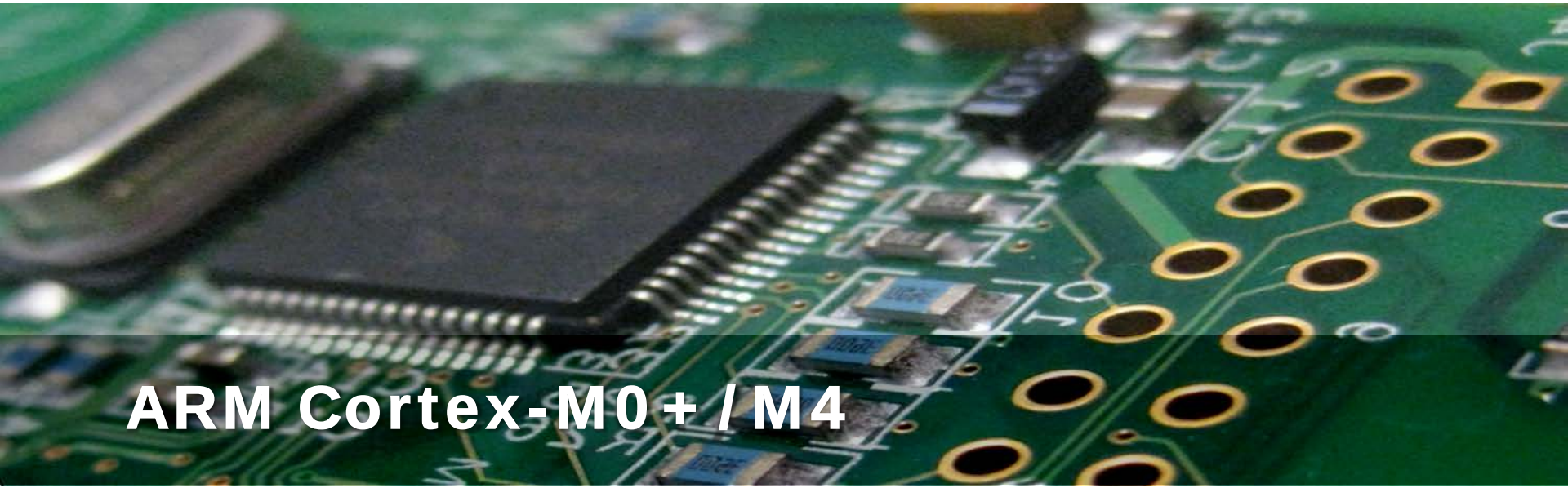
Lucerne University of
Applied Sciences and Arts
HOCHSCHULE
LUZERN
Technik & Architektur



# ARM Cortex-M0+ / M4

*“There is an ARM for that…"*

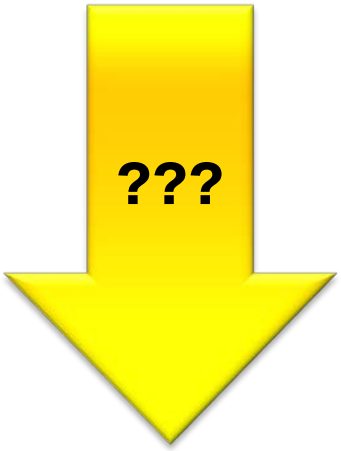
## Scriptum: Cortex-M Interrupts

**Prof. Erich Styger**
*erich.styger@hslu.ch*
*+41 41 349 33 01*

# Learning Goals

- Problem: A new core to learn about

- Goal
  - ARM Cortex cores (A, R, M)
  - Instruction Set
  - Memory Map
  - Interrupts
  - Registers



# ARM Cortex Processors

- ARM Inc.,
  - Cambridge, UK
  - IP License
  - STM, NXP, TI, Atmel, EM, ...

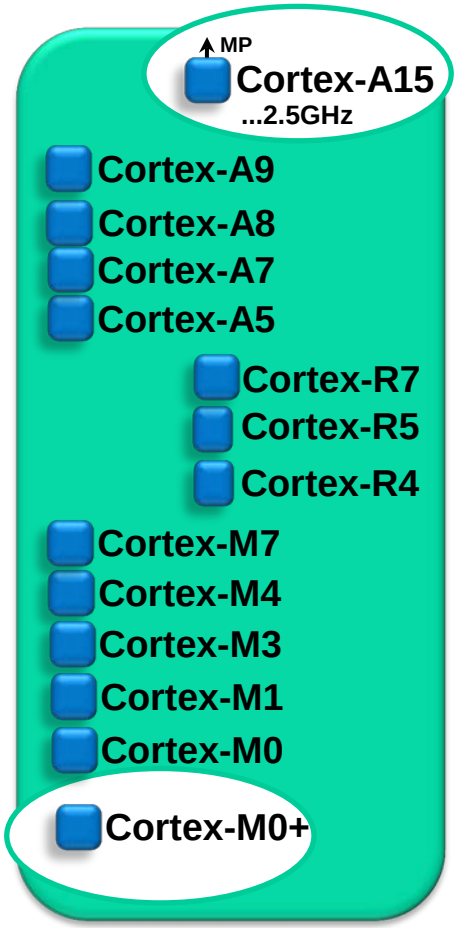
- 32bit, RISC

- ARM Cortex™-**A** family:
  - Applications processors

used in smartphones

- ARM Cortex-**R** family:
  - Embedded processors for real-time signal processing, control applications

- ***ARM Cortex-M family:***
  - ***Microcontroller-oriented processors for MCU and SoC applications***




*Source: ARM Inc.*

# Cortex-M Processor Family

## Six Cores, two Architectures

ARMv8_M zurzeit noch nicht auf dem Markt

Lucerne University of Applied Sciences and Arts
HOCHSCHULE LUZERN
Technik & Architektur

| | | |
|---|---|---|
| Cortex-M7 | High performance, hardware multiply/divide, enhanced DSP, floating point unit, special tightly coupled memory. | ARMv7-M architecture |
| Cortex-M4(F) | High performance data processing & I/O control. Support hardware divide, MAC, bit field processing, DSP. Floating point unit optional (Cortex-M4F). | |
| Cortex-M3 | High performance data processing & I/O control. Support hardware divide, MAC (Multiply Accumulate), bit field processing. | |
| Cortex-M0+ | General data processing, high performance I/O control, mixed signal ASICs, replacement for 8/16-bit MCUs | ARMv6-M architecture |
| Cortex-M0 | General data processing, I/O control, mixed signal ASICs, replacement for 8/16-bit MCUs | |
| Cortex-M1 | For FPGA designs only. Optimized for FPGA and can work in most FPGA devices | |

*Source: ARM Inc.*

Price: M0: 40cents, M3-7: 3-4dollar

# Cortex-M Comparison

Heute: meistens M0+

bei vielen Multiplikationen wird empfohlen: Hardware multiply 1 cycle -> faster!

Lucerne University of Applied Sciences and Arts
HOCHSCHULE LUZERN
Technik & Architektur

| ARM Cortex-M | | | | | |
|---|---|---|---|---|---|
| | M0 | M0+ | M3 | M4 | M7 |
| Thumb | Most | Most | Entire | Entire | Entire |
| Thumb-2 | Subset | Subset | **Entire** | **Entire** | **Entire** |
| Hardware multiply | 1 or 32 cycles | 1 or 32 cycles | **1 cycle** | **1 cycle** | **1 cycle** |
| Hardware divide | No | In Software! No | **Yes** | **Yes** | **Yes** |
| Saturated math | No | No | **Yes** | **Yes** | **Yes** |
| DSP Extensions | No | No | No | **Yes** | **Yes, enhanced** |
| Floating-point | No | No | No | **Optional single precision** | **Yes** |
| Tightly coupled memory | No | No | No | No | yes |
| Architecture | ARMv6-M | ARMv6-M | ARMv7-M | ARMv7-M | ARMv7-M |
| CoreMark/MHz | 2.33 | 2.42 | 3.32 | 3.40 | 5.04 |

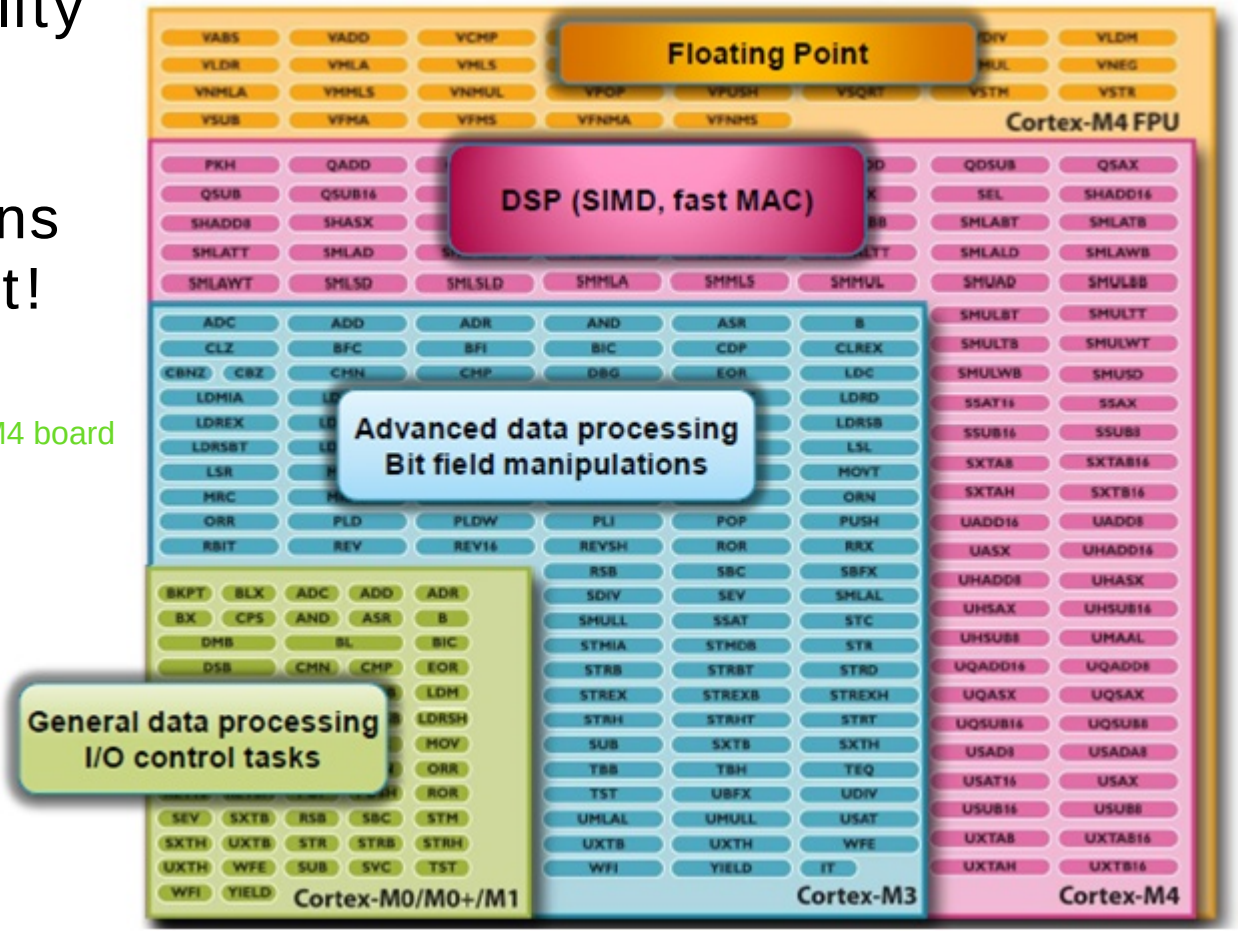
*Source: http://www.anandtech.com*

30% of the silicon is for the floating point (also 30% of the price)

# Instruction Set

- Upward compatibility
- THUMB(2):
  - Code Density
  - 16bit Instructions
  - **Odd** Address bit!

you can download M0 with a Cortex-M4 board
-> it's compatibility



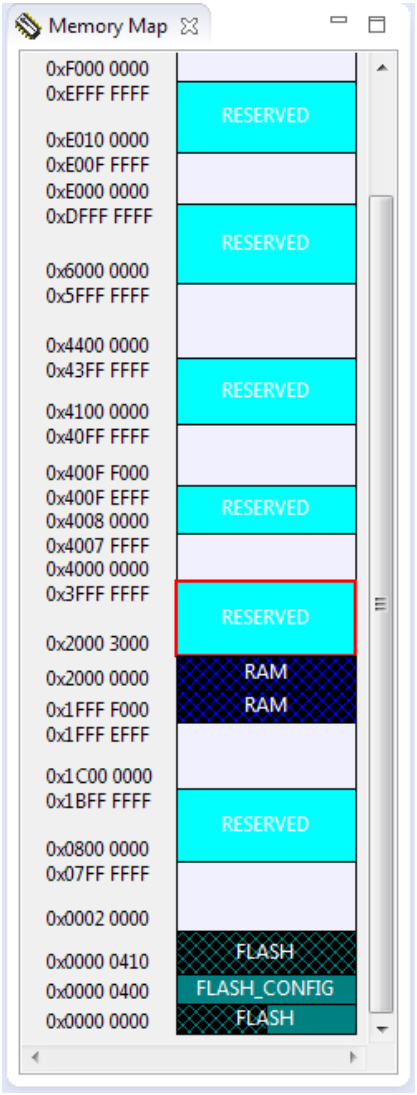

*Source: ARM Inc.*

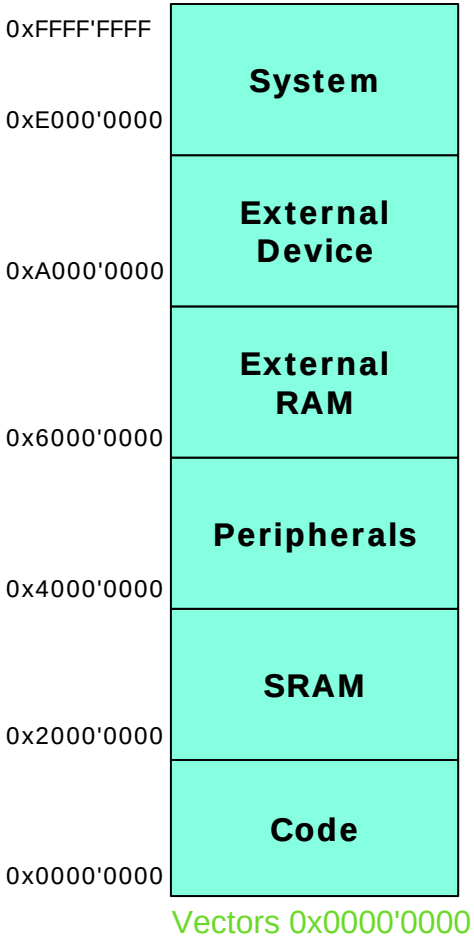
# Memory Map

- System
  - NVIC, Debug, MPU,...
- SRAM/Code
  - Mapping vendor specific

- See: Processor Expert
  Memory Map view

the mapping is standard -> makes it easier

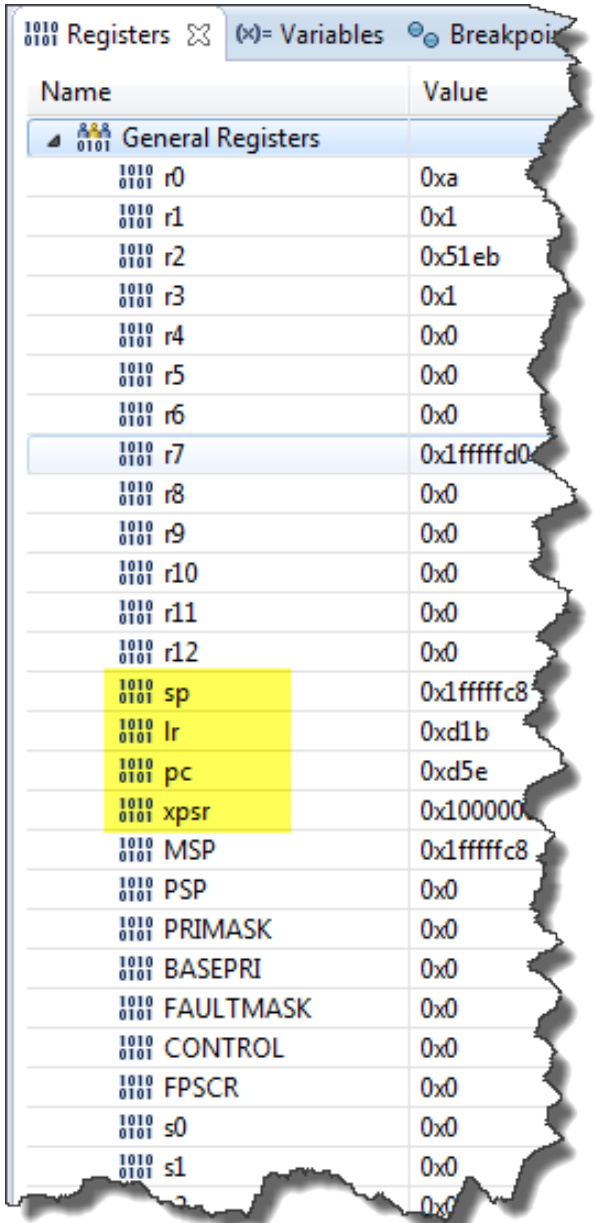


# CPU Registers

- R0-R12: GPR (32bit)
- R13
  - Main Stack Pointer (MSP) (priv. Access, default)
  - Process Stack Pointer (PSP)
- R14 Link Register (LR)
  - Odd bit: THUMB
- R15 Program Counter (PC)
- CPSR (Current Program Status Register) condition code, carryflag etc.

BC -> LR (linke register)



# ARM Cortex Exception Types

1-3: can't change priority (lower the number, the higher the priority!)

- 1-15: Core System Exception Types
- >15: 'External' interrupts
- The lower the number, the higher the urgency!

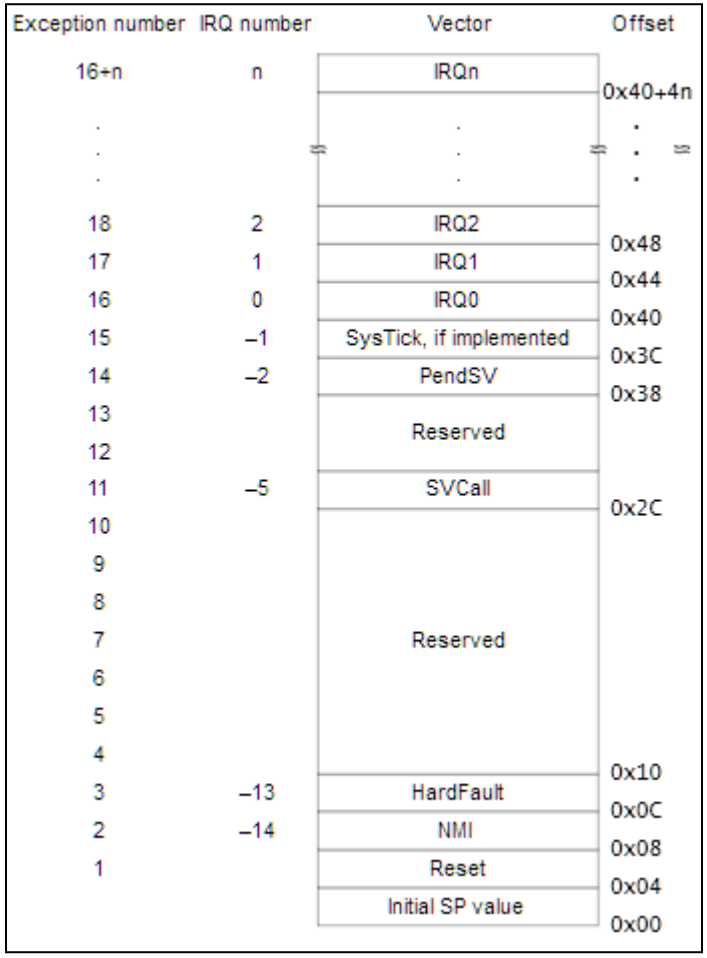
| Exception | Vector | Core | Priority |
|---|---|---|---|
| 1 | Reset | M0/M4/M3/M7 | -3 |
| 2 | NMI | M0/M4/M3/M7 | -2 |
| 3 | HardFault | M0/M3/M4/M7 | -1 |
| 4 | MemManageFault | M3/M4/M7 | User |
| 5 | BusFault | M3/M4/M7 | User |
| 6 | UsageFault | M3/M4/M7 | User |
| 7-A | Reserved | | |
| B | SVCall | M0/M3/M4/M7 | User |
| C | Debug Monitor | M3/M4/M7 | User |
| D | Reserved | | |
| E | PendSV | M0/MM3/M4/M7 | User |
| F | SysTick | M0(optional)/M3/M4/M7 | User |
| 10-... | IRQ0, IRQ1, ... (Vendor) | M0: 0x10-0x47<br>M4/M7: IRQ0-239 | User |

NMI Non maskable interrupt

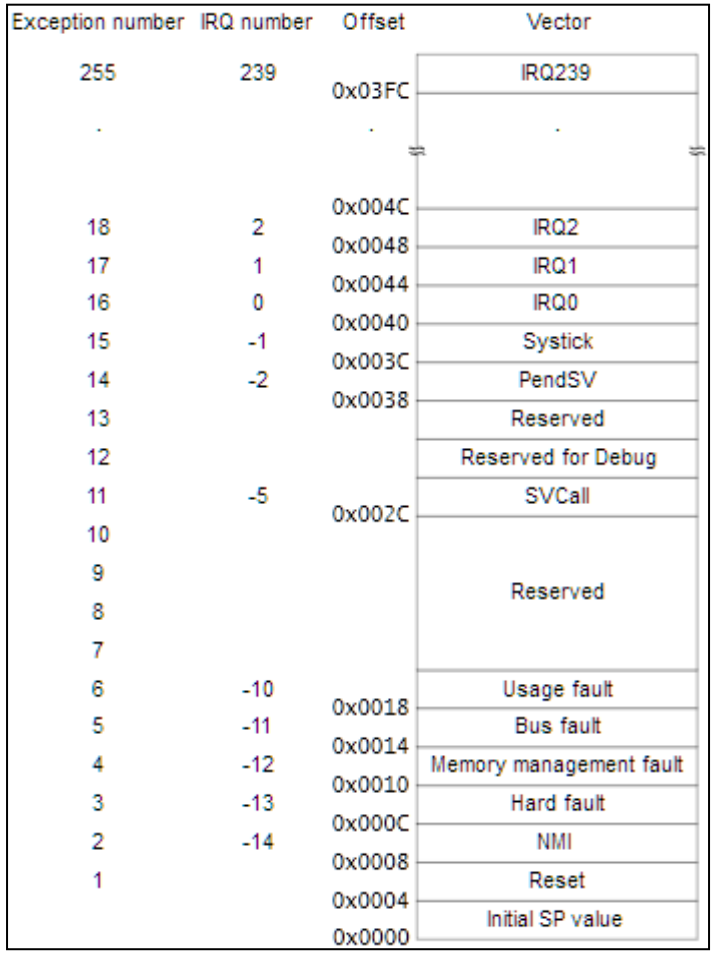
Hardware exception! fault, crash!

supervisor call

# Interrupts and Vector Table




M0/M0+




M4/M7

*Source: ARM Info Center*

# PRIMASK

interrupts are masked and disabled

- 1-Bit Register (Cortex M0 and M4)
- Masks interrupts
- NMI and Hard Faults are not maskable (negative prio!)
- M0: PRIMASK only
- M4: PRIMASK and BASEPRI (priority masking)
  - BASEPRI: **masks** interrupts ≥ BASEPRI value

PRIMASK I = 1 -> all interrupts are disabled

```
__asm("cpsid i"); /* disable */
```

| | |
|---|---|
| primask | 0x1 |

```
__asm("cpsie i"); /* enable */
```

| | |
|---|---|
| primask | 0x0 |

BASEPRI I = 4 --> interrupts 4 and higher are disabled
it's possible to combine PRIMASK and BASEPRI

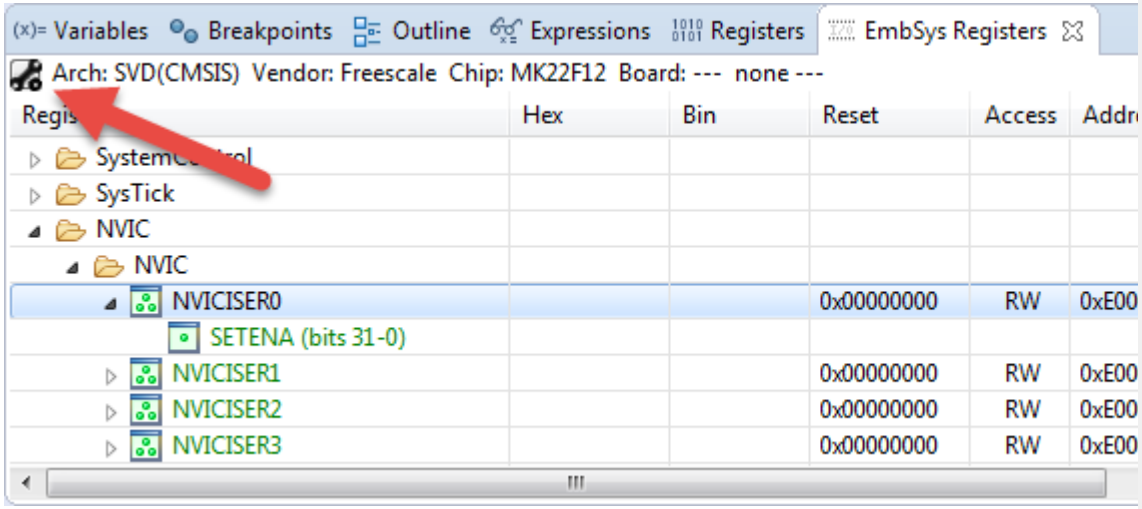
Registers | (x)= Variables | Breakpoints

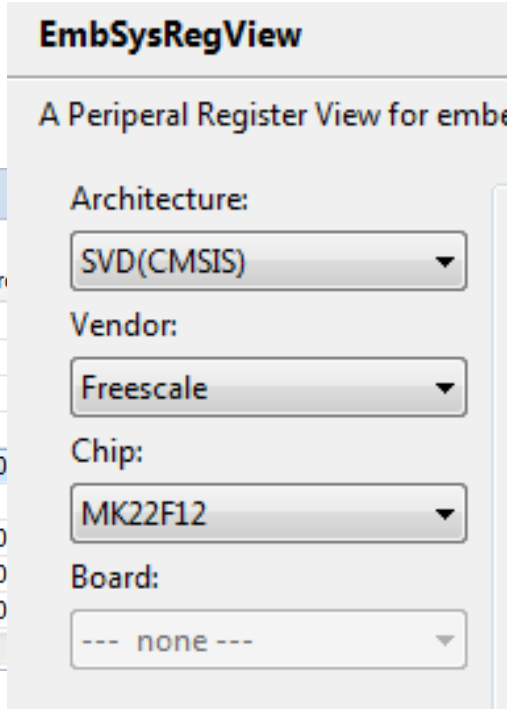
| Name | Value |
|---|---|
| MSP | 0x1ffffff8 |
| PSP | 0x0 |
| PRIMASK | 0x0 |
| BASEPRI | 0x0 |
| FAULTMASK | 0x0 |
| CONTROL | 0x0 |
| FPSCR | 0x0 |
| s0 | 0x0 |

enable

# EmbSys Registers View

- Menu *Window > Show View > Other > Debug*
- Wrench icon to load CMSIS SVD files for CPU
  - Change manually for MKL25Z4 and MK22F12
- Double click (green) to edit



# Cortex-M Interrupt Priorities

- 8bit Priority Register (2 bits min for M0, 3 bits min for M3/M4)
  - #bits vendors specific (easier to port software)
  - Up to 128 pre-emption levels (main prio)
- Example: 3 bits: 8 priorities
  - 0x00, 0x20, 0x40, 0x60, 0x80, 0xA0, 0xC0, 0xE0

| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|
| Implemented | | | Not implemented (read 0) | | | | |

- M3, M4: Pre-empt prio + sub-priorities possible
  - Multiple interrupts with same priority (lower subprio value handled first)
  - Otherwise: lower vector number (higher prio!) wins

| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|
| Preempt Prio | | Sub Prio | Not implemented (read 0) | | | | |

Subpriority: 3.1, 3.2 -> higher prio means lower number

3.1, 3.1 -> the lower vector number wins! the one with the lower IRQ numer wins!

# NVIC: Cortex-M0 (NXP Kinetis KL)

minimum 2 bits are implemented:

- 2 bits, 4 Prios: 0, 0x40, 0x80, 0xC0 (or (0<<6), (1<<6)

| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|
| Implemented | | Not implemented (read 0) | | | | | |

- 32 user interrupts (0x10-0x47)
- 1 bit each (1 32bit register)
  - NVIC_ISER (Interrupt Set Enable Register)
  - NVIC_ICER (Interrupt Clear Enable Register)
  - NVIC_ISPR (Interrupt Set Pending Register)
  - NVIC_ICPR (Interrupt Clear Pending Register)
- 8 bits each (8 32bit register)
  - NVIC_IPRx (Interrupt Priority Registers)

# Cortex-M0 NVIC

| NVIC | | | Nested Vectored Interrupt Controller |
|---|---|---|---|
| NVIC_ISER | RW | 0xE000E100 | Interrupt Set Enable Register |
| NVIC_ICER | RW | 0xE000E180 | Interrupt Clear Enable Register |
| NVIC_ISPR | RW | 0xE000E200 | Interrupt Set Pending Register |
| NVIC_ICPR | RW | 0xE000E280 | Interrupt Clear Pending Register |
| NVIC_IPR0 | RW | 0xE000E400 | Interrupt Priority Register 0 |
| NVIC_IPR1 | RW | 0xE000E404 | Interrupt Priority Register 1 |
| NVIC_IPR2 | RW | 0xE000E408 | Interrupt Priority Register 2 |
| NVIC_IPR3 | RW | 0xE000E40C | Interrupt Priority Register 3 |
| NVIC_IPR4 | RW | 0xE000E410 | Interrupt Priority Register 4 |
| NVIC_IPR5 | RW | 0xE000E414 | Interrupt Priority Register 5 |
| NVIC_IPR6 | RW | 0xE000E418 | Interrupt Priority Register 6 |
| NVIC_IPR7 | RW | 0xE000E41C | Interrupt Priority Register 7 |

| NVIC_ISER | | 0x... | RW | 0xE000E100 | Interrupt Set Enable Register |
|---|---|---|---|---|---|
| SETENA0 (bit 0) | | | | | DMA channel 0 transfer complete interrupt set-enabl... |
| SETENA1 (bit 1) | | | | | DMA channel 1 transfer complete interrupt set-enabl... |
| SETENA2 (bit 2) | | | | | DMA channel 2 transfer complete interrupt set-enabl... |
| SETENA3 (bit 3) | | | | | DMA channel 3 transfer complete interrupt set-enabl... |
| SETENA4 (bit 4) | | | | | Reserved iv 20 interrupt set-enable bit |

# NVIC: Cortex-M4 (NXP Kinetis K22F)

- 4 bits, 16 Prios: 0, 0x10, 0x20, 0x30, .. 0xF0

| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|
| Implemented | | | | Not implemented (read 0) | | | |

- 82 user interrupts (0x10-0x61)
- 1 bit each (4 32bit register)
  - NVIC_ISER (Interrupt Set Enable Register)
  - NVIC_ICER (Interrupt Clear Enable Register)
  - NVIC_ISPR (Interrupt Set Pending Register)
  - NVIC_ICPR (Interrupt Clear Pending Register)
  - **NVIC_IABR** (Interrupt Active Bit Register)
- 8 bits each (106 32bit register)
  - NVIC_IPRx (Interrupt Priority Registers)

# Cortex-M4: PRIGROUP

configure how man subPrio

- 0...7: Position of SubPrio

| Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---|---|---|---|---|---|---|---|
| Preempt Prio | | | Sub Prio | Not implemented (read 0) | | | |

**PRIGROUP: 4**

| | | | | | |
|---|---|---|---|---|---|
| SystemControl | | | | | System Control Registers |
| SystemControl | | | | | System Control Registers |
| ACTLR | | 0x... | RW | 0xE000E008 | Auxiliary Control Register, |
| CPUID | | 0x... | RO | 0xE000ED00 | CPUID Base Register |
| ICSR | | 0x... | RW | 0xE000ED04 | Interrupt Control and State Register |
| VTOR | | 0x... | RW | 0xE000ED08 | Vector Table Offset Register |
| AIRCR | 11111010000001010000... | 0x... | RW | 0xE000ED0C | Application Interrupt and Reset Control Register |
| VECTRESET (bit 0) | 0 | | | | no description available |
| VECTCLRACTIVE (bit 1) | 0 | | | | no description available |
| SYSRESETREQ (bit 2) | 0 | | | | 0: no system reset request |
| PRIGROUP (bits 10-8) | 000 | | | | Interrupt priority grouping field. This field determines... |
| ENDIANNESS (bit 15) | 0 | | | | 0: Little-endian |
| VECTKEY (bits 31-16) | 1111101000000101 | | | | Register key |

# PEx: ARM Core Interrupt Settings

- Settings in CPU component properties
- Vector table in vectors.c (Generated_Code)
- 0x00: initial SP
- 0x04: initial PC

```
__attribute__ ((section (".vectortable"))) const tVectorTable __vect_table = { /* Interrup

  /* ISR name                              No. Address      Pri Name
  &__SP_INIT,                          /* 0x00  0x00000000   -   ivINT_Initial_Stack_Pointe
  {
  (tIsrFunc)&__thumb_startup,          /* 0x01  0x00000004   -   ivINT_Initial_Program_Count
  (tIsrFunc)&Cpu_INT_NMIInterrupt,     /* 0x02  0x00000008   -2   ivINT_NMI
  (tIsrFunc)&Cpu_INT_Hard_FaultInterrupt, /* 0x03  0x0000000C   -1   ivINT_Hard_Fault
  (tIsrFunc)&Cpu_Interrupt,            /* 0x04  0x00000010   -   ivINT_Reserved4
  (tIsrFunc)&Cpu_Interrupt,            /* 0x05  0x00000014   -   ivINT_Reserved5
  (tIsrFunc)&Cpu_Interrupt,            /* 0x06  0x00000018   -   ivINT_Reserved6
  (tIsrFunc)&Cpu_Interrupt,            /* 0x07  0x0000001C   -   ivINT_Reserved7
  (tIsrFunc)&Cpu_Interrupt,            /* 0x08  0x00000020   -   ivINT_Reserved8
  (tIsrFunc)&Cpu_Interrupt,            /* 0x09  0x00000024   -   ivINT_Reserved9
  (tIsrFunc)&Cpu_Interrupt,            /* 0x0A  0x00000028   -   ivINT_Reserved10
  (tIsrFunc)&Cpu_INT_SVCallInterrupt, /* 0x0B  0x0000002C   0   ivINT_SVCall
  (tIsrFunc)&Cpu_Interrupt,            /* 0x0C  0x00000030   -   ivINT_Reserved12
  (tIsrFunc)&Cpu_Interrupt,            /* 0x0D  0x00000034   -   ivINT_Reserved13
  (tIsrFunc)&Cpu_INT_PendableSrvReqInterrupt, /* 0x0E  0x00000038   -2   ivINT_PendableSr
```

Properties | Methods | Events | Build options | Resources

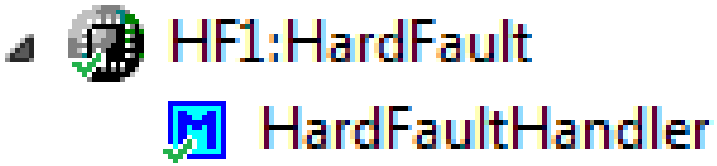
| Name | Value |
|---|---|
| Component name | Cpu |
| CPU type | MKL25Z128VLK4 |
| ▷ **Clock settings** | |
| Initialization priority | interrupts enabled |
| Watchdog disable | yes |
| ▷ **Internal peripherals** | |
| ◢ **CPU interrupts/resets** | |
| ◢ **NMI interrupt** | Enabled |
| Interrupt | INT_NMI |
| ◢ **Hard Fault** | Enabled |
| Interrupt | INT_Hard_Fault |
| ▷ **Bus Fault** | Disabled |
| ▷ **Usage Fault** | Disabled |
| ◢ **Supervisor Call** | Enabled |
| Interrupt | INT_SVCall |
| Priority | maximal priority |
| ◢ **Pendable Service** | Enabled |
| Interrupt | INT_PendableSrvReq |
| Priority | maximal priority |
| ▷ **MCG Loss of lock** | Disabled |
| ▷ **Low power mode settings** | |
| ◢ **Clock configurations** | 1 |
| ◢ **Clock configuration 0** | |
| ▷ **Very low power mode** | Disabled |
| ◢ **Clock source setting** | configuration 0 |

# Fault Exception

- Bus Fault
  - Instruction or data error, stacking error
- Memory Fault
  - Write to Read-Only, outside of memory map
- Usage Fault
  - Illegal instruction, invalid ISR return, unaligned memory access
- Hard Faults
  - Can be called by above fault conditions
  - Debugging: see
  - http://mcuoneclipse.com/2012/11/24/debugging-hard-faults-on-arm-cortex-m/
  - http://mcuoneclipse.com/2012/12/28/a-processor-expert-component-to-help-with-hard-faults/

# Hard Fault Handler

- Hard Fault stack frame different from normal ISR frame
- Interrupt handler (or debugger) to read special registers and location to find out location of problem
  - Link register
  - PC register
- HardFault Processor Expert component

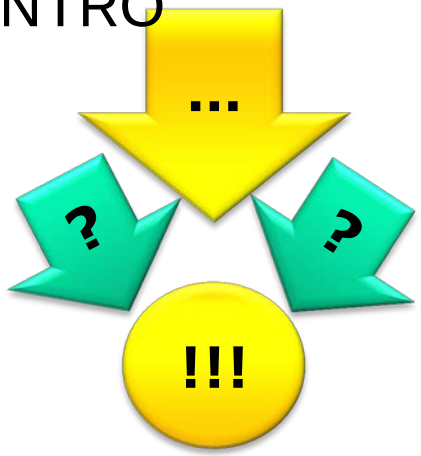


# Summary

- ARM Inc. licenses the configurable core
- Vendors implement the device/peripherals
- Instruction Set
- Memory Map

- 16 predefined interrupts
- Up to 8bit interrupt priority bits
  - The lower the number, the higher the priority
  - No sub-prio on NXP M0+/M4(F) we use in INTRO
- Hard Faults

- 32bit GPR Registers
- SP, PC, LR
- PRIMASK (interrupt)



# Lab: Hard Faults

- Use HardFault handler component
- Create a hard fault
- Locate hard fault with debugger