typically accessed by tool software only
Cortex-A9 Processor Embedded Trace Macrocell
Zynq Device JTAG – One Cable or Two

- Introduction to CoreSight Technology
- Zynq Device JTAG – One Cable or Two
- Vivado Logic Analyzer Integration
- Hardware and Software Co-Simulation
- Zynq All Programmable SoC Virtual Platform
- Summary
Two available JTAG chains

- PJTAG: for programming the ARM Cortex™-A9 processors
  - JTAG-only slave boot method: debug/non-secure only

- JTAG: for configuring the programmable logic
  - Soft PL logic-based using the BSCAN component
  - Used in conjunction with the Vivado logic analyzer and software debugger
Debug and Development Mode (1)

Two TAP controllers
- Xilinx TAP
  - Used to access PL BSCAN component
- ARM DAP
  - Used to access the PS (ARM)

Key features
- On-board flash programming
- Vivado logic analyzer support
- Direct system address space access through DAP-AP port
- External trace capture using MIO in PS, or EMIO in PL
Debug and Development Mode (2)

Topology

- Cascade mode
  - Device is accessed by a Xilinx download cable
  - Facilitates independent booting of PS and configuration of PL

- Independent mode
  - ARM DAP is pinned out using PL I/O—four additional pins needed
  - PL can be configured using Xilinx TAP - third-party cable
  - PS can be debugged using ARM DAP - Xilinx cable
Vivado Logic Analyzer Integration

- Introduction to CoreSight Technology
- Zynq Device JTAG – One Cable or Two
- Vivado Logic Analyzer Integration
- Hardware and Software Co-Simulation
- Zynq All Programmable SoC Virtual Platform
- Summary
The Vivado logic analyzer is comprised of two basic items:

- Debug cores that sample and inject signals into the programmable logic
- Flows for generating and inserting these debug cores into a design and for controlling and viewing these cores once downloaded
Vivado logic analyzer can be used in three contexts

- Verification: the process of determining the compliance of a given design to a (written) specification
- Debugging: the process of finding and correcting the source of this discrepancy
- Data capture: used to collect real-world data for input into a simulation
Vivado Logic Analyzer Cores and Capabilities

Vivado logic analyzer cores and capabilities
- Integrated Logic Analyzer (ILA) core
  - Probe inputs for triggering and data
  - Probes attach to user net, schematic or HDL
  - Programmable trace depth
  - Flexible triggering options

- Virtual I/O (VIO) core
  - Used to provide stimulus (inputs)
  - View slow moving signals (outputs)

- Cross trigger with CoreSight technology in the PS
  - Hardware event forces a program breakpoint
  - Software breakpoint triggers logic analyzer
ILA Instantiation Example in Vivado IDE Block Diagram Editor
Hardware and Software Co-Simulation

- Introduction to CoreSight Technology
- Zynq Device JTAG – One Cable or Two
- Vivado Logic Analyzer Integration
- **Hardware and Software Co-Simulation**
- Zynq All Programmable SoC Virtual Platform
- Summary
ISim Hardware Co-Simulation

- Bridge the gap between simulation and on-chip debug
  - Offload whole or part of a design to the programmable logic
  - Re-use HDL testbench to provide stimuli

- Accelerate RTL simulation by up to 100X
  - Most acceleration seen for compute-intensive designs
ISim Hardware Co-Simulation Overview

- Design Top-level
- Instance co-simulated in FPGA
- Ethernet P2P or JTAG
- Xilinx FPGA board
- Xilinx ISim Simulator
Benefits

- Accelerate functional simulation up to 100x compared to software-only simulation
- Allows HDL testbench reuse for testing a design running in hardware
- Allows incremental programmable logic testing of blocks of a design
- Enables IP simulation in hardware versus slower gate-level simulation models
- Easy to use: available inside the ISim environment as a core feature
There is NO RTL simulation model for the Zynq All Programmable SoC PS

Simulation of the PS is accomplished by ISim hardware co-simulation

- All ISim accesses to the PS will be to actual hardware
- PL logic can also be included in hardware co-simulation

PS ISim hardware co-simulation is also referred to as Hardware-in-the-Loop (HIL) simulation
Performance Comparison (JTAG)

Behavioral simulation time for Wi-MAX DUC CFR design – run on ML505 (XC5VLX50T)

<table>
<thead>
<tr>
<th>ISim</th>
<th>ISim HW Co-Sim</th>
<th>Speed Up</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simulation</td>
<td>4550 s</td>
<td>314 s</td>
</tr>
</tbody>
</table>

15x speed-up = More design turns per day

Accelerations up to 100x compared to software only simulation
Use Model 1: Pure Logic-Based Designs

- **Design characteristics**
  - Composed of only LUTs, flip-flops, block RAMs, and DSP primitives
  - Single clock
  - Computational intensive
  - All ports controlled by ISim and accessible from the software testbench

- **Advantages of ISim hardware co-simulation**
  - Simulation acceleration
  - In-hardware functional verification
  - Bit-and-cycle accurate with respect to pure software simulation
Use Model 2: Hybrid Designs

Design characteristics
- Can be composed of hard IP blocks, DCMs/PLLs/MMCMs, and MGTs
- Multiple clocks
  - One clock in lockstep with software simulation
  - Other clock is free running
- Some ports can be mapped to external I/Os that are neither controlled by ISim nor accessible from the software testbench

Advantages of ISim hardware co-simulation
- Simulation acceleration
- In-hardware functional verification
- Hybrid simulation allowing non-trivial software and hardware interactions
Hybrid Simulation Architecture

- Software testbench drives the simulation of Module 1
- Rest of the system is free running in hardware, interacting with external peripherals
Zynq All Programmable SoC Virtual Platform

- Introduction to CoreSight Technology
- Zynq Device JTAG – One Cable or Two
- Vivado Logic Analyzer Integration
- Hardware and Software Co-Simulation
- Zynq All Programmable SoC Virtual Platform
- Summary
What is the Zynq All Programmable SoC Virtual Platform?

- NO RTL simulation model for the Zynq All Programmable SoC

- Virtual platform advantages
  - Fast functional model of an embedded platform (real or envisaged)
  - Executes the same software binaries as the actual hardware
  - Runs on traditional desktops, laptops, or servers

- Zynq All Programmable SoC virtual platform features
  - A fast instruction set simulator for the ARM® Cortex-A9 MPCore processor
  - Models register interface for all Zynq-7000 PS devices
  - Models functions of key IOP and other PS peripherals
  - Extensible using TLM (C or SystemC models)
    - Peripherals, accelerators, and any other IP in the programmable logic or on the board
Zynq-7000 All Programmable SoC Virtual Platform (1)
Zynq All Programmable SoC Virtual Platform (2)

- Zynq All Programmable SoC virtual platform
  - A fast instruction set simulator for the ARM Cortex-A9 MPCore processor
    - ARM ISA, NEON SIMD, VFPv3, Thumb, Thumb2, Jazelle
  - Models register interface for all Zynq-7000 devices
  - Models function of key Zynq-7000 All Programmable SoC and on-board devices
  - Extensible using TLM (C or SystemC models)
    - Peripherals, accelerators and any other IP in the programmable logic or on the board
    - Enabled with system creator (SC) tools
Virtual Platform Tools

- Two tool bundles to choose from: you can choose any seat-combination of two tool bundles
  - Different target users (developer vs. creator)
  - Different tool capabilities (standard debug vs. comprehensive debug and system views and an array of SystemC model development tools)

- Each bundle connects to the same Zynq-7000 All Programmable SoC virtual platform
Facilitates full simulation of the PS and modeled hardware

Does not allow simulation of custom user logic in PL that has not been modeled

Operation in SDK is similar to debugging in GDB
System Creator ("SC")

- Provides for PS and PL simulation
- PS simulation similar to that of the SD version
- PL simulation and debugging facilitated with SystemC modeling
- Uses Cadence system creator tools
System Creator: Key Features (1)

**Tools for the target**
- Waveform viewer (logic analyzer): Shows individual register, interrupt, or memory values over time associated with lines of code (post-run)
- Memory viewer: Shows the memory value for every hardware location, mapped to hardware/device and executing lines of code
- Register window: Shows every Zynq All Programmable SoC register as they are modified during software execution
- Watch window: Shows value of every hardware and software watch points

**Tools for the model**
- Schematic tracer: Shows the schematic of the model with static and dynamic parameters
- Virtual platform logging: Provides log information about the model execution on the host and the execution of the virtual hardware on the simulator
Tools for simultaneous model/target usage
- Design browser: Shows each of the virtual platform submodels and key associated parameters
- Source browser: Main debugger window; correlates the Cortex-A9 processor lines of code to hardware parameters
- TLM transaction analysis: Provides model timing and transaction sequence information correlated to executing Cortex-A9 processor code
System creator runs on a Linux platform

Zynq All Programmable SoC target interface
- A serial console connected to the virtual platform model is available from the host; use to connect exactly like real hardware
- Waveform graphics of PS/PL interfaces displayed on the host

Hardware host to model "bridges"
- Ethernet
- USB

Graphical user support for
- Multi-core debug
- Register waveforms
- Model transaction results
- Software line of code maps to hardware state
- Performance analysis charting
QEMU Simulation

➤ **QEMU: open-source emulator/simulator**
  - System emulation
  - OS (Linux) user emulation

➤ **Available for many processors**
  - ARM Cortex-A9 processor
    - Zynq All Programmable SoC PS support
    - Pathways for custom PL logic support
  - MicroBlaze processor

➤ **Primarily supported through third parties**
Summary

- Introduction to CoreSight Technology
- Zynq JTAG – One Cable or Two
- Vivado Logic Analyzer Integration
- Hardware/Software Co-Simulation
- Zynq All Programmable SoC Virtual Platform

**Summary**
Apply Your Knowledge

1. Identify the various JTAG chain configurations for the Zynq All Programmable SoC

2. Describe the difference between the SC and SD versions of the Zynq All Programmable SoC simulation virtual platforms
Summary (1)

- The ARM CoreSight technology debug unit is used to access the components of the PS via the PJTAG.

- PL debugging is facilitated by the traditional JTAG BSCAN primitive and soft logic IP such as the Vivado logic analyzer.

- Both JTAG chains can be cascaded or independent:
  - Cascaded: Use the Xilinx USB Platform cable for hardware/software download and debug.
  - Independent: Allows for third-party cable download and debug of PS.
Summary (2)

➤ The Vivado logic analyzer and software debugger provides for simultaneous hardware/software debug

➤ Hardware co-simulation can drastically reduce simulation times
  – The PS is always co-simulated in hardware

➤ The Zynq All Programmable SoC virtual platform tool from Cadence provides full software simulation
Lab 4: Debugging on the Zynq All Programmable SoC

Introduction

In this lab you will be introduced to the basic operations of the Vivado logic analyzer.

Objectives

After completing this lab, you will be able to:

- Mark nets of interest, in a Vivado Design Suite block diagram, for debug
- Insert an ILA debug core into a synthesized design
- Set up an ILA for cross-triggering with a Cortex™-A9 processor
- Import a software application for debugging
- Operate the Vivado logic analyzer
- Operate the SDK software debugger
- Perform cross-triggering between the Vivado logic analyzer and SDK software debugger