



# The Arm Cortex-M4 Processor Architecture

# Module Syllabus

- Arm architectures and processors
  - What is Arm architecture?
  - Arm processor families
  - Arm Cortex-M series
  - Cortex-M4 processor
  - Arm processors vs. Arm architectures
- Arm Cortex-M4 Processor
  - Processor overview
  - Block diagram
  - Registers

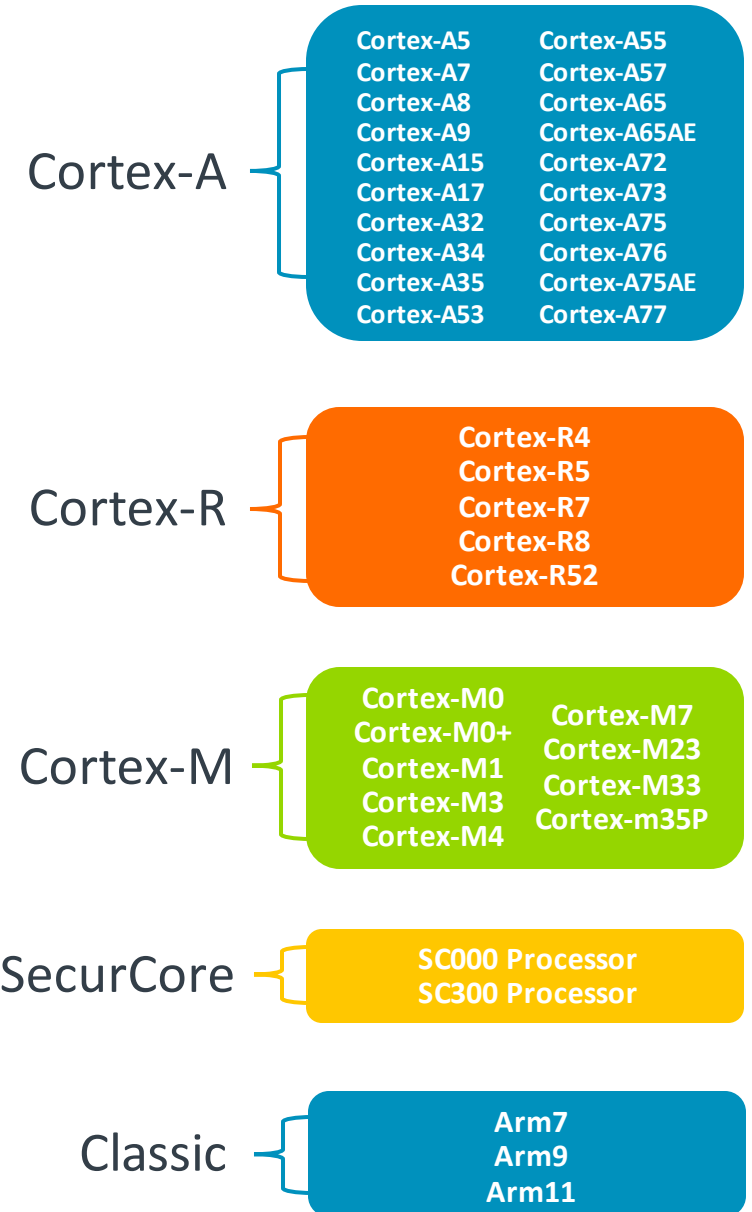
# Arm Architectures and Processors

- Arm architecture is a family of RISC-based processor architectures
  - Well known for its power efficiency
  - Widely used in mobile devices, e.g. smartphones and tablets
  - Designed and licensed by Arm to a wide ecosystem of partners
- Arm
  - The company that designs Arm-based processors
  - Arm does not manufacture, but it licenses designs to semiconductor partners who add their own intellectual property (IP) on top of Arm's OP, which they then fabricate and sell to customers
  - Arm also offers IP other than processors, such as physical IPs, interconnect IPs, graphics cores and development tools



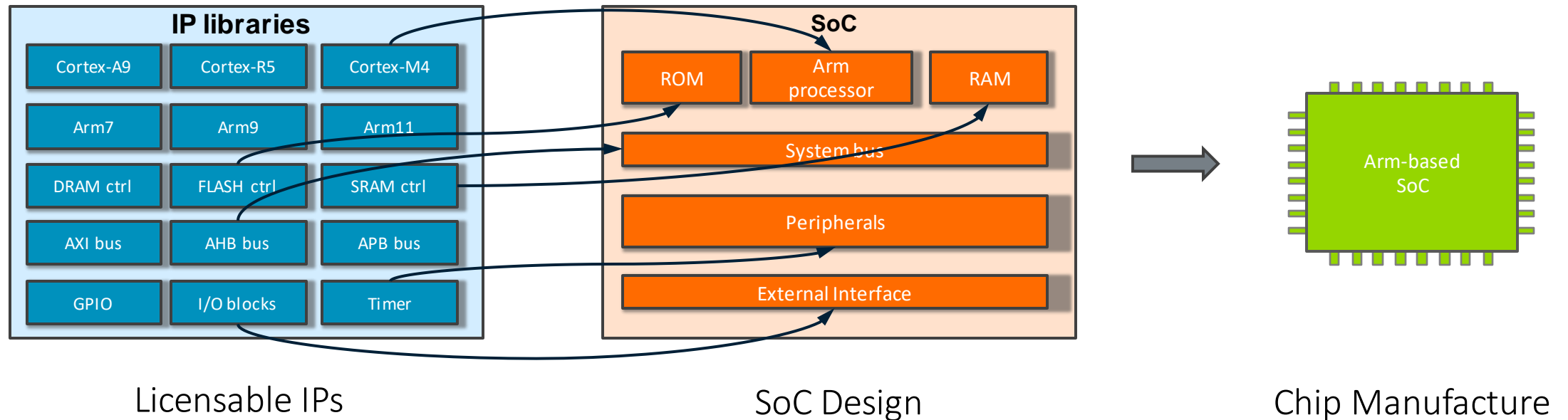
# Arm Processor Families

- Cortex-A series (Application)
  - High performance processors capable of full operating system (OS) support
  - Applications include smartphones, digital TV, smart books
- Cortex-R series (Real-time)
  - High performance and reliability for real-time applications
  - Applications include automotive braking systems, powertrains
- Cortex-M series (Microcontroller)
  - Cost sensitive solutions for deterministic microcontroller applications
  - Applications include microcontrollers, smart sensors
  - SecurCore series for high security applications
- Earlier classic processors including Arm7, Arm9, Arm11 families



# How to Design an Arm-based SoC

1. Select a set of IP cores from Arm and/or other third-party IP vendors
2. Integrate IP cores into a single-chip design
3. Give design to semiconductor foundries for chip fabrication



# Arm Cortex-M Series

- Energy-efficiency
  - Low energy cost, long battery life
- Smaller code
  - Lower silicon costs
- Ease of use
  - Faster software development and reuse
- Embedded applications
  - Smart metering, human interface devices, automotive and industrial control systems, white goods, consumer products and medical instrumentation

Cortex-M

Cortex-M0  
Cortex-M0+  
Cortex-M1  
Cortex-M3  
Cortex-M4  
Cortex-M7  
Cortex-M23  
Cortex-M33  
Cortex-m35P



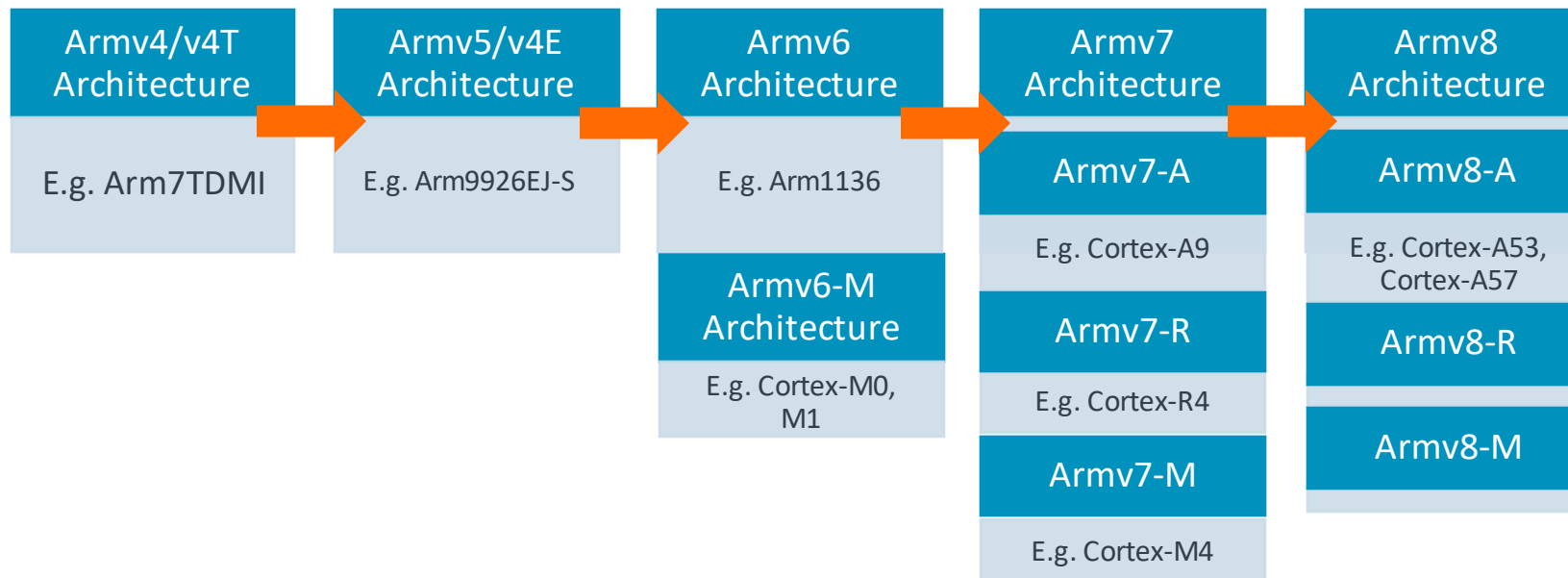
# Arm Processors vs. Arm Architectures

## Arm architecture

- Describes the details of instruction set, programmer's model, exception model, and memory map
- Documented in the 'Architecture Reference Manual'

## Arm processor

- Developed using one of the Arm architectures
- More implementation details, such as timing information
- Documented in processor's 'Technical Reference Manual'



# Arm Cortex-M Series Family

Processor	Arm Architecture	Core Architecture	Thumb	Thumb-2	Hardware Multiply	Hardware Divide	Saturated Math	DSP Extensions	Floating Point
Cortex-M0	Armv6-M	Von Neumann	Most	Subset	1 or 32 cycle	No	No	No	No
Cortex-M0+	Armv6-M	Von Neumann	Most	Subset	1 or 32 cycle	No	No	No	No
Cortex-M3	Armv7-M	Harvard	Entire	Entire	1 cycle	Yes	Yes	No	No
Cortex-M4	Armv7E-M	Harvard	Entire	Entire	1 cycle	Yes	Yes	Yes	Optional
Cortex-M7	Armv7E-M	Harvard	Entire	Entire	1 cycle	Yes	Yes	Yes	Optional



# Cortex-M4 Processor Overview

- Cortex-M4 processor
  - Introduced in 2010
  - Designed with a large variety of highly efficient signal processing features
  - Features extended single-cycle multiply accumulate instructions, optimized SIMD arithmetic, saturating arithmetic, and an optional floating-point unit
- High performance efficiency
  - 1.25 DMIPS/MHz (Dhrystone million instructions per second/MHz) at the order of  $\mu$ Watts/MHz
- Low power consumption
  - Longer battery life – especially critical in mobile products
- Enhanced determinism
  - The critical tasks and interrupt routines can be served quickly in a known number of cycles

# Cortex-M4 Processor Features

- 32-Bit reduced instruction set computing (RISC) processor
- Harvard architecture
  - Separated data bus and instruction bus
- Instruction set
  - Includes the entire Thumb-1 (16-bit) and Thumb-2 (16/32-bit) instruction sets
- 3-stage + branch speculation pipeline
- Supported interrupts
  - Non-maskable interrupt (NMI) + 1 to 240 physical interrupts
  - 8 to 256 interrupt priority levels

# Cortex-M4 Processor Features

- Supports sleep modes
  - Up to 240 wake-up interrupts
  - Integrated wait for interrupt (WFI) and wait for event (WFE) instructions and sleep on exit capability
  - Sleep and deep sleep signals
  - Optional retention mode with arm power management kit
- Enhanced instructions
  - Hardware divide (2-12 cycles)
  - Single-cycle 16, 32-bit MAC, single-cycle dual 16-bit MAC
  - 8, 16-bit SIMD arithmetic

# Cortex-M4 Processor Features


- Debug
  - Optional JTAG & serial-wire debug (SWD) ports
  - Up to eight breakpoints and four watchpoints
- Memory protection unit (MPU)
  - Optional eight-region MPU with sub regions and background regions

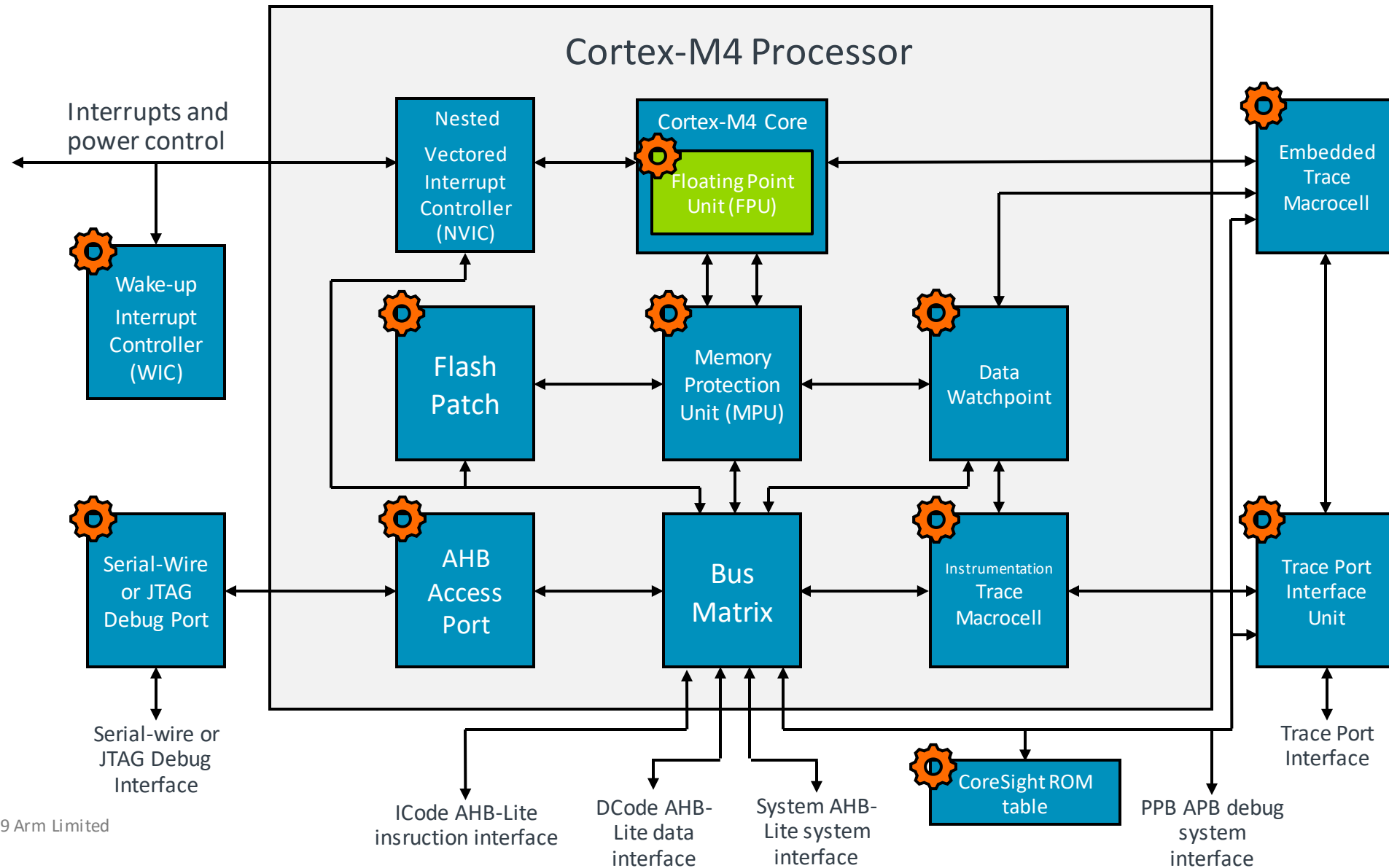
# Cortex-M4 Processor Features

- The Cortex-M4 processor is designed to meet the challenges of low dynamic power constraints while retaining a light footprint
  - 180ULL ultra low power process: 151  $\mu\text{W}/\text{MHz}$
  - 90LP low power process: 32.82  $\mu\text{W}/\text{MHz}$
  - 40LP low power process: 12.26  $\mu\text{W}/\text{MHz}$

Arm Cortex-M4 Implementation Data			
Process	180ULL (7-track, typical 1.8v, 25C)	90LP (7-track, typical 1.2v, 25C)	40G (9-track, typical 0.9v, 25C)
Dynamic power	151 $\mu\text{W}/\text{MHz}$	32.82 $\mu\text{W}/\text{MHz}$	12.26 $\mu\text{W}/\text{MHz}$
Floor planned area	0.44 $\text{mm}^2$	0.119 $\text{mm}^2$	0.028 $\text{mm}^2$

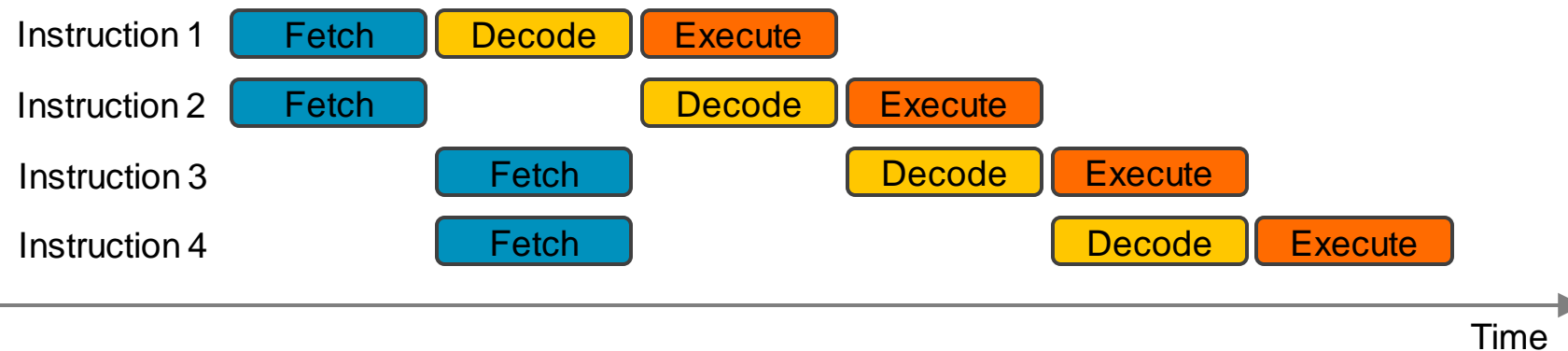
# Cortex-M4 Block Diagram

 Optional component



# Cortex-M4 Block Diagram

- Processor core
  - Contains internal registers, the ALU, data path, and some control logic
  - Registers include 16x 32-bit registers for both general and special use
- Processor pipeline stages
  - Three-stage pipeline: fetch, decode, and execution
  - Some instructions may take multiple cycles to execute, in which case the pipeline will be stalled
  - Speculatively prefetches instructions from branch target addresses
  - Up to two instructions can be fetched in one transfer (16-bit instructions)



# Cortex-M4 Block Diagram

- Nested vectored interrupt controller (NVIC)
  - Up to 240 interrupt request signals and an NMI
  - Automatically handles nested interrupts, such as comparing priorities between interrupt requests and the current priority level
- Wake-up interrupt controller (WIC)
  - For low-power applications, the microcontroller can enter sleep mode by shutting down most of the components
  - When an interrupt request is detected, the WIC can inform the power management unit to power up the system
- Memory protection unit (MPU)
  - Used to protect memory content, e.g., make some memory regions read-only or preventing user applications from accessing privileged application data



# Cortex-M4 Block Diagram

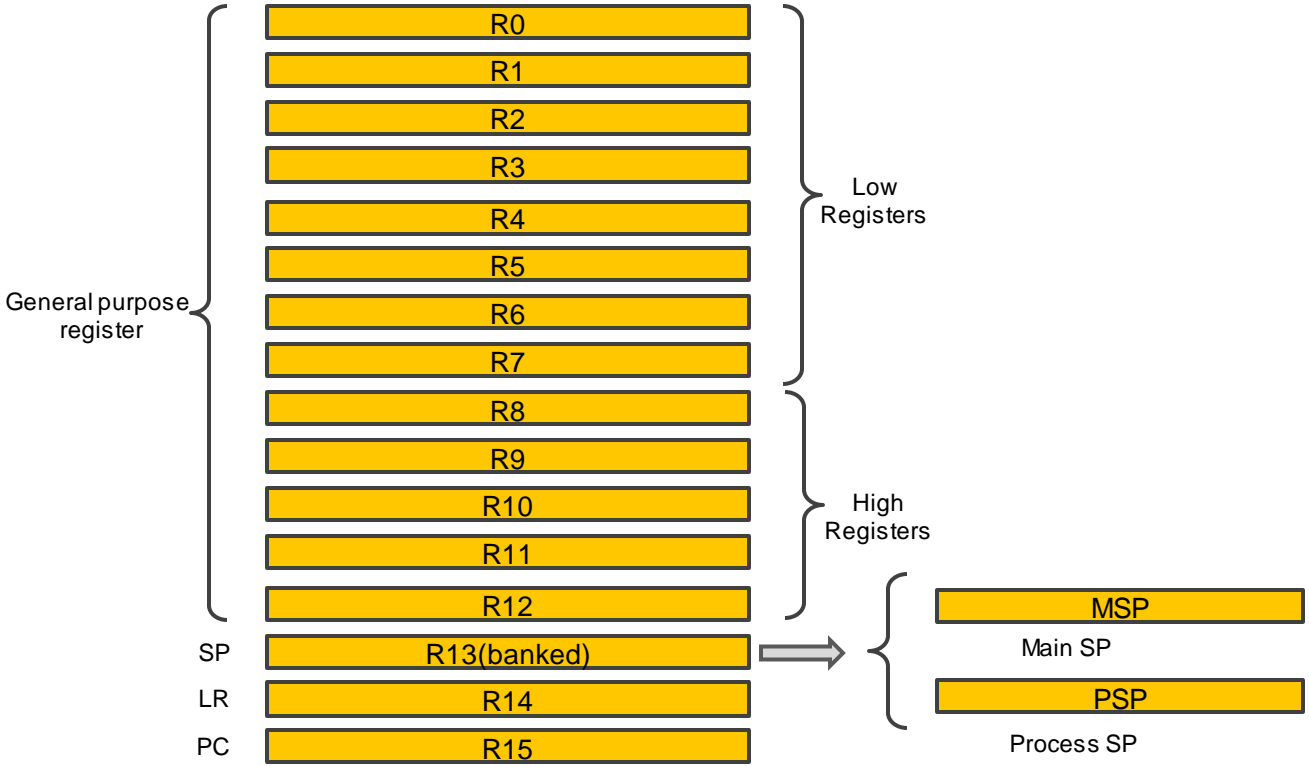
- Bus interconnect
  - Allows data transfer to take place on different buses simultaneously
  - Provides data transfer management, e.g. write buffer, bit-oriented operations (bit-band)
  - May include bus bridges (e.g. AHB-to-APB bus bridge) to connect different buses into a network using a single global memory space
  - Includes the internal bus system, the data path in the processor core, and the AHB LITE interface unit
- Debug subsystem
  - Handles debug control, program breakpoints, and data watchpoints
  - When a debug event occurs, it can put the processor core in a halted state, so developers can analyse the status of the processor, such as register values and flags, at that point

# Arm Cortex-M4 Processor Registers

- Processor registers
  - The internal registers are used to store and process temporary data within the processor core
  - All registers are inside the processor core, so they can be accessed quickly
  - Load-store architecture
    - To process memory data, they have to first be loaded from memory to registers, processed inside the processor core using register data only, and then written back to memory if needed
- Cortex-M4 registers
  - Register bank
    - 16x 32-bit registers (thirteen are used for general-purpose)
  - Special registers

# Cortex-M4 Registers

Register bank

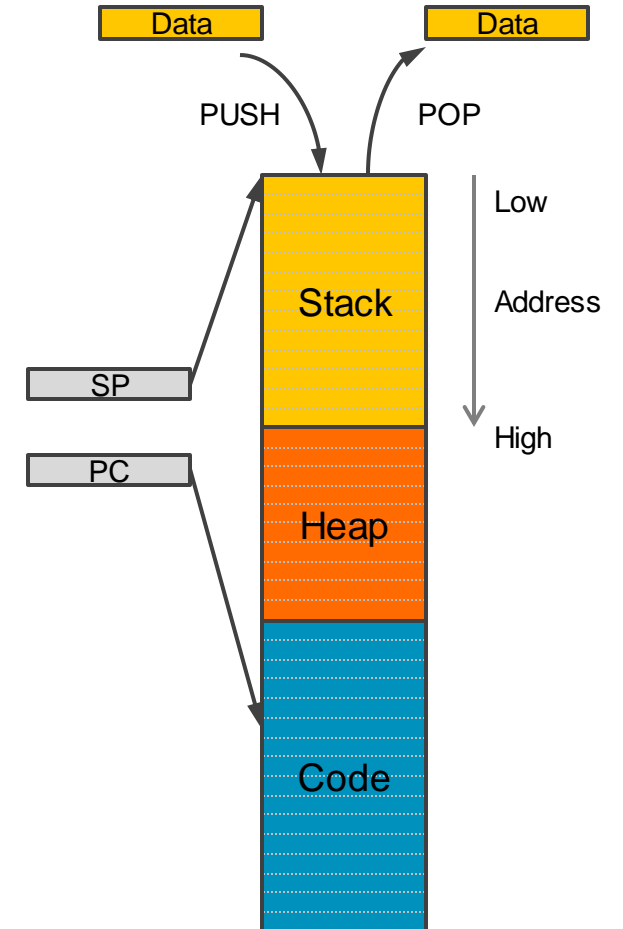


Special registers



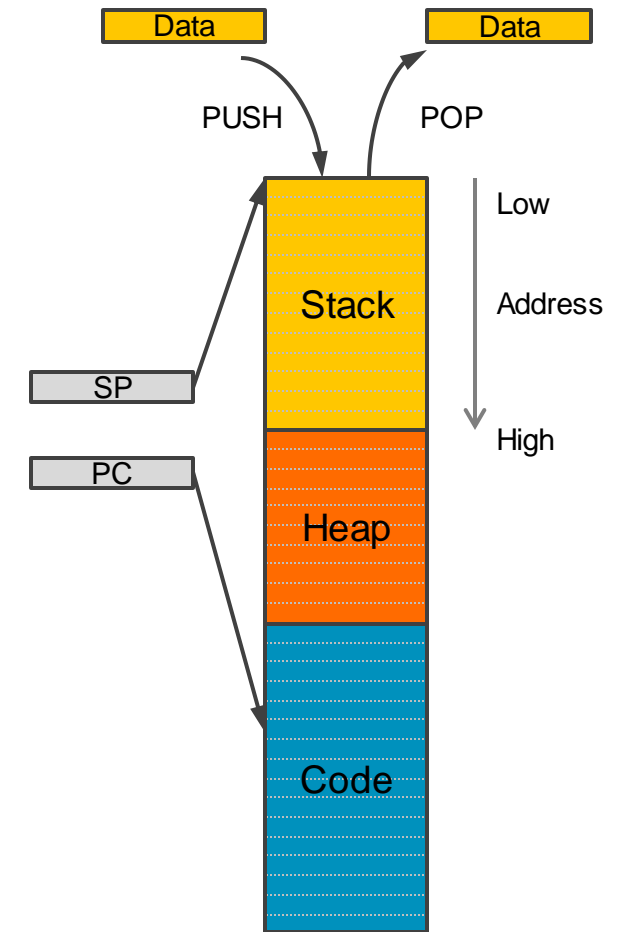
# Cortex-M4 Registers

- R0-R12: general purpose registers
  - Low registers (R0-R7) can be accessed by any instruction
  - High registers (R8-R12) sometimes cannot be accessed e.g. by some Thumb (16-bit) instructions
- R13: Stack Pointer (SP)
  - Records the current address of the stack
  - Used for saving the context of a program while switching between tasks
  - Cortex-M4 has two SPs: Main SP, used in applications that require privileged access e.g. OS kernel and process SP, used in base-level application code (when not running an exception handler)



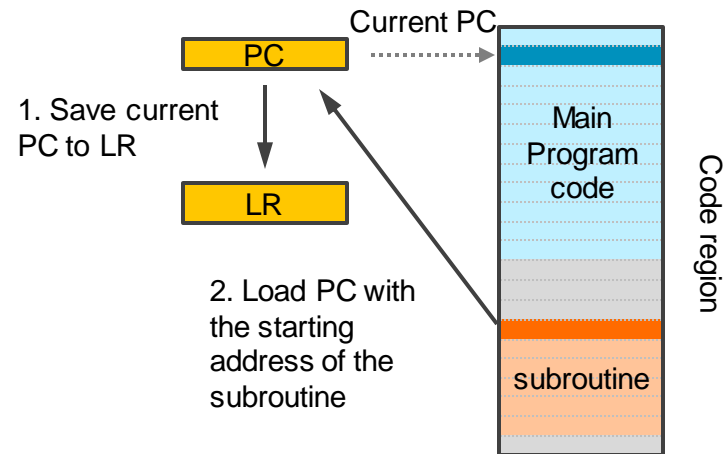
# Cortex-M4 Registers

- Program Counter (PC)
  - Records the address of the current instruction code
  - Automatically incremented by four at each operation (for 32-bit instruction code), except branching operations
  - A branching operation, such as function calls, will change the PC to a specific address, while saving the current PC to the LR

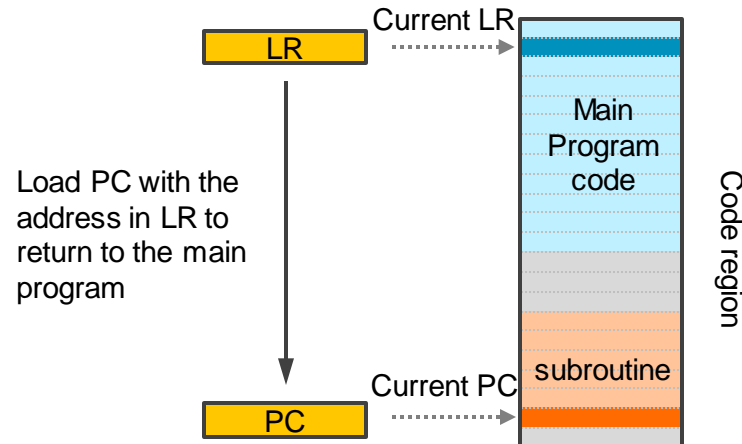


# Cortex-M4 Registers

- R14: Link Register (LR)
  - The LR is used to store the return address of a subroutine or a function call
  - The PC will load the value from the LR after a function is finished



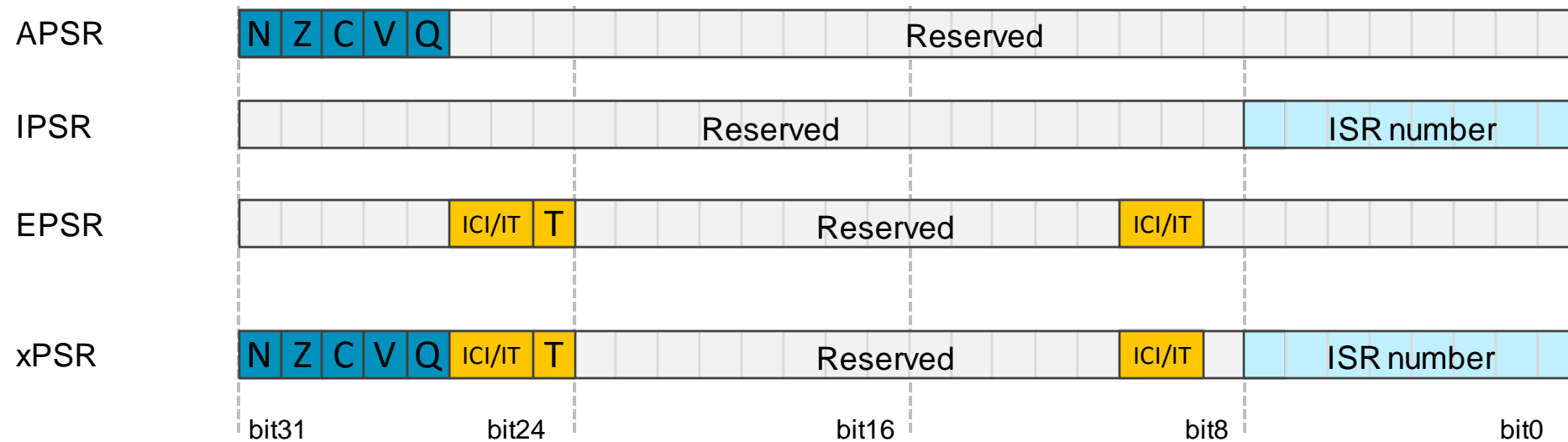
Call a subroutine



Return from subroutine to main program

# Cortex-M4 Registers

- xPSR, combined program status register (PSR)
  - Provides information about program execution and ALU flags
  - Application PSR (APSR)
  - Interrupt PSR (IPSR)
  - Execution PSR (EPSR)



# Cortex-M4 Registers

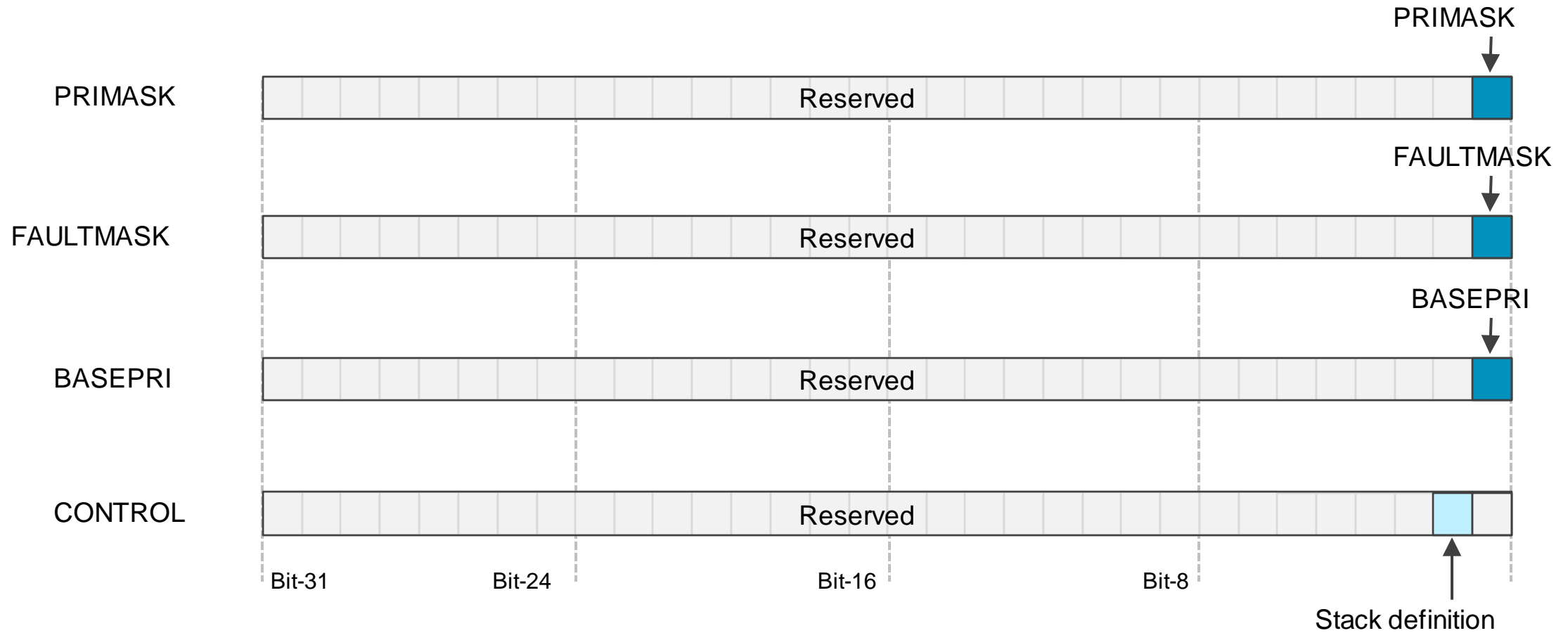
- APSR
  - N: negative flag: set to one if the result from the ALU is negative
  - Z: zero flag: set to one if the result from the ALU is zero
  - C: carry flag: set to one if an unsigned overflow occurs
  - V: overflow flag: set to one if a signed overflow occurs
  - Q: stick saturation flag: set to one if saturation has occurred in saturating arithmetic instructions, or overflow has occurred in certain multiply instructions
- IPSR
  - Interrupt service routine (ISR) number: current executing ISR number
- EPSR
  - T: Thumb state: always one since Cortex-M4 only supports the Thumb state
  - IC/IT: Interrupt-continuable instruction (ICI) bit, IF-THEN instruction status bit



# Cortex-M4 Registers

- Exception mask registers
  - 1-bit PRIMASK
    - If set to one, will block all interrupts apart from NMI and the hard fault exception
  - 1-bit FAULTMASK
    - If set to one, will block all the interrupts apart from NMI
  - 1-bit BASEPRI
    - If set to one, will block all interrupts of the same or lower level (only allowing for interrupts with higher priorities)
- CONTROL: special register
  - 1-bit stack definition
    - Set to one to use the PSP
    - Clear to zero to use the MSP

# Cortex-M4 Registers



# Resources

- Cortex-M4 Technical Reference Manual: <https://developer.arm.com/docs/100166/0001>
- Cortex-M4 Devices Generic User Guide: <https://developer.arm.com/docs/dui0553/b>