

ARM CORTEX-A8 SYSTEM DESIGN

Ref : 004753A

Duration : 4 days

OBJECTIVES

- This course takes an **in depth** look at the considerations you will need to take into account when designing a system containing a Cortex-A8 core
- It is aimed at :**
 - Software engineers who not only want to obtain details of how to write software to run on the Cortex-A8, but also wish to obtain an understanding of hardware design issues
 - Hardware engineers who need to understand how to design Cortex-A8 based systems, but also wish to obtain an understanding of the issues of writing software to run on that system

RELATED COURSES

- ARM-7/ARM-9 System Design (Ref.002879A)
- ARM1176 System Design (Ref.003772A)

PARTNERS

- This training course is approved by ARM

PREREQUISITES

- A basic awareness of the ARM is **highly recommended** especially the knowledge of ARM V4T and V5TE instruction sets
- A basic understanding of microprocessors and microcontrollers is recommended
- A basic understanding of digital logic or hardware / ASIC design issues would be useful but not essential
- A basic understanding of assembler or C programming would be useful but not essential
- A basic awareness ARM cores is useful but not essential

PRACTICAL LABS

- For on-site courses, labs can be run under the following environments : Eclipse/RVDS, GNU/Lauterbach simulator
- For open courses, labs are run under Eclipse/RVDS



Contact

Tel : +33 (0)5 62 13 52 32
 Fax : +33 (0)5 61 06 72 60
training@mvd-fpga.com

**Course also available
 customized**

Next sessions, see : <http://www.mvd-fpga.com/en/formationsCalend.html>

TOPICS

First day

INTRODUCTION TO CORTEX-A8

- Block diagram
- ARMv7-A architecture
- Operating modes
- ARM instruction set
- Thumb-2 instruction set
- Thumb-2EE instruction set, replacement of Jazelle
- Program Status register
- Exceptions
- System control coprocessor
- Configurable options

THUMB-2 INSTRUCTION SET

- General points on syntax
- Data processing instructions
- Branch and control flow instructions
- Memory access instructions
- Exception generating instructions
- If...then conditional blocks
- Stack in operation
- Exclusive load and store instructions
- Accessing special registers
- Coprocessor instructions
- Memory barriers and synchronization
- Interworking ARM and Thumb states

INSTRUCTION PIPELINE

- Superscalar pipeline operation
- Studying how instructions are processed step by step
- Instruction cycle timing
- Branch prediction mechanism, BTB and GHB usage

- Guidelines for optimal performance
- Return stack
- Instruction Memory Barrier
- Prefetch queue flush

NEON TECHNOLOGY

- Overview of the NEON media coprocessor
- 10-stage NEON pipeline, processing pipelines, load/store pipeline
- Data types
- NEON instruction set
- VFPv3 architecture
- General purpose registers
- NEON vectorizing compiler support
- NEON coding examples

Second day

UNALIGNED DATA AND MIXED-ENDIAN DATA SUPPORT

- Understanding how unaligned word transfers are handled
- Setting the endian mode for data transfers through CPSR[E]
- Understanding how the bus interface unit re-orders bytes when a big endian transfer is performed

MEMORY MANAGEMENT & PROTECTION

- Introduction to page management
- V7 virtual memory architecture, 16-MB supersection support
- Understanding protection domains
- TLB reload mechanism
- TLB lockdown
- Page table in trusted and untrusted worlds
- TLB software read for debug purposes
- Abort exception management, syndrome registers

- Imprecise aborts

TRUSTZONE

- TrustZone conceptual view
- Secure to non secure permitted transitions
- Related CP15 registers
- L1 and L2 secure state indicators, memory partitioning

HARDWARE IMPLEMENTATION

- Clock domains, CLK, PCLK, ATCLK
- Using clock enable to determine the ratio between input clock and operation clock
- Reset domains, power-on reset, debug and ETM reset
- Power control, dynamic power management
- Wait For Interrupt architecture
- Debugging the processor while powered down

DESIGN FOR TEST

- MBIST, L1 and L2 MBIST controllers, CAMBIST controller
- MBIST registers, selecting test patterns
- ATPG test features
- Wrapper Boundary Register
- Reset handling

Third day

LEVEL 1 CACHES

- Cache organization
- Virtual indexing, physical tagging
- Hash Virtual Address Buffer
- Hardware support for virtual aliasing conditions
- Parity protection
- Write buffer
- L1 caches software read for debug purposes

LEVEL 2 CACHE

- Cache organization
- Physical indexing, physical tagging
- L2 cache transfer policy
- Parity / ECC protection
- Write buffer
- L2 Preload Engine [PLE]
- START, STOP and CLEAR commands
- L2 cache software read for debug purposes

DEBUG UNIT

- Coresight specification overview

- CP14 and memory-mapped registers, utilization of an APB slave interface
- APB port access permissions
- Embedded core debug
- Invasive debug : breakpoints and watchpoints
- Vector catch
- Debug exception
- External debug interface
- Understanding how the Debug unit, the Embedded Trace Macrocell and the Cross-Triggering Interface interact

Fourth day

EMBEDDED TRACE MACROCELL

- Overview
- Exporting the compressed trace information
- Benefits of an Embedded Trace Buffer
- Defining trace trigger conditions
- Context ID tracing
- Instrumentation instructions

THE PERFORMANCE MONITOR UNIT

- Event counting principle
- Configuring the 4 event counters
- Event selection

CROSS-TRIGGER INTERFACE

- ETM, debug logic and PMU interacting with each other
- Trigger inputs and outputs
- Channel interface
- Programmer's model

AXI PROTOCOL

- Topology : direct connection, multi-master, multi-layer
- PL300 AXI interconnect
- Separate address/control and data phases
- AXI channels, channel handshake
- Transaction ordering, out of order transaction completion
- Read and write burst timing diagrams
- Cortex-A8 external memory interface, ID encoding
- Exclusive resource management

APB

- Address decoding stages
- APB interconnect
- APB in AMBA3

DOCUMENTATION

Training manuals will be given to attendees during training in print.

