FTF FREESCALE TECHNOLOGY FORUM 2014

# MPC57xx e200zx Core Differences

## FTF-AUT-F0345

Shanaka Yapa | Automotive MCU Apps Eng

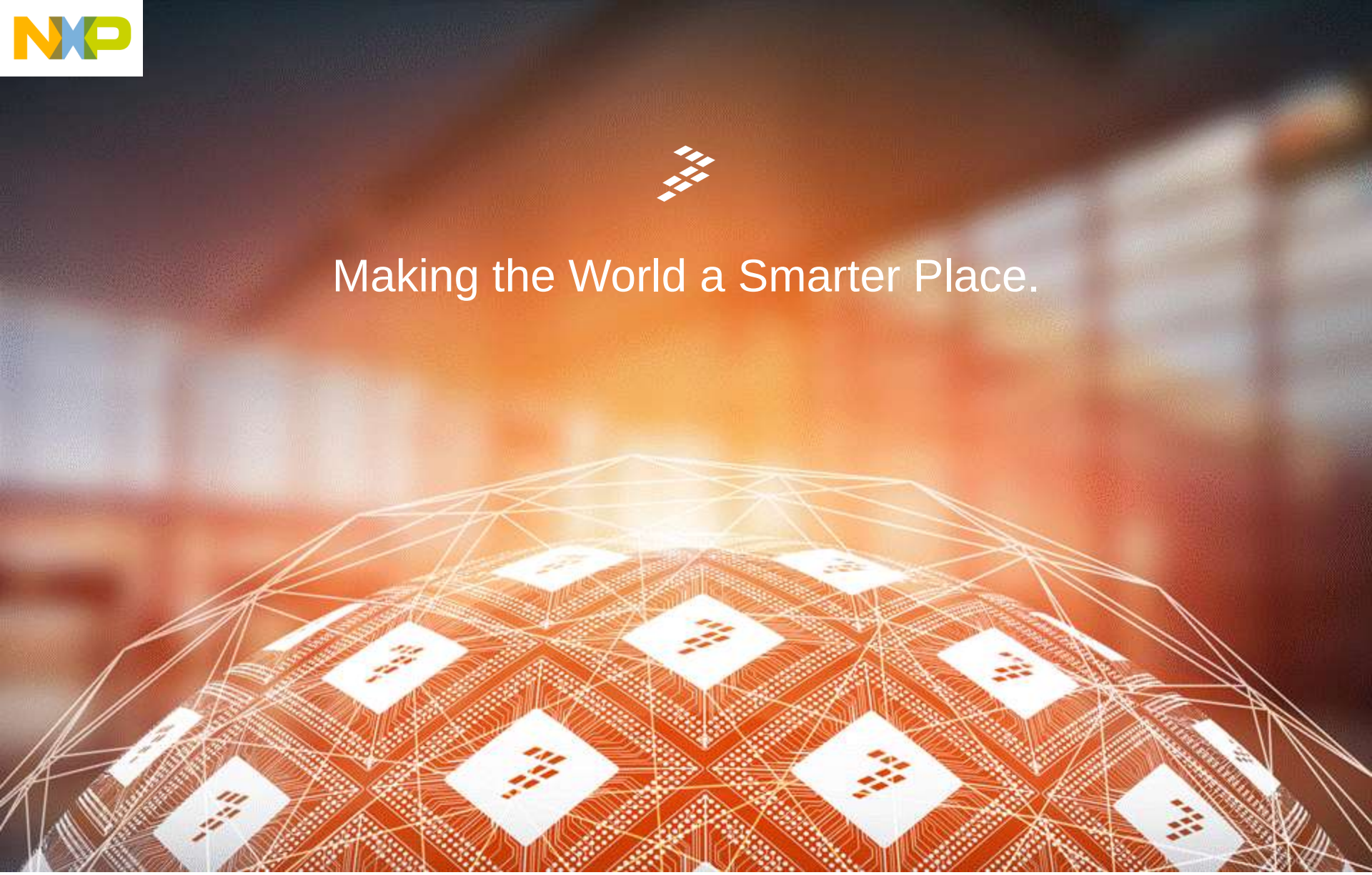
APR.2014

freescale™

External Use

# Agenda

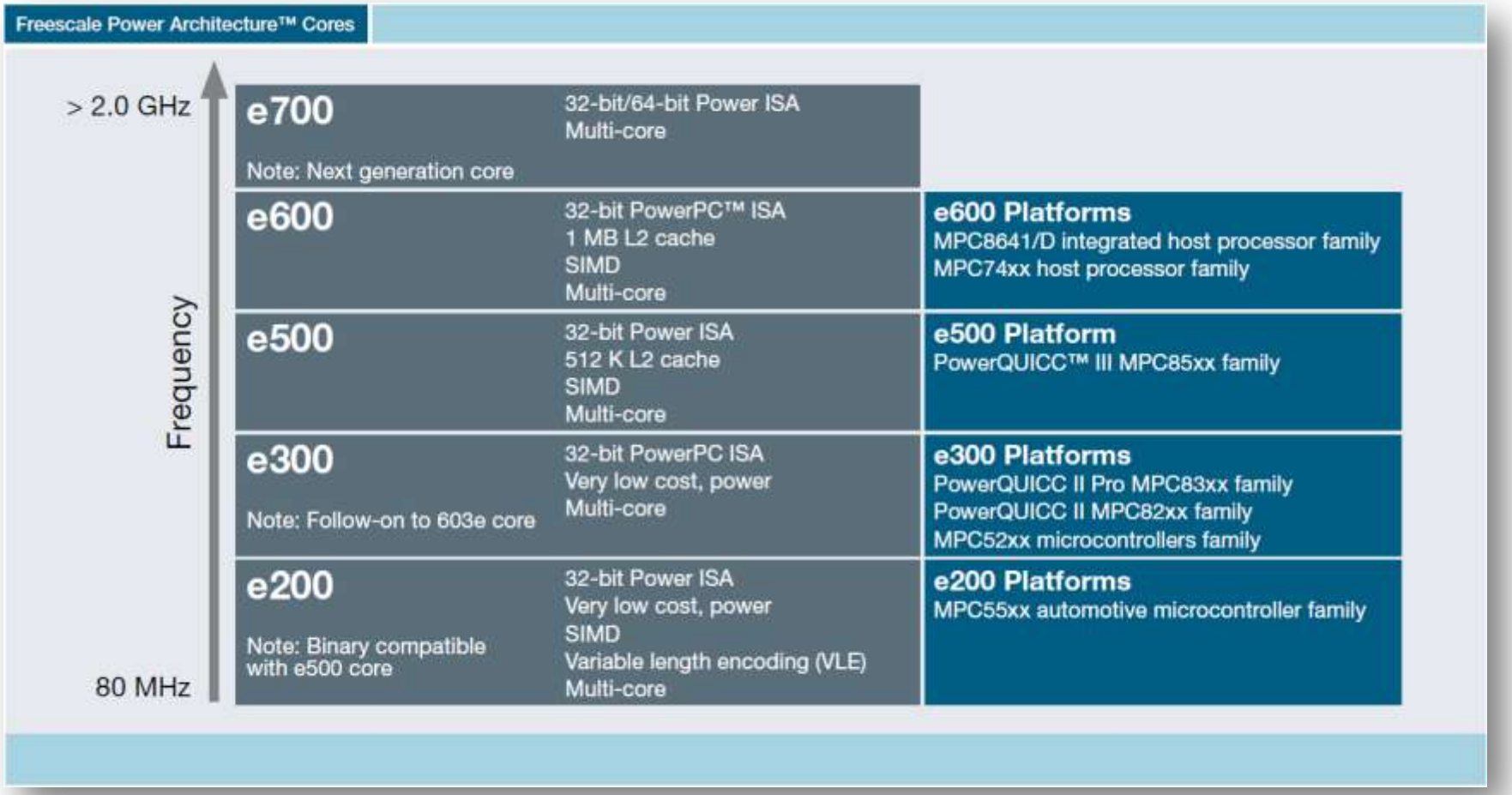
- Abstract
- Automotive MCU Roadmap
- Automotive Power Architecture® MCU Technologies
- MPC57xx Automotive MCU Family & Core Implementation
- Differences in 57xx Core
  - E200zx core execution options
  - E200zx bus interface & memory options
  - E200zx debug options
- Discussion Q&A


Making the World a Smarter Place.

# Abstract

- Freescale is embedding various Power Architecture cores in product lines such as e600x, e500x, e300x and e200x cores
- Automotive e200x is implemented for low power/ high performance and better die size utilization
- Depending on application subsets, Freescale varies features in the core according to market requirements

# Freescale Power Architecture Cores




- Ref: http://www.freescale.com/files/32bit/doc/white_paper/E200CORELCNWP.pdf

# Roadmap

# Qorivva Automotive MCU Roadmap


- eTPU timer system
- GTM timer system
- No eTPU/GTM timer system
- Chassis/Radar focus
- Cluster focus
- Camera focus

Separate security module

Supports functional safety

## Gen 1* (0.25/0.35 µm)

- MPC566
- MPC565
- MPC564
- MPC563
- MPC555
- MPC562
- MPC561

## Gen 2 (130 nm)

- MPC5566
- MPC5554
- MPC5567
- MPC5565
- MPC5561
- MPC5553
- MPC5534
- MPC5510

## Gen 3 (90 nm)

- MPC5676R
- MPC567xF
- MPC567xK
- MPC566xG/E
- MPC564xS
- MPC564xL
- MPC564xB/C
- MPC564xA
- MPC563xM
- MPC560xS
- MPC560xP
- MPC560xE
- MPC560xB/C/D

## Gen 4 (55 nm)

- MPC5777M
- MPC5777C
- MPC5775K
- MPC5746M
- MPC574xG
- MPC574xC
- MPC5746R
- MPC574xP

Production

Execution

1998 * Not considered Qorivva devices.

2003

2008

2012

# Automotive Power Architecture MCU Technologies

| Family | Manufacturing technology | Transistor geometry | Cores used |
|---|---|---|---|
| **MPC555** | CDR1 | 350nm(0.35um) | RCPU (Automotive RISC) |
| **MPC56x Family** | CDR3 | 250nm(0.25um) | |
| **Qorivva MPC55xx Family** | HiP7 | 130nm(0.13um) | e200z0, e200z1, e200z3, e200z6 |
| **Qorivva MPC56xx Family** | C90 | 90nm | e200z0, e200z3, e200z4, e200z6, e200z7 |
| **Qorivva MPC57xx Family** | C55 | 55nm | e200z0, e200z2, e200z4, e200z7 |

# MPC57xx Automotive MCU Family & Core Implementation

# Core Usage vs. MPC57xx Automotive MCU Family

| Device | Core 0 | Core 1 | Core 2 | Lock-step Core1 |
|---|---|---|---|---|
| MPC5748/7/6G Calypso | e200z4204n3† | e200z4204n3 | e200z210n3 | NA |
| MPC5744P Panther | e200z4251n3 | NA | NA | Core 0 (e200z424) |
| MPC5746R Rainier | e200z425n3 | e200z425n3† | NA | Core 0 (e200z424) |
| MPC5746M McKinley | e200z410n3 | e200z410n3 | e200z425Bn3† | Core 0 (e200z409) |
| MPC57777M Matterhorn | e200z710n3 | e200z710n3 | e200z425Bn3† | Core 0 (e200z709) |
| MPC5777C Cobra55 | e200z759n3† | e200z759n3 | NA | Core 1 (e200z758) |
| MPC5775K Racerunner | e200z4201n3† | e200z7260n3 | e200z7260n3 | Core 0 (e200z419) |

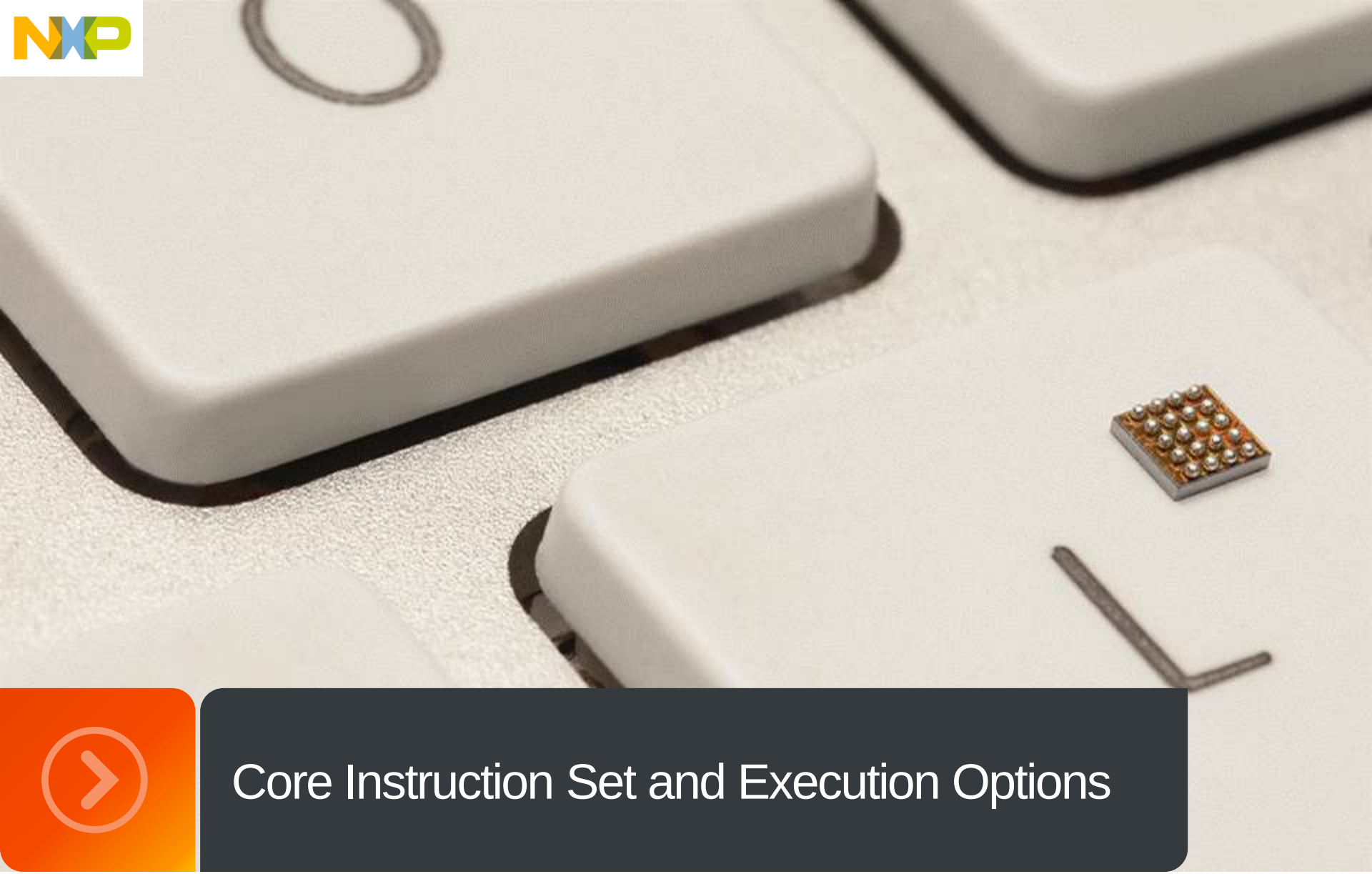
- † Initial default boot core that executes code from the Boot Assist Flash (BAF). Debuggers have access to this core initially as well.

# Differences in Core System in MPC57xx

- Core instruction set and execution options
  (signal processing instruction options, saturated math instructions, floating point type options, instruction issue option, lock-step option, etc.)

- Core bus interface options
  (memory protection options, end-to-end error correction code options, crossbar bus width, local memory options, and cache options)

- Core debug options
  (Nexus class, timestamp option, trace port width, and whether it has the fixed JTAG Nexus register access sequence)



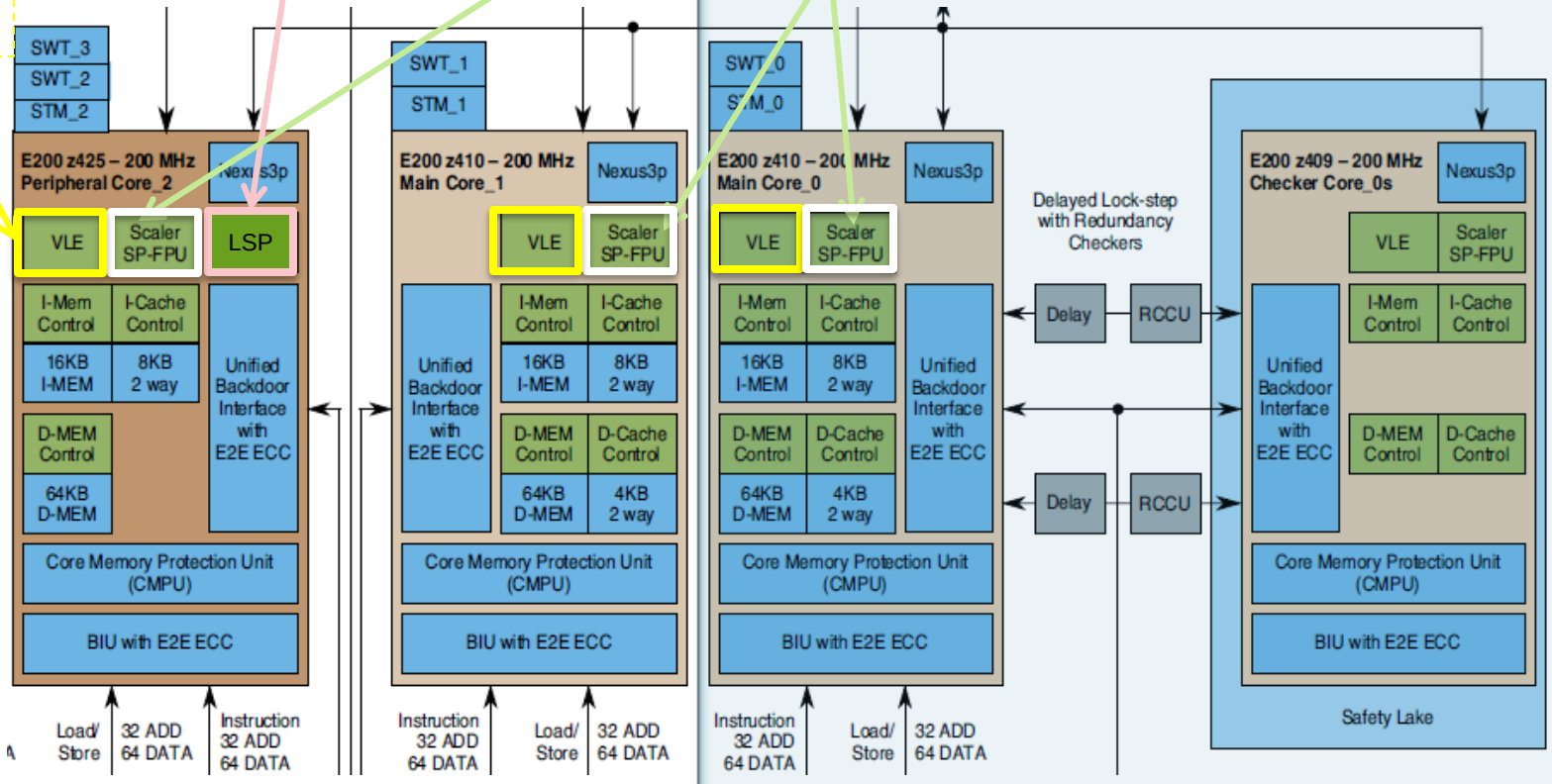
# Core Instruction Set and Execution Options

# MPC5746M Core Overview

Variable Length Encoding

Signal processing

Floating point operations




e200z425Bn3

Initial boot-up core

e200z410n3

e200z410n3

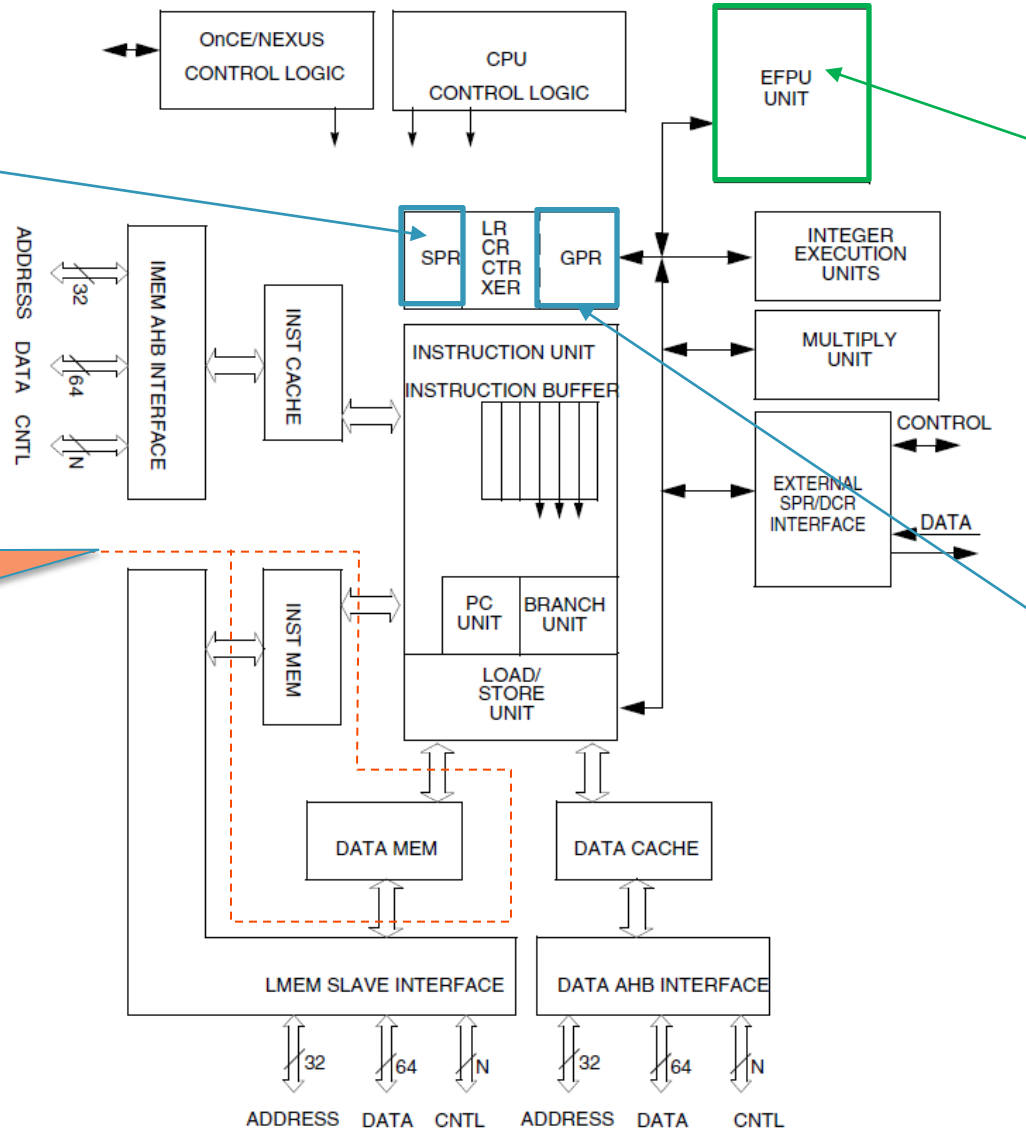
Lock-step

VLE : Variable length encoding

Ref: MPC5746M reference manual rev 4

# Core Architecture

Special Purpose Registers




FPU
Supports SPE x.x
Or
Supporst LSP

**S**ignal **P**rocessing **E**ngine or
**L**ightweight **S**ignal **P**rocessing

Core Local Memories with 0 wait access

32 General Purpose Registers



Ref: MPC5746M reference manual rev 4

# Instruction Sets

- **Book E Support**
  - Previous e200z3 through e200z7 cores supported both the Power Architecture Book E instruction set and a more size-efficient Variable Length Encoded (VLE) instruction set. The previous e200z0 cores used on some of the MPC5500 and MPC5600 devices supported only the VLE instruction set. Most new MPC57xx devices support only the VLE instruction set; however, there are few parts where we support both Book E & VLE instruction sets.

- **Dual Issue**
  - Some versions of the cores can issue two instructions and execute two instructions at a time. Other versions of the core will only execute a single instruction at a time. There are limitations on the instructions that can execute concurrently. Each core in an MCU could be defined as either dual- or single-issue variation for a particular MCU.

- **Lock-step Type**
  - For safety applications, two cores execute instructions in parallel and compare the results to identify errors in the processor core. To reduce current requirements of the two cores running in parallel, one of the cores can be delayed clock. This spreads the peak current requirements of the core. Not all cores support the lock-step option. In the previous MPC56xx Family, the cores implemented a true lock-step where the two lock-step cores execute completely in parallel, with no delay.

# Instruction Sets

- **General Purpose Registers**
  - Although all of the e200zx cores are 32-bit Power Architecture cores, some versions of the cores support 64-bit registers that can be used by the Signal Processing Engine (SPE) Auxiliary Processing Unit (APU). Not all devices support the SPE APU.
- **Signal Processing Instruction support**
  - There are several options for signal processing extensions to the core, either the Lightweight Signal Processing Unit or the Signal Processing Extension. These are shown in the following table. Some cores do not support any signal processing instructions but will support the full Book E instruction set.
- **Saturation Instruction Support**
  - To better support AutoSAR math functions, saturated math instructions are supported on some variations/versions of the cores. This allows optimization of some AutoSAR system calls to a single instruction.
- **Floating Point**
  - Basic floating point instructions are supported on some variations of the cores. These can be defined as part of the base Power Architecture core. Either scalar or vector floating point options are supported with single-precision data format. Vector mode is implemented in the SPE APU.

# LSP / SPE / SPE2 Comparison

# Register Set

- LSP
  - 32x32-bit register set
- SPE
  - 32x64-bit register set
- SPE2
  - 32x64-bit register set

LSP will have more register pressure because of less register space. Algorithms need to be carefully planned out to balance loads/stores with math operations. Data arrangement becomes more important.

# Single Instruction Multiple Data (SIMD)

- LSP
  - 2-way for byte and halfword
  - 1-way for word (2-way using register pairs)
  - No floating point
- SPE
  - 2-way for byte, halfword, and word
  - 2-way for floating point
- SPE2
  - 8-way for byte
  - 4-way for halfword
  - 2-way for word
  - 2-way for floating point

# Floating point

- LSP
  - No vector floating point
  - Floating point is separate unit
- SPE
  - Vector floating point
  - Floating point built in to SPE unit
- SPE2
  - Vector floating point
  - Floating point built in to SPE unit

SPE/SPE2 floating point algorithms ported to LSP device will run significantly slower.

Devices without LSP will still have floating point.

# Saturation Instruction Support

# Instruction Support for Arithmetic Saturation

- Applies to basic arithmetic operators: add, sub, multiply, divide
- For specific boundary cases, limit result to max/min value
  - Consider a signed 32-bit addition: 0x7FFF_FFFF + 0x0000_0001
    - Addition of max positive number + 1 crosses a numeric boundary
    - Standard binary arithmetic generates result of 0x8000_0000, but this is the maximum negative number
    - These types of undetected numeric discontinuities can have disastrous implications. See http://cs.furman.edu/digitaldomain/themes/risks/risks_numeric.htm
    - For this example, a saturated signed addition returns max positive number ADDSS (signed saturation) (0x7FFF_FFFF, 0x0000_0001) = 0x7FFF_FFFF

# AutoSAR Fixed Point Math Routines

- 55-page specification
- Defines API, functions
- *Hundreds* of functions
- For example, 36 ADD functions defining virtually every combination of input and output variables as signed/unsigned, 8-,16- or 32-bit values
- Simplest example:

```
sint32 Mfx_Add_s32s32_s32(sint32, sint32);
```

"Return-value shall be saturated to boundary values in the event of underflow or overflow."

AUTOSAR

Specification of Fixed Point Math Routines
V1.2.0
R4.0 Rev 3

## 1 Introduction and functional overview

This specification specifies the functionality, API and the configuration of the AUTOSAR library dedicated to arithmetic routines for fixed point values.

This mathematical library (MFX) contains the following routines :

- addition
- subtraction
- absolute value
- absolute value of differences
- multiplication
- division
- combination of multiplication and division
- combination of multiplication and shift right
- combination of division and shift left
- modulo
- limitation

Some of these functions are proposed too for $2^n$ Scaled Integers :

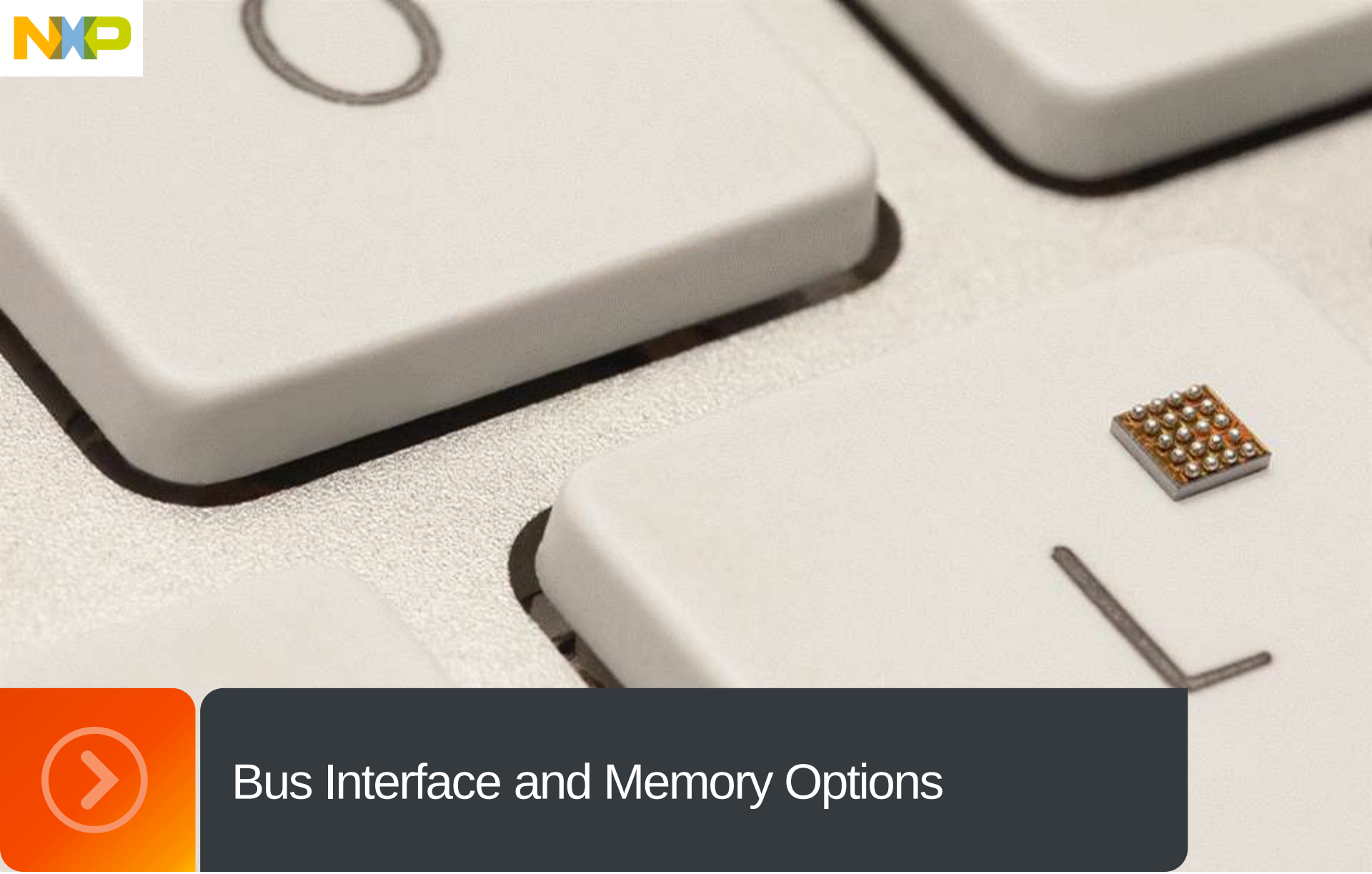
- addition
- subtraction
- absolute value
- absolute value of differences
- multiplication
- division
- conversion (specific to $2^n$ Scaled Integers)

All routines are re-entrant and can be used by multiple runnables at the same time.

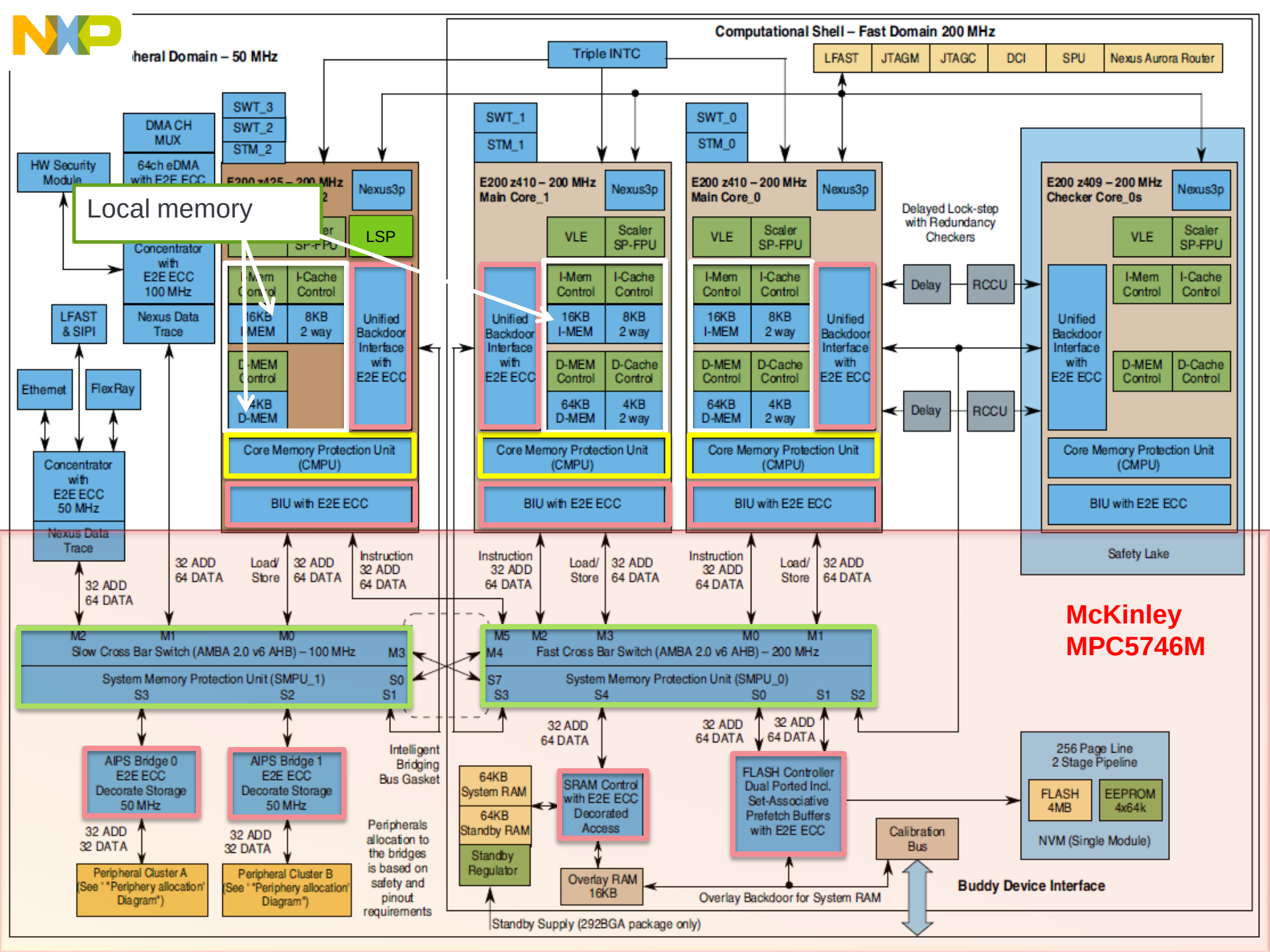
# Core Instruction Set and Execution Options

| Device | Instantiation | Core Name | Book E/VLE | Dual/Single Issue | Lockstep | GPR | Signal processing | Saturation | Floating point |
|---|---|---|---|---|---|---|---|---|---|
| Calypso MPC5748/7/6G | Core2 | e200z210n3 | VLE | Single | No | 32x32 | No | No | No |
| | Core0/1 | e200z4204n3 | VLE | Dual | No | 32x32 | No | No | Scalar |
| Panther MPC5744P | Core0 | e200z4251n3 | VLE | Dual | Yes | 32 x 32 | LSP | No | Scalar |
| Rainier MPC5746R | Core 0/1 | e200z425n3 | VLE | Dual | Yes | 32 x 32 | LSP | Yes | Scalar |
| McKinley MPC5746M | Core 0/1 | e200z410n3 | VLE | Single | Yes | 32 x 32 | No | Yes | Scalar |
| | Core 2 | e200z425Bn3 | VLE | Dual | No | 32 x 32 | LSP | Yes | Scalar |
| Matterhorn MPC57777M | Core 0/1 | e200z710n3 | VLE | Single | Yes | 32 x 32 | No | Yes | Scalar |
| | Core 2 | e200z425Bn3 | VLE | Dual | No | 32 x 32 | LSP | Yes | Scalar |
| Cobra55 MPC5777C | Core 0/1 | e200z759n3 | BookE /VLE | Dual | Yes | 32 x **64bit** | SPE1.1 | No | **Vector** |
| Racerunner MPC5775K | Core0 | e200z4201n3 | VLE | Dual | Yes | 32 x 32 | No | No | Scalar |
| | Core1/2 | e200z7260n3 | BookE /VLE | Dual | No | 32 x **64bit** | SPE2 | No | **Vector** |

N indicates the initial boot-up core



# Bus Interface and Memory Options


Computational Shell – Fast Domain 200 MHz
Triple INTC
LFAST
JTAGM
JTAGC
DCI
SPU
Nexus Aurora Router
SWT_3
SWT_2
STM_2
SWT_1
STM_1
SWT_0
STM_0
DMA CH MUX
64ch eDMA with E2E ECC
HW Security Module
Local memory
Nexus3p
LSP
E200 z410 – 200 MHz Main Core_1
E200 z410 – 200 MHz Main Core_0
E200 z409 – 200 MHz Checker Core_0s
VLE
Scaler SP-FPU
I-Mem Control
I-Cache Control
16KB I-MEM
8KB 2 way
D-MEM Control
D-Cache Control
64KB D-MEM
4KB 2 way
Unified Backdoor Interface with E2E ECC
Core Memory Protection Unit (CMPU)
BIU with E2E ECC
Delayed Lock-step with Redundancy Checkers
Delay
RCCU
Safety Lake
Concentrator with E2E ECC 100 MHz
Nexus Data Trace
LFAST & SIPI
Ethernet
FlexRay
Concentrator with E2E ECC 50 MHz
Nexus Data Trace
32 ADD 64 DATA
Load/ Store
Instruction 32 ADD 64 DATA
Slow Cross Bar Switch (AMBA 2.0 v6 AHB) – 100 MHz
System Memory Protection Unit (SMPU_1)
Fast Cross Bar Switch (AMBA 2.0 v6 AHB) – 200 MHz
System Memory Protection Unit (SMPU_0)
M0
M1
M2
M3
M4
M5
S0
S1
S2
S3
S4
S7
McKinley MPC5746M
AIPS Bridge 0 E2E ECC Decorate Storage 50 MHz
AIPS Bridge 1 E2E ECC Decorate Storage 50 MHz
32 ADD 32 DATA
Peripheral Cluster A (See "Periphery allocation" Diagram")
Peripheral Cluster B (See "Periphery allocation" Diagram")
Intelligent Bridging Bus Gasket
Peripherals allocation to the bridges is based on safety and pinout requirements
64KB System RAM
64KB Standby RAM
Standby Regulator
SRAM Control with E2E ECC Decorated Access
Overlay RAM 16KB
Overlay Backdoor for System RAM
Standby Supply (292BGA package only)
FLASH Controller Dual Ported Incl. Set-Associative Prefetch Buffers with E2E ECC
Calibration Bus
256 Page Line 2 Stage Pipeline
FLASH 4MB
EEPROM 4x64k
NVM (Single Module)
Buddy Device Interface

- **Memory Protection Unit (MPU)**
  - The Memory Protection Unit (MPU) allows memory regions to be protected from being accessed by certain cores, protected from being modified by certain cores, protected from core execution, for enabling or disabling of being cached, and for other safety protection features.

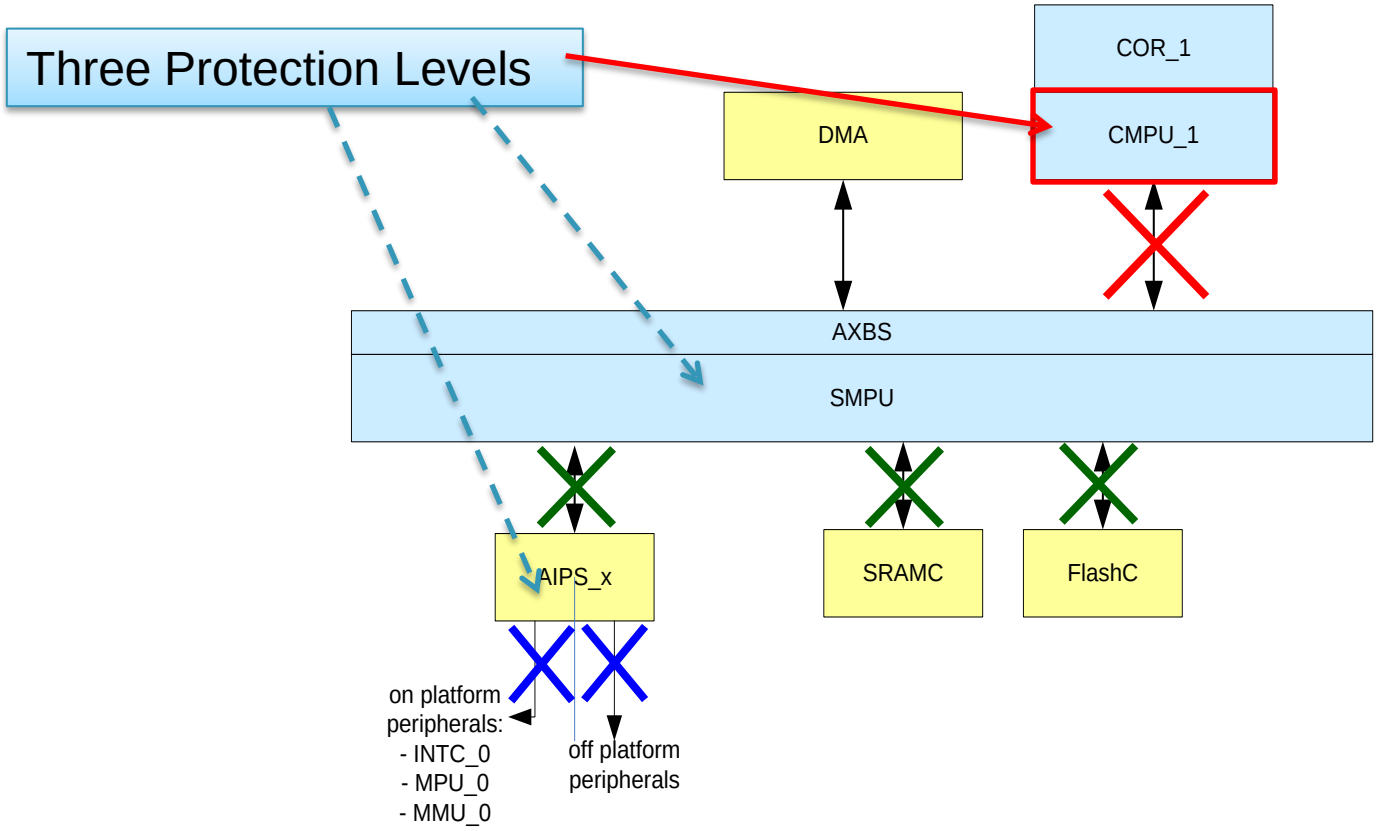
- **Memory Management Unit (MMU) –**
  - The Memory Management Unit is similar to the Memory Protection unit in that it allows memory regions to be protected from being accessed by certain cores, protected from being modified, and for enabling/disabling of the cache for that memory region. However, it also supports the capability of remapping a logical (or virtual) memory address into a physical hardware memory address, including the capability to map addresses based on a Process ID. The MMU is much more complicated than the MPU.

- **End-to-End Error Correction Coding (e2eECC) –**
  - End-to-end (e2e) Error Correction Coding (ECC) provides an additional layer of safety by including ECC on all bus transactions. The ECC for the transfer is generated on the transmitting end of the transaction and checked at the receiving end. e2eECC is an optional feature that can be implemented with the core and system interfaces.

- **Cross-Bar (XBAR) interface –**
  - In general, the e200zx cores have traditionally implemented a 64-bit bus interface to the Cross-Bar switch. On the newer e200zx cores, the instruction bus interface remains 64-bit, however, the data load/store bus on some versions of the cores are implemented with a 32-bit interface to save power and

# MPU

# MPUs: Protection Against Software Storage Interference




**Core-MPU** **System-MPU** **AIPS-Prot.**

# Core MPU (CMPU) Capabilities

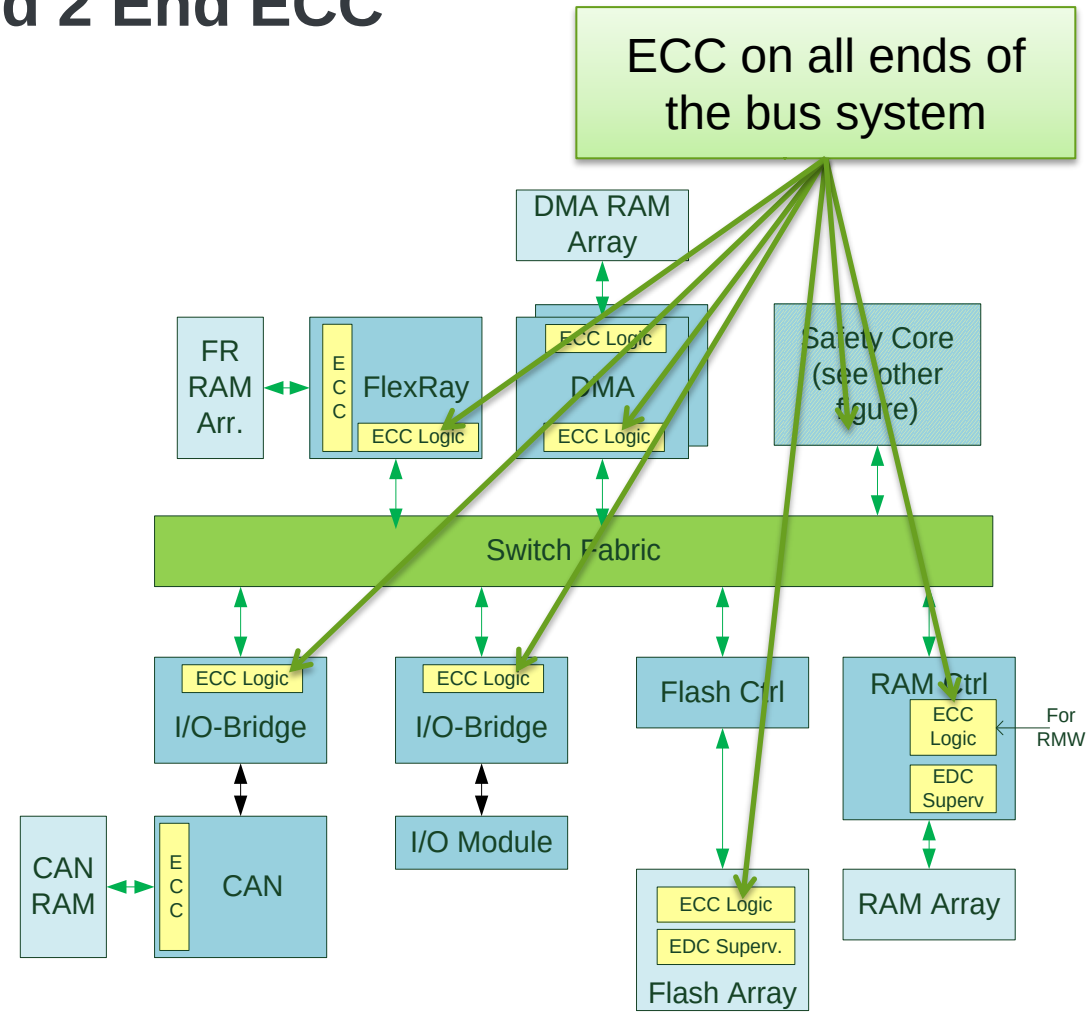
- The core MPU supports the following key features:
  - Process ID aware
  - 24-entry fully associative region descriptor table in Panther
  - Protection control:
    - Ability to set access permissions and memory attributes on a per-region basis
    - Capability of masking upper address bits in the range comparison
    - Capability of bypassing permissions checking for selected access types
    - Per-entry write-once logic for entry protection
  - SW instructions for reading/writing MPU entries (mpure, mpuwe) and special purpose register for the MPU assist registers

# System MPU (SMPU) Capabilities

- The System MPU supports the following key features:
  - Bus master ID aware (not process ID aware)
  - 16 region descriptor table in Panther
  - Protection control:
    - Ability to set access permissions and memory attributes on a per-region basis
    - Region size from 1 byte to 4 GBytes
    - Per-entry write-once logic for entry protection

# E2E ECC

# End 2 End ECC




- ECC for master-slave accesses via XBAR
- Every memory with ECC
- ECC on address and data
- EDC after ECC, redundancy of ECC logic to avoid failure in ECC itself

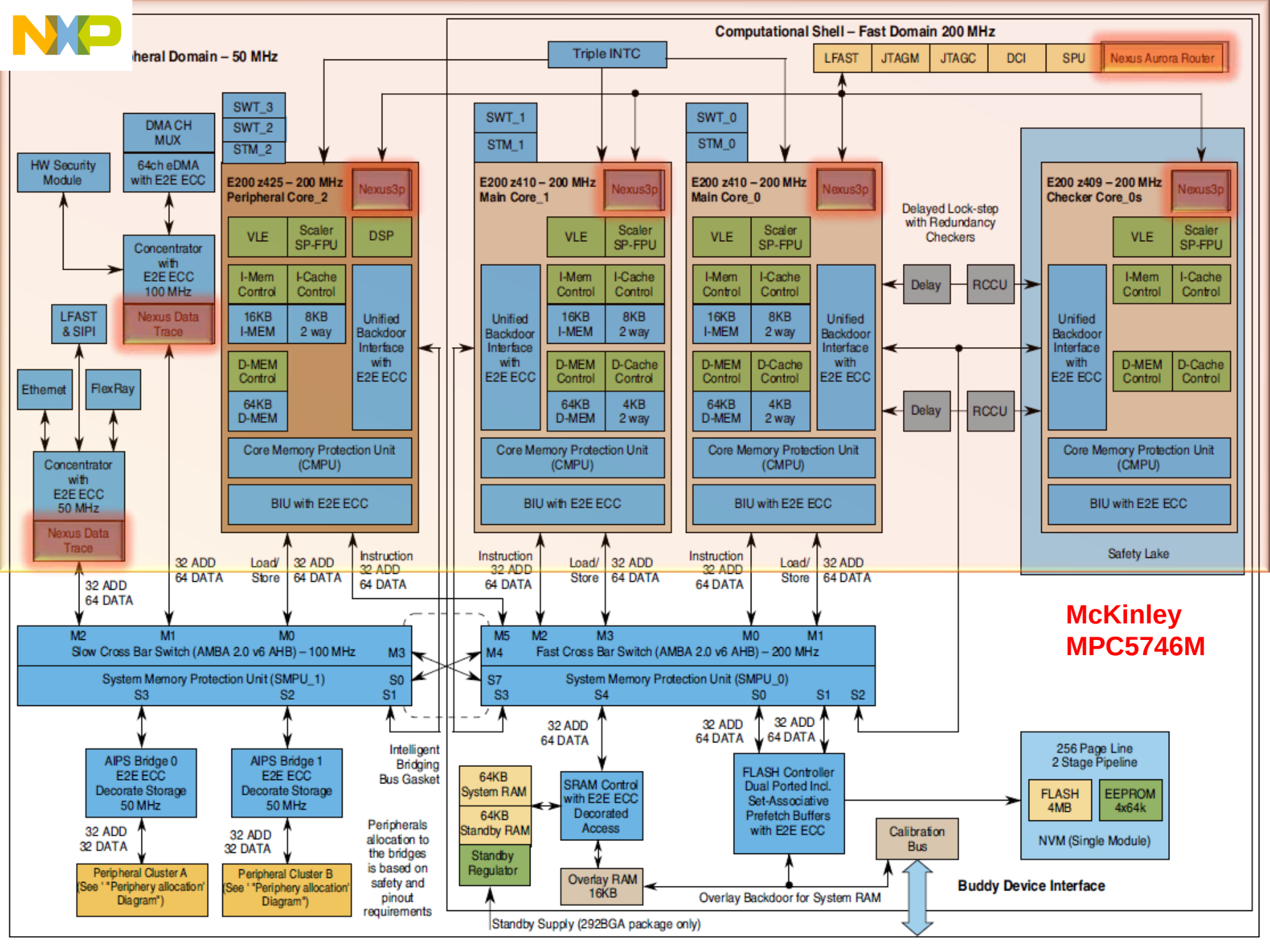
- **Local Data Memory (DTCM) :**
  - Some cores support a fast local data memory (SRAM). This is sometimes referred to as a tightly coupled memory (DTCM). The size of this SRAM is definable for a given core variant. Not all devices implement local memory. The local data memory allows for fast access of variables that are required frequently by a single core with 0 wait.
- **Local Instruction Memory (ITCM) :**
  - Some cores support a fast local instruction memory (SRAM). This is sometimes referred to as a tightly coupled memory (ITCM). The size of this SRAM is definable for a given core variant. Not all devices implement local memory. The local instruction memory can be loaded with frequently executed software routines. This is similar to the concept of locking some regions of cache for frequently used subroutines or functions.
- **Data Cache (D-Cache) :**
  - Data cache allows fast access to recently used data from the load/store bus from memory space outside of the core complex, typically the device internal flash or external1 (to the device) memory. Different sizes of D-cache can be instantiated (as defined for a device) in a particular core.
- **Instruction Cache (I-Cache) :**
  - Instruction cache allows fast access to recently loaded instructions from memory (typically flash). Different sizes of I-Cache can be instantiated (as defined for a device) in a particular core.

# Bus Interface and Memory Option

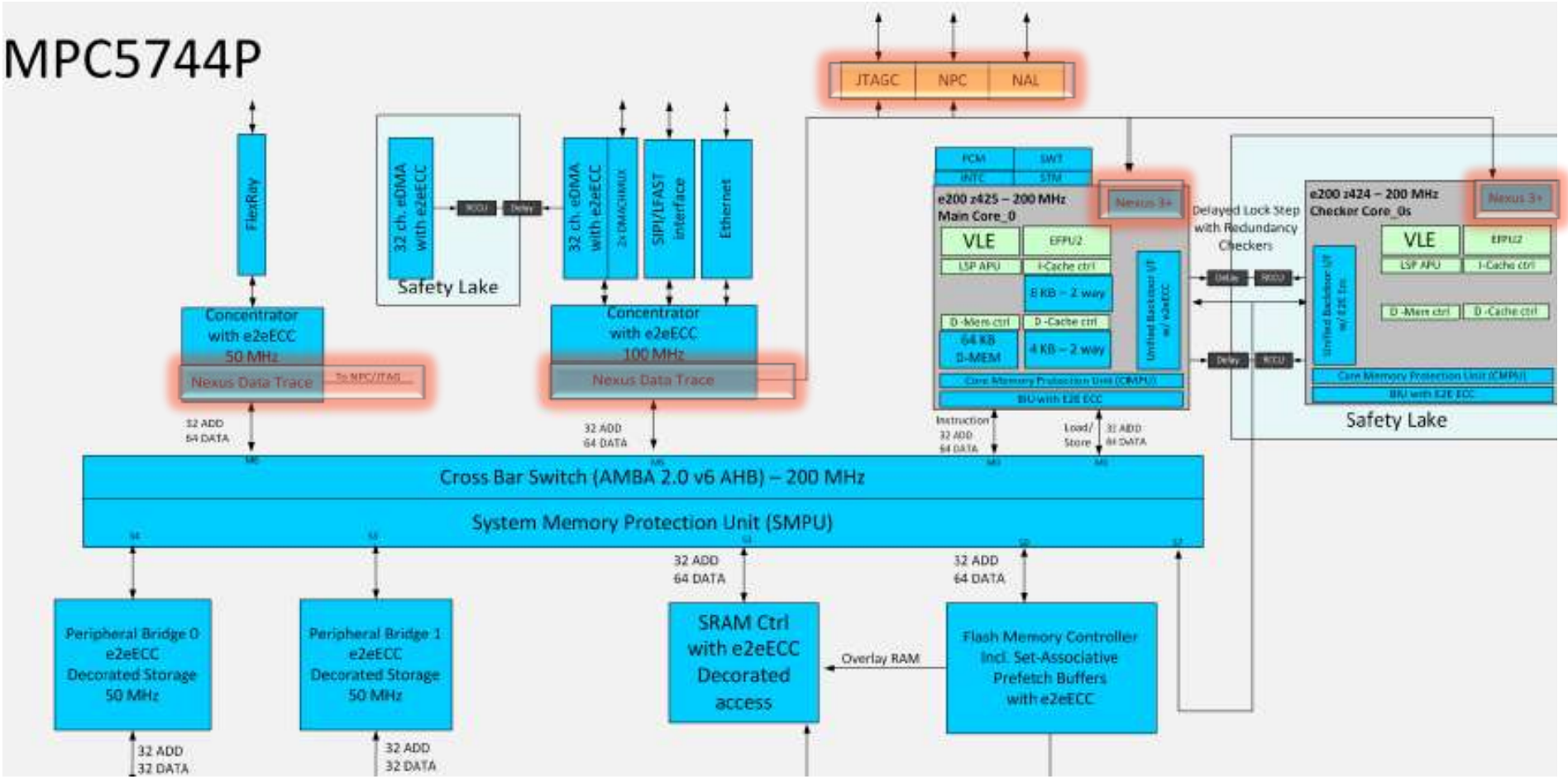
| Device | Revision | Instantiation | Core Name | MPU | e2eECC | XBAR Bus | DTCM | ITCM | D-cache | I-cache |
|---|---|---|---|---|---|---|---|---|---|---|
| Calypso MPC5748/7/6G | 1.0 | Core2 | e200z210n3 | No | Yes | 64I/64D | No | No | No | No |
| | | Core**0**/1 | e200z4204n3 | No | Yes | 64I/64D | No | No | 4Kb | 8Kb |
| Panther MPC5744P | 2.1 | Core**0** | e200z4251n3 | 24 | Yes | 64I/64D | 64Kb | No | 4Kb | 8Kb |
| Rainier MPC5746R | 2.0 | Core 0/**1** | e200z425n3 | 24 | Yes | 64I/64D | 32Kb | 16K | No | 8Kb |
| McKinley MPC5746M | 2.0 | Core 0/1 | e200z410n3 | 24 | Yes | 64I/64D | 64Kb | 16K | 4Kb | 8Kb |
| | | Core 2 | e200z425Bn3 | 24 | Yes | 64I/32D | 64Kb | 16K | No | 16Kb |
| Matterhorn MPC57777M | 2.0 | Core 0/1 | e200z710n3 | 24 | Yes | 64I/64D | 64Kb | 16K | 4Kb | 16Kb |
| | | Core 2 | e200z425Bn3 | 24 | Yes | 64I/32D | 64Kb | 16K | No | 16Kb |
| Cobra55 MPC5777C | 1.0 | Core 0/**1** | e200z759n3 | **MMU** | Yes | 64I/64D | 64Kb | No | 16Kb | 16Kb |
| Racerunner MPC5775K | 2.0 | Core0 | e200z4201n3 | 24 | Yes | 64I/64D | 64Kb | No | 4Kb | 8Kb |
| | | Core1/2 | e200z7260n3 | 24 | Yes | 64I/64D | 64Kb | No | 16Kb | 16Kb |

N : indicate the initial boot-up Core

# Debug Options


Computational Shell – Fast Domain 200 MHz
Triple INTC
LFAST
JTAGM
JTAGC
DCI
SPU
Nexus Aurora Router
SWT_3
SWT_2
STM_2
SWT_1
STM_1
SWT_0
STM_0
DMA CH MUX
64ch eDMA with E2E ECC
HW Security Module
Concentrator with E2E ECC 100 MHz
Nexus Data Trace
LFAST & SIPI
Ethernet
FlexRay
Concentrator with E2E ECC 50 MHz
Nexus Data Trace
E200 z425 – 200 MHz Peripheral Core_2
Nexus3p
VLE
Scaler SP-FPU
DSP
I-Mem Control
I-Cache Control
16KB I-MEM
8KB 2 way
D-MEM Control
64KB D-MEM
Unified Backdoor Interface with E2E ECC
Core Memory Protection Unit (CMPU)
BIU with E2E ECC
E200 z410 – 200 MHz Main Core_1
E200 z410 – 200 MHz Main Core_0
D-Cache Control
4KB 2 way
Delayed Lock-step with Redundancy Checkers
Delay
RCCU
E200 z409 – 200 MHz Checker Core_0s
Safety Lake
32 ADD 64 DATA
Load/ Store
Instruction 32 ADD 64 DATA
M2 M1 M0 M3
Slow Cross Bar Switch (AMBA 2.0 v6 AHB) – 100 MHz
System Memory Protection Unit (SMPU_1)
S3 S2 S1 S0
M5 M4 M2 M3 M0 M1
Fast Cross Bar Switch (AMBA 2.0 v6 AHB) – 200 MHz
System Memory Protection Unit (SMPU_0)
S7 S3 S4 S0 S1 S2
McKinley
MPC5746M
AIPS Bridge 0 E2E ECC Decorate Storage 50 MHz
AIPS Bridge 1 E2E ECC Decorate Storage 50 MHz
32 ADD 32 DATA
Peripheral Cluster A (See "Periphery allocation" Diagram")
Peripheral Cluster B (See "Periphery allocation" Diagram")
Intelligent Bridging Bus Gasket
Peripherals allocation to the bridges is based on safety and pinout requirements
64KB System RAM
64KB Standby RAM
Standby Regulator
SRAM Control with E2E ECC Decorated Access
Overlay RAM 16KB
Standby Supply (292BGA package only)
FLASH Controller Dual Ported Incl. Set-Associative Prefetch Buffers with E2E ECC
Overlay Backdoor for System RAM
Calibration Bus
256 Page Line 2 Stage Pipeline
FLASH 4MB
EEPROM 4x64k
NVM (Single Module)
Buddy Device Interface

# Panther Architecture Overview




Ref: MPC5744P reference manual rev 3

- **Nexus Class support –**
  - The IEEE-ISTO 5001 standard supports multiple "Classes" of support. The e200zx cores can be instantiated with different levels of support. An overview of the Nexus Class definitions can be found in AN4088 "MPC5500/MPC5600 Nexus Support Overview".

- **Nexus Timestamp Support**
  - The IEEE-ISTO 5001 standard allows Nexus messages to have timestamps appended to the message to better track the exact timing of the messages.
  - Timestamps are an optional feature. When implemented on the e200zx cores, the timestamps can either be implemented as buffered or non-buffered. In general, the unbuffered mode is implemented on devices with a Nexus Aurora Router (NAR) as the Nexus output controller. Devices that instantiate the Nexus Port Controller (NPC) use the buffered timestamps.
  - This difference is due to the minimal time that messages spend in the core Nexus buffers on NAR devices since the NAR implements dedicated input queues from each Nexus client on the device. However, the NPC only acts as a switch from the output of the individual Nexus clients directly to the Nexus output port. The NPC implements no buffers for holding messages. On NPC-based devices, each client implements the buffers that hold Nexus messages while other Nexus clients are transmitting their message out of the device.

# Nexus Class Support

# Nexus Classes of Features

| | | |
|---|---|---|
| Static Debug<br>r / w regs. & mem.<br>start/stop processor<br>hw / sw breakpoints | **Class 1 (JTAG only)** | – Enter a debug mode from reset or user code<br>– Read/ write user registers or memory in debug mode<br>– Single step instructions in user mode and re- enter debug mode<br>– Ability to set breakpoints or watchpoints<br>– Stop program execution on instruction/ data breakpoint and enter debug mode (minimum 2 breakpoints)<br>– Device identification |
| Watchpoint Msg<br>Ownership Trace Msg<br>Program Trace Msgs | **Class 2** | – All class 1 features<br>– Monitor process ownership while process runs in real- time (Ownership trace)<br>– Trace program flow in real time |
| Read / Write Access<br>Data Trace Msgs | **Class 3** | – All class 2 features<br>– Read/ write memory locations while processor runs in real-time<br>–Trace data reads & writes while processor runs in real- time |
| Memory Substitution<br>Port Replacement | **Class 4** | – All class 3 features<br>– Start data or program traces upon **watch-poin**t occurrence<br>– Ability to stall processor when trace buffers are full<br>– Program execution from Nexus port (not supported on MPC5500)<br>•Substitutes instructions and data in memory<br>– Port replacement |

- **JTAG Nexus State Machine Reset**
  - In some cases, the Nexus state machine in the e200z*x core does not get properly reset* when the JTAG TAP is changed to a different JTAG client (to a different core or other client). Previously, the Nexus JTAG state machine was reset only if the JTAG IR changed the CMD/DATA. It has been modified to reset the Nexus state machine anytime the JTAG state machine goes through the UPDATE_IR state.

- **Data Acquisition Message (DQM) Data Tag (DQTAG) packet type**
  - The Data Tag (DQTAG) packet of the Data Acquisition Message (DQM) is defined by the IEEE-ISTO 5001 standard as a Fixed length field. Some versions of the cores incorrectly implement the DQTAG as a Variable length packet instead of the Fixed length field.

# Debug Feature Options

| Device | Instantiation | Core Name | NEXUS class support | Time stamp | NEXUS port Width (MDO) | JTAG Nexus state Machine reset | DQM DQTAG Packet type |
|---|---|---|---|---|---|---|---|
| Calypso MPC5748/7/6G | Core2 | e200z210n3 | 3+ | Yes | 16/12 | Fixed | Fixed |
| | Core**0**/1 | e200z4204n3 | 3+ | Yes | 16/12 | Fixed | Fixed |
| Panther MPC5744P | Core**0** | e200z4251n3 | 3+ | No | 30 or 4 | Fixed | Fixed |
| Rainier MPC5746R | Core 0/**1** | e200z425n3 | 3+ | No | 30 or 4 | Fixed | Fixed |
| McKinley MPC5746M | Core 0/1 | e200z410n3 | 3+ | Yes | 30 | Fixed | Fixed |
| | Core 2 | e200z425Bn3 | 3+ | Yes | 30 | Fixed | Fixed |
| Matterhorn MPC57777M | Core 0/1 | e200z710n3 | 3+ | No | 30 | Fixed | Fixed |
| | Core 2 | e200z425Bn3 | 3+ | Yes | 30 | Fixed | Fixed |
| Cobra55 MPC5777C | Core 0/**1** | e200z759n3 | 3+ | No | 16 or 12 | Fixed | Fixed |
| Racerunner MPC5775K | Core0 | e200z4201n3 | 3+ | Yes | 30 or 16 | Fixed | Fixed |
| | Core1/2 | e200z7260n3 | 3+ | Yes | 30 or 16 | Fixed | Fixed |

N indicates the initial boot-up core
MDOs are package dependent

# References

- AN4802 Rev 0 http://cache.freescale.com/files/microcontrollers/doc/app_note/AN4802.pdf by Randy Dees
- http://www.freescale.com/files/32bit/doc/white_paper/E200CORELCNWP.pdf
- MPC5746M RM

# Thank You


www.Freescale.com


www.Freescale.com