

## MPC 55XX IMPLEMENTATION

Ref : 00A

Duration : n days

### OBJECTIVES

- The course uses a MPC5554 or MPC5567 board
- The e200 core is studied in detail, especially the MMU, the cache and the SPE instruction set
- The course explains how to develop a generic interrupt handler
- The training highlights data flows between core and peripherals through the internal crossbar switch
- The host programming of eTPU and eMIOS is viewed in details

### RELATED COURSES

- CAN bus training (reference 002601A)
- C language for real-time and embedded applications (002603A)
- eTPU programming (reference 003199A)

### PARTNERS

- This training course is approved by FREESCALE

### PREREQUISITES

- Experience of a microcontroller is mandatory
- Knowledge of CAN and TPU is recommended

**WIND RIVER****NeoMore**

### Contact

Tel : 05 62 13 52 32  
Fax : 05 61 06 72 60  
training@mvd-fpga.com

Course also available  
customized

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

### TOPICS

#### MPC55XX OVERVIEW

- Automotive MPC55XX roadmap
- Internal architecture of the Copperhead (MPC5554)
- Functional pin multiplexing
- Memory map, internal register space

#### e200 CORE

- Differences between the new Book E architecture and the classic PowerPC architecture
- The instruction pipeline
- Integer and floating point execution units
- SPE instruction set, signal processing capability, new data types
- Vector and scalar floating point
- The MMU, 32-entry fully associative TLB, page size selection
- Hardware assist for TLB miss exception
- Page attributes WIMGE
- Process protection, variable number of PID registers and sharing
- TLB initialization
- The 32-kB unified L1 cache, pseudo round-robin replacement algorithm, 8-way set associativity
- 8-entry store buffer
- Cache-related instructions
- ABI : sections
- Book E exception handling
- Core timers
- Nexus emulation
- Watchpoint logic

#### THE INTERRUPT CONTROLLER

- Up to 504 on-chip module interrupt sources
- Software vs hardware vector mode
- Hardware acceleration for ISRs : use of 9-bit vectors
- Preemption, priority management
- External IRQs

#### HARDWARE IMPLEMENTATION

- FMPLL
- Configuration pins
- Reset configuration halfword
- Boot assist module, 4 different boot modes
- MMU configuration after BAM executes
- Initialization sequence
- External bus interface, pinout
- Memory controller with support for SDR flash and SRAM
- Compatibility with the external bus of the MPC5XX
- Support for external master accesses to internal addresses
- Burst support
- Chip-select programming

#### ON-CHIP MEMORIES

- 2 MB on-chip flash
- Integrated ECC
- Censorship protection
- Read while write operation
- Erase and program sequences
- 111 kB on-chip SRAM : general purpose SRAM, cache and eTPU RAMs

#### eDMA AND CROSSBAR

- Autonomous IO control

### DOCUMENTATION

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

- Parallel memory bus architecture, concurrent accesses
- Programmable master priorities on a per-slave basis
- 64 independent channels with link capability
- Parking on slave ports
- Transfer control descriptors, inner and outer loops, modulo feature
- Scatter / gather feature
- DMA channel arbitration
- DMA error reporting

#### THE eTPUs

- Real time hardware events processing, scheduling, priority scheme
- Microengine operation
- New arithmetic, logical and control instructions
- Angle clock hardware
- DMA support
- Dual eTPU shared resources
- Introduction to the eTPU functions QOM, NITC, PWM, SIOP, UART
- Channel service max latency time calculation
- eTPU development tools, Ashware debugger

#### eMIOS

- Introduction to time functions supported by the 24 unified channels
- DMA request per channel
- Pin serialization / deserialization
- eMIOS interrupt requests
- Double action submodules
- PWM submodules, center aligned PWM
- Windowed programmable time accumulation
- Quadrature decode

#### eQADC

- Analog inputs multiplexing
- 12-bit AD resolution
- Queue management, trigger sources
- Conversion queue priority scheme
- Conversion cycle times
- eQADC command / data flow
- Hardware interface
- ADC error correction

#### DSPI

- SPI protocol explanation, master / slave operation
- Command queue
- Flexible programming transfer attributes on a per-frame basis
- Transmit and receive sequences

#### eSCI

- UART basics
- Double buffering
- Wake up mode
- Transmit and receive sequences
- Support for LIN master operation

#### The FlexCAN controllers

- CAN protocol basics
- Message buffer structure
- Mask registers
- Listen-only mode capability
- Receive and Transmit processes
- Error counters