Software Application development at device level

Module : Introduction to ARM Microcontrollers

Agenda

1. How to choose Microcontrollers for IoT devices
2. ARM family of Microcontrollers
3. Design of RISC Microcontrollers
4. ARM Bus Architecture
5. ARM Cortex M Processors
6. Embedded Software development process
7. ARM Cortex M3/M4 Architecture
8. CMSIS

Criteria for choosing Microcontroller for IoT device

- Power efficiency
- SoC kind of design for size consideration
- Better CPU performance
- Deterministic handling of events & exceptions
- Ease of use
- Availability of development ecosystem
- Cost

ARM family of processors

Processor Core	Architecture
ARM7TDMI family	v4T
ARM9TDMI family	v4T
ARM9E family	v5TE
ARM10E family	v5TE
ARM11 family	v6, v6T2
Cortex Family	V7
- ARM Cortex A	v7A, v8A
- ARM Cortex R	v7R, v8R
- ARM Cortex M	v7M, v8M

Ref : https://developer.arm.com/architectures/cpu-architecture

ARM Cortex™ family of processors

Processor Core	Intended Usage
- ARM Cortex A Series	Application processors used for computationally intensive application use cases like Smart phones, servers, automotive infotainment systems etc. – Support for OS
- ARM Cortex R	Real Time Processors used for building real time systems. Typical use cases are safety critical systems used in automobiles, medical equipments etc.
- ARM Cortex M	Microcontroller based processors used for building deeply embedded systems where power consumption and size matters. Typical use cases are IoT end points, embedded devices etc.

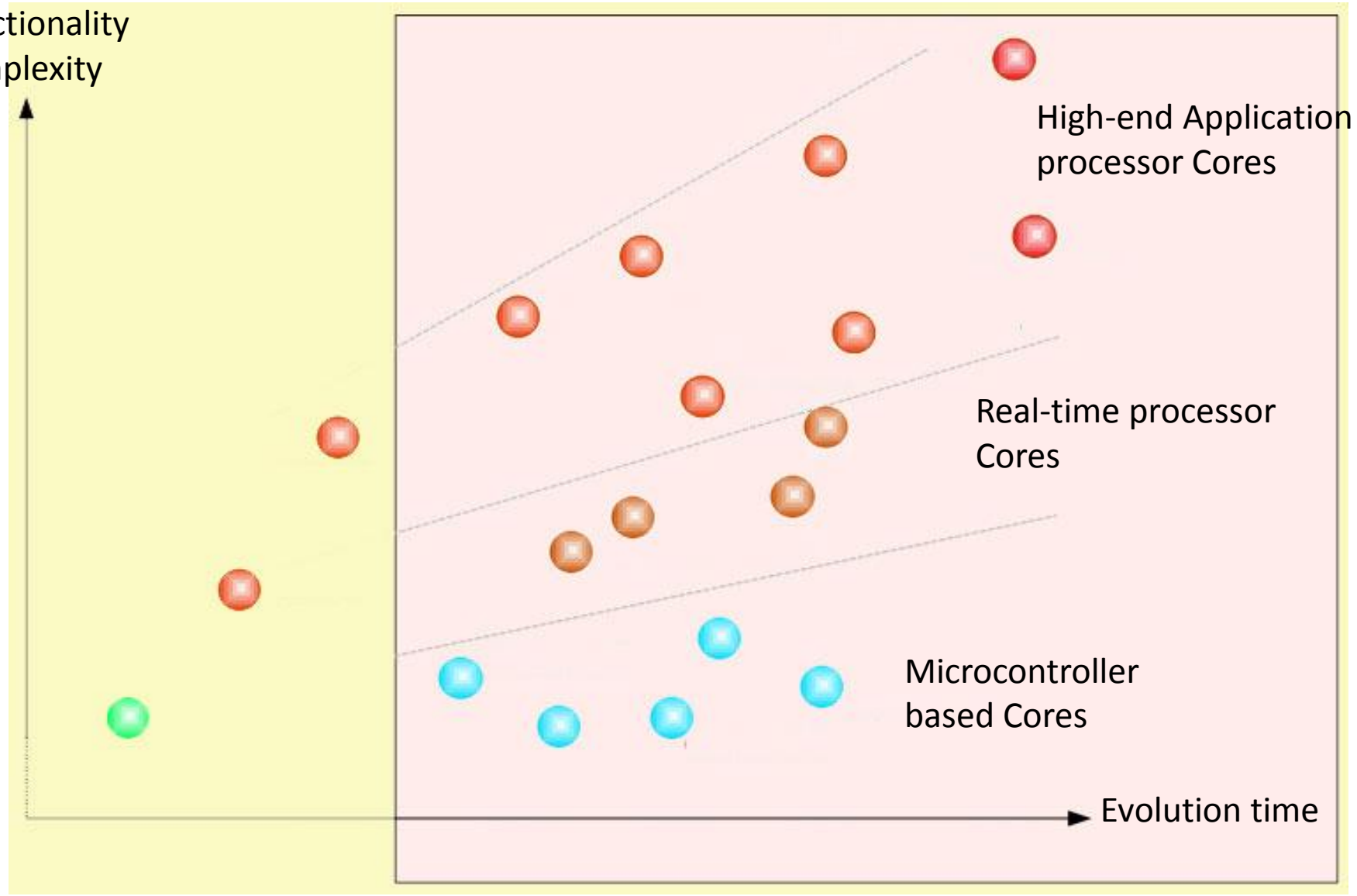
Ref : https://developer.arm.com/architectures/cpu-architecture

Evolution of ARM processor cores

Processor Core Examples	ARM Architecture
ARM7TDMI, ARM920T, Intel StrongARM	V4/v4T
ARM926EJ, ARM966, Intel xscale	V5/v5E
ARM1136, ARM1156T-2	V6
Cortex-M0+, Cortex-M0, Cortex-M1	V6-M
Cortex-A8, Cortex-A9, Cortex-A15	v7A
Cortex-R4, Cortex-R5, Cortex-R7	V7R
Cortex-M3, Cortex-M4, Cortex-M7	v7M

Ref : https://developer.arm.com/architectures/cpu-architecture

Diversity of ARM Processor Cores



Design rules for RISC Microcontrollers

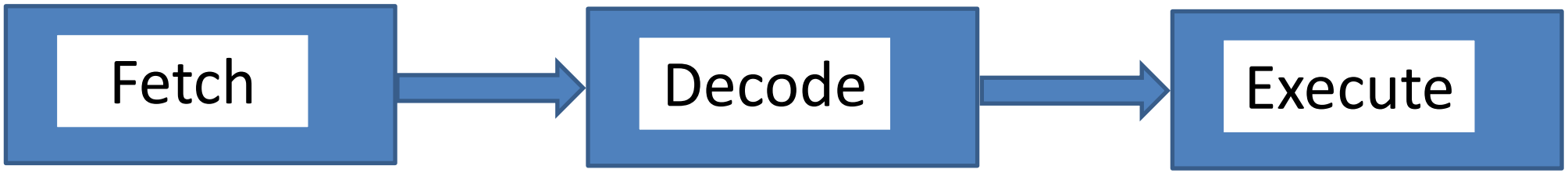
- Instructions
  - reduced number of instructions and classes
  - single cycle execution
  - complex compiler
  - fixed length instruction for pipeline efficiency
- Instruction level Pipeline
  - instructions are broken into smaller parts called stages
  - parallel execution of stages
  - maximize throughput
- Registers
  - large number of general purpose registers
- Load Store architecture

Load Store Architecture

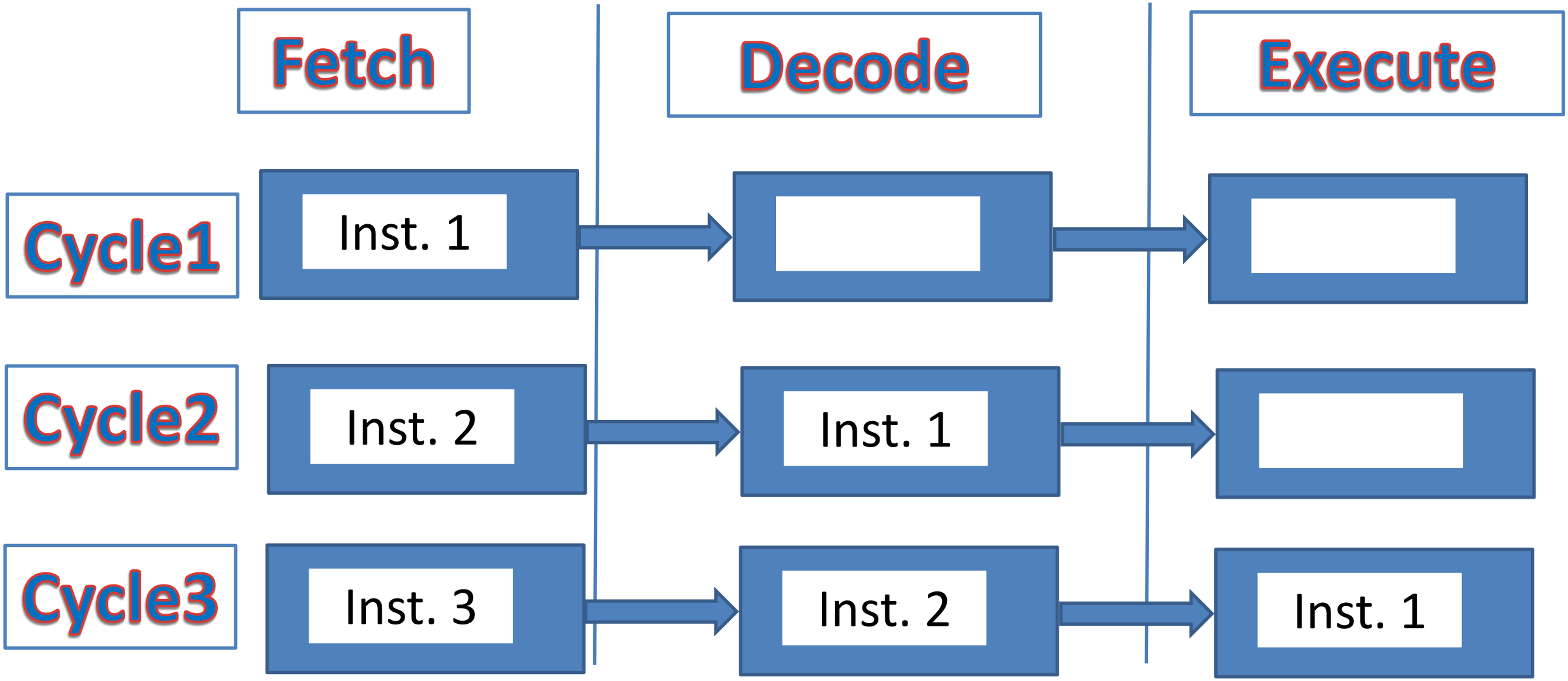
- Memory access is done only through Load Store instructions
- Processor can process data available only in CPU registers
- Data is taken from memory to register using LDR instruction
- After processing the data in registers, results are stored back to memory using STR instruction
- Read-Process-Write
- ARM
- Better performance by improving pipeline efficiency

Concept of Pipelining

- Mechanism adopted by RISC processors to execute instructions
- Speeds up
- Increase throughput
- but how ?



Instruction Level Pipelining (ILP)



ARM microcontroller’s internal BUS Architecture

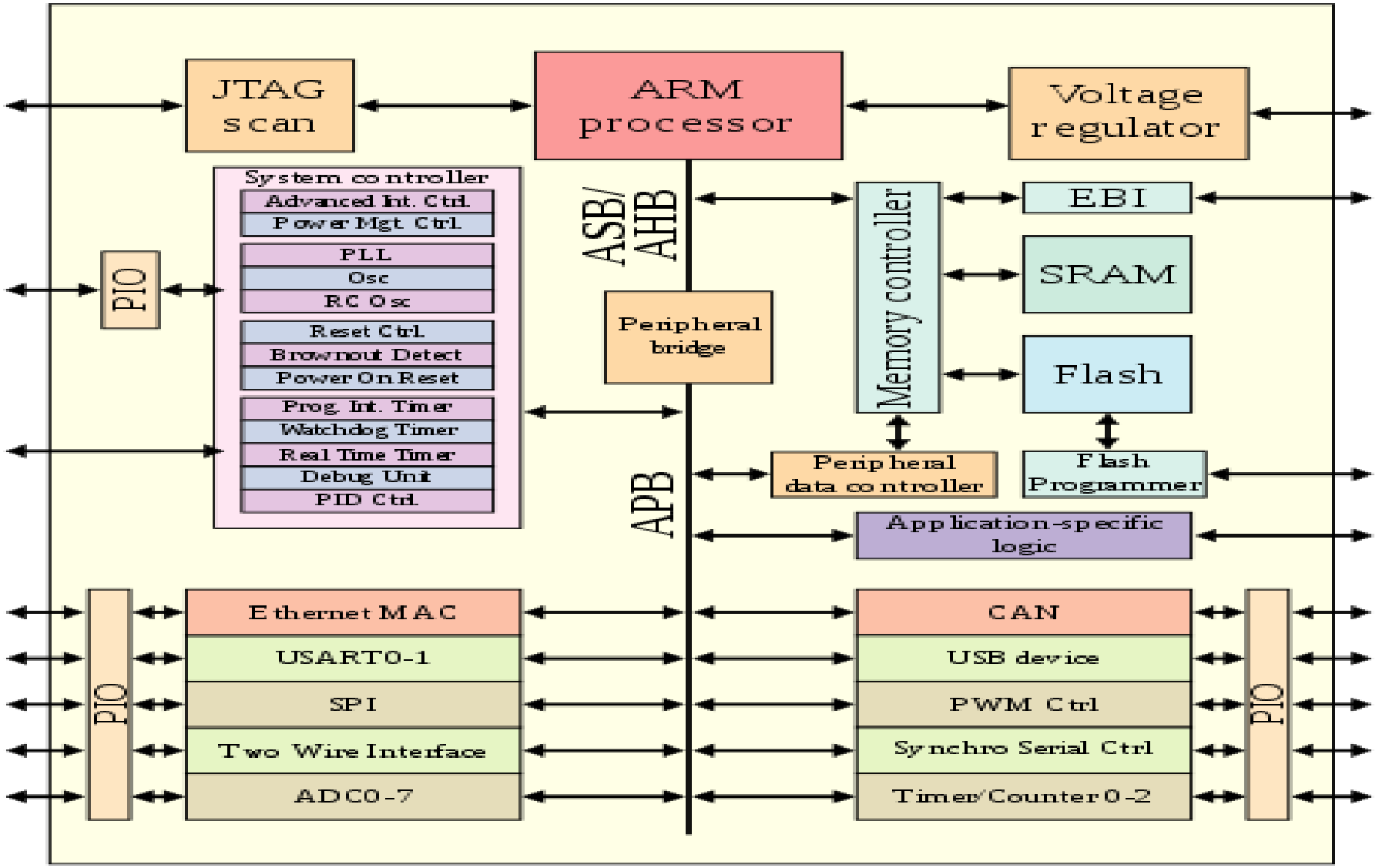
ARM Bus Types and Standards

- ARM microcontrollers use AMBA bus standard
- The Advanced Microcontroller Bus Architecture (AMBA) specification defines an on chip communications standard for designing high-performance embedded microcontrollers
- 3 distinct buses are defined within the AMBA specification:
  1. Advanced High-performance Bus (AHB)
  2. Advanced System Bus (ASB)
  3. Advanced Peripheral Bus (APB)

ARM Bus Types and Standards

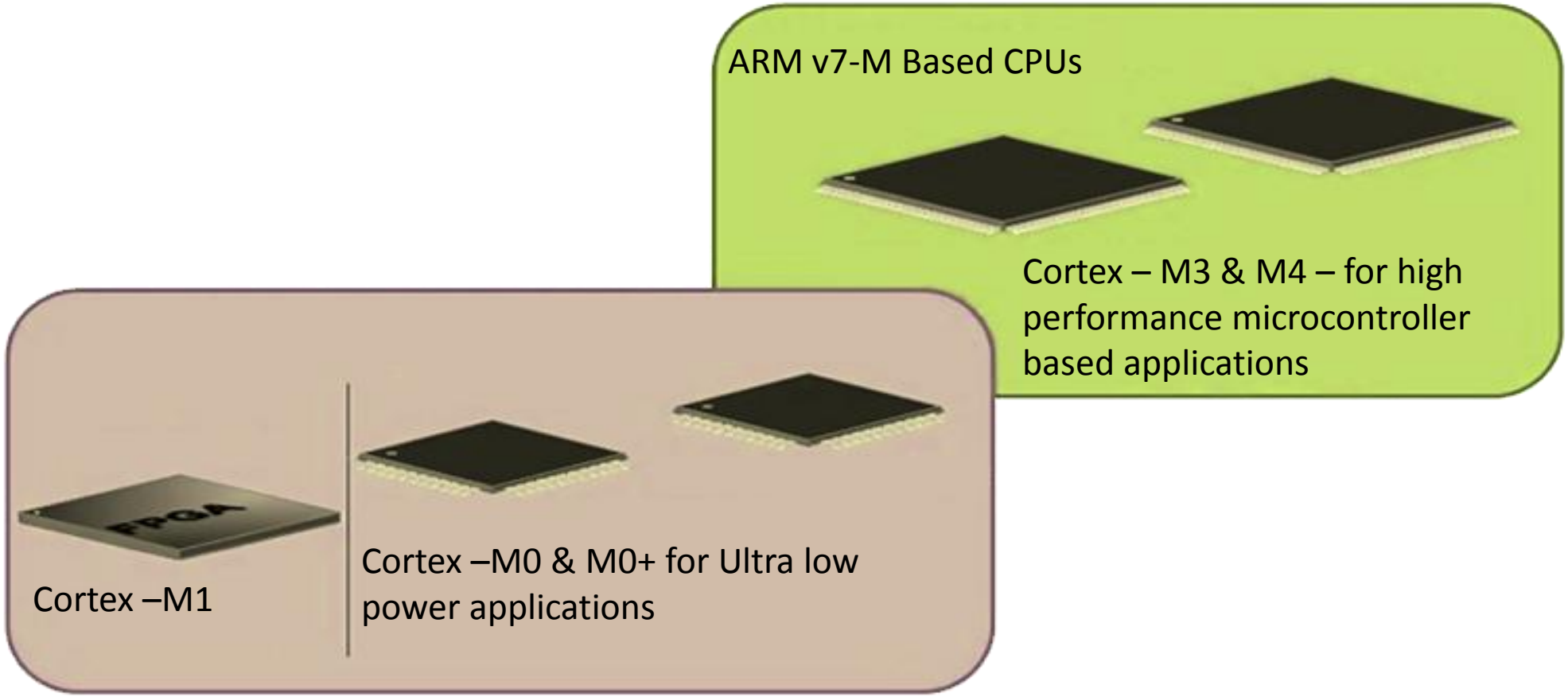
- AHB : The AMBA AHB is used mainly for interconnecting processor with on-chip/ off-chip memories for high performance & high speed connectivity.
- ASB : The AMBA ASB is the previous version of AHB and can be used for interconnecting processor with on-chip/off-chip memories and system components.
- APB : The AMBA APB is used for connecting with low bandwidth, low complexity , low power system peripherals. AMBA APB is optimized for minimal power consumption and reduced interface complexity to support peripheral functions.

ARM Microcontroller based SoC



Ref : https://en.wikipedia.org/wiki/ARM_architecture

Cortex M processor family



ARM Cortex M series features

- 3 stage pipeline design
- Harvard bus architecture with unified single memory space : instructions and data use the same address space
- 32 – bit addressing , supporting 4GB of memory space
- On-chip bus interfaces based on ARM AMBA
- Interrupt controller called NVIC, 240 interrupts , 256 priorities
- Support for OS related features
- Sleep modes for low power operations
- Supports optionally MPU
- bit banding

Advantages of Cortex M processor

- low power
  - 200 mA/MHz run mode
  - 100 mA/MHz in sleep mode
- performance
  - 3 CoreMarK/Mhz and 1.25 DMIPS/MHz
- energy efficiency
- code density
- interrupts
  - interrupt latency is only 12 clock cycles Maximum
- ease of use
- scalability
- debug features
- os support
- software portability and reusability

Instruction Set Architecture (ISA)

- Cortex M microcontroller belonging to v7-M architecture have support for Thumb v2 called as Thumb2 ISA
- Thumb2 ISA supports both 16 bit & 32 bit instructions
- The earlier version Thumb supported only 16 bit instructions
- Thumb v2 based CPUs have only one state called thumb state
- Previous Thumb architecture based CPUs had thumb state & ARM state

Embedded Software Development Process

1. Create a Project using any of the ARM microcontroller development IDEs such as Keil or IAR etc.
2. Add the relevant and required driver library files, middleware if any and compile your application code
3. After compiling and building the project for the targeted hardware, flash the binary image using a debugger
4. Using on/off board debugger, debug the code. Debuggers could be based on either JTAG or SWD standard.

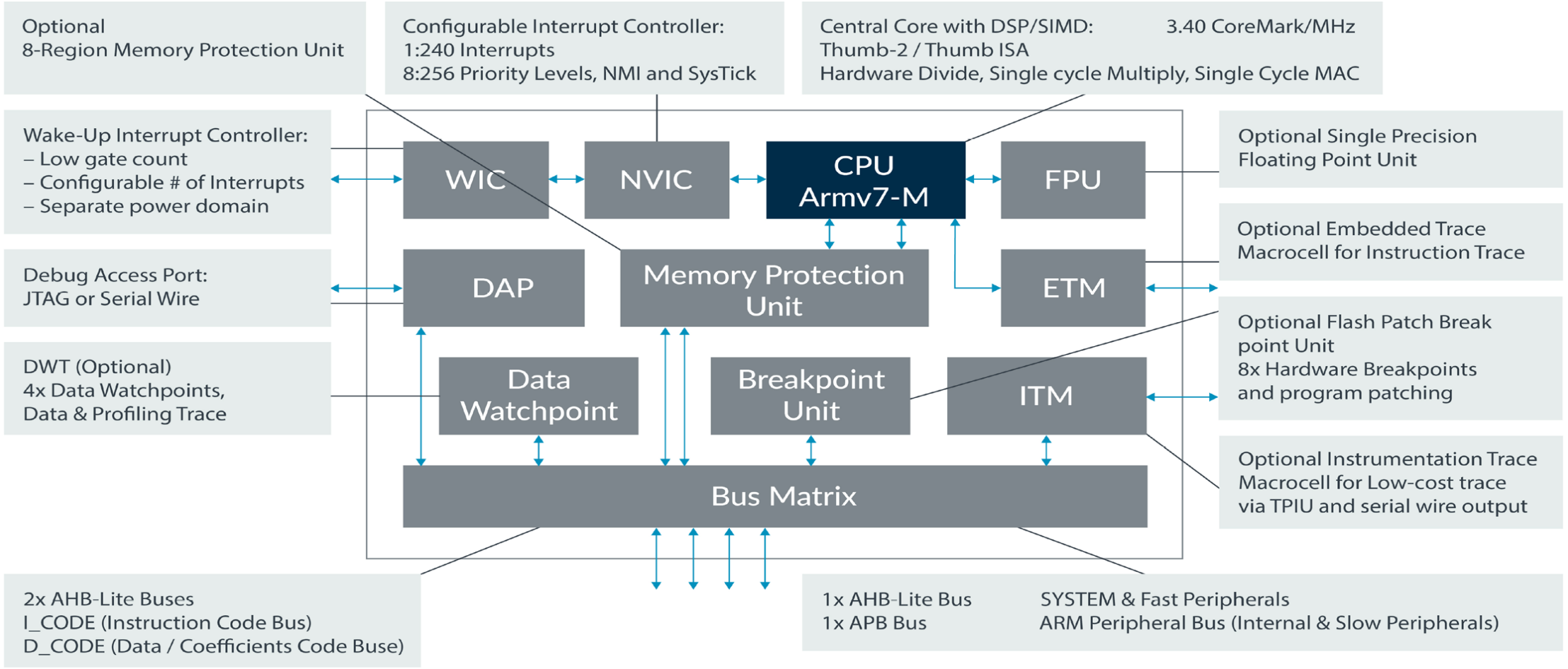
Tools in development suite

Tool	Description
C Cross Compiler	To compile C code into object file of target architecture
Assembler	To assemble assembly code into object file of target architecture
Linker & Build	Tool used for combining multiple object files into executable of target architecture defined with memory configuration
Flasher	Tool used to download executable binary image generated into FLASH memory of target board
Debugger	A tool to control operation of the microcontroller for doing debug operations
Simulator	A tool used for debugging microcontroller without the actual hardware

Cortex M4

- A low-power processor that features low gate count, low interrupt latency, and low-cost debug
- Cortex-M4F includes floating point arithmetic functionality
- Supports DSP
- Implements the ARMv7-M architecture profile
- 32 bit RISC
- Internal registers in the register bank, data path and bus interfaces all 32bits
- Thumb - 2 -> ISA
- 3 stage pipeline design
- Harvard bus architecture with separate memory address buses for instructions and data

Cortex M4 Block Diagram



Ref : ARM Cortex M4 Data Sheet
https://developer.arm.com/ip-products/processors/cortex-m/cortex-m4

ARM Cortex M4 Bus interfaces

Bus	Description
I_CODE	Primarily for instruction and exception vector fetches from program memory – FLASH (0x0 to 0x1FFFFFFF) – AHB_Lite
D_CODE	Used for fetching data from program memory – FLASH (0x0 to 0x1FFFFFFF) – AHB_Lite
SYSTEM	Used for accessing RAM and some of the on-chip system peripherals (0x20000000 to 0xFFFFFFFF) except for PPB – AHB_Lite
PPB	Private peripheral bus includes – APB + access to SCS and NVIC (0xE0000000 to 0xFFFFFFFF)
DAP	Debug access port – JTAG/SWD

ARM Cortex M3/M4 Features

- NVIC with upto 240 IRQ# and 8 to 256 levels of priority
- support for various features for OS implementation
- sleep mode and various low power features
- optional MPU and FPU
- support for bit-data accesses (bit band)
- variable length of instruction with rich ISA
- DSP – SIMD and fast MAC
- 32-bit addressing, supporting 4GB of memory space
- on-chip bus interfaces based on ARM AMBA bus standard

ARM Cortex M3/M4 Performance

- Better Clock Per Instruction (CPI) ratio/DMIPS
- Multiple bus interfaces allow parallel access to instruction & data memories
- Higher clock frequencies due to pipelined bus interface in memory system
- Thumb -2 ISA provides a highly efficient instruction set allowing complex operations to be performed using less number of instruction.
- I_CODE bus being 32 bit or higher; and most of instructions being 16 bit allows two or more instructions to be fetched at a time. Enabling better performance & energy efficiency.

ARM Cortex M3/M4 Code Density

- Good code density as compared to other 32 bit processor
- Thumb-2 technology allows 16-bit and 32-bit instructions to work together without switching the states
- Efficient data access due to multiple addressing modes related with memory
- Single instruction can access multiple memory locations
- Hardware division instruction supported along with multiply and accumulate (MAC) kind of instructions.
- Instructions for bit field processing
- SIMD instruction support - DSP
- Optional support for single precision floating point instructions

ARM Cortex M3/M4 Low Power

- Current consumption under 200 mA/MHz (approximately 0.36 mW/MHz for a supply voltage of 1.8 volt) and some of them can even run at under 100 mA/MHz
- Less power utilization due to higher code density
- Supports multiple low power modes of operation

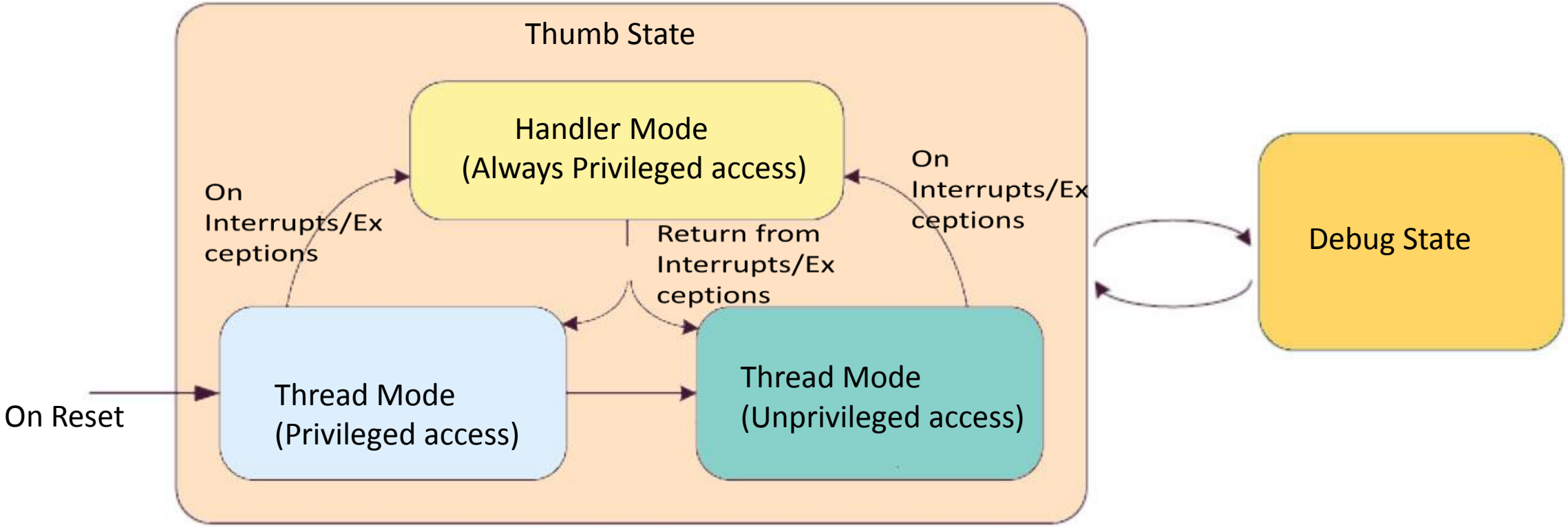
ARM Cortex M3/M4 Memory System

- Total of 4GB of linear memory map
- Harvard bus architecture
- High speed & low latency bus transfer interface implemented with pipelined AHB Lite bus interface architecture
- Bit band feature for RAM & peripheral regions
- Support for both little endian or big endian memory systems
- Optional MPU support

ARM Cortex M3/M4 - Interrupt & Exception Handling

- Interrupts and Exceptions managed by NVIC
- Supports up to 240 interrupts & 15 System Exceptions
- Programmable priority levels for interrupts & Exceptions
- NVIC automatically handles prioritization of interrupt/exception execution
- **Relocatable** Exception vector table
- Highly deterministic interrupt latency – Max. of 12 Clock cycles under all circumstances

Operating Modes & States in Cortex M3/M4



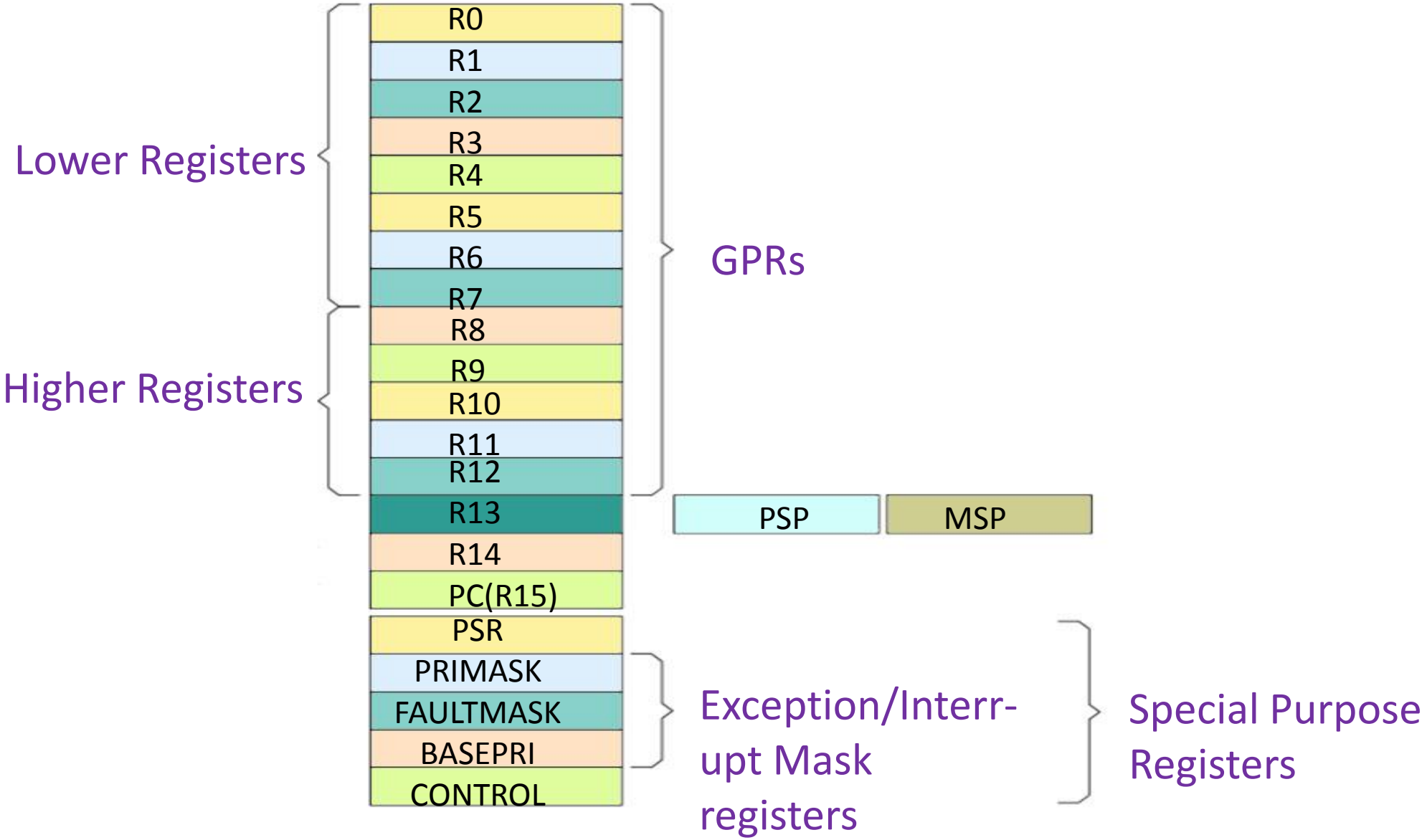
Operating Modes & States Contd.

- Handler Mode : Processor enters handler mode immediately after encountering interrupt/exception. As handler mode is always with privileged access used for critical handling of events, ISRs and OS routines.
- Thread Mode : Entered on reset, initially with privileged access, can be used for basic start-up operations and then optionally can be driven into unprivileged access level for executing user programs.
- Thumb State : Operating state of the processor for running instructions

Operating Modes & States Contd.

- Debug state : Processor enters debug state while undergoing debug operations by the debugger like when the debugger wants to do step-by-step execution, after hitting breakpoints etc.
- There are only two states – Debug state & Thumb state
- There are two modes – Handler mode & Thread mode
- For designing a secured & robust embedded software usually an OS – EOS/RTOS makes uses of these modes of operation provided by the CPU

ARM Cortex M – Processor Registers



Registers in ARM Cortex M3/M4 Processors

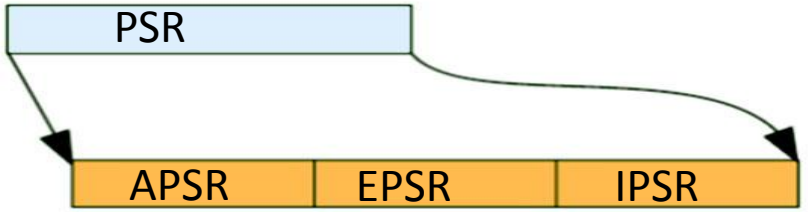
- R0 – R12 are general purpose registers
- R13 – stack pointer register ; it can either MSP or PSP
- On Reset MSP is selected as SP unless changed through CONTROL register
- Handler mode doesn’t have the option to change SP register; always uses MSP
- Thread mode has an option to switch SP register to either MSP or PSP
- Usage of PSP is to separate stack regions for user applications and system/interrupt/OS executions

Special registers in Cortex M3/M4

PSR – Program Status Register

APSR – Application Program Status Register

IPSR – Interrupt Program Status Register



PRIMASK
FAULTMASK
BASEPRI

Interrupt/Exception related registers

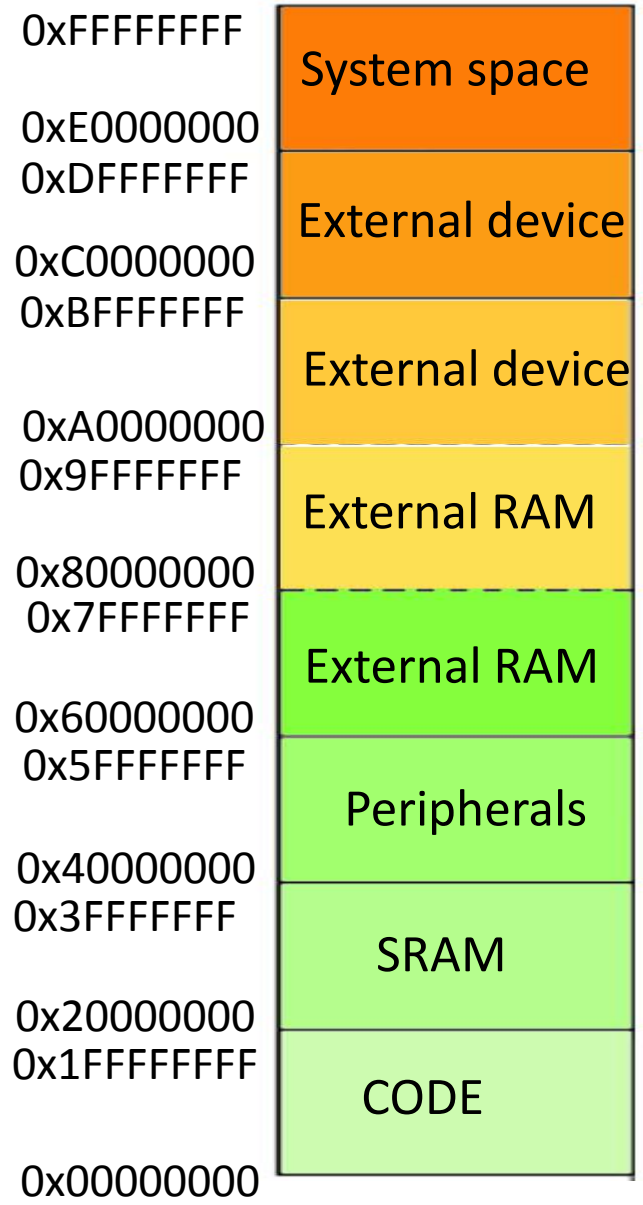
CONTROL

CONTROL register applicable only in thread mode for SP selection and Privilege selection

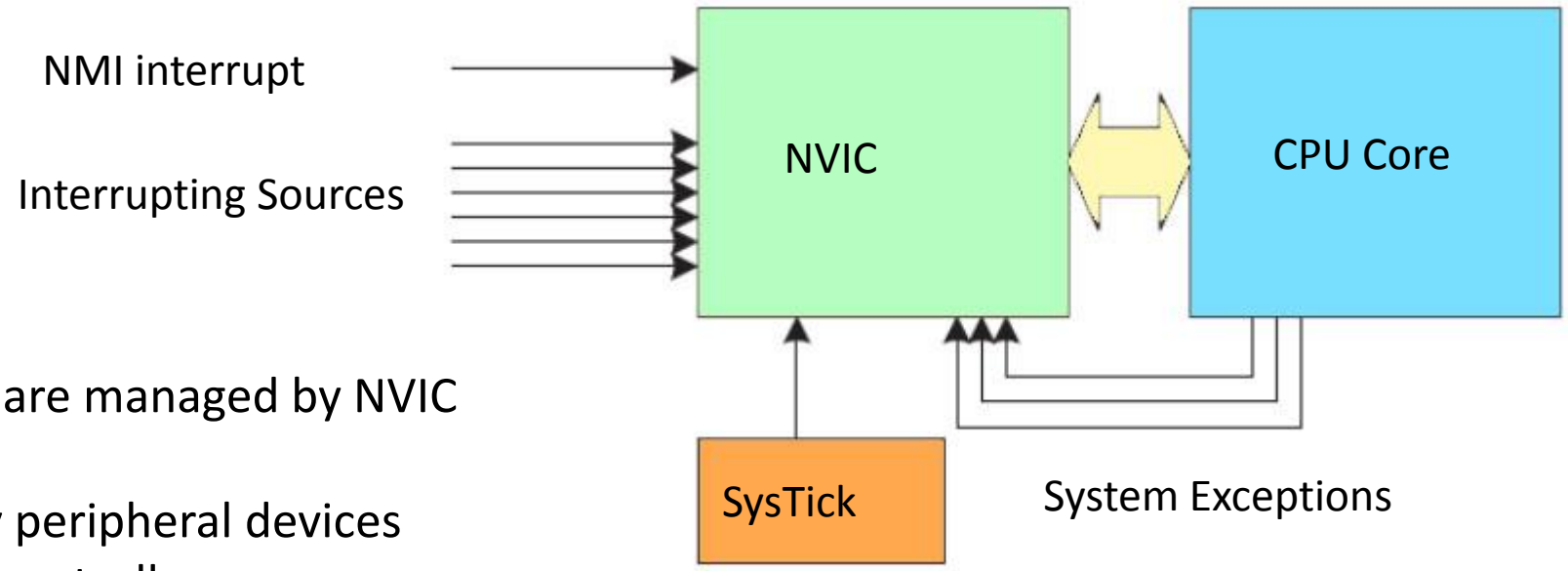
Memory Map in ARM Cortex M3/M4 Processors

System Space Region – Break-up

Private Peripheral Region	0xE0000000 – 0xE00FFFFF - System control space and debug registers
Vendor Specific Peripheral Region	0xE0100000- 0xFFFFFFFF



Interrupts & Exceptions ARM Cortex M3/M4

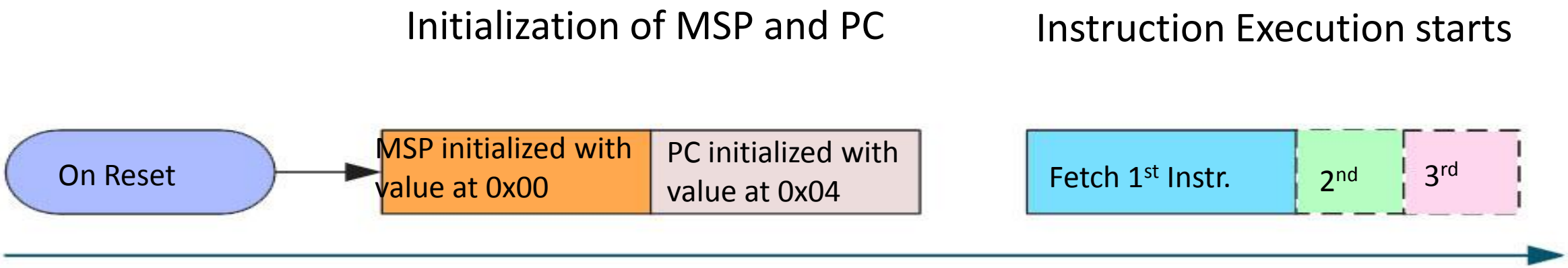


- All interrupts & Exceptions are managed by NVIC
- Interrupts are triggered by peripheral devices internal or external to microcontroller
- Exceptions are generated by CPU core itself whenever it encounters any run time errors

NVIC

- NVIC registers are located in system control space of the memory map
- NVIC takes care of configuration of interrupts & exceptions, their setting of priorities & masking capabilities
- features:
  - Ensures Max. interrupt latency of 12 Clock Cycles
  - Support for nesting of interrupt handlers
  - Support for vectored interrupts & Exceptions
  - Support for pending of interrupts

Sequence of operations after Reset



- Reset everything – Power on Reset
- Reset CPU and Peripherals – System Reset
- Reset CPU – Processor Reset

CMSIS

- Cortex Microcontroller Software Interface Standard – Software development standard defined by ARM to provide a standard software development infrastructure.
- CMSIS helps in maintaining compatibility across tools of different vendors and also between applications developed with this standard.
- CMSIS project includes :
  - CMSIS core, CMSIS RTOS, CMSIS DSP, CMSIS DAP

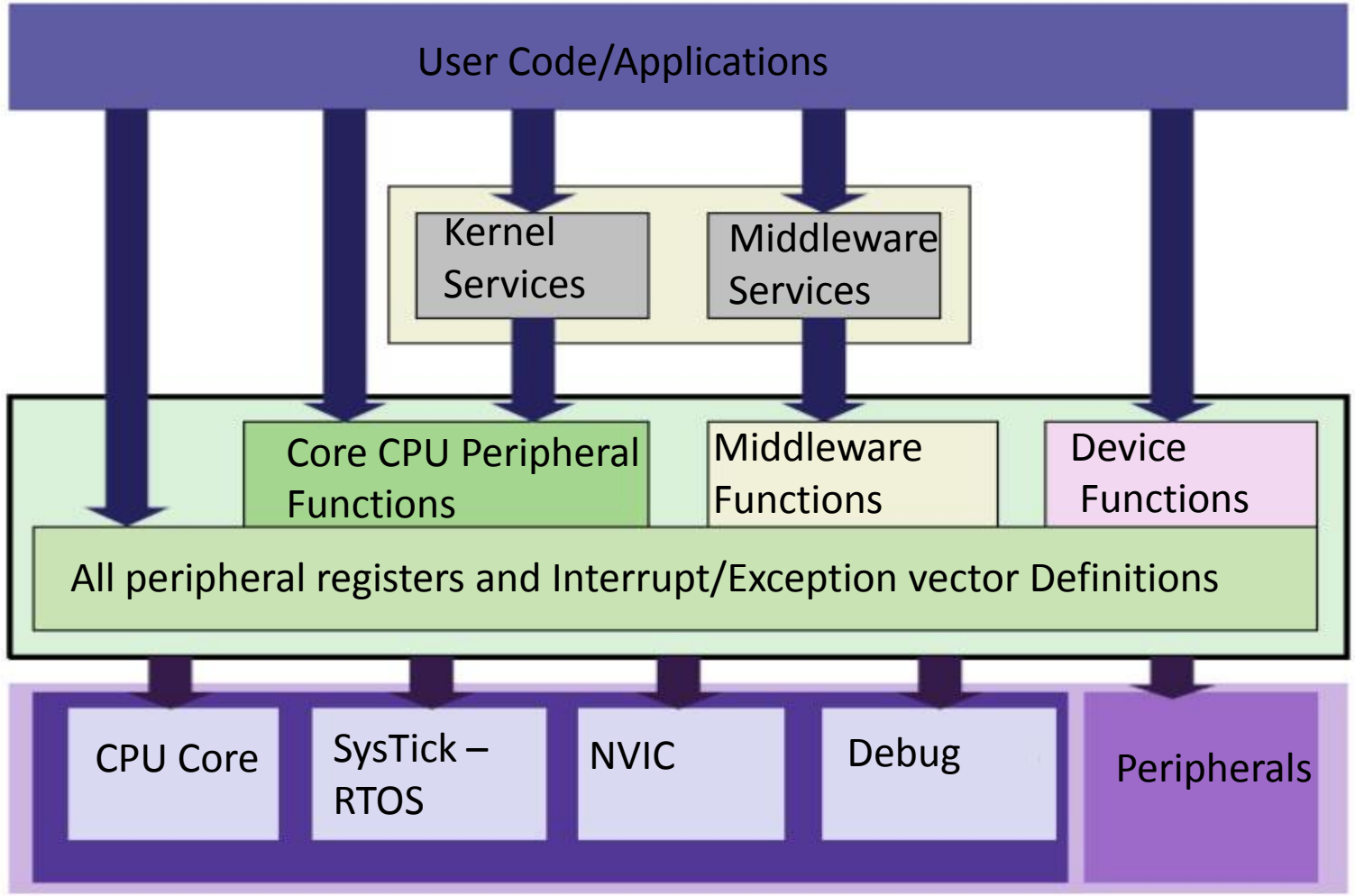
CMSIS – Blocks and Structures

Applications

RTOS Layer

CMSIS Layer

Hardware Layer
Cortex M -
Microcontrollers



Use of CMSIS

- Code reusability
- Code compatibility
- Ease of development
- Vendor and toolchain independent
- Open standard

Advantages of CMSIS

- Helps in software portability across Cortex M Microcontroller family of processors
- Faster time to market
- Consistent standard for vendors, library/middleware developers and application developers
- Adopted by all vendors
- CMSIS results in better code density/code footprint
- As serves as a base standard for RTOS developers

references

- http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.ddi0403e.d/index.html
- https://www.st.com/en/microcontrollers-icroprocessors/stm32f429-439.html?querycriteria=productId=LN1806
- The Definitive Guide to ARM Cortex-M3 and Cortex-M4 Processors Third Edition
- http://soc.eecs.yuntech.edu.tw/Course/SoC/doc/amba.pdf
- http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.dai0321a/BIHGJICF.html
- https://www.anandtech.com/show/8400/arms-cortex-m-even-smaller-and-lower-power-cpu-cores
- ARM System Developers Guide Designing and Optimizing System Software by Andrew N Sloss