HX5 - AMD Zynq All Programmable SoC: Hardware and Software Design

This course explains how to design a System on Chip (SoC) based on the AMD Zynq-7000 All Programmable SoC

Objectives

- Describing how to build a complete Embedded System based on the Zynq (Processing System with an ARM Cortex-A9MP Core + FPGA)
- Describing the Zynq Implementation, the Vivado IP Integrator tool and the Software Development Kit (SDK) tools to create a hardware platform and the software to program it
- Working with AMD (Xilinx) tools like Chipscope and the SDK Remote Debugging to debug the Software and the Hardware
- Booting the Linux Kernel on the platform and Executing Linux OS based Applications
- Creating a User IP and the corresponding Linux Driver and integrating it to the System

Prerequisites

- Basic knowledge on processor and FPGA technology
- Knowledge of VHDL and C languages

Course Environment

- Theoretical course
  - PDF course material (in English) supplemented by a printed version for face-to-face courses.
  - Online courses are dispensed using the Teams video-conferencing system.
  - The trainer answers trainees' questions during the training and provide technical and pedagogical assistance.
- Practical activities
  - Practical activities represent from 40% to 50% of course duration.
  - Code examples, exercises and solutions
  - For remote trainings:
    - One Online Linux PC per trainee for the practical activities.
    - The trainer has access to trainees' Online PCs for technical and pedagogical assistance.
    - QEMU Emulated board or physical board connected to the online PC (depending on the course).
    - Some Labs may be completed between sessions and are checked by the trainer on the next session.
  - For face-to-face trainings:
    - One PC (Linux ou Windows) for the practical activities with, if appropriate, a target board.
    - One PC for two trainees when there are more than 6 trainees.
  - For onsite trainings:
    - An installation and test manual is provided to allow preinstallation of the needed software.
    - The trainer come with target boards if needed during the practical activities (and bring them back at the end of the course).
- Downloadable preconfigured virtual machine for post-course practical activities
- At the start of each session the trainer will interact with the trainees to ensure the course fits their expectations and correct if needed

Target Audience

- Any embedded systems engineer or technician with the above prerequisites.

Course Outline

Zynq Device Overview

- Processing System
- Programmable Logic
- Interfacing and signals
- Interconnects
- Memory
- Interrupts

Embedded System Design: Starting with a simple « Hello world » project

- Tools Introduction :
  - Vivado Design Suite
  - SDK
- Development Flow Introduction
  - Hardware Development
  - Software Development
  - Verification (Simulation and Debug)
  - Downloading the bitstream

Exercise: Creating the Hardware and the Software to send strings on a serial port

Embedded System Design Using the PS and the Programmable Logic

- Adding an existing IP (from the AMD Xilinx Library) to the design
- Dealing with interrupts
- Developing with SDK
- Debugging with SDK
- Software profiling

Exercise: Enhancing the previous Platform (Adding Interrupt Controller, GPIO, RAM)
Exercise: Developing the software dealing with interrupts

Chipscope - Hardware Debug

- Introduction to Chipscope Pro
- Implementing an AXI monitor into the design to analyze AXI4-Lite Bus transactions
- Retrieving the on-chip signals waveforms using Chipscope Pro Analyzer
- Clarifying trigger conditions

Exercise: Connecting a Chipscope Analyzer to the AXI bus

Linux Booting and Application debugging

- The linux kernel
- Linux booting, boot Methods
- Linux OS based application software

Exercise: Linux Booting through different methods
Exercise: Debugging a Linux application using SDK Remote profiling

System Design with a DMA and the Processing System High Performance Slave Port

- Integrating the AXI CDMA
- Standalone Application
- Linus OS based Application

Exercise: Running a Standalone CDMA Application
Exercise: Running a Linux CDMA Application

Custom Peripheral (IP) Creation and Insertion

- Creating a Peripheral IP
- Importing the Peripheral
- Linux Base Device Driver Development
- Loading Module into running kernel
- Application execution

Exercise: Creating our own Intellectual Property and Device Driver for Linux OS; and executing the application