Embedded C/C++ SDSoC Development Environment and Methodology

Course Description

This one-day course is structured to help designers new to the SDSoC™ development environment to quickly create accelerated systems. The focus is on utilizing the tools to accelerate an existing design at the system architecture level, not on the optimization of the accelerator microarchitectures.
Several optional modules are provided to quickly provide students with the necessary background on both hardware and software. The first half of this class is Level 1 while the afternoon's topics are at Level 2.

Level: Embedded 1 (morning), Embedded 2 (afternoon)

Training Duration: 1 day

Who Should Attend?

Anyone interested in quickly adding hardware acceleration to a software system.

Prerequisites:

- Understanding of Zynq®-7000 architecture (with emphasis on ACP, HP ports, and internal routing)
- Comfort with the C programming language
- Familiarity with the Vivado® Design Suite, Vivado HLS tool, and Xilinx SDK

Software Tools:

- SDx™ development environment 2016.3

Hardware:

- Architecture: Zynq-7000 All Programmable SoC*
- Demo board: Zynq-7000 All Programmable SoC ZC702 or ZedBoard*

Skills Gained: After completing this training, you will be able to:

- Identify candidate functions for hardware acceleration by using the TCF profiling tool
- Use the System Debugger's capabilities to control the execution flow and examine memory and variables during a debug session
- Move designated software functions to hardware and estimate the performance of the accelerator and the effect on the entire system
- Override tool defaults to improve the performance of the individual accelerators and the overall system

Course Outline

1. Zynq AP SoC Architecture Support for Accelerators [Optional]

2. Software Overview [Optional]

3. SDSoC Tool Overview

4. SDSoC Tool Design Best Practices

5. Application Profiling

6. Application Debugging

7. Understanding Estimations in the SDSoC Tool

8. Blocking and Non-Blocking Implementations in the SDSoC Tool

9. Implementing Multiple Accelerators in the SDSoC Tool

10. SDSoC Tool Platform Creation

11. Hardware/Software Event Tracing

Lab Description

- SDSoC Tool Overview
Introduces the purpose, underlying structures, and basic functionality of the SDSoC development environment through a combination of lecture and demonstration. Student will cement their knowledge with a lab that reinforces the concepts provided in the lecture and demo.

- Application Profiling
Profiling is the process that identifies how the processor is spending its time. Through profiling, the user can quickly identify which functions must be optimized or moved to hardware to satisfy the performance requirements.

- Application Debugging
Through the use of the System Debugger, students will learn how to follow the control flow in an executing application and see the effects of the code on memory to successfully debug software issues.

- Understanding Estimations in the SDSoC Tool
Once a function is moved to hardware, questions remain: Will the accelerator fit in hardware? Will it fun fast enough? Estimations can provide the answers.

- Blocking and Non-Blocking Implementations in the SDSoC Tool
Addresses how the processor behaves while the accelerator is producing solutions—does it wait or continue on?

- Implementing Multiple Accelerators in the SDSoC Tool
There are times when moving a single function to hardware is not enough—multiple functions must be moved to hardware, or one accelerator must be duplicated. Here students will learn to control how the tool produces the accelerators.

- SDSoC Tool Platform Creation
Describes how to create a custom SDSoC platform starting from a hardware system built using the Vivado Design Suite, and a software run-time environment, including operating system kernel, boot loaders, file system, and libraries.

- Hardware/Software Event Tracing
Hardware/software event trace helps the user to understand the performance of their application given the workload, hardware/software partitioning, and system design choices. Such information helps the user to optimize and improve system implementation.