ĐẠI HỌC QUỐC GIA TP.HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA ĐIỆN-ĐIỆN TỬ
BỘ MÔN KỸ THUẬT ĐIỆN TỬ

Embedded System Design

Chapter 2: Microcontroller Series (Part 1)

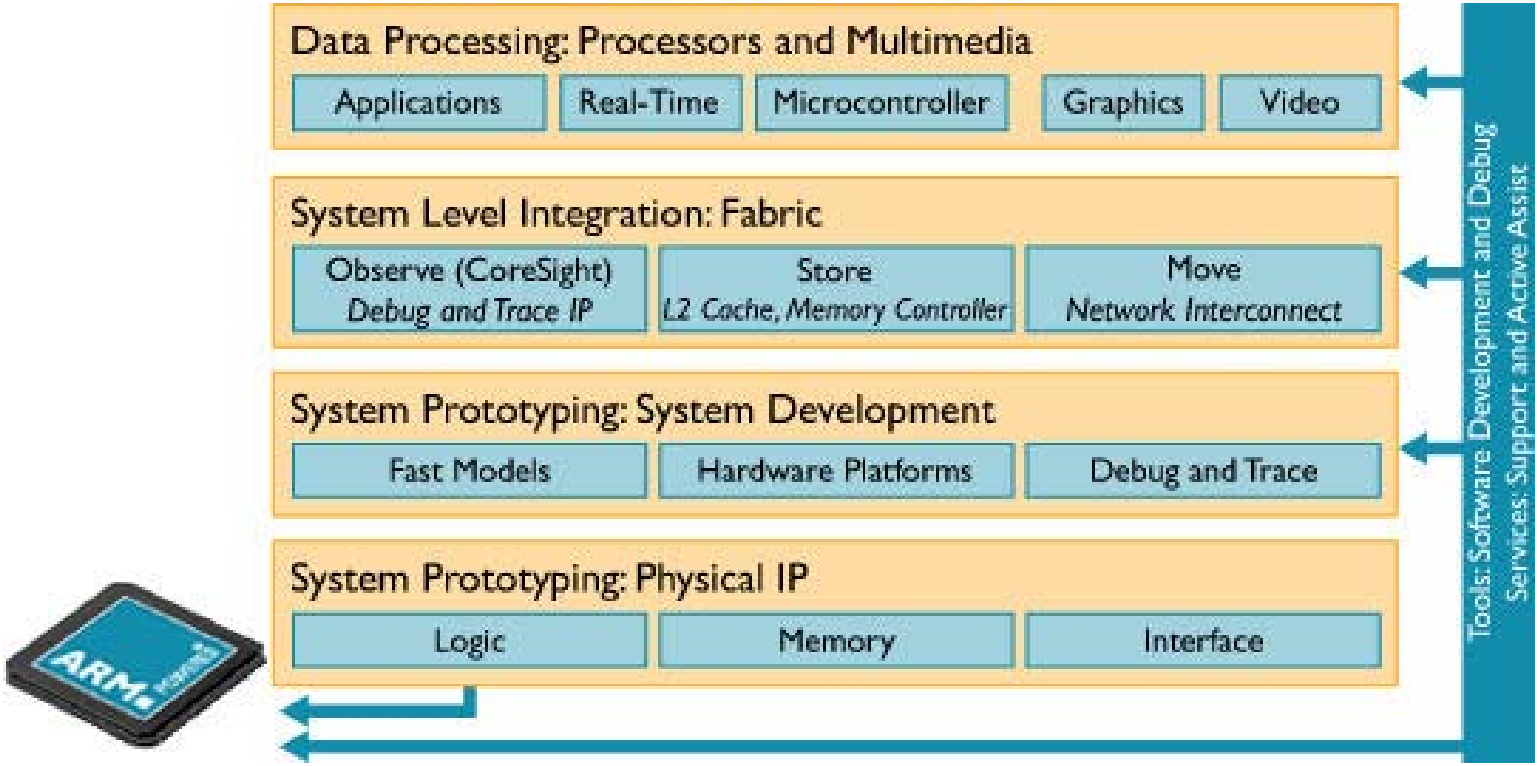
1. Introduction to ARM processors
2. ARM Cortex-M3

References

- Textbook
  - Joseph Yiu, “The Definitive Guide to the ARM Cortex-M3”, Elsevier Newnes, 2007
- Websites
  - www.arm.com
  - www.st.com
  - www.ti.com
  - www.nxp.com
  - www.thegioiic.com
  - www.arm.vn
  - www.tme.vn
  - www.proe.vn

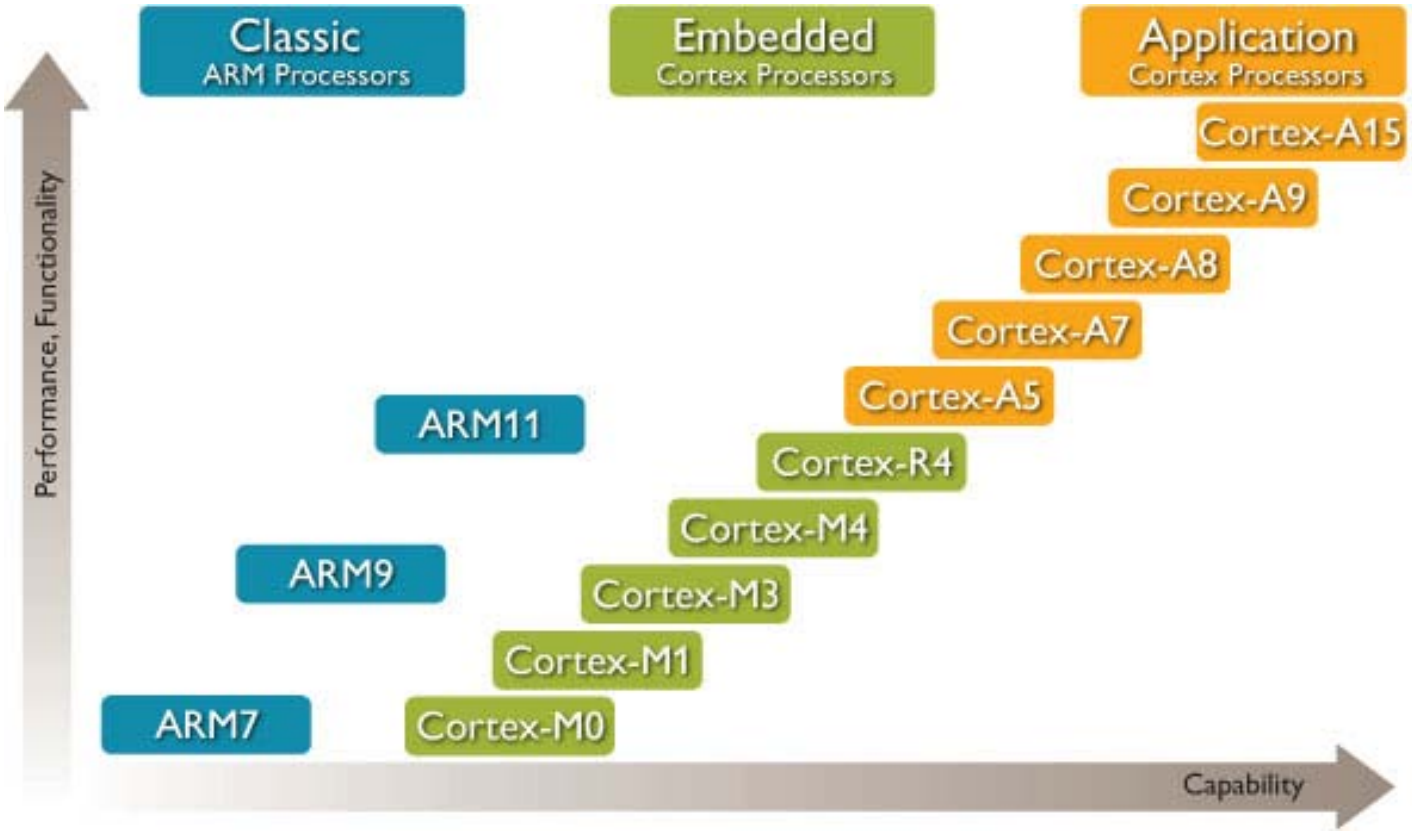
1. Introduction to ARM processors

- ARM (Advanced RISC Machine)
  - is the industry's leading provider of 32-bit embedded microprocessors
  - offering a wide range of processors that deliver high performance, industry leading power efficiency and reduced system cost



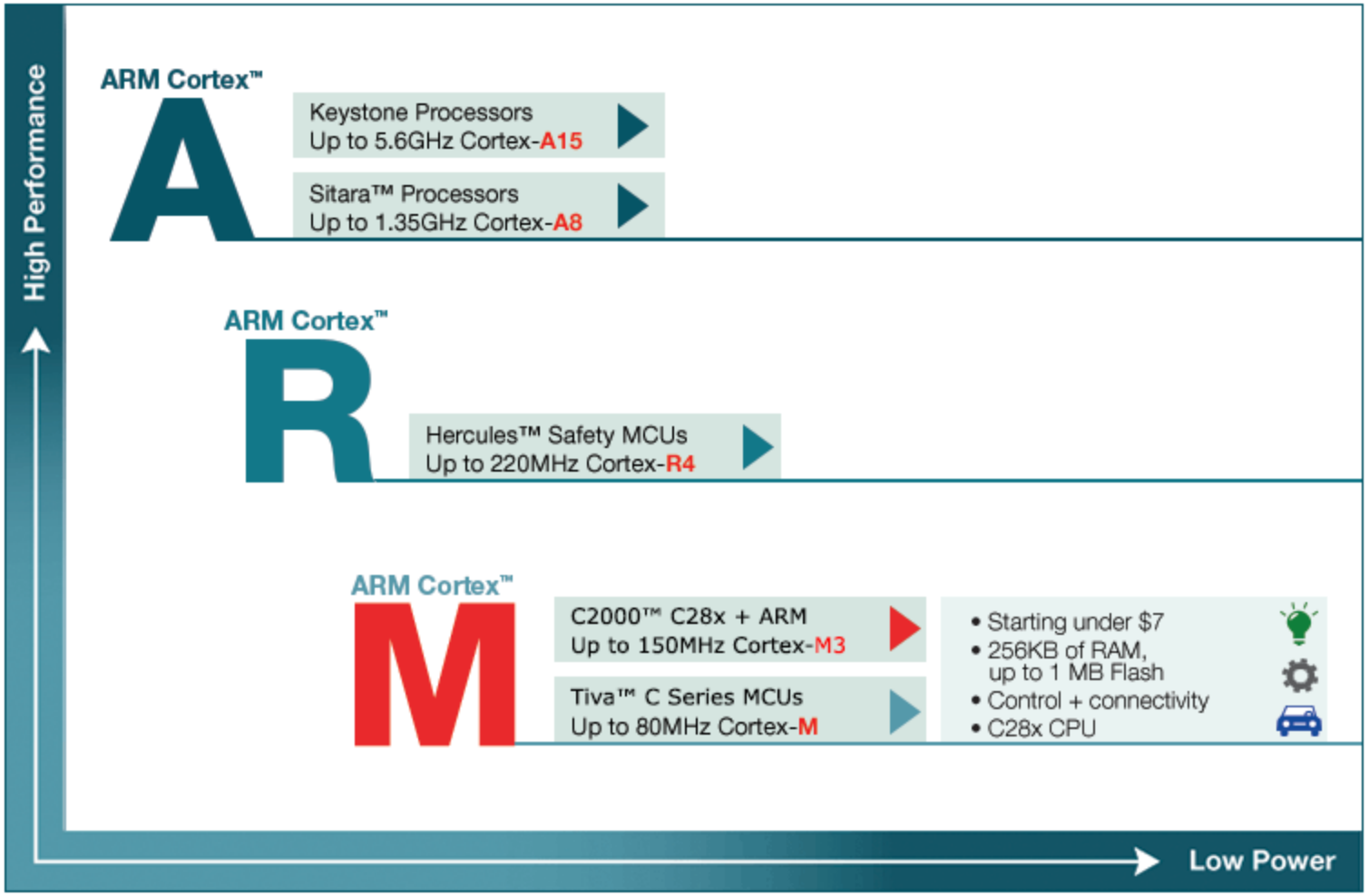
1. Introduction to ARM processors

- **Cortex™-A Series -** High performance processors for open Operating Systems
- **Cortex-R Series** - Exceptional performance for real-time applications
- **Cortex-M Series** - Cost-sensitive solutions for deterministic microcontroller applications

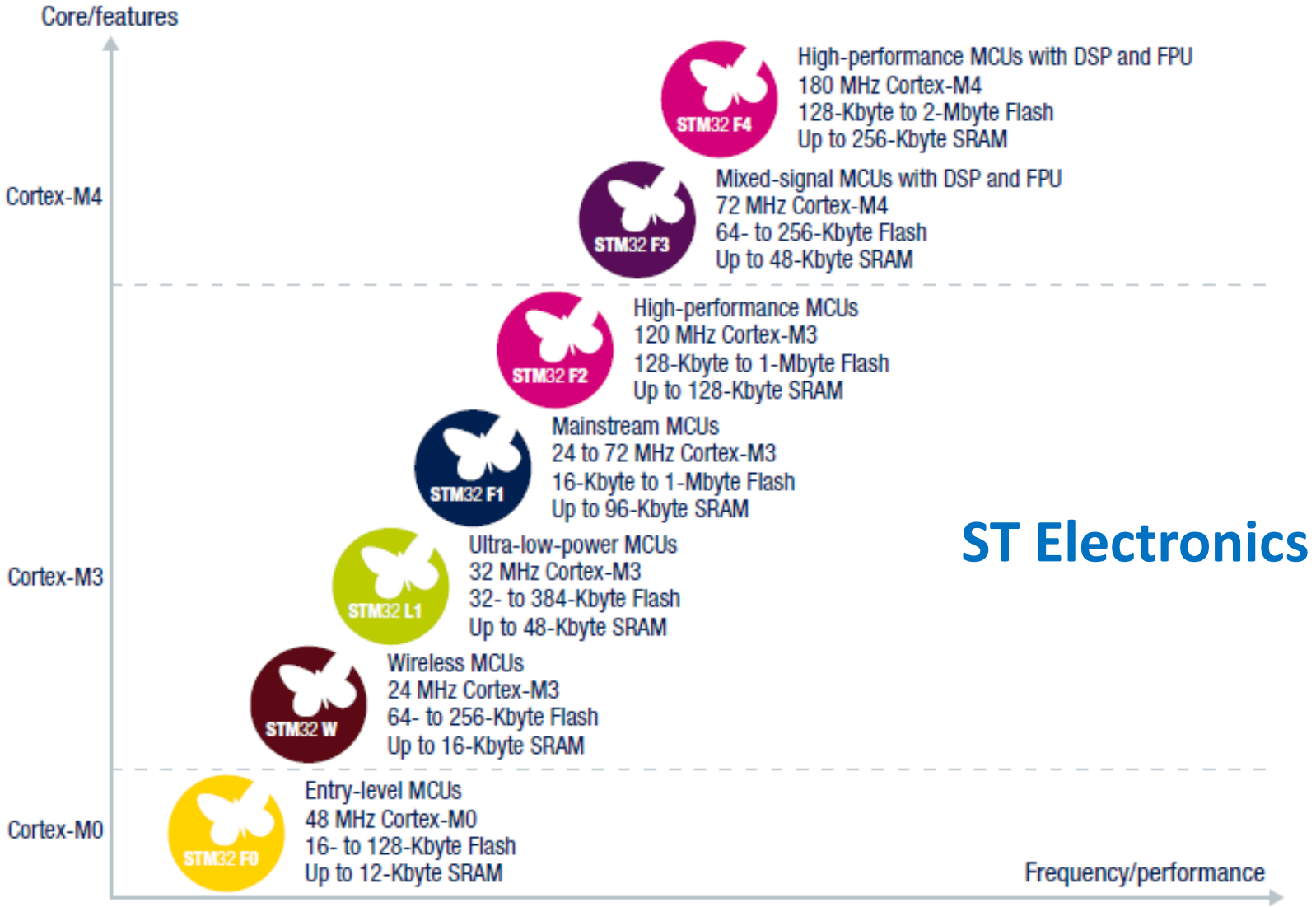


1. Introduction to ARM processors

- TI's ARM microcontroller overview

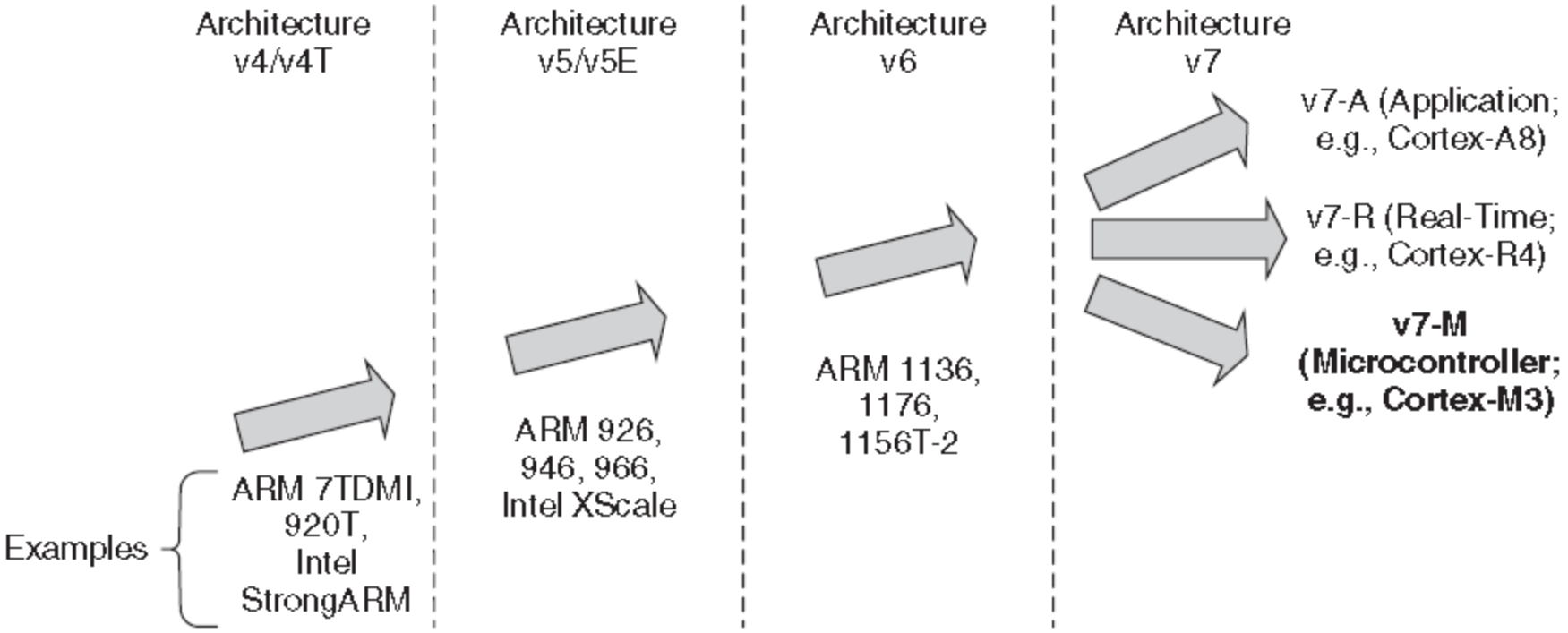


1. Introduction to ARM processors



1. Introduction to ARM processors

- The Cortex processor families are the first products developed on architecture v7
- Cortex-M3 processor is based on one profile of the v7 architecture



1. Introduction to ARM processors

- Cortex-M Series:
  - Easy to use: Global standard across multiple vendors, code compatibility, unified tools and OS support
  - Low cost: smaller code, high density instruction set, small memory requirement
  - High performance: deliver more performance per MHz, enable richer features at lower power
  - Energy efficiency: run at lower MHz or with shorter activity periods
- Cortex-M Series applications
  - Microcontrollers
  - Mixed signal devices
  - Smart sensors
  - Automotive body electronics and airbags

2. ARM Cortex-M3

- 32-bit embedded processor
- Improved code density
- Enhanced determinism, quick interrupts
- Low power consumption
- Lower-cost solutions (less than US$1)
- Wide choice of development tools

LPC1754FBD80

AT91SAM7S64-AU

STM32F103RCT6

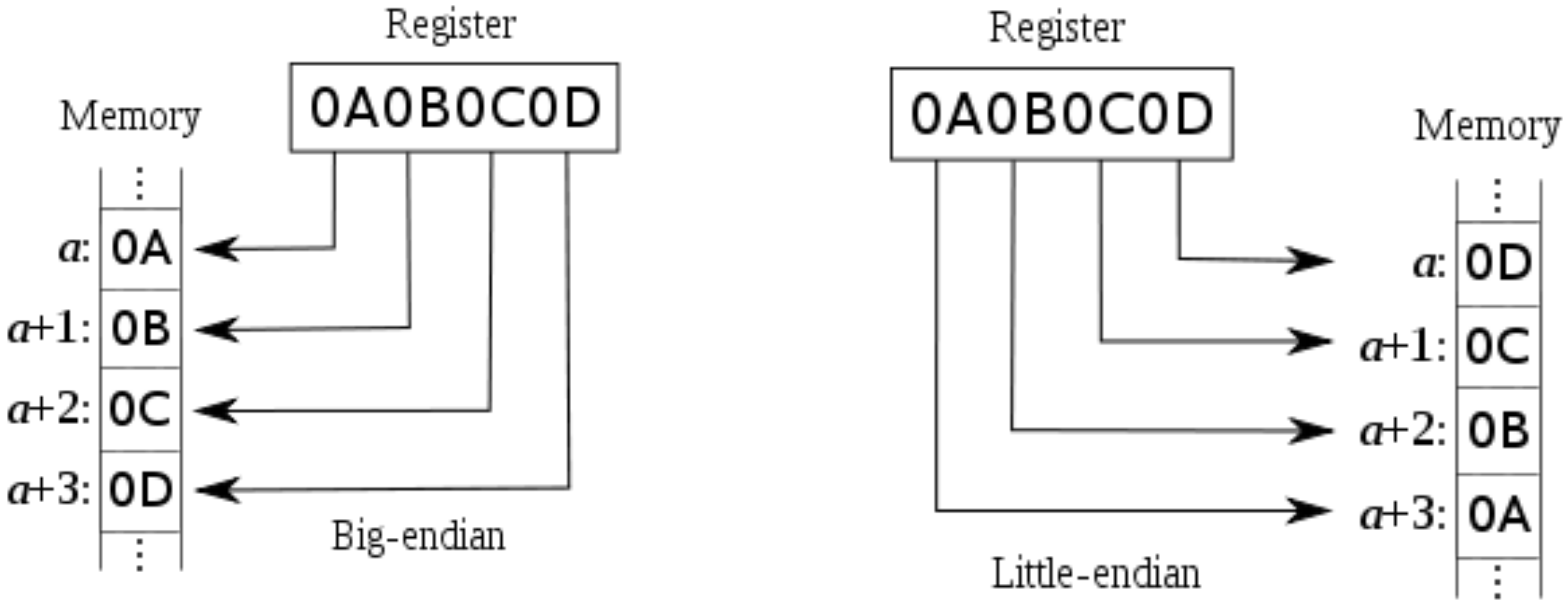
LM3S3749

2.1 ARM Cortex-M3 - Architecture

- 32-bit microprocessor
  - 32-bit data path
  - 32-bit register bank
  - 32-bit memory interface
- Harvard architecture
  - 3-stage pipeline
  - separate instruction bus and data bus
  - share the same memory space, difference length of code and data
- Interrupts
  - 1 to 240 physical interrupts, plus NMI
  - 12 cycle interrupt latency
- Instruction Set
  - Thumb (entire)
  - Thumb-2 (entire)

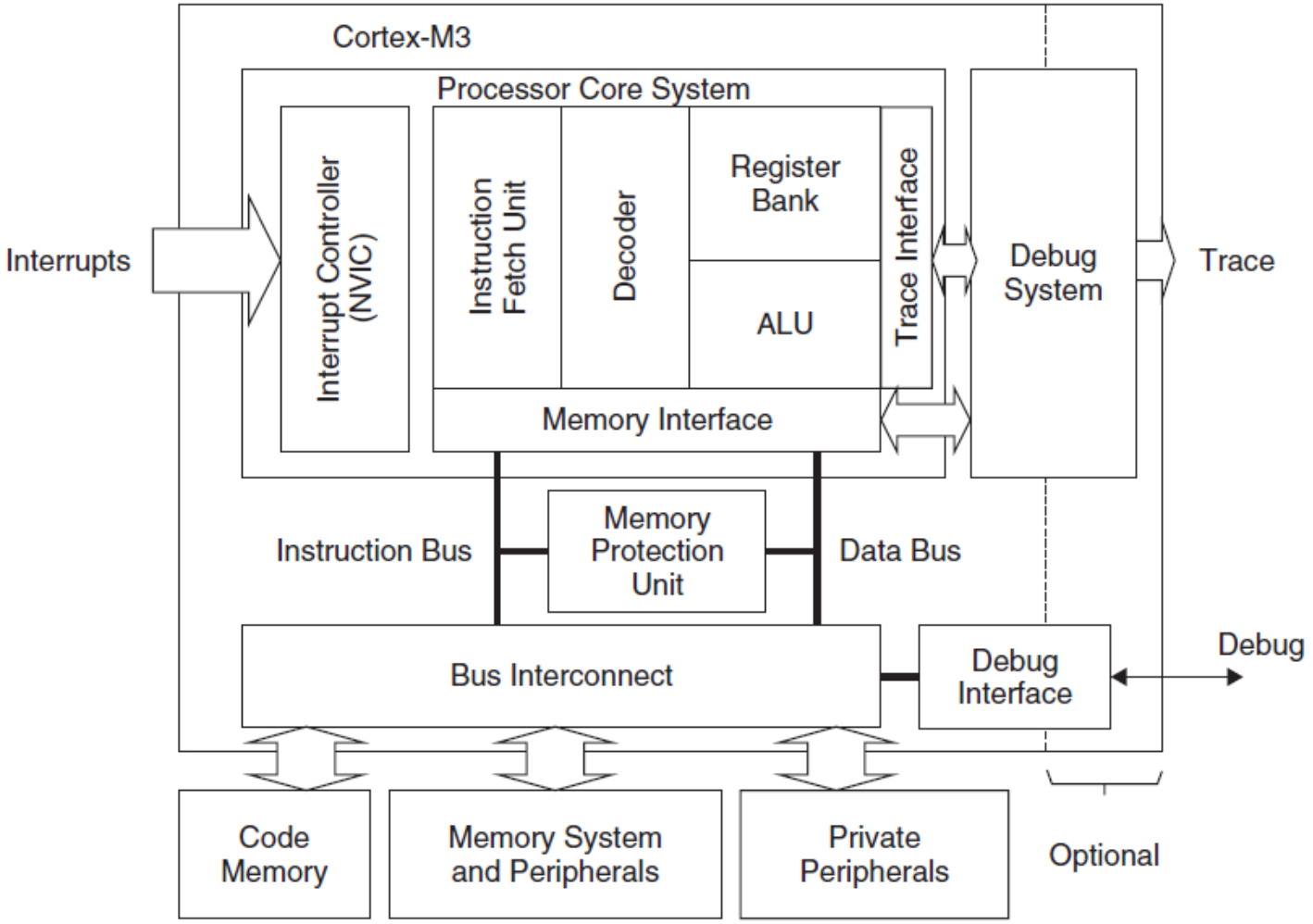
2.1 ARM Cortex-M3 - Architecture

- Endian mode
  - both little Endian and big Endian are supported
  - In most cases, Cortex-M3-based microcontrollers will be little endian.
  - Instruction fetches are always in little endian
  - PPB accesses are always in little endian.

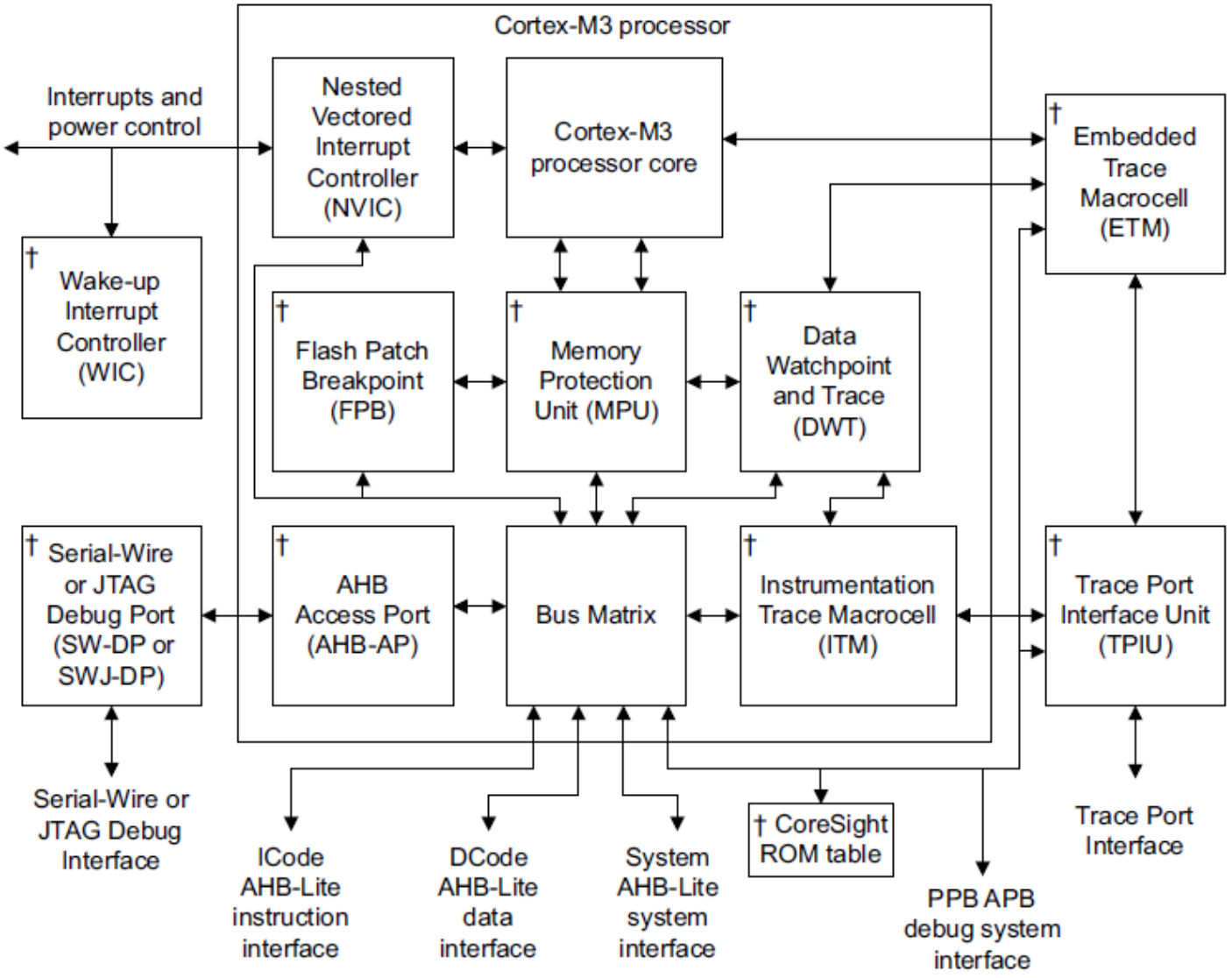


2.1 ARM Cortex-M3 - Architecture

Harvard architecture



2.1 ARM Cortex-M3 – Architecture



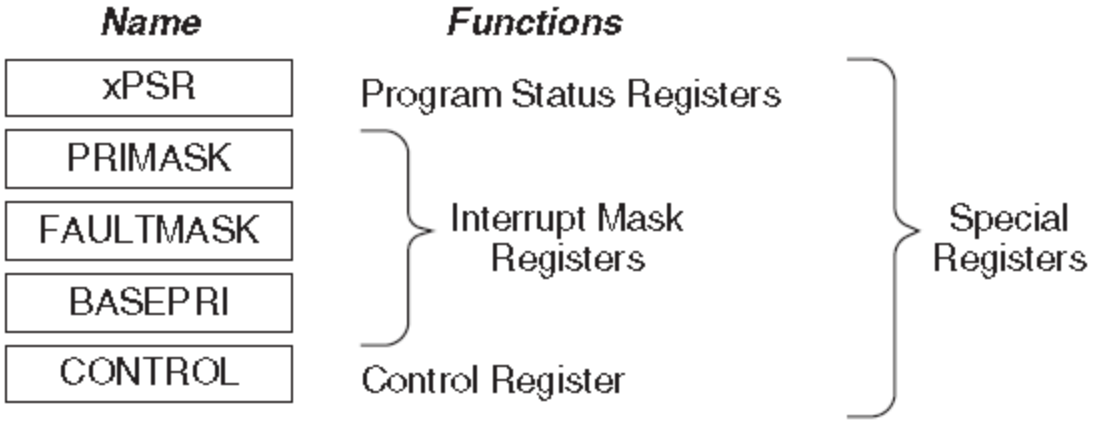
† Optional component

2.2 ARM Cortex-M3 - Registers

- Registers
  - R0 – R12: general purpose registers
  - R13: stack pointers
  - R14: link register (store return address)
  - R15: program counter
- Special Registers
  - Program Status Registers (PSRs)
  - Interrupt Mask Registers (PRIMASK, FAULTMASK, BASEPRI)
  - Control Register (CONTROL)

2.2 ARM Cortex-M3 - Registers

- Special registers in the Cortex-M3



Register	Function
xPSR	Provide ALU flags (zero flag, carry flag), execution status, and current executing interrupt number
PRIMASK	Disable all interrupts except the nonmaskable interrupt (NMI) and HardFault
FAULTMASK	Disable all interrupts except the NMI
BASEPRI	Disable all interrupts of specific priority level or lower priority level
CONTROL	Define privileged status and stack pointer selection

2.2 ARM Cortex-M3 - Registers

- **Program Status Registers**
  - Application PSR (APSR)
  - Interrupt PSR (IPSR)
  - Execution PSR (EPSR)

	31	30	29	28	27	26:25	24	23:20	19:16	15:10	9	8	7	6	5	4:0
APSR	N	Z	C	V	Q											
IPSR												Exception Number				
EPSR						ICI/IT	T			ICI/IT						

Figure 3.3 Program Status Registers (PSRs) in the Cortex-M3

	31	30	29	28	27	26:25	24	23:20	19:16	15:10	9	8	7	6	5	4:0
xPSR	N	Z	C	V	Q	ICI/IT	T			ICI/IT		Exception Number				

Figure 3.4 Combined Program Status Registers (xPSR) in the Cortex-M3

2.2 ARM Cortex-M3 - Registers

- **The Control register**
  - has two bits
  - used to define the privilege level and the stack pointer selection.

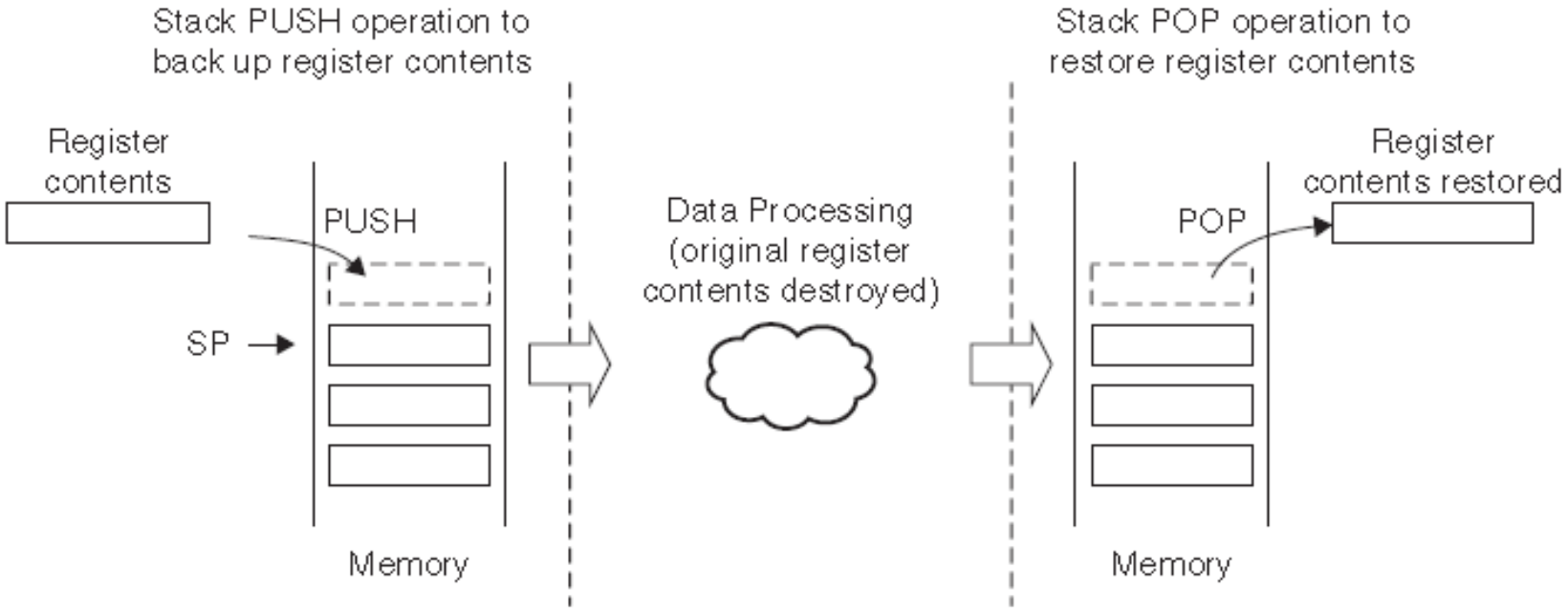
Bit	Function
CONTROL[1]	Stack status: 1 = Alternate stack is used 0 = Default stack (MSP) is used If it is in the Thread or base level, the alternate stack is the PSP. There is no alternate stack for handler mode, so this bit must be zero when the processor is in handler mode.
CONTROL[0]	0 = Privileged in Thread mode 1 = User state in Thread mode If in handler mode (not Thread mode), the processor operates in privileged mode.

2.2 ARM Cortex-M3 - Registers

Name	Type[a]	Required privilege[b]	Reset value	Description
R0-R12	RW	Either	Unknown	*General-purpose registers* on page 2-4
MSP	RW	Privileged	See description	*Stack Pointer* on page 2-4
PSP	RW	Either	Unknown	*Stack Pointer* on page 2-4
LR	RW	Either	0xFFFFFFFF	*Link Register* on page 2-4
PC	RW	Either	See description	*Program Counter* on page 2-4
PSR	RW	Privileged	0x01000000	*Program Status Register* on page 2-4
ASPR	RW	Either	Unknown	*Application Program Status Register* on page 2-5
IPSR	RO	Privileged	0x00000000	*Interrupt Program Status Register* on page 2-6
EPSR	RO	Privileged	0x01000000	*Execution Program Status Register* on page 2-6
PRIMASK	RW	Privileged	0x00000000	*Priority Mask Register* on page 2-8
FAULTMASK	RW	Privileged	0x00000000	*Fault Mask Register* on page 2-8
BASEPRI	RW	Privileged	0x00000000	*Base Priority Mask Register* on page 2-9
CONTROL	RW	Privileged	0x00000000	*CONTROL register* on page 2-9

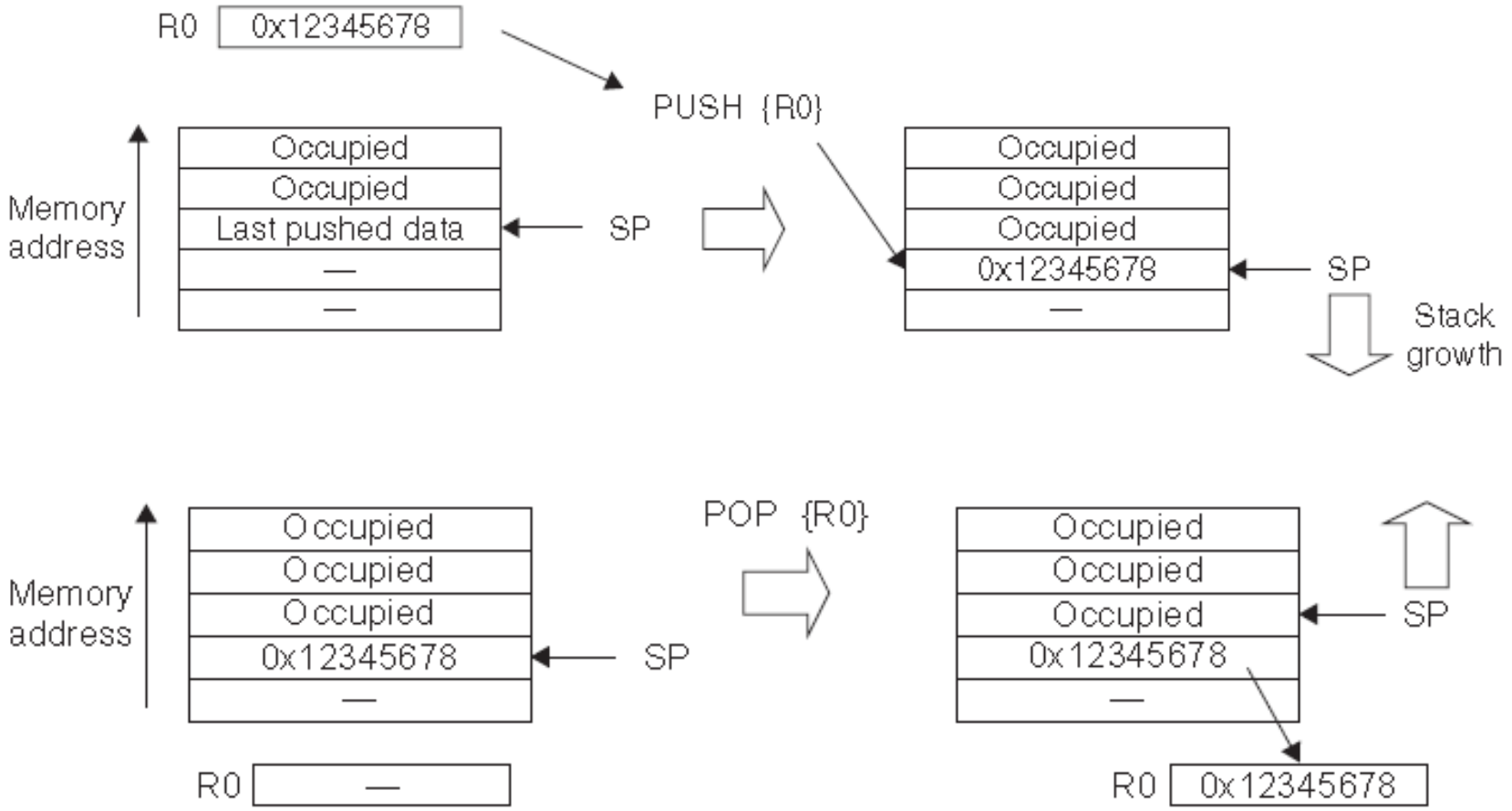
2.3 ARM Cortex-M3 - Stack Pointer

- Cortex-M3 processor has two stack pointers
  - Main Stack Pointer (MSP): used by the OS kernel, exception handlers
  - Process Stack Pointer (PSP): Used by the base-level application code
- When using the register name R13, you can only access the current stack pointer



2.3 ARM Cortex-M3 – PUSH & POP

- The stack pointer (SP) points to the last data pushed to the stack memory,
- The SP decrements before a new PUSH operation



2.4 ARM Cortex-M3 – Operation Modes

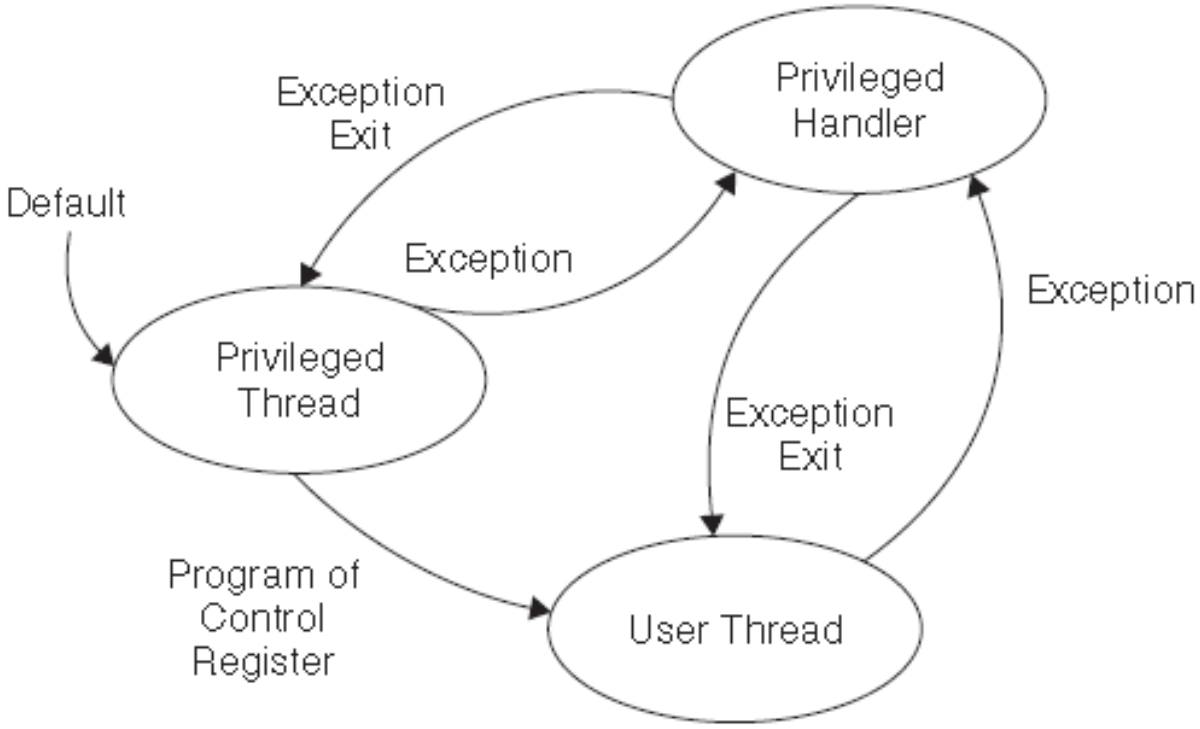
Cortex-M3 has 2 modes and 2 privilege levels

- Thread mode: Used to execute application software
- Handler mode: Used to handle exceptions
- Unprivileged:
  - has limited access to the MSR and MRS instructions, and cannot use the CPS instruction
  - cannot access the system timer, NVIC, or system control block
  - might have restricted access to memory or peripherals
- Privileged: can use all the instructions and has access to all resources

	Privileged	*User*
When running an exception	Handle Mode	
When running main program	Thread Mode	Thread Mode

2.4 ARM Cortex-M3 – Operation Modes

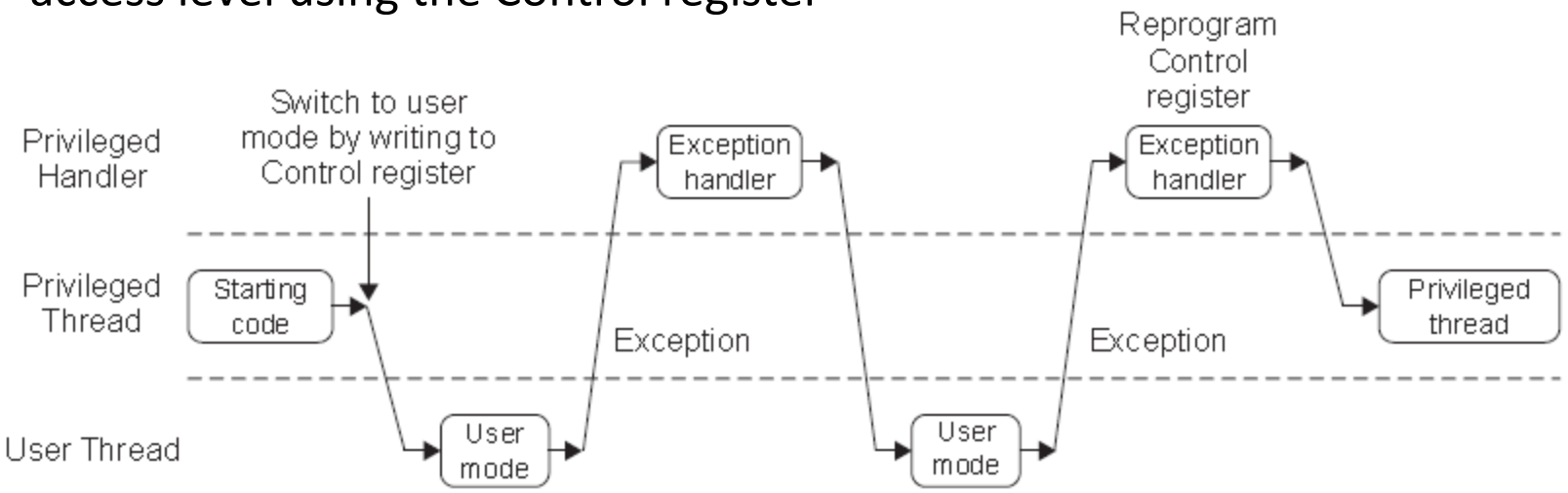
- Mode transitions



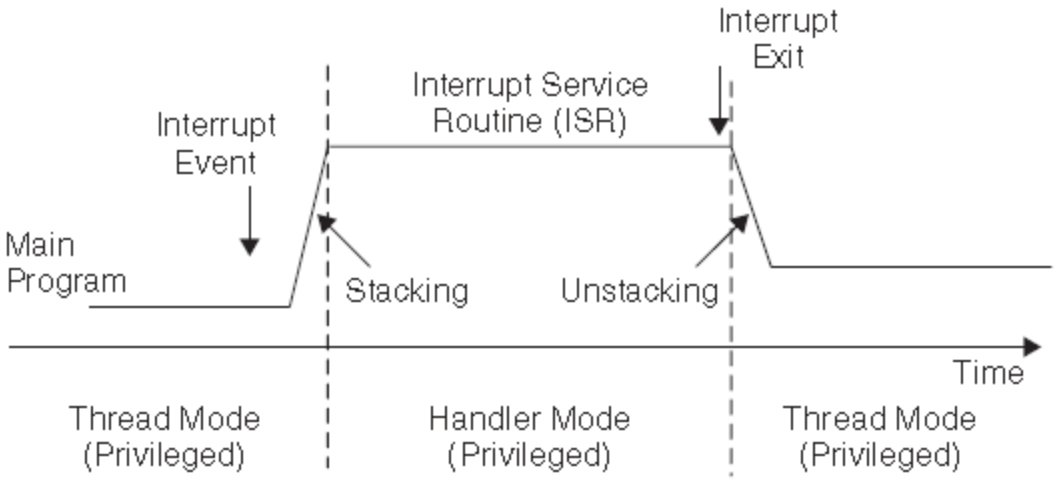
Allowed Operation Mode Transitions

2.2 ARM Cortex-M3 – Operation Modes

- Software in a privileged access level can switch the program into the user access level using the Control register



- Switching Processor Mode at Interrupt

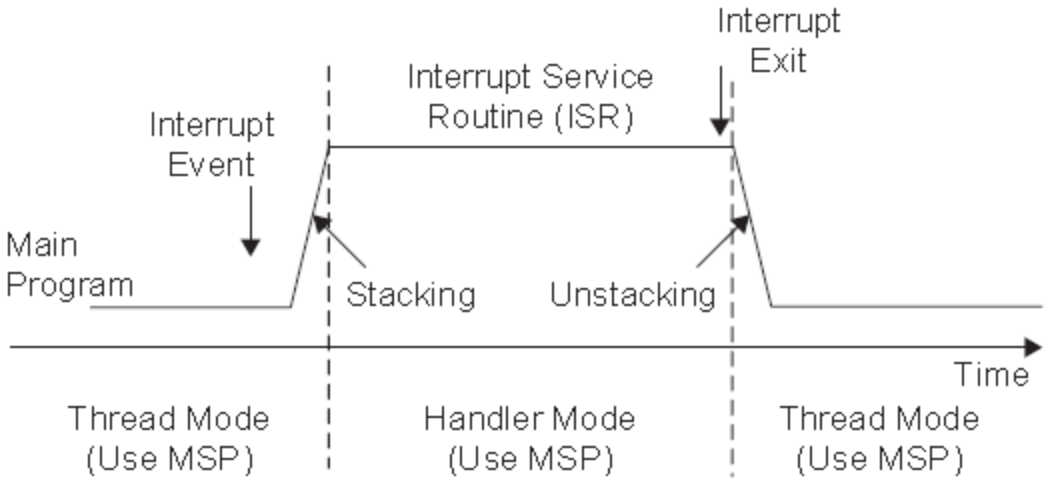


2.5 ARM Cortex-M3 - Interrupt

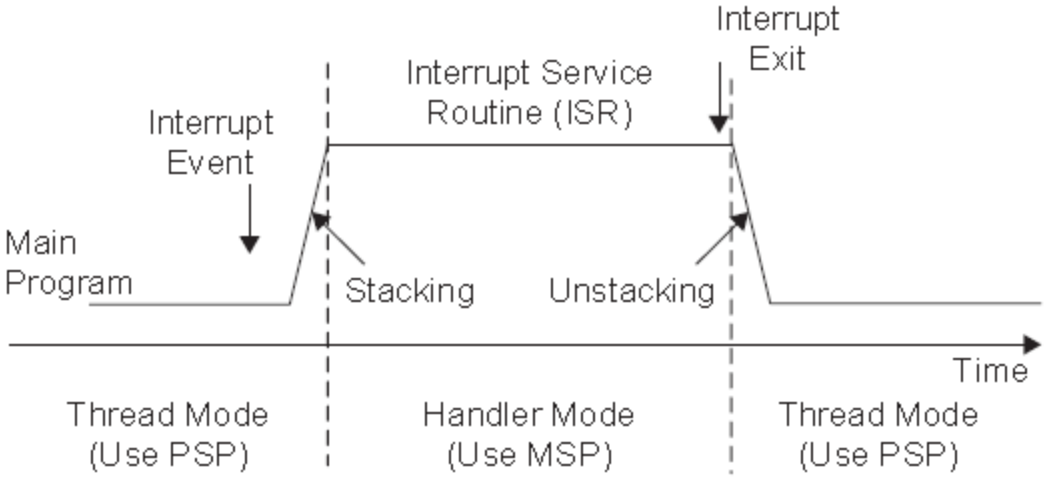
- The Built-In Nested Vectored Interrupt Controller (NVIC) provides a number of features:
  - **Nested interrupt support:** different priority levels
  - **Vectored interrupt support:** interrupt vector table in memory
  - **Dynamic priority changes support:** Priority levels of interrupts can be changed during run time.
  - **Reduction of interrupt latency:** automatic saving and restoring some register contents, reducing delay in switching
  - **Interrupt masking:** Interrupts and system exceptions can be masked

2.5 ARM Cortex-M3 - Interrupt

- Control[1] = 0: Both Thread Level and Handler Use Main Stack

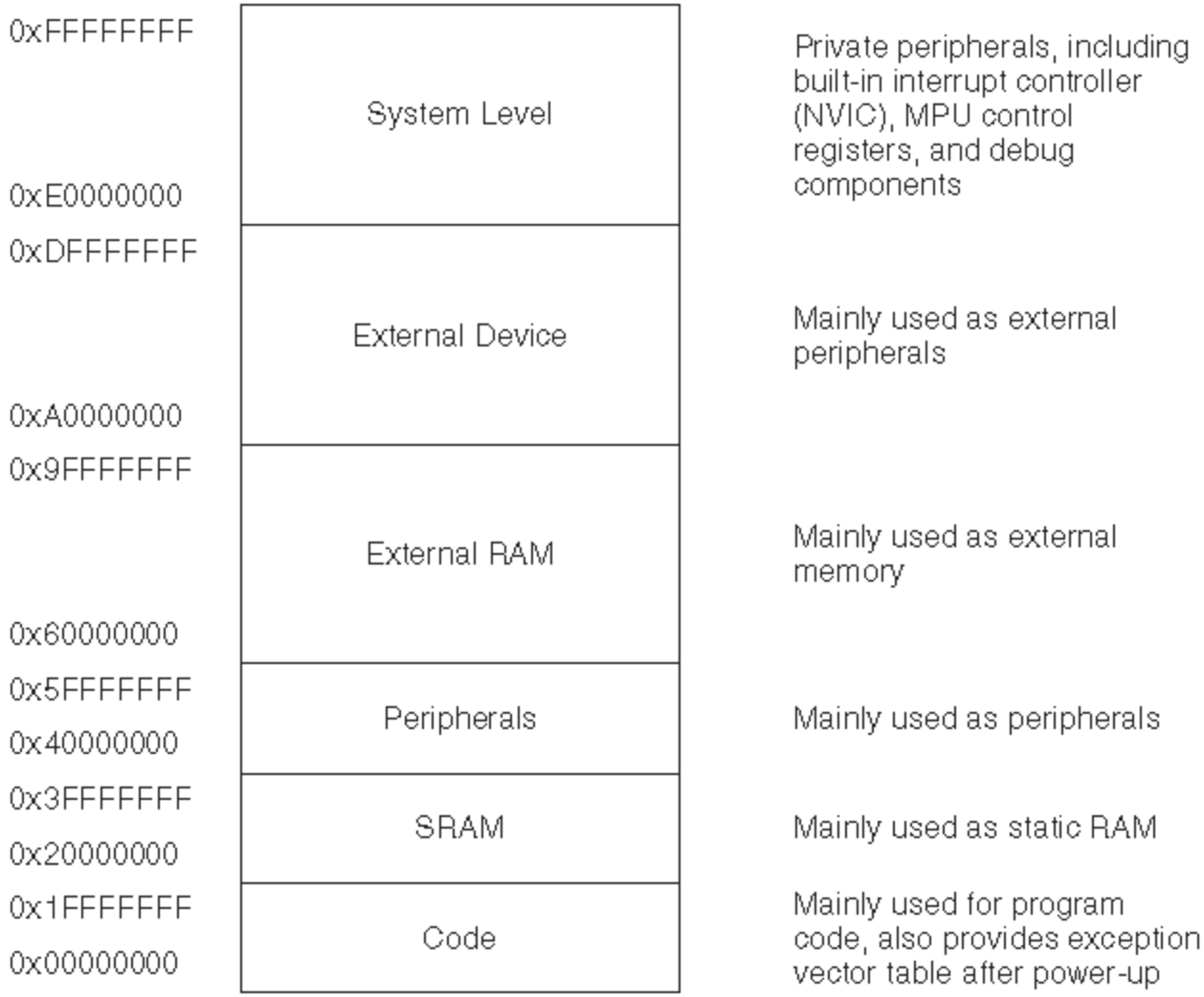


- Control[1] = 1: Thread Level Uses Process Stack and Handler Uses Main Stack



2.6 ARM Cortex-M3 – Memory Map

- The 4 GB memory space can be divided into the ranges shown in Figure:



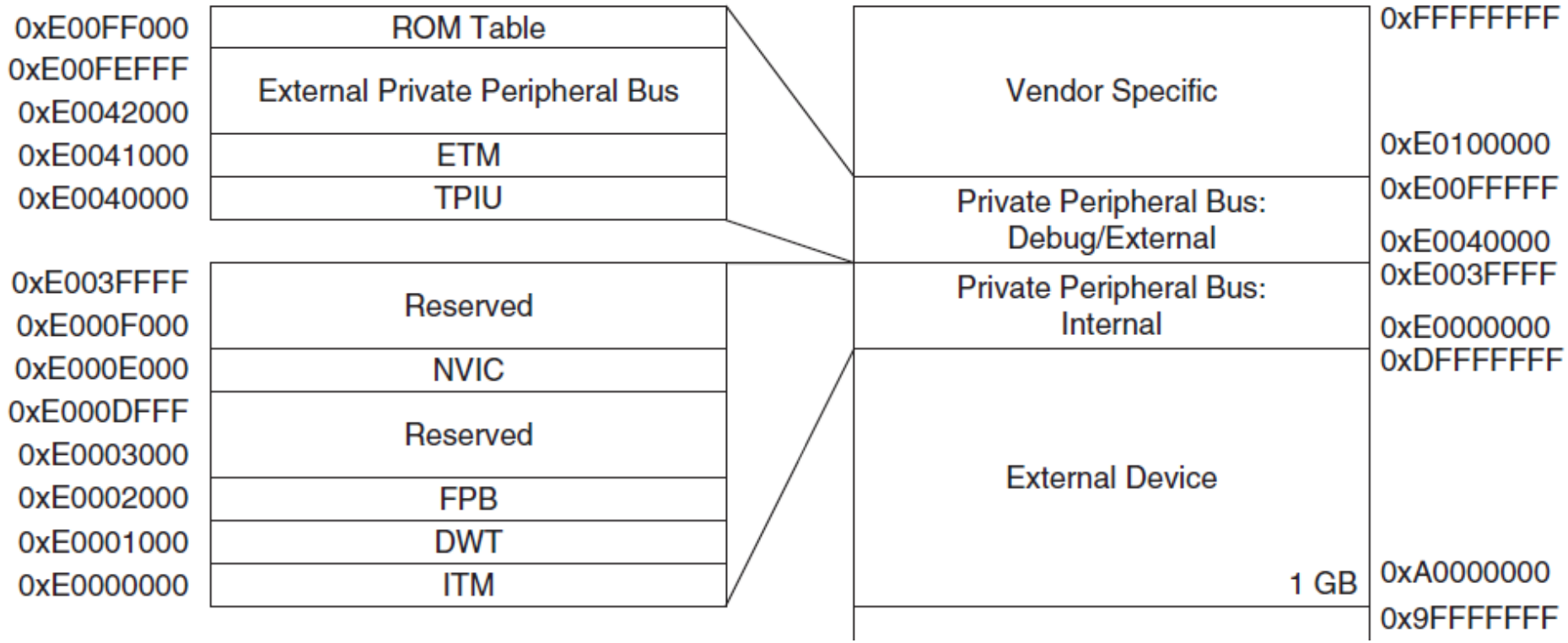
2.6 ARM Cortex-M3 – Memory Access

Address range	Memory region	Memory type[a]	XN[a]	Description
0x00000000-0x1FFFFFFF	Code	Normal	-	Executable region for program code. You can also put data here.
0x20000000-0x3FFFFFFF	SRAM	Normal	-	Executable region for data. You can also put code here.This region includes bit band and bit band alias areas, see Table 2-13 on page 2-16.
0x40000000-0x5FFFFFFF	Peripheral	Device	XN	This region includes bit band and bit band alias areas, see Table 2-14 on page 2-16.
0x60000000-0x9FFFFFFF	External RAM	Normal	-	Executable region for data.
0xA0000000-0xDFFFFFFF	External device	Device	XN	External Device memory.
0xE0000000-0xE00FFFFF	Private Peripheral Bus	Strongly-ordered	XN	This region includes the NVIC, System timer, and system control block.
0xE0100000-0xFFFFFFFF	Device	Device	XN	Implementation-specific.

Execute Never (XN) Means the processor prevents instruction accesses.

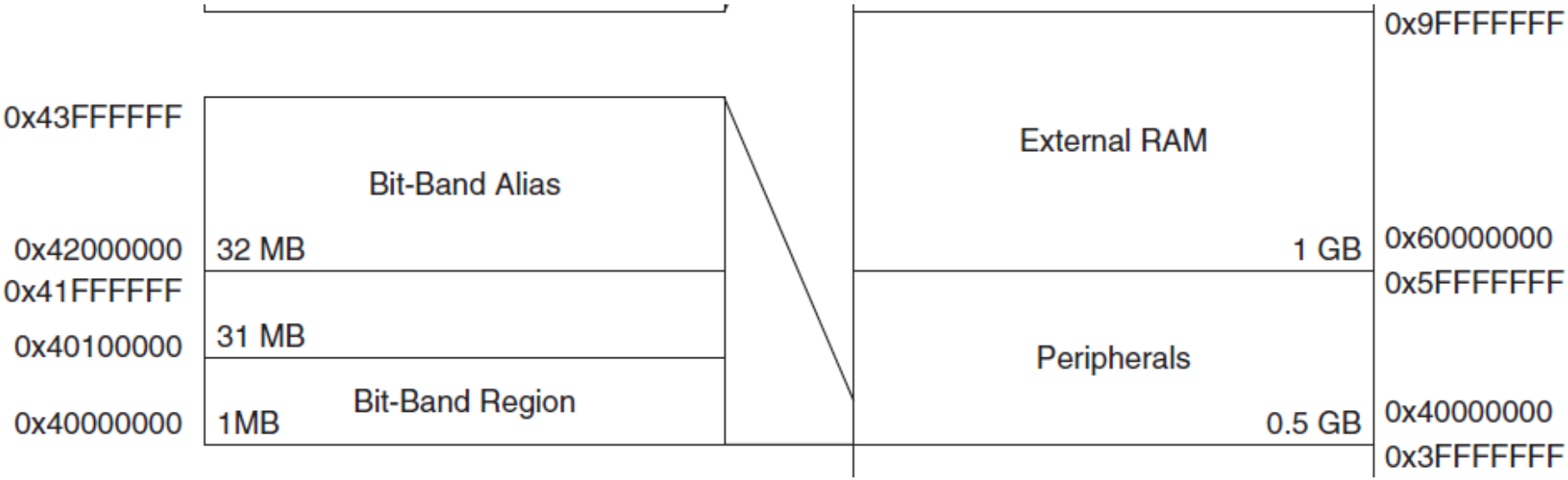
2.6 ARM Cortex-M3 – Memory Access

- System level memory
- External device memory



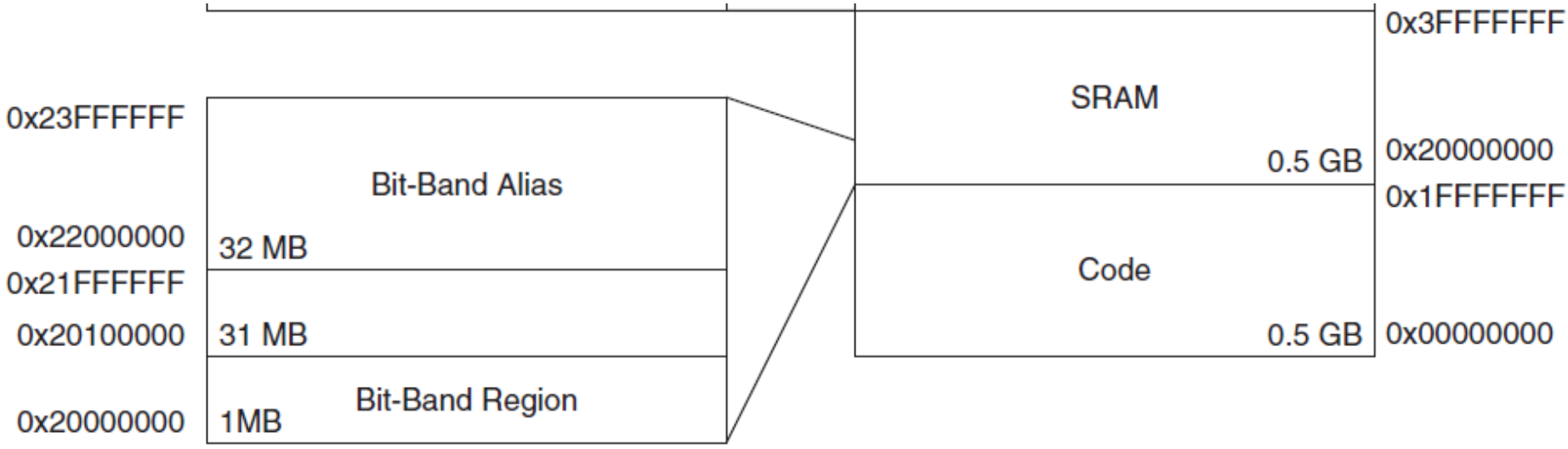
2.6 ARM Cortex-M3 – Memory Access

- Bit-Band region



2.6 ARM Cortex-M3 – Memory Access

- Bit-Band region



2.6 ARM Cortex-M3 – Memory Access

- Default memory access permissions

Memory Region	Address	Access in User Program
Vendor specific	0xE0100000–0xFFFFFFFF	Full access
ROM Table	0xE00FF000–0xE00FFFFF	Blocked; user access results in bus fault
External PPB	0xE0042000–0xE00FEFFF	Blocked; user access results in bus fault
ETM	0xE0041000–0xE0041FFF	Blocked; user access results in bus fault
TPIU	0xE0040000–0xE0040FFF	Blocked; user access results in bus fault
Internal PPB	0xE000F000–0xE003FFFF	Blocked; user access results in bus fault
NVIC	0xE000E000–0xE000EFFF	Blocked; user access results in bus fault, except Software Trigger Interrupt Register that can be programmed to allow user accesses
FPB	0xE0002000–0xE0003FFF	Blocked; user access results in bus fault
DWT	0xE0001000–0xE0001FFF	Blocked; user access results in bus fault
ITM	0xE0000000–0xE0000FFF	Read allowed; write ignored except for stimulus ports with user access enabled
External Device	0xA0000000–0xDFFFFFFF	Full access
External RAM	0x60000000–0x9FFFFFFF	Full access
Peripheral	0x40000000–0x5FFFFFFF	Full access
SRAM	0x20000000–0x3FFFFFFF	Full access
Code	0x00000000–0x1FFFFFFF	Full access

2.7 ARM Cortex-M3 – Bus Interface & MPU

- Bus Interface
  - Code memory bus for code memory, consist of two buses: I-Code and D-code
  - System bus: for memory and peripherals
  - Private peripheral bus: for private peripherals such as debugging components
- Memory Protection Unit (MPU)
  - allow access rules for privilege access and user program access
  - When an access rule is violated, a fault exception is generated
  - MPU is setup by an OS
  - allowing data used by privileged code to be protected from untrusted user programs

2.8 ARM Cortex-M3 – Instruction set

- Instruction Set:
  - support thumb-2 instruction set.
  - allow 32-bit and 16-bit instructions
- Traditional ARM processor has two operation states:
  - 32-bit ARM state
  - 16-bit Thumb state
- To get the best of both worlds, many applications have mixed ARM and Thumb codes
- Advantages over traditional ARM processor of ARM Cortex M3
  - No state switching overhead, saving both execution time and instruction space
  - No need to separate ARM code and Thumb code source files
  - easier to write software, because there is no need to worry about switching code between ARM and Thumb

2.9 ARM Cortex-M3 - Exceptions

- The interrupt features in the Cortex-M3 are implemented in the NVIC

Exception Number	Exception Type	Priority (Default to 0 if Programmable)	Description
0	NA	NA	No exception running
1	Reset	−3 (Highest)	Reset
2	NMI	−2	Nonmaskable interrupt (external NMI input)
3	Hard fault	−1	All fault conditions, if the corresponding fault handler is not enabled
4	MemManage fault	Programmable	Memory management fault; MPU violation or access to illegal locations
5	Bus fault	Programmable	Bus error (Prefetch Abort or Data Abort)
6	Usage fault	Programmable	Exceptions due to program error
7–10	Reserved	NA	Reserved

2.2 ARM Cortex-M3 - Exceptions

11	SVCall	Programmable	System service call
12	Debug monitor	Programmable	Debug monitor (break points, watchpoints, or external debug request)
13	Reserved	NA	Reserved
14	PendSV	Programmable	Pendable request for system device
15	SYSTICK	Programmable	System tick timer
16	IRQ #0	Programmable	External interrupt #0
17	IRQ #1	Programmable	External interrupt #1
...	...	...	...
255	IRQ #239	Programmable	External interrupt #239

- The number of external interrupt inputs is defined by chip manufacturers.
- A maximum of 240 external interrupt inputs can be supported

Questions

1. What are differences among Cortex- A, -R, and -M series?
2. What is Harvard architecture?
3. How many general purpose registers in Cortex-M3 are there?
4. How many special registers in Cortex-M3 are there?
5. What are difference between little endian and big endian?
6. How many allowed operation modes in Cortex-M3 are there?
7. What is functions of control register in Cortex-M3?
8. What are features of NVIC in Cortex-M3?
9. What is the maximum memory space of Cortex-M3?
10. What is the size of memory space for external RAM?
11. Can program code can be executed from an external RAM region?
12. Does Cortex-M3 support ARM code?
13. Why is Thumb-2 instruction set more advanced than previous?
14. In which areas of memory are instructions not permitted to be executed?
15. Can a non-maskable interrupt be preempted by an exception?