
ARM SOFTWARE DEVELOPMENT USING REALVIEW

Ref : 002580A

Duration : 3 days

OBJECTIVES

- The main objective of this training is to allow attendees to be autonomous in developing software on an ARM 7 or 9 platform.
- The course explains ARM7 and ARM9 pipeline operation
- The workbook provides the student with an introduction to the tools provided with the ARM Developer Suite version 1.2 (ADS) or ARM RealView Developer Suite version 3.1 (RVDS)
- The ARM assembly instructions are viewed in detail
- The course focuses on cache operation
- Thumb / ARM interworking is described
- ARM debug solutions are explained

RELATED COURSES

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

PARTNERS

- This training course is approved by ARM

PREREQUISITES

- A basic understanding of microprocessors and microcontrollers is recommended
- 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 : CodeWarrior/ADS/AXD, Eclipse/RVDS, Keil μ Vision, GNU/Lauterbach simulator, or IAR Workbench
- 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

THE ARM ARCHITECTURE

- Overview of ARM
- ARM operation modes
- The ARM registers set, register organization summary according to the current mode
- Program Status Registers
- Exception handling, vector table, automatic switch into ARM mode
- Instruction sets : ARM branches and subroutines

ARM PROCESSOR CORE

- ARM7TDMI core signals
- ARM7TDMI block diagram
- The ARM7TDMI instruction pipeline
- ARM7TDMI memory interface
- ARM9TDMI datapaths
- ARM9TDMI pipeline
- Example ARM9TDMI system
- Overview of ARM9E-S, ARM10, StrongARM and Xscale

RealView DEVELOPPER SUITE (RVDS) OVERVIEW

- Using the core tools
- C/C++ compilers key features
- Supplied libraries
- Codewarrior introduction
- Debugging with multi-ICE

RVDS INTRODUCTORY WORKBOOK

- Compiling and running an example
- Creating a header file
- Creating a new project
- Viewing registers and memory

Second day

ARM AND THUMB INSTRUCTION SETS

- Conditional execution and flags
- Branch instructions
- The barrel shifter
- Immediate constants
- Single register data transfer
- Block data transfer
- Stack management
- Coprocessor instructions
- Register access in Thumb
- ARM architecture V5TE new instructions
- Assembler workbooks

ARM / THUMB INTERWORKING

- Switching between states
- Branch exchange example
- Mixing ARM and Thumb subroutines
- ARM to thumb veneer
- Thumb-to-ARM veneer
- Interworking calls
- Interworking using Codewarrior

EXCEPTION HANDLING

- Exception return instructions
- Exception priority
- Vector table instructions
- Chaining exception handlers
- Register usage in exception handlers
- FIQ vs IRQ
- Example C interrupt handler
- Software managed interrupt controller
- Issues when re-enabling interrupts
- C nested interrupt example
- Invoking SWIs
- Data abort with memory management

- The return address

COMPILER HINTS AND TIPS

- Automatic optimisation
- Instruction scheduling
- Tail-call optimisation
- Parameter passing
- Array and structure access
- Loop termination
- Division operations
- Inline assembler
- Stack usage
- Global data layout

Third day

INITIALIZING CACHED PROCESSORS

- Cache basics, associativity, cache lockdown
- Programmer's model
- Cache flushing
- Write buffer, cache write strategy
- Memory management, virtual to physical address mapping
- TLB and translation tables, level 1 and level 2 descriptors
- Address generation with process ID register
- Memory protection, MPU configuration steps
- System control coprocessor

- Example initialisation code
- Tightly coupled memory

EMBEDDED SOFTWARE DEVELOPMENT

- ROM or RAM at 0x0 ?
- ROM/RAM remapping
- Exception vector table
- Reset handler
- Initialisation : stack pointers, code and data areas
- C library initialisation
- Scatterloading
- Linker placement rules
- Long branch veneers
- C library functionality
- Placing the stack and heap
- Debugging ROM image

ARM DEBUG SOLUTIONS

- Debugging with multilCE
- Watchpoints, hardware breakpoints, software breakpoints
- Debug communication channel
- Semihosting
- EmbeddedICE-RTT logic
- Real Time trace
- Instruction trace, data trace
- Trace capture

DOCUMENTATION

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