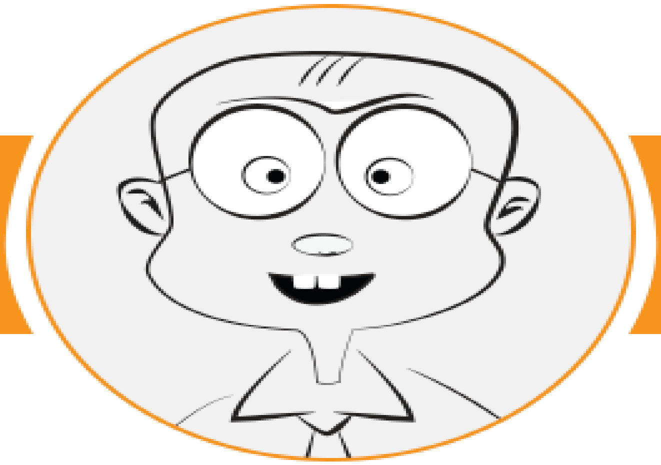


LECTURE 4: ARM Cortex-M architecture overview

EMBEDDED SYSTEM COURSE

Learning Goals

Introduce about the ARM Cortex M processor.

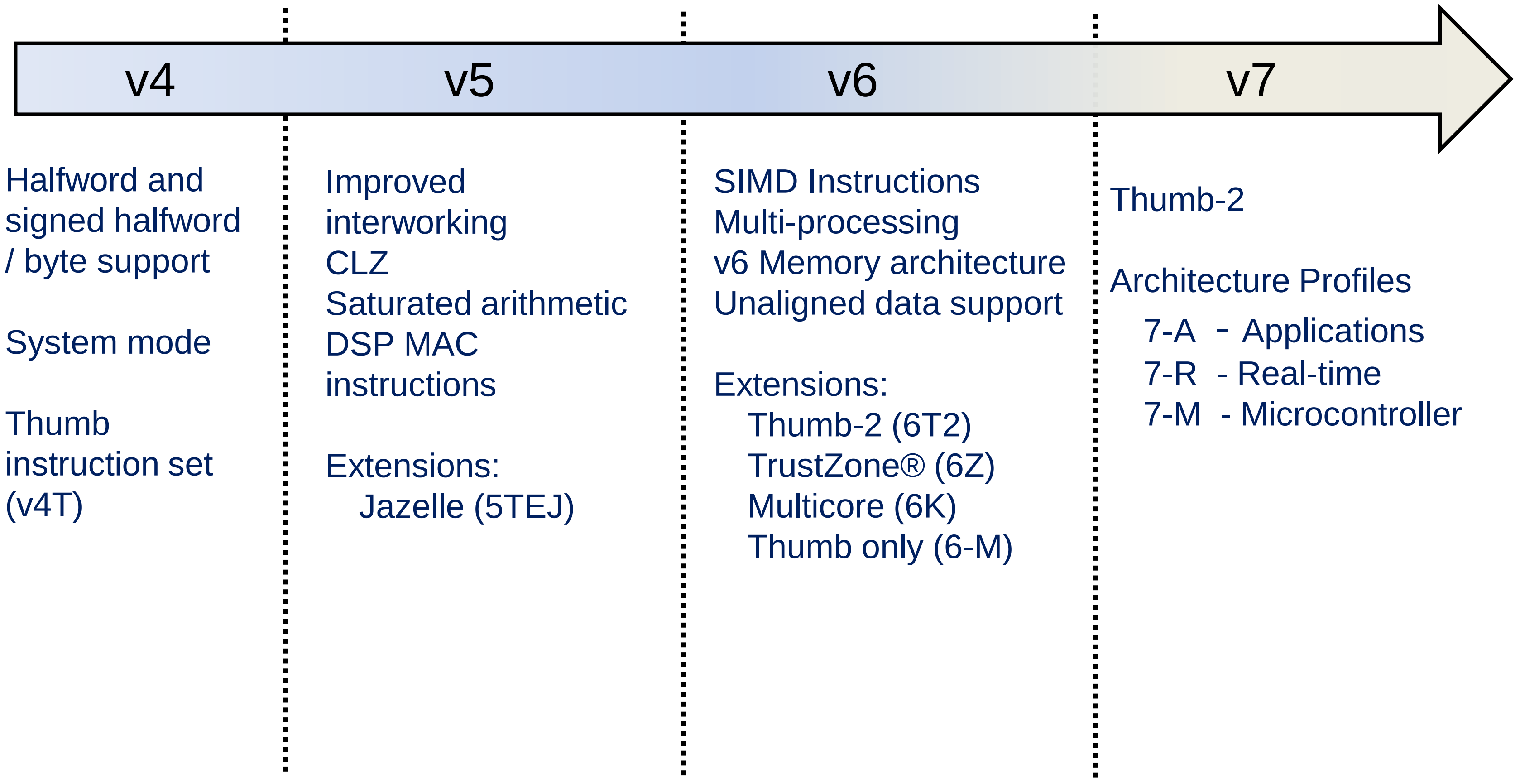
Explain some core components in Cortex-M including NVIC, SysTick timer and Floating Point Unit.

Explain about the basic concepts on Cortex-M instruction set.

Table of contents

Table of contents

Development of the ARM Architecture



v4	v5	v6	v7
Halfword and signed halfword / byte support System mode Thumb instruction set (v4T)	Improved interworking CLZ Saturated arithmetic DSP MAC instructions Extensions: Jazelle (5TEJ)	SIMD Instructions Multi-processing v6 Memory architecture Unaligned data support Extensions: Thumb-2 (6T2) TrustZone® (6Z) Multicore (6K) Thumb only (6-M)	Thumb-2 Architecture Profiles 7-A - Applications 7-R - Real-time 7-M - Microcontroller
		Cortex M0/M0+ ***Cortex M1***	***Cortex M3/M4***

General Information about the Cortex-M

ARM CORTEX

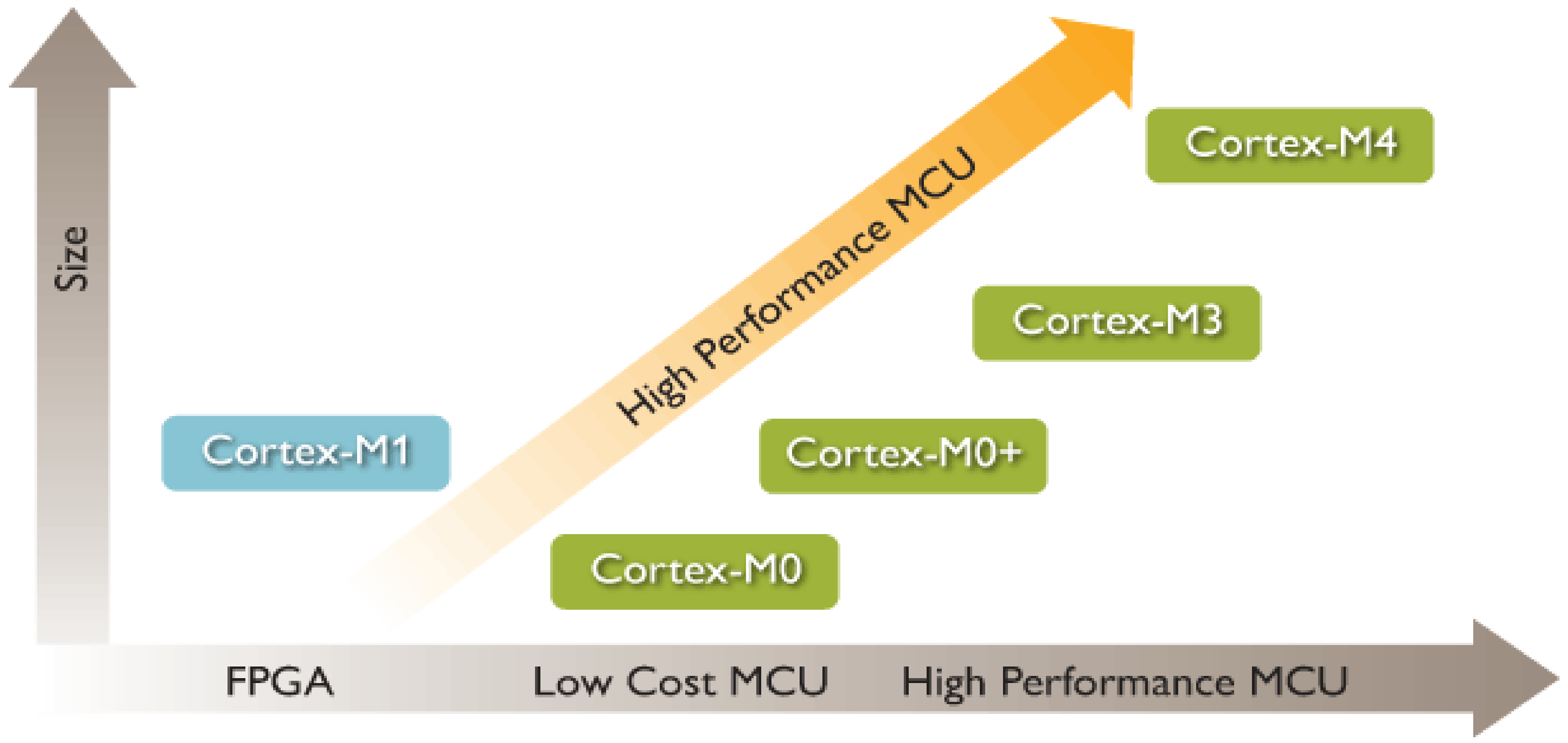
The ARM Cortex family includes processors based on the three distinct profiles of the ARMv7 architecture.

The A profile for sophisticated, high-end applications running open and complex operating systems

The R profile for real-time systems

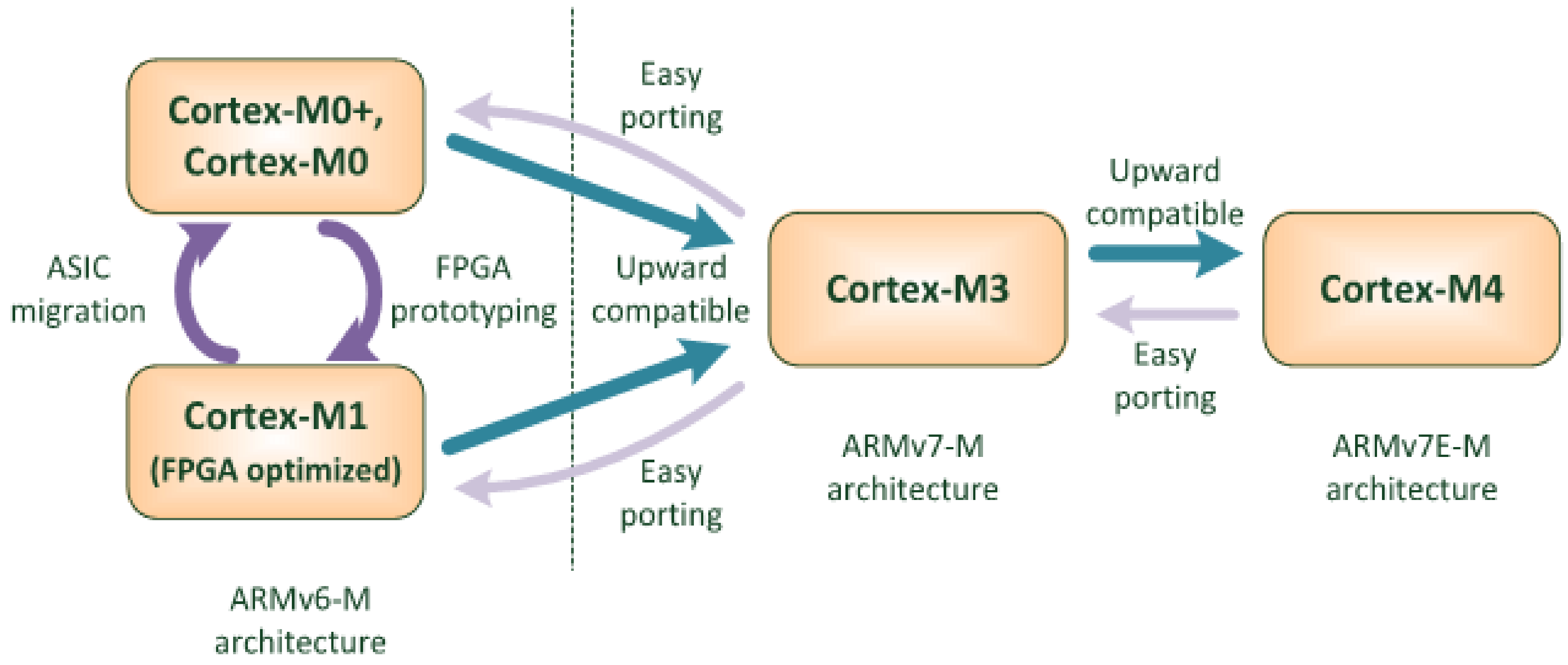
The M profile optimized for cost-sensitive and microcontroller applications

General Information about the Cortex-M



Embedded Processors

General Information about the Cortex-M



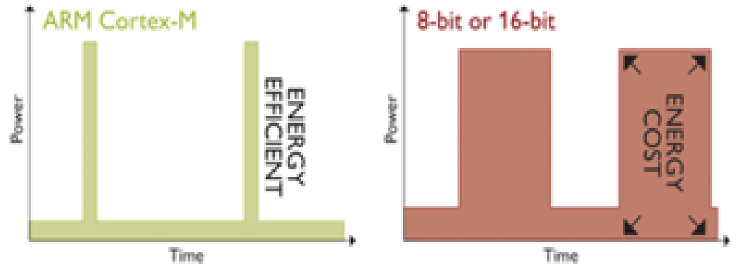
Cortex-M compatibility

Cortex M processor

General Information about the Cortex-M

Cortex-M Advantages:

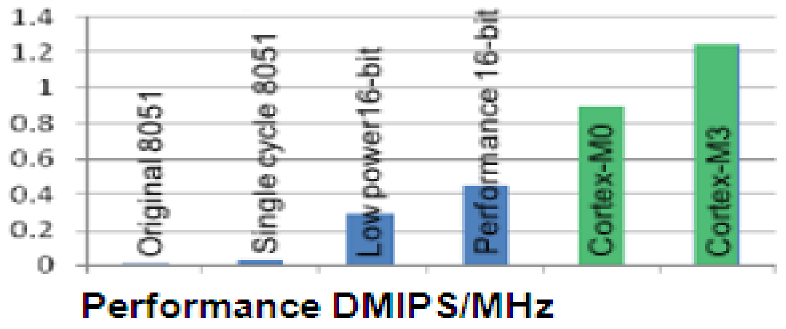
1. Energy efficiency



2. Smaller code

3. Ease of use

4. High performance



General Information about the Cortex-M

Cortex M0

- 32-bit RISC processor
- 3-stage pipeline von Neumann architecture
- ARMv6-M architecture
- 16-bit Thumb instruction set with Thumb-2 technology
- Load-Store Architecture
- 56 Instructions
- Low power support

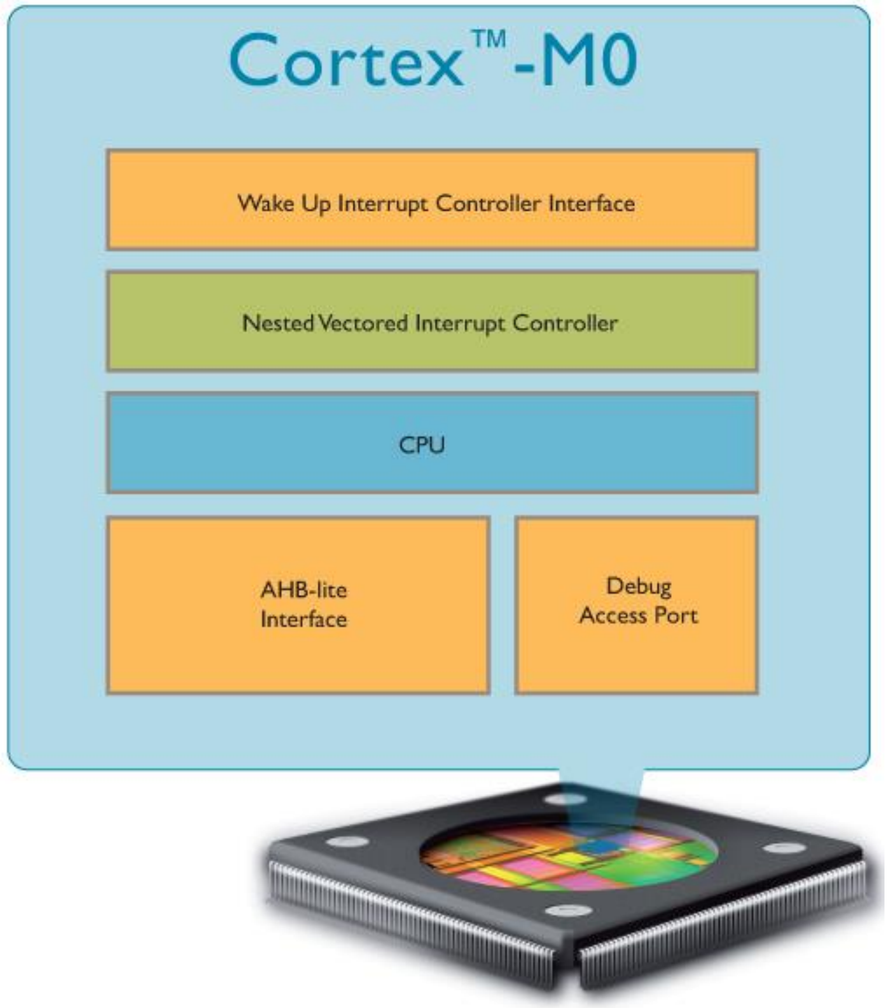
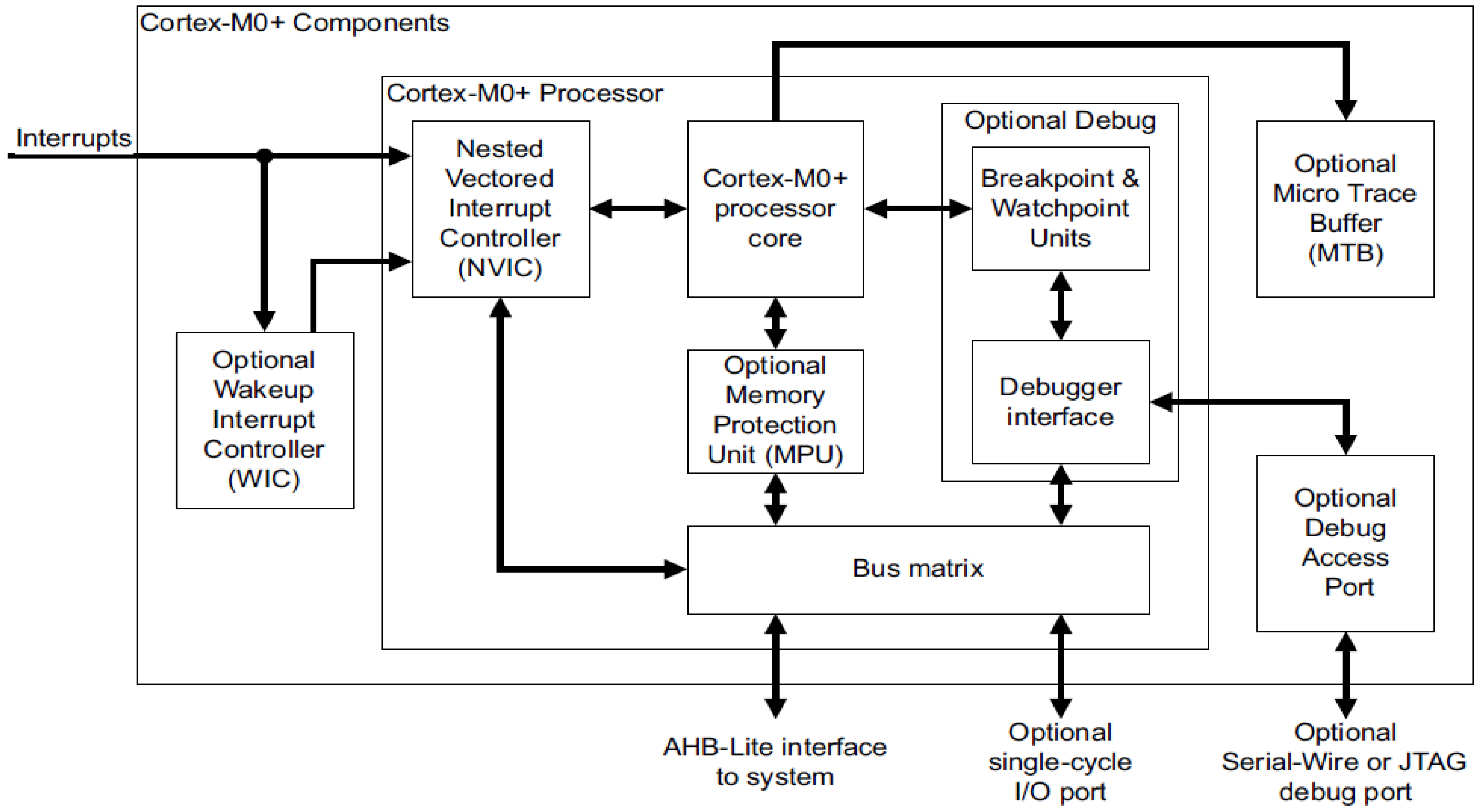

Cortex™-M0
Wake Up Interrupt Controller Interface
Nested Vectored Interrupt Controller
CPU
AHB-lite Interface
Debug Access Port

Table of contents

Introduction to the architecture

Simplified Block Diagram


Cortex-M0+ Components
Cortex-M0+ Processor
Interrupts
Nested Vectored Interrupt Controller (NVIC)
Cortex-M0+ processor core
Optional Debug
Breakpoint & Watchpoint Units
Optional Micro Trace Buffer (MTB)
Optional Wakeup Interrupt Controller (WIC)
Optional Memory Protection Unit (MPU)
Debugger interface
Optional Debug Access Port
Bus matrix
AHB-Lite interface to system
Optional single-cycle I/O port
Optional Serial-Wire or JTAG debug port

Introduction to the architecture

Cortex-M0 Functional Blocks

- ARMv6-M Thumb instruction set
- NVIC: 32 external interrupt inputs
- Debug: 4 HD breakpoints, 2 watchpoints.
- Bus interfaces: 32-bit AMBA-3 AHB-Lite system interface

Introduction to the architecture

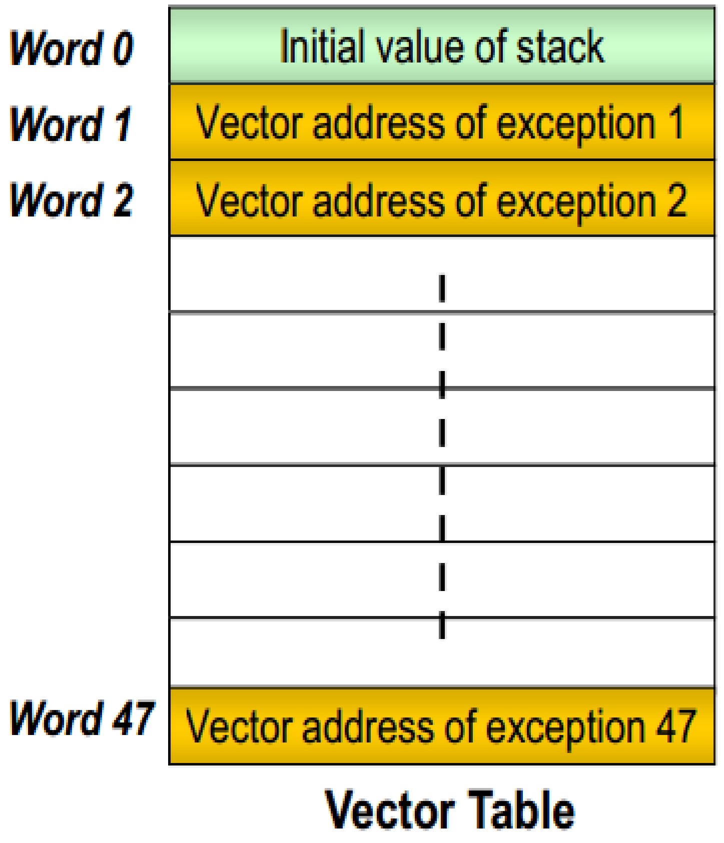
Memory model

- 32-bit address space
- Virtual memory is not supported in ARMv6-M
- Instruction fetches are always half-word-aligned
- Data accesses are always naturally aligned

Introduction to the architecture

Exception model

- Each exception has exception number, priority number and vector address
- An exception may be an interrupt or a hardware error
- Vector table base address is fixed at 0x00


Word 0
Initial value of stack
Word 1
Vector address of exception 1
Word 2
Vector address of exception 2
Word 47
Vector address of exception 47
Vector Table

Introduction to the architecture

System control space

Consists of the following groups:

- CPUID space.
- System control, configuration

  and status.
- SysTick system timer
- Nested Vectored Interrupt Controller (NVIC)

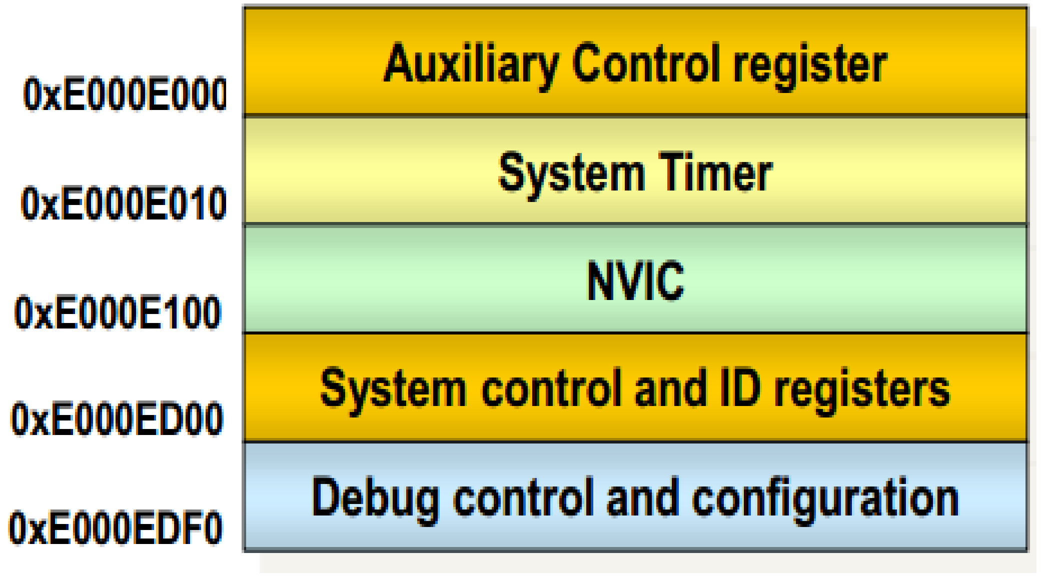
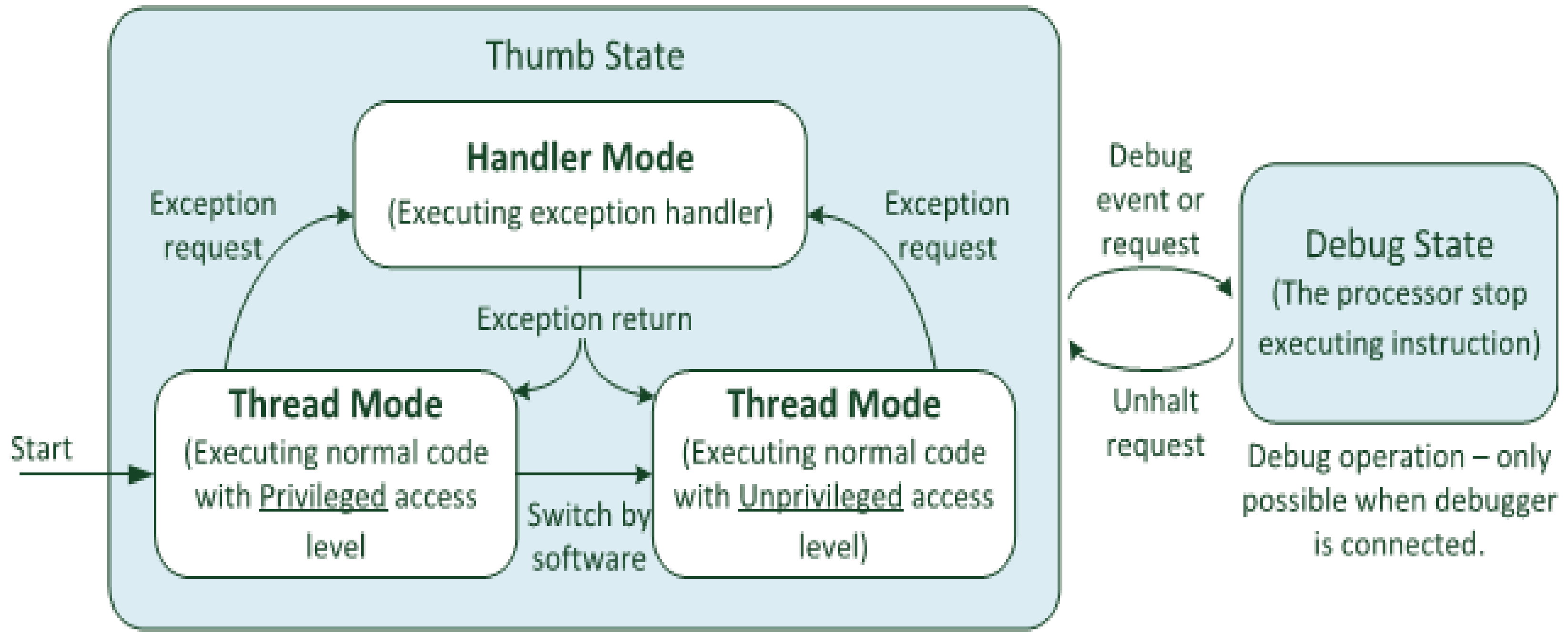

0xE000E000
0xE000E010
0xE000E100
0xE000ED00
0xE000EDF0
Auxiliary Control register
System Timer
NVIC
System control and ID registers
Debug control and configuration

Table of contents

Programmer's Model

Operation Modes & States

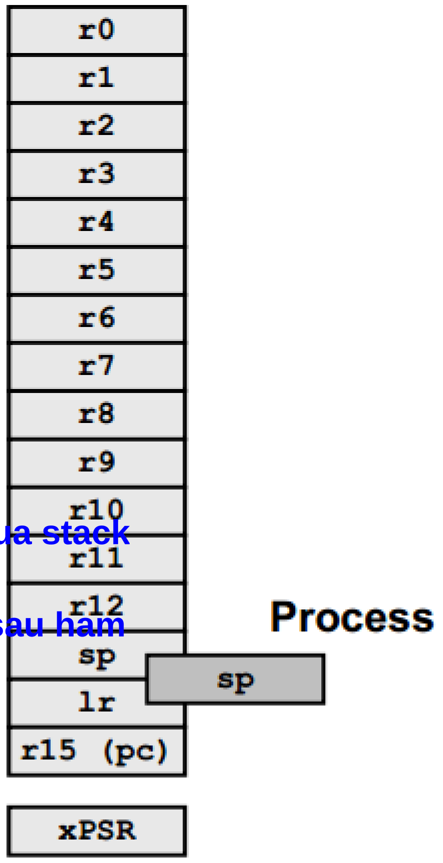


Thumb State
Handler Mode
(Executing exception handler)
Exception request
Exception request
Exception return
Thread Mode
(Executing normal code with Privileged access level
Thread Mode
(Executing normal code with Unprivileged access level)
Start
Switch by software
Debug event or request
Unhalt request
Debug State
(The processor stop executing instruction)
Debug operation – only possible when debugger is connected.

Programmer's Model

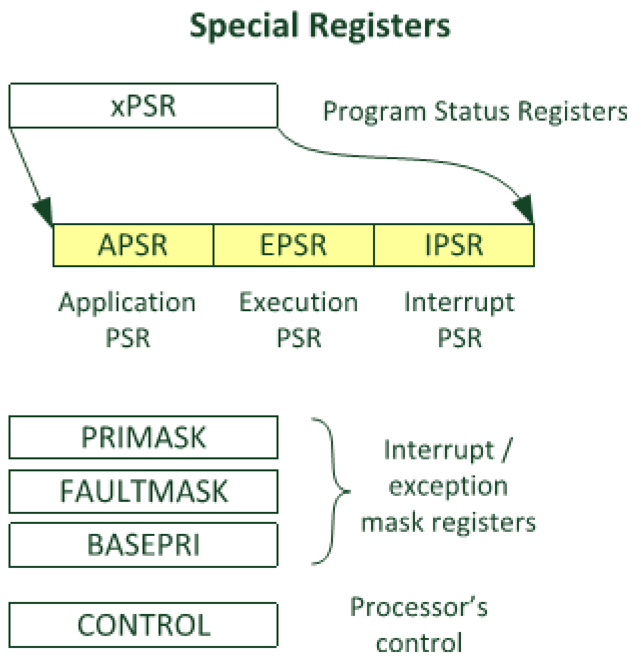
Core Registers

- All registers are 32 bits wide
- 13 general purpose registers
- Registers r0 – r7 (Low registers)
- Registers r8 – r12 (High registers)
- 3 registers with special meaning/usage
  - ✓ Stack Pointer (SP) – r13 luwu address hien tai data cuoi cung cua stack
  - ✓ Link Register (LR) – r14 luwu tru sia chi cua cau lenh tiep theo sau ham
  - ✓ Program Counter (PC) – r15 tro vao dia chi hien tai
- Special-purpose registers
  - ✓ xPSR shows a composite of the content of
  - ✓ APSR, IPSR, EPSR



Programmer's Model

Special Registers



APSR, IPSR, and EPSR

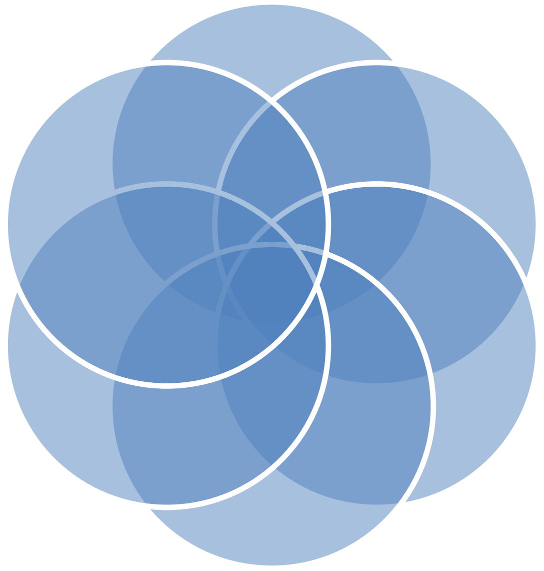
	31	30	29	28	27	26:25	24	23:20	19:16	15:10	9	8	7	6	5	4:0
xPSR	N	Z	C	V	Q	ICI/IT	T		GE*	ICI/IT		Exception Number				

Programmer's Model

Stack

Full descending: stack pointer indicates the last stacked item on the stack memory.

Two stacks, two independent stack pointers.



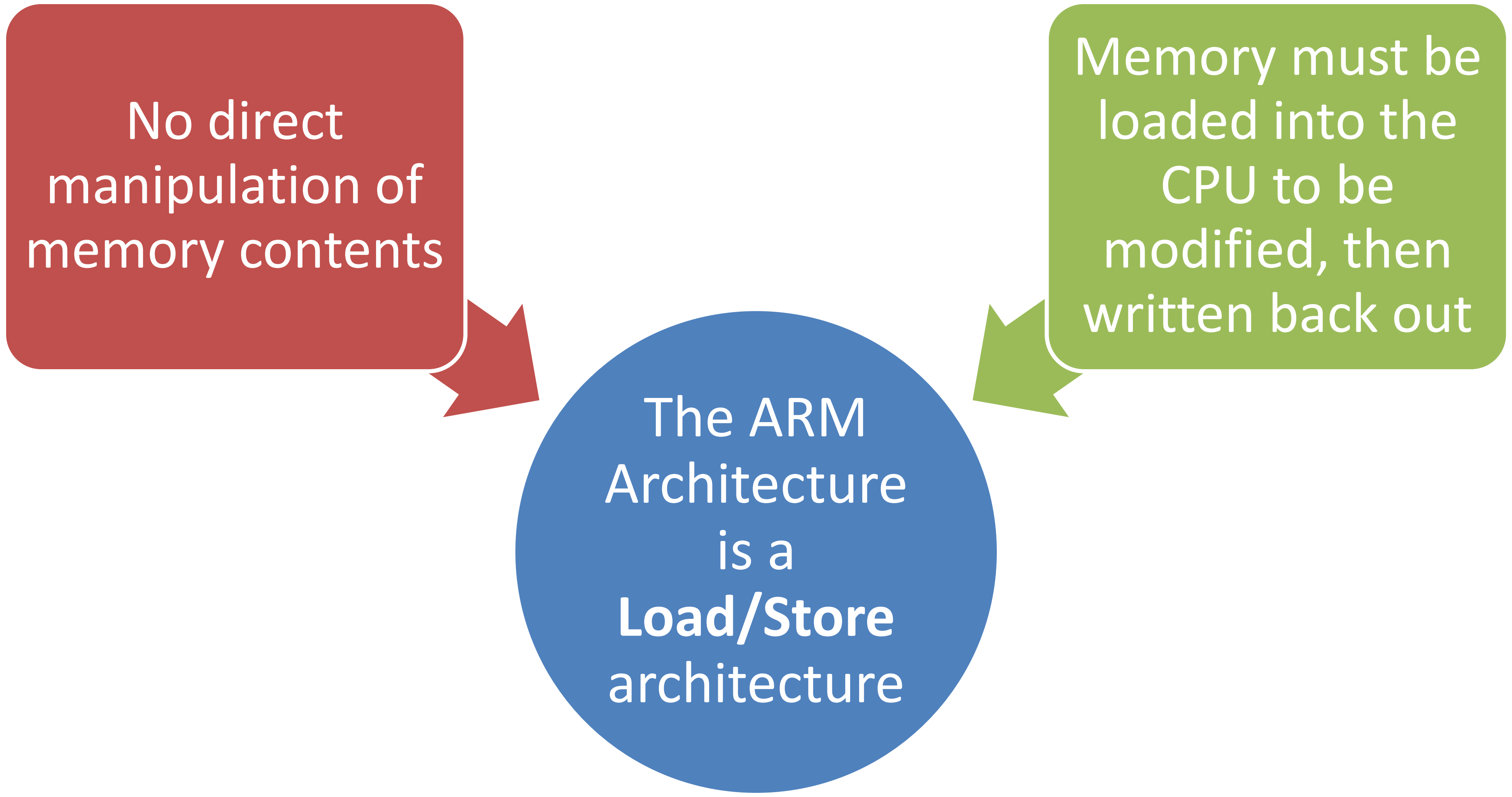
In an OS environment, ARM recommends that threads running in Thread mode use the process stack and the kernel and exception handlers use the main stack

Handler mode always uses the MSP (Main Stack Pointer)

Thread mode can use MSP (Main Stack Pointer) by default, or PSP (Process Stack Pointer).

Table of contents

Instruction Set Architecture



No direct manipulation of memory contents

Memory must be loaded into the CPU to be modified, then written back out

The ARM Architecture is a **Load/Store** architecture

Instruction Set Architecture

- ***Thumb-2 Instruction set***
  - Variable-length instructions
    - ARM instructions are a fixed length of 32 bits
    - Thumb instructions are a fixed length of 16 bits
    - Thumb-2 instructions can be either 16-bit or 32-bit
  - Thumb-2 gives approximately 26% improvement in code density over ARM
  - Thumb-2 gives approximately 25% improvement in performance over Thumb

Cortex M0 ISA Overview:

- ARMv6-M supports Thumb-2 technology

  (The ARM instruction set is not supported)

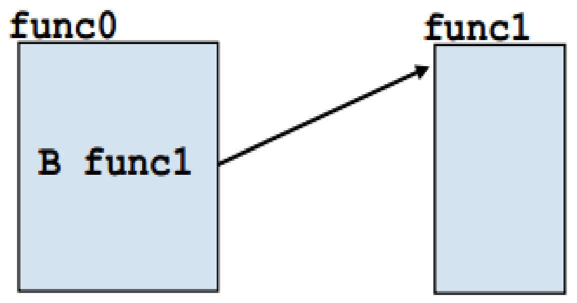
- Thumb-2 technology supports mixed 16-bit/32-bit instructions
- Small number of additional 32-bit instructions supported
- Conditional execution is supported
- Optimized for compilation from C
  - ✓ Thumb-2 instructions are not designed to be written by hand
  - ✓ Easy to learn due to small number of mnemonics

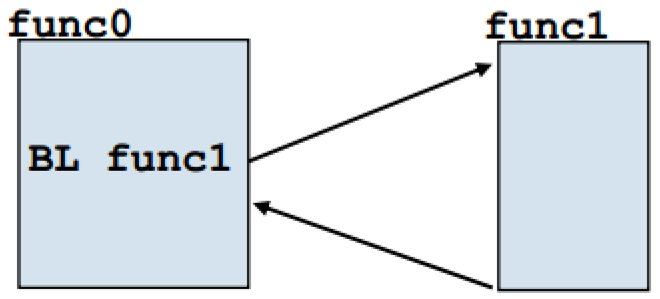
Instruction Classes

- Branch instructions
- Data-processing instructions
- Load and store instructions
- Status register access instructions
- Miscellaneous instructions

Branch instructions

- B – Branch
  - ✓ Absolute branch to a target address, relative to Program Counter (PC)
  - ✓ +/- 256 bytes range, conditional execution supported
  - ✓ +/- 1MB range, no conditional execution supported
- BL – Branch with Link
  - ✓ Branch to a subroutine – Link register is updated
  - ✓ +/- 16MB range, relative to Program Counter (PC)





Instruction Set Architecture

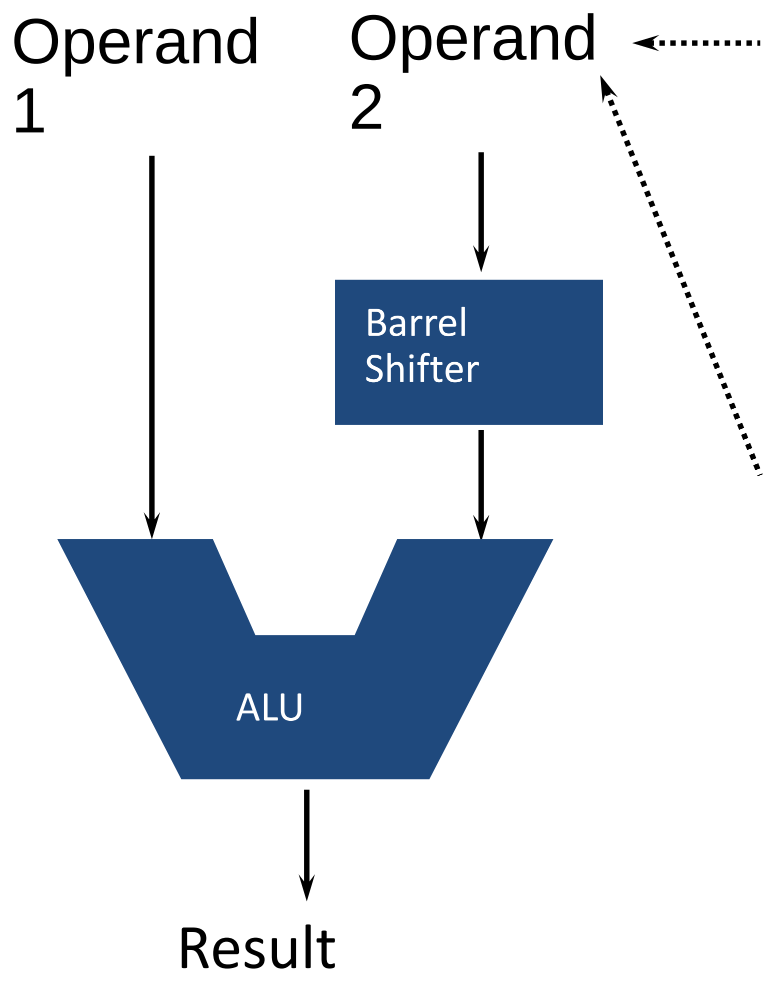
Data Processing Instructions:

- Consist of:
  - Arithmetic: **ADD ADC SUB SBC RSB RSC**
  - Logical: **AND ORR EOR BIC**
  - Comparisons: **CMP CMN TST TEQ**
  - Data movement: **MOV MVN**
- These instructions only work on registers, NOT memory.

- Syntax:

  <Operation>{<cond>}{S} Rd, Rn, Operand2
  - Comparisons set flags only - they do not specify Rd
  - Data movement does not specify Rn
  - Second operand is sent to the ALU via barrel shifter.

Using Barrel Shifter:



Register, optionally with shift operation

- Shift value can be either be:
  - 5 bit unsigned integer
  - Specified in bottom byte of another register.
- Used for multiplication by constant

Immediate value

- 8 bit number, with a range of 0-255.
  - Rotated right through even number of positions
- Allows increased range of 32-bit constants to be loaded directly into registers

Load and store instructions:

LDR	**STR**	Word
LDRB	**STRB**	Byte
LDRH	**STRH**	Halfword
LDRSB		Signed byte load
LDRSH		Signed halfword load

- Memory system must support all access sizes

- Syntax:
  - **LDR**{<cond>}{<size>} Rd, <address>
  - **STR**{<cond>}{<size>} Rd, <address>

e.g. **LDREQB**

Status Register Access Instructions

MRS/MSR - **Move data between a general purpose register and status register**

- **MRS (Register ← Status Register)**
- **MSR (Status Register ← Register)**

- Syntax:
  - **LDR**{<cond>}{<size>} Rd, <address>
  - **STR**{<cond>}{<size>} Rd, <address>

Table of contents

Summary

- All the ARM Cortex M processors are 32-bit RISC (Reduced Instruction Set Computing) processors. They have:
    - 32-bit registers
    - 32-bit internal data path
    - 32-bit bus interface

- The Cortex-M processors contain the core of the processor, NVIC, the SysTick timer, and optionally the floating point unit (for Cortex-M4).

- All the ARM Cortex -M processors are based on Thumb technology, which allows a mixture of 16-bit and 32-bit instructions to be used within

Question & Answer

Thanks for your attention !

Copyright