Designing with i.MX8M SoC

The i.MX 8M family of processors brings together high-performance computing, power efficiency, enhanced system reliability and embedded security needed to drive the growth of fast-growing edge node computing, streaming multimedia, and machine learning applications.
At the heart is a scalable core complex of up to four Arm Cortex-A53 cores running up to 1.5GHz plus Cortex-M4 based real-time processing domain at 266MHz.
The i.MX 8M also packs-in 4K video acceleration, 3D graphics GPU and advanced audio capabilities to enable audio-rich applications.

Course Description

i.MX8M is a feature rich processor, encapsulating many sub-modules, which makes it a challenge to comprehend. ***'Designing with i.MX8M SoC'*** is a 4-day training that aims providing a deep introduction with the i.MX8M SoC and bringing you up to speed so you could focus on your business work as soon as possible.
We believe that practical work enhances the learning experience and provides an added value to our students, therefore this training blends both, technical data overview and practical lab work on a hardware platform. During the lab work, you'll take an active part in coding and building functional use-cases, both on Cortex-M4 and Cortex-A53. You'll experience developing applications and kernel modules under Linux, and software under free-RTOS.

Course Goals

- Introduce i.MX8M architecture
- Introduce heterogenous ARM cores: Cortex-A53 and Cortex-M4
- Introduce i.MX8M multimedia, graphics and audio capabilities
- Introduce the clock management, reset, power management
- Introduce the i.MX8M memory architecture and capabilities
- Introduce the i.MX8M Boot process including secure boot
- Introduce the developments tools, used to develop software on both micro-processors

Target Audience

Software engineers that would like developing software and BSP for platforms based on i.MX8M SoC.

Course Duration

4 days

Target platform

Variscite's DART-MX8M Evaluation Kit is used as the development platform for the lab.

Course Prerequisites

- Computer architecture background
- Experience in developing embedded systems
- C knowledge
- Familiarity with ARM architecture is an advantage
- Familiarity with Linux is an advantage

Lab prerequisites

- Windows/Linux (tested on Ubuntu) 64-bit based laptop
  - Minimum free space required: 50GB
- Physical Ethernet connection
- Virtualization (VT-x/AMD-v) is enabled on BIOS level
- Virtualbox 6.0.0 or higher installed

Day 1

➢ Introduction to the i.MX8M Family

- ❖ i.MX applications processor values
- ❖ i.MX processor portfolio
- ❖ i.MX8M key features and main target applications
- ❖ i.MX8M, Mini, Nano capabilities and differences
- ❖ i.MX8M software support and qualification levels

Lab #1: Board and Tools Bring-up

➢ CPU Platform

- ❖ Cortex-M4
  - Cortex-M4 platform
  - M-profile instructions
  - Core register set
  - Processor pipeline
  - Memory map
  - Modes privilege and stacks
  - Interrupts and exceptions
  - Nested Vector Interrupt Controller (NVIC)
  - Memory Protection Unit (MPU)
  - Power management
  - Cortex-M4 use cases

Lab #2: Interrupts – configure an interrupt under MCUXpresso

- ❖ What's new in ARMv8-A
  - Privilege levels
  - A32 vs A64
  - AArch64 registers
  - A64 instruction set
  - AArch64 exception model
  - GIC
  - AArch64 memory model

Lab #3: Interrupts – configure an interrupt under Linux

- Cortex-A53
  - Cortex-A53 platform
  - Cortex-A53 pipeline
  - Cache overview
  - Data cache coherency
  - Memory Management Unit (MMU)
  - Interrupt and bus interfaces
  - Power management

Lab #4: Configure the MPU with Different Access Permissions and Identify Stack Overflow

Day 2

i.MX8M Memory System Overview

- Internal memory (Cache, TCM, OCRAM, ROM, ROMCP)
- Memory controllers overview

i.MX8M DDR Controller

- Controller overview
- AXI port interface
- Initialization sequence
- DDR calibration

i.MX8M ECSPI Controller

- Controller overview
- External signals
- Modes of operation
- Initialization

Lab #5: SPI loopback using user-space SPIDEV

i.MX8M QuadSPI Controller

- Controller overview
- Modes of operation
- Programmable sequence engine
- Flash programming – IP interface
- Flash read – IP and AHB interface

i.MX8M uSDHC Controller

- Controller overview
- External signals
- Transfer modes

i.MX8M GPMI Controller

- Controller overview
- AHB-to-APBH bridge with DMA
- DMA scripting language

i.MX8M Timers

- GPT overview
- Input capture channel
- Output compare channel

i.MX8M Messaging Unit (MU)

- MU overview
- MU memory mapping
- Messaging mechanism
- Messaging use-cases
- MU interrupts
- Remote Processor Msg (RPMSG)
- RPMSG channel
- MU Linux driver

Lab #6: Open a communication channel and transfer messages between the Cortex-A53 and Cortex-M4

Day 3

i.MX8M Clock Management

- Clock Control Module (CCM) overview
- CCM block diagram
- PLLs
- Crystal Oscillator (XTALOSC)
- Low Power Clock gating

Lab #7: CPU governor and frequency control

i.MX8M System Reset Controller (SRC)

- SRC overview
- Reset inputs
- Reset and power-up sequence
- Rest outputs
- External POR
- Internal POR
- Reset inputs & outputs
- Parallel reset requests
- Boot mode control

i.MX8M Power Management

- General Power Controller (GPC) overview
- GPC main features
- Processors modes (RUN, Low power, WAIT, STOP, Deep Sleep)
- Low power mode process (entering and exiting)
- Power Gating Controller (PGC) overview
- Time Slot Control Mechanism
- Power control for A53 platform
- Power control for the M4 platform
- Thermal Management Unit (TMU)

Lab #8: Enter low power mode

i.MX8M SDMA

- SDMA overview
- SDMA block diagram
- SDMA main features
- SDMA Core
- SDMA Scheduler
- Burst DMA unit
- Peripheral DMA unit
- SDMA security support
- SDMA clocks and low power modes
- Linux DMA Engine driver

Lab #9: Measuring the memcpy Function Using SDMA, Cortex-A53, and Cortex-M4

i.MX8M Boot

- System boot overview
- Boot modes
- Boot security configuration
- Device Configuration Data (DCD)
- Boot block activation
- Exception and interrupt handling during boot
- Cortex-A and Cortex-M boot process
- Boot devices supported
- Program image
- Serial downloader
- High Assurance Boot (HAB)

Day 4

i.MX8M Multimedia Overview

- Multimedia components
    - Display interfaces
        - Display Controller Subsystem (DCSS)
        - Enhanced LCD interface (eLCDIF)
    - GPU
    - MIPI_DSI
    - MIPI_CSI
    - Audio interface

- Enhanced LCD Interface (eLCDIF)
    - eLCDIF functional description
    - Write/Read data path
    - eLCDIF interrupts
    - MPU interface
    - VSYNC interface
    - DOTCLK interface
    - DVI interface
    - Alpha blending

Lab #10: Set either DCSS or eLCDIF controller to drive LCD

- Display Controller Subsystem (DCSS)
    - DCSS data path
    - DCSS HDR pipeline
        - HDR10
        - Dolby Vision
        -
    - DCSS frontend
    - Display Timing Generator
    - Context loader
    - Graphics decompression
    - Video decompression
    - Display Prefetch and Resolve Unit
    - Scaler & RTRAM
    - RTRAM_CTRL interface

- GPU fundamentals
    - What is a GPU?
    - Graphics pipeline overview

    - Fixed & programmable pipelines
    - Supported APIs
    - GPU performance

Lab #11: General Purpose GPU programming with OpenCL

- ❖ Video Processing Unit (VPU)
    - VPU G1 overview
    - VPU G2 overview
    - Gstreamer overview
        - Gstraemer pipeline
        - Gstreamer elements & pads
        - Gstreamer communication

Lab #12: Decode and play a VP9 4K Video