LOGTEL
It's more than knowledge

Authorized Training Provider
XILINX®

Embedded Systems Software Design

Course Description

This two-day course introduces you to software design and development for the Xilinx Zynq® All Programmable System on a Chip (SoC) using the Xilinx Software Development Kit (SDK). You will learn the concepts, tools, and techniques required for the software phase of the design cycle.
Topics are comprehensive, covering the design and implementation of the board support package (BSP) for resource access and management of the Xilinx Standalone library. Major topics include device driver use, user application debugging and integration. Practical implementation tips and best practices are also provided throughout to enable you to make good design decisions and keep your design cycles to a minimum. You will have enough practical information to start developing software applications for the ARM® Cortex™-A9 and MicroBlaze™ processors.
Additionally, this course covers developing software applications for a Xilinx embedded system based on a MicroBlaze processor.

Level: Embedded Software 3

Training Duration: 3 days

Who Should Attend?

Software design engineers interested in system design and implementation and software application development and debugging using the Xilinx Standalone library.

Prerequisites:

- C or C++ programming experience, including general debugging techniques
- Conceptual understanding of embedded processing systems including device drivers, interrupt routines, writing / modifying scripts, user applications, and boot loader operation.

Software Tools:

- Vivado® Design or System Edition 2016.1

Hardware:

- Architecture: Zynq-7000 All Programmable SoC and 7 series FPGAs*
- Demo board: Zynq-7000 All Programmable SoC ZC702 or ZedBoard or Kintex®-7 FPGA KC705 board*

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

- Implement an effective software design environment for a Xilinx embedded system using the Xilinx SDK tools
- Write a basic user application (under Standalone or Linux) using the Xilinx Software Development Kit (SDK) and run it on an embedded system platform
- Use Xilinx debugger tools to troubleshoot user applications
- Apply software techniques to improve operability
- Maintain and update software projects with changing hardware

Course Outline

1. Overview of Embedded Software Development

2. Embedded UltraFast Design Methodology

3. Zynq7000 All-Programmable SoC Architecture Overview (Lab)

4. MicroBlaze Processor Architecture Overview (Lab)

5. Driving the SDK Tool (Lab)

6. System Debugger (Lab)

7. Standalone Software Platform Development (Lab)

8. C Coding Support for Standalone

9. Memory File System (Standalone) (Lab)

10. Using Linker Scripts (Lab)

15. Introduction to Interrupts

16. Interrupts: Software Considerations (Lab)

17. Operating Systems: Introduction and Concepts

Lab Description

Exploring the Architecture of the Zynq-7000 All Programmable SoC – Begin with the Vivado IP integrator to create the hardware design for the Zynq All Programmable SoC and customize the processor.

Exploring the Architecture of the MicroBlaze Processor – Begin with Vivado IP integrator to create a basic MicroBlaze processor design and explore hardware/performance trade-offs.

Driving the SDK Tool – Explore the basic behavior of the Eclipse-based IDE tool that will be used in most of the software development labs.

Software Debugging – Launch the SDK debug perspective step through the various capabilities of the system debugger, including controlling the flow of execution and monitoring and modifying memory.

Application Development – Create a simple software application project with the provided source files. Verify proper BSP settings and linker script generation. Use API documentation to complete the software application. Verify proper operation in hardware.

File Systems – Implement a Standalone software platform that incorporates the XilMFS memory file system. Develop an application that performs file-related tasks on external memory.

Embedded Systems Software Design.

Course Outline

Cont.

18. Linux: A High-Level Introduction

19. Linux Software Application Development Overview (Lab)

20. Writing Code in the Xilinx Linux Environment

21. Booting Overview (Lab)

22. Profiling Overview (Lab)

23. Understanding Device Drivers

24. Custom Device Drivers (Lab)

Lab Description

Linker Script – Configure the linker script and observe increased performance when utilizing different memory sections.

Software Interrupts – Replace a software timing loop with an interrupt-driven timer. Add the timer software and implement an interrupt handler for the timer. Download into hardware and test the application.
Linux Application Development – Access the general-purpose input/output (GPIO) that is connected to the evaluation board.

Booting Loading from Flash Memory – Take a provided bitstream and application and construct your own bootable image.

Software Profiling – Profile a program, interpret reports, and verify the performance with multiple calls.

Writing a Device Driver – Create the skeleton driver framework and add a provided LCD device driver to the BSP.