# Designing with ALTERA SoC.

**Obiettivi:**
Il corso ha lo scopo di fornire tutte le necessarie informazioni per programmare le moderne FPGA SoC di Altera quali Cyclone V o Arria10. Questa famiglie di FPGA rappresentano al momento lo stato dell'arte poiche' integrano al loro interno grandi quantita' di logica insieme a processori ARM dual core, permettendo quindi lo sviluppo di applicazioni innovative, molto potenti e compatte. Il corso trattera' sia gli aspetti teorici di introduzione all'architettura sia comprendera' attivita' pratiche su demo boards Altera. La prima parte trattera' gli argomenti piu' di natura hardware mentre la seconda sara' piu' rivolta verso lo sviluppo degli applicativi software per le CPU ARM compreso anche lo sviluppo sotto il sistema operativo Linux. Il corso verra' tenuto da un docente della ditta HandsOnTraining Lmt che e' leader in Europa in questo genere di attivita' formativa.

**Target:**
Il corso si rivolge principalmente a tecnologi e tecnici interessati ad apprendere l'utilizzo di queste famiglie di logiche programmabili.

**Date:**
Il corso avra' una durata complessiva di 32 ore, avra' inizio lunedi' 12 maggio alle ore 14 ed avra' termine venerdi' 16 maggio alle ore 13. Per i giorni di martedi', mercoledi' e giovedi' e' previsto un quick lunch in un'aula adiacente a quella del corso.

**Organizzazione logistica:**
Il corso si terra‡ presso l'aula 133, I^ piano della Sezione di Pisa dell'Istituto Nazionale di Fisica Nucleare di Pisa, in Largo Pontecorvo 3. Si trattera' di un corso teorico-pratico basato sull'utilizzo di demo boards Altera appositamente sviluppate (equipaggiate con FPGA Cyclone V). Saranno disponibili circa 10 schede. Ad alcuni partecipanti sara' richiesto di portare il proprio portatile con installato il compilatore Quartus e il sistema di sviluppo SoC. Le eventuali licenze saranno fornite direttamente dall'istruttore con congruo anticipo. Una copia delle dispense del corso sara' a disposizione di ogni partecipante.
Il corso sar‡ tenuto il lingua inglese.

**Metodologia didattica:**
Corso teorico- pratico con introduzioni teoriche seguite da brevi esercitazioni pratiche

**N∞ partecipant**i:
Il numero massimo di partecipanti e' limitato a 20.

**Responsabile**:
Franco Spinella – tel. 050-2214203
franco.spinella@pi.infn.it

**Segreteria**:
Erika Cioli - tel. 050 2214329
erika.cioli@pi.infn.it

**INFORMAZIONI LOGISTICHE**
Hotel vicini alla sede della riunione da prenotare a cura dei partecipanti:

10-15 minuti a piedi

Hotel Verdi
http://www.verdihotel.it/
prezzo da Ä 50 a Ä 70 (chiedere prezzo in convenzione INFN )

Royal Victoria Hotel ***
http://www.royalvictoria.it/
prezzi da Ä 66 a Ä 102

Hotel Minerva ***
http://www.hotelminervapisa.it/
prezzi da Ä 60,00 a Ä 70,00

Hotel Di Stefano ***
http://www.hoteldistefano.it/index.html
prezzi da Ä 89,00

Bed and Breakfast Relais Dei Fiori
http://www.relaisdeifiori.com/
prezzi da Ä 75,00

Hotel Leonardo***
http://www.hotelleonardopisa.it
prezzi da Ä 50 a Ä 60

Hotel Villa Kinzica***
http://www.hotelvillakinzica.com/
prezzi da Ä 60

Hotel Ariston
http://www.hotelariston.pisa.it/
prezzi da Ä 70

pi˘ di 30 minuti a piedi

Grand Hotel Bonanno ****
http://www.grandhotelbonanno.it
prezzi da Ä 75 a Ä 90

## PROGRAMMA

This course provides all theoretical and practical know-how to design ALTERA SoC devices under Quartus II software.
The course combines 60% theory and 40% practical work on ALTERA SoC evaluation boards.
The course starts with SoC families overview and their capabilities, continues with deep methodic training of the SoC architecture.
The course teaches the HPS architecture and its building blocks, how to manage SoC system, how to configure system based on SoC, how to transfer data through the Bus system and internal interconnect, how to connect external memories, how to build a system with Qsys, how to handle interrupts, and how to use efficiently pin muxing.
The second part of the course focuses on practical use of simulation models (BFMs), and creating SoC test-benches.
The third part of the course teaches the software development flow for the SoC, how to perform boot process with and without operating system, how to write preloader and bootloader, how to use the flash programmer and OS support.
The course ends with SoC debug interfaces overview, how and when to use them, crosstriggering between CPUs and FPGAs and how to use the system console manager.
Course Duration
4.5 days

Goals
1. Become familiar with ALTERA SoC families, their capabilities and when to use them
2. Understand SoC design hardware and software flow from specification to programming and final verification on the board
3. Integrate IPs into the SoC design (also custom peripherals)
4. Configure the SoC system (clocks, PLLs, Resets, Peripherals)
5. Use efficiently the Qsys tool
6. Use Bus Functional Models (BFMs) to simulate SoC behavior
7. Program and use SignalTap II to verify the SoC functionality
8. Understand and choose the right Boot scheme
9. Handle Interrupts using the Generic Interrupt Controller (GIC)
10. Connect external memories to the SoC
11. Become familiar with the boot process and choose the right Boot scheme
12. Be able to develop preloader and bootloader
13. Work with operating system and bare-metal

Intended Users
Hardware, software and system engineers who would like to design with ALTERA SoC technology

Course Material
1. Simulator: Modelsim
2. Synthesizer and Place & Route: Quartus II
3. ALTERA Cyclone V SoC Evaluation board
4. Course book (including labs)

Table of Contents

o HPS peripherals overview
o On-chip ROM features
o On-chip RAM features
o SDRAM controller features
o HPS SDRAM controller configuration
o Maximizing SDRAM performance
o Considerations when accessing HPS SDRAM from FPGA
• Direct Memory Access Controller (DMA)
o DMA overview
o DMAC data transfer features
o DMAC peripheral flow control features
o HPS DMA capabilities
o When to use and not to use HPS DMA
Day #2
• Hardware Design Flow
o Typical design flow
o Qsys tool
o Automatic interconnect generation
o Create Quartus II project for SoC device
o Start a new system in Qsys
o Add IP to Qsys system
o Add custom components
o Methods to connect components
o HPS in Qsys
o HPS-Nios II system block diagram
o Generate completed system
o Hardware/software design flow overview
o Generated software handoff files
• Avalon and AXI Standards
o Qsys-supported standard interfaces
o Advantages of using standard interfaces
o Avalon-MM interfaces
o AXI overview
o AXI features
o Handshake examples
o AXI write transaction
o AXI read transaction
o Component editor – AMBA support
o AXI specification
o Qsys memory-mapped packet format
o Which protocol to choose: Avalon or AXI?
• HPS Component Configuration
o Hard processor system component
o General options & Boot control
o Events
o General Purpose I/O (GPIO)
o Debug APB
o System Trace Macrocell
o Cross Trigger Interface (CTI)
o Trace port interface
o Boot from FPGA
o AXI bridges
o FPGA-HPS bridge interfaces
o Accessing HPS memory from FPGA
o FPGA-to-HPS SDRAM interface
o Resets
o DMA control
o Interrupts
o GIC overview (SGIs, PPIs, SPIs)
o Peripheral pin multiplexing
o HPS I/O muxing overview
o Ethernet
o Other peripheral options (QSPI, SPI master, UART)
o Pin usage & conflicts
o HPS pin assignments
o HPS clocks
o SDRAM embedded memory interface
LAB #1: Creating an ARM based SoC system using Qsys

• HPS Simulation
o Bus Functional Models (BFMs)
o Simulation flow
o Slave component testing
o Master component testing
o HPS system testing
o HPS simulation support – interfaces
o Generate Testbench Qsys system
o Testbench directory structure
o Qsys Testbench system – HPS system
o Writing the test – AXI BFM API overview
o Testbench example
o Using conduit BFMs
o Run simulation script
Day #3
• Software Design Introduction
o SoC EDS overview
o ARM DS-5 Altera edition
o Altera SoC EDS additional deliverables
o Licensing
o Installed files
o SoC EDS folder structure
o Golden Hardware Reference Design (GHRD)
o GHRD in Qsys
o GHRD SD image
• Software Design Flow
o Software initial board bring-up flow
o Software OS/application development flow
o Hardware/software design flow overview
o Generated software handoff files
o Coordinated multi-channel delivery
o Rocketboards
o Typical compiler flow
o Make overview
o Example application
o Build manually
o Build using makefile
o Invoking makefile
o Licensing the Exclipse for DS-5
o Populated workspace
o Importing projects
o Compiling projects
o Debugging and running application
o Perspectives (registers view, variables view, functions view, APP console view)
HPS Booting
o HPS boot stages
o Boot schemes – independent
o Boot schemes – FPGA first
o Boot schemes – HPS first
o Boot sources
o Boot scenarios
o HPS Power On/Reset
o HPS Boot ROM (cold and warm)
o HPS preloader
o HPS user Bootloader
o HPS Linux OS start up
o Boot image & Flash image
o Physically bad boot images
o Logically bad boot images
o What happens when we ìRun Outî of images?
o Booting CPU 1
o HPS state on Preloader entry
o Multiprocessing – AMP vs SMP
HPS Flash Programmer
o HPS Flash programmer overview
o Two parts of HPS Flash programmer
o Command line utilities

PreLoader
o PreLoader introduction
o PreLoader tools
o PreLoader flow
o Launch PreLoader support package generator GUI
o BSP editor
o PreLoader command line
o PreLoader support package files
o Compiling PreLoader
o PreLoader generated files
LAB #2: Creating the PreLoader for an ARM based SoC
• User Bootloader
o User Bootloader introduction
o What is U-Boot?
• Hardware Libraries
o What is 'Bare-metal' programming
o Benefits of Bare-metal solution
o Disadvantages of Bare-metal
o Bare-metal usage scenarios
o SoC Boot phases
o Hardware libraries (HWLibs)
o HWLibs components
o APIs in HWMgr
o HWLibs documentation
o SoCAL address map
o SoCAL component interface
o SoCAL HPS config register
o HWMgr API reference
o Example API
o Bare-metal project
LAB #3: creating and debugging a Bare-metal system for an ARM based SoC with HWLibs
Day #4
• OS Support
o Real-Time operating system – what is it?
o HPS real-time restrictions
o Ways to improve HPS real-time performance
o Other real-time options
o Why use RTOS?
o Planned embedded operating systems
o Introduction the Yocto project
o Yocto project components
o Yocto project application development and debug
o Benefits of Yocto project
o How does Altera use it?
o How is Altera's Linux BSP organized?
o Altera in the Linux community
o Getting Linux and support
• Building the Yocto Project
o Concretely
o What is a recipe?
o What is a layer?
o Setting up the Yocto project environment
o How does the Yocto project find the source code to build?
o How to build Altera's BSP with the Yocto project
o Useful Bitbake options for debugging new recipes
o Perform an incremental buold to SoC Linux kernel
LAB #4: Boot and add Linux application
• Device Tree
o Innovative approach
o What's a Linux device tree?
o Why use a device tree?
o How to use a device tree
o Device tree generator
Day #5
• Hardware Debug Overview
o Debug interfaces (JTAG, Ethernet)
SignalTap II Logic debug

System console
FPGA adaptive debugging
o System console overview
Usage examples
System console interfaces
System console GUI launch
System console services
Service types
o SignalTap II cross triggering
Cross triggering
Cross Triggering Interface (CTI)
Altera SoC debug architecture
Export CTI to custom hardware
SignalTap II configuration for cross trigger
o ARM DS-5 debugger
Debug perspective – registers view
Run debugger and SignalTap II Logic Analyzer
LAB #5: exercise the FPGA using the system console tool
LAB #6: debugging hardware using SignalTap II Logic Analyzer
• Debugging Advanced Features
o Visualization of SoC peripherals (unified display, FPGA adaptive)
o Cortex-A9 debug
o Synthesizing CPUs on FPGA
o Real-Time trace
o Synchronization of hardware event with trace
o Trace capture buffer
o DS-5 debugger DTSL trace setup
o Event viewer setting
o Viewing events
LAB #7: cross triggering and debug