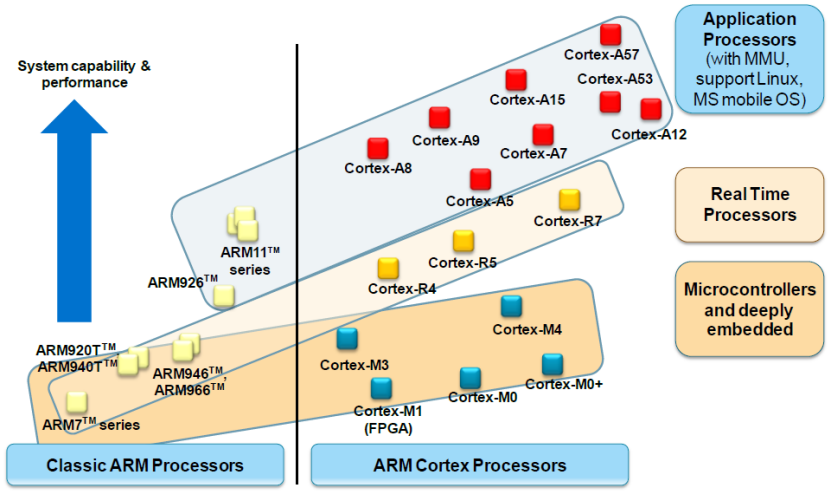
# NuMicro
# Overview of Cortex-M

NuMicro@nuvoton.com




# ARM architecture road map

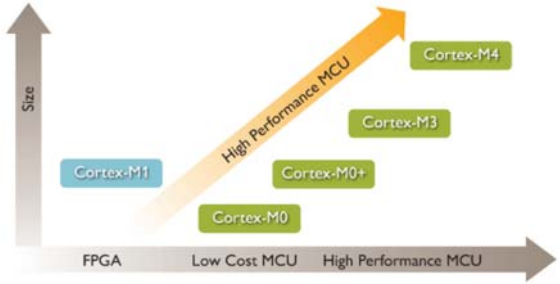



*Reference from ARM*



# Cortex-M Processor Family (1/3)

| Cortex-M0 | Cortex-M0+ | Cortex-M3 | Cortex-M4 |
|---|---|---|---|
| Low cost, ultra low power | Best energy-efficiency (single cycle I/O interface + vector table relocations) | Powerful embedded processor (hardware divider and Multiply-Accumulate) | All the features on the Cortex-M3 + DSP instructions + Floating Point Unit |
| deeply embedded applications | deeply embedded applications | Performance, general purpose | Performance, general purpose DSP task |




***All binary upwards compatible***

*Reference from ARM*

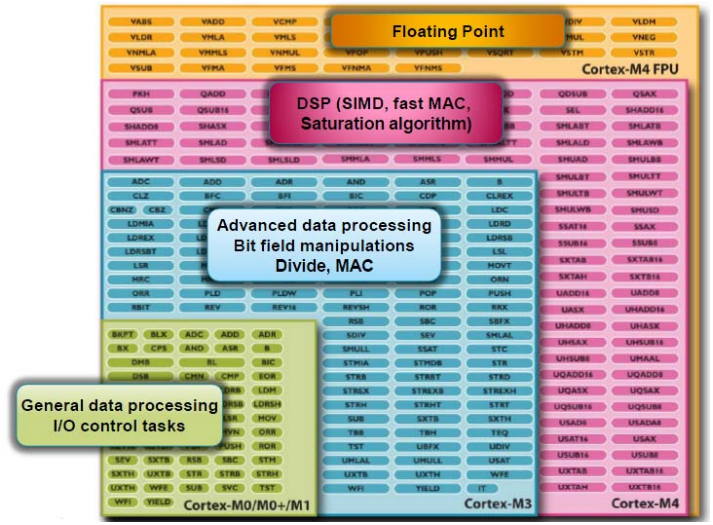


# Cortex-M Processor Family (2/3)

| | M0 | M0+ | M3 | M4 |
|---|---|---|---|---|
| **DMIPS/MHz (v2.1) - official)** | 0.84 | 0.93 | 1.25 | 1.25 |
| **CoreMark/MHz (v1.0)** | 2.33 | 2.42 | 3.32 | 3.40 |
| **Number Interrupts** | 1 ~ 32 + NMI | 1 ~ 32 + NMI | 1 ~ 240 + NMI | 1 ~ 240 + NMI |
| **Interrupt Priorities** | 4 | 4 | 8 ~ 256 | 8 ~ 256 |
| **Interrupt Latency** | 16 cycles | 15 cycles | 12 cycles | 12 cycles |
| **Hardware Multiply** | 1 or 32 cycles | 1 or 32 cycles | 1 cycle | 1 cycle |
| **Hardware Divide** | No | No | 2 – 12 cycles | 2 – 12 cycles |
| **DSP Extension** | No | No | No | Yes |
| **FPU** | No | No | No | Optional Single precision |
| **MPU** | No | Optional 8 region MPU | Optional 8 region MPU | Optional 8 region MPU |
| **Trace option** | No | Optional Micro Trace Buffer | Optional ETM, DWT and ITM | Optional ETM, DWT and ITM |
| **Core Architecture** | Von Neumann | Von Neumann | Harvard | Harvard |
| **ARM Architecture** | ARMv6-M | ARMv6-M | ARMv7-M | ARMv7E-M |

*Reference from ARM*

# Cortex-M Processor Family (3/3)



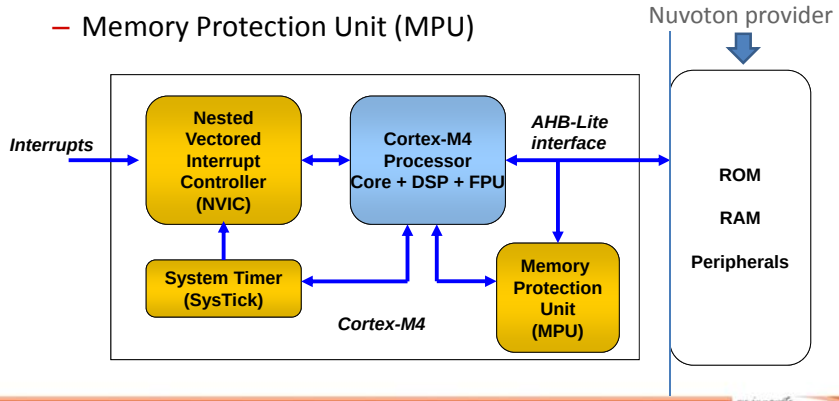

*Reference from ARM*

***Instruction Set***



# Cortex-M4F Function Block

- **Cortex-M4 processor includes**
  - Cortex-M4F processor core (DSP + FPU)
  - Nested Vectored Interrupt Controller (NVIC)
  - System Timer (SysTick)
  - Memory Protection Unit (MPU)





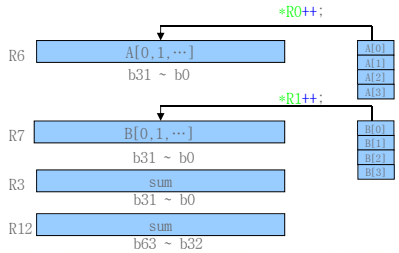

# Features of Cortex-M4F Processor (1/4)

- **Conceptually M4 is M3 + DSP instructions, with optional FPU. Key features include:**
  - ARMv7 architecture with an instruction set of
    - Thumb, Thumb-2, 1-cycle 32-bit hardware multiply, 2-12 cycle 32-bit hardware divide, saturated math support, DSP extension (1-cycle MAC and SIMD arithmetic), FP extension
  - 3-stage pipeline with branch speculation
  - 1~240 physical interrupts plus NMI, 12-cycle latency
  - Integrated sleep modes
  - 8 region memory protection unit (MPU)



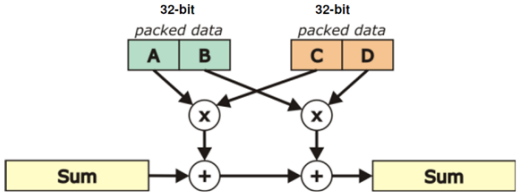
# Features of Cortex-M4F Processor (2/4)

- ***Most operations are dominated by MACs***
  - FIR Filter $y[n]=\sum_{k=0}^{N-1} h[k]x[n-k]$
  - IIR or recursive filter $y[n]=b_0x[n]+b_1x[n-1]+b_2x[n-2]+a_1y[n-1]+a_2y[n-2]$
  - FFT Butterfly (radix-2) $Y[k_1]=X[k_1]+X[k_2]$, $Y[k_2]=(X[k_1]-X[k_2])e^{-j\varphi}$
- **Single Cycle MAC**
  - 32 x 32 + 64 -> 64 operation
    - `sum += *A++ * *B++;`
  - Two 16 x 16 operations



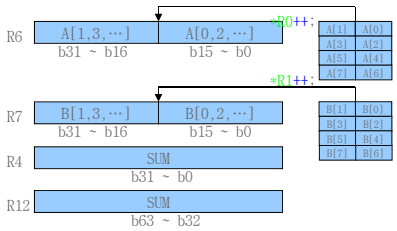
# Features of Cortex-M4F Processor (3/4)

- **SIMD Instructions**
  - Handle two 16-bit data or four 8-bit data in parallel.




- **Single Cycle SIMD**

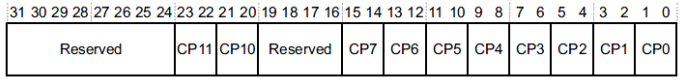
```
sum = __SMLALD(*__SIMD32(pSrcA)++
               , *__SIMD32(pSrcB)++
               , sum);
```





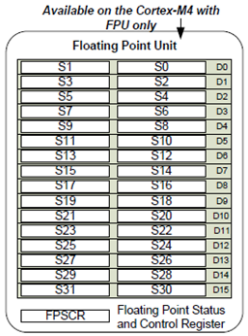

# Features of Cortex-M4F Processor (4/4)

- **FPU boosts performance by using hardware to handle single precision floating point operations.**



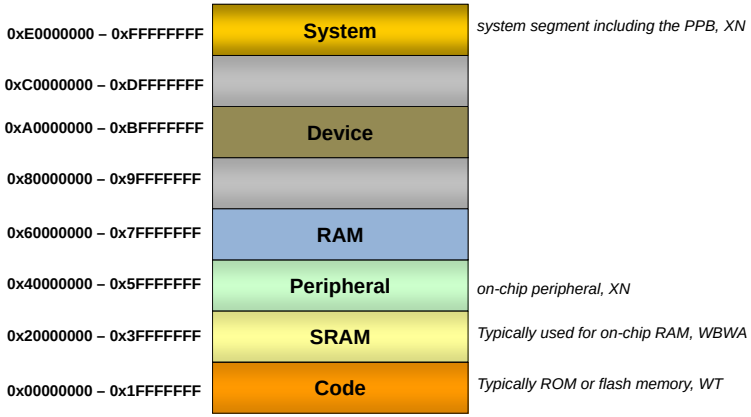

| | |
|---|---|
| 00 | Access denied |
| 01 | Privileged access only |
| 10 | Reserved |
| 11 | Full access |

```
SCB->CPACR |= ((3UL << 10*2) |          /* set CP10 Full Access */
               (3UL << 11*2)  );         /* set CP11 Full Access */
```






# Cortex-M address map



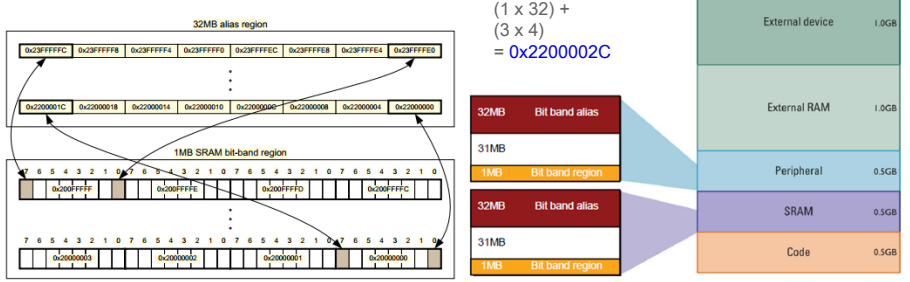

*Note 1 : Event entry points (vectors), system control, and configuration are defined at physical addresses*
*Note 2 : A multi-word access which crosses a 0.5GB address boundary is UNPREDICTABLE*



# Bit-banding

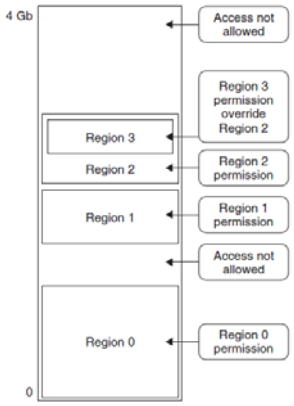
- **Writes to a word address in the bit band alias affect a single bit in the bit band region.**
- **Bit 0 of the stored register is written to the appropriate bit.**
- **bit_word_addr = bit_band_base +**
  **(byte_offset x 32) +**
  **(bit_number × 4)**

▪E.g. : 0x20000000[11] = 0x22000000 +
(1 x 32) +
(3 x 4)
= 0x2200002C

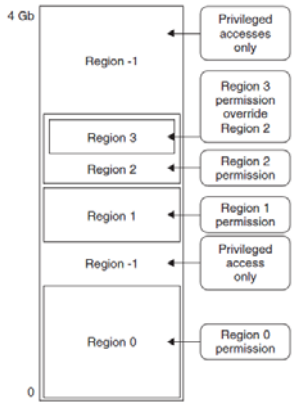


# Memory Protection Unit

- **Optional 8 region MPU with sub regions and background region – to prevent memory access violations.**




*Without background region*




*With background region*

| AP Value | Privileged Access | User Access |
|---|---|---|
| 000 | No access | No access |
| 001 | Read/Write | No access |
| 010 | Read/Write | Read only |
| 011 | Read/Write | Read/Write |
| 100 | Unpredictable | Unpredictable |
| 101 | Read only | No access |
| 110 | Read only | Read only |
| 111 | Read only | Read only |



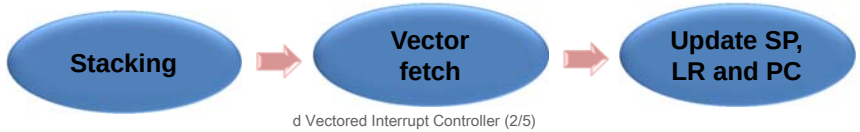
# Vector Table Offset

- **The vector table starting address can be changed by updating Vector Table Offset Register (VTOR @ 0xE000ED08).**
  - Relocating vector table to SRAM
    - To allow dynamically changing exception handler entrance points.
    - For faster vector fetch.
  - Relocating vector table to different one in ROM (or flash)
    - Different stage of program execution can have different exception handlers.

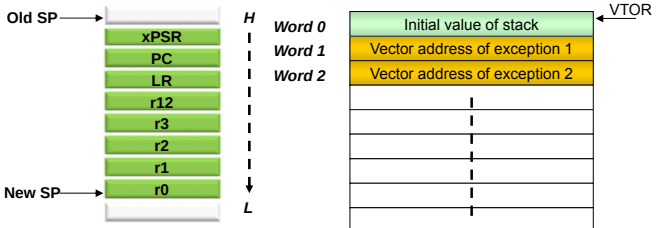


# Exception Model (1/2)

- **An exception occurred and is higher priority than the current level, and not blocked by any masking registers**




d Vectored Interrupt Controller (2/5)




*S0 ~ S15 and FPSCR will be pushed to stack if FP exists and CONTROL.FPCA is 1*




**Vector Table**

***SP*** : updated for stacking, then MSP will be used

**PSR** : exception number.

**PC** : vector handler

**LR** : updated to **EXC_RETURN**



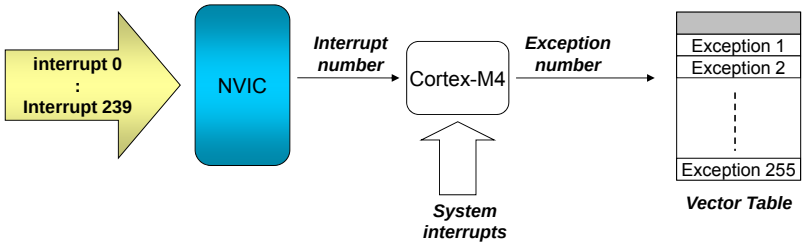
# Exception Model (2/2)

- **Exceptions defined in Cortex-M4**

| Exception number | Exception | Priority level |
|---|---|---|
| 0 | MSP initial value | |
| 1 | Reset | -3 |
| 2 | NMI | -2 |
| 3 | HardFault | -1 |
| 4 | MemManage | configured by register SHPR1 |
| 5 | Bus Fault | configured by register SHPR1 |
| 6 | Usage Fault | configured by register SHPR1 |
| 11 | SVCall | configured by register SHPR2 |
| 14 | PendSV | configured by register SHPR3 |
| 15 | SysTick | configured by register SHPR3 |
| 16 | External Interrupt (0) | configured by register NVIC_IPRx |
| … | … | … |
| 255 | External Interrupt (239) | configured by register NVIC_IPRx |

# Nested Vectored Interrupt Controller(1/5)

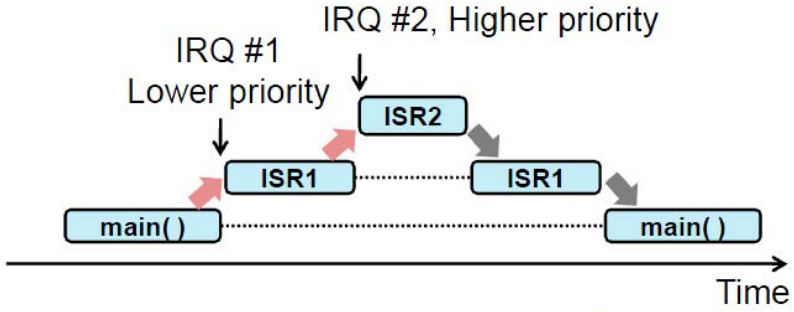
- **Supports up to 240 discrete interrupts which can be either level-sensitive or pulse-sensitive.**
- **NVIC interrupts can be enabled/disabled, pended/un-pended and prioritized by setting NVIC control registers**






# Nested Vectored Interrupt Controller (2/5)

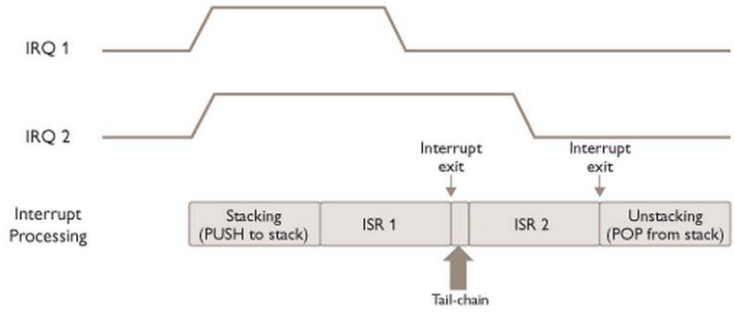
- **Automatic nested interrupt handling**
  - By assigning different priorities to each interrupt, the NVIC can support Nested Interrupts automatically without any software intervention. (*automatic hardware stacking and unstackina*)






# Nested Vectored Interrupt Controller (3/5)

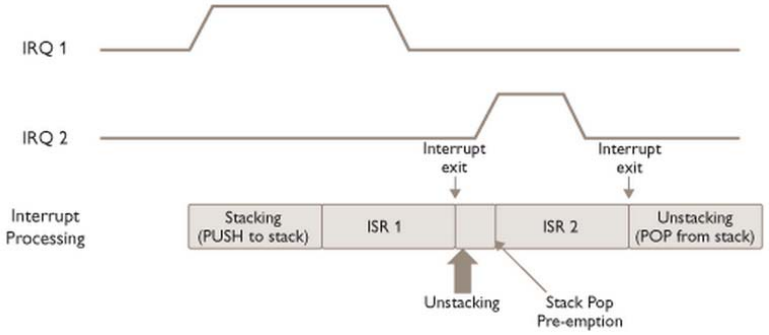
- **Tail chaining**
  - Skip the stack Pop and services the new interrupt immediately.





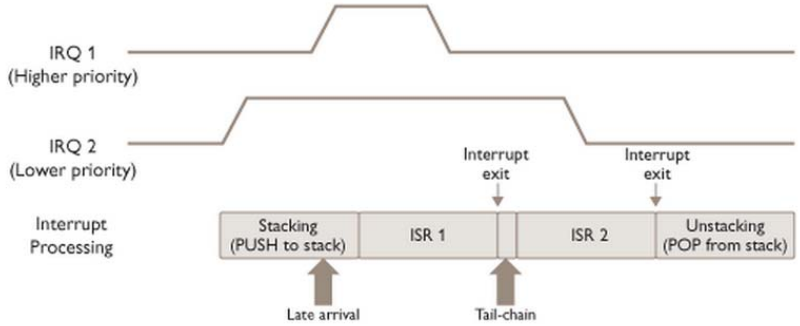

# Nested Vectored Interrupt Controller (4/5)

- **Stack pop pre-emption**
  - Abandons the stack Pop and services the new interrupt immediately.



## Nested Vectored Interrupt Controller (5/5)

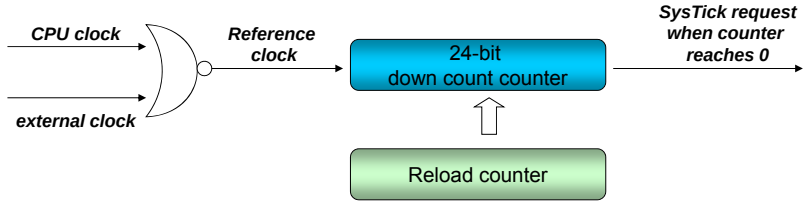
- **Late Arrivals**
  - If a higher priority interrupt arrives during the stacking of a lower priority interrupt, the processor fetches a new vector address and processes the higher priority interrupt first.






## System timer - SysTick

- **SysTick provides a simple, 24-bit clear-on-write, decrementing, wrap-on-zero counter.**
- **When enabled, the timer will count down. When the counter transitions to zero, the COUNTFLAG status bit is set.**
- **The reference clock can be the core clock or an external clock source.**






# Q & A

## Thank You