ARM Cortex-M0 Introduction

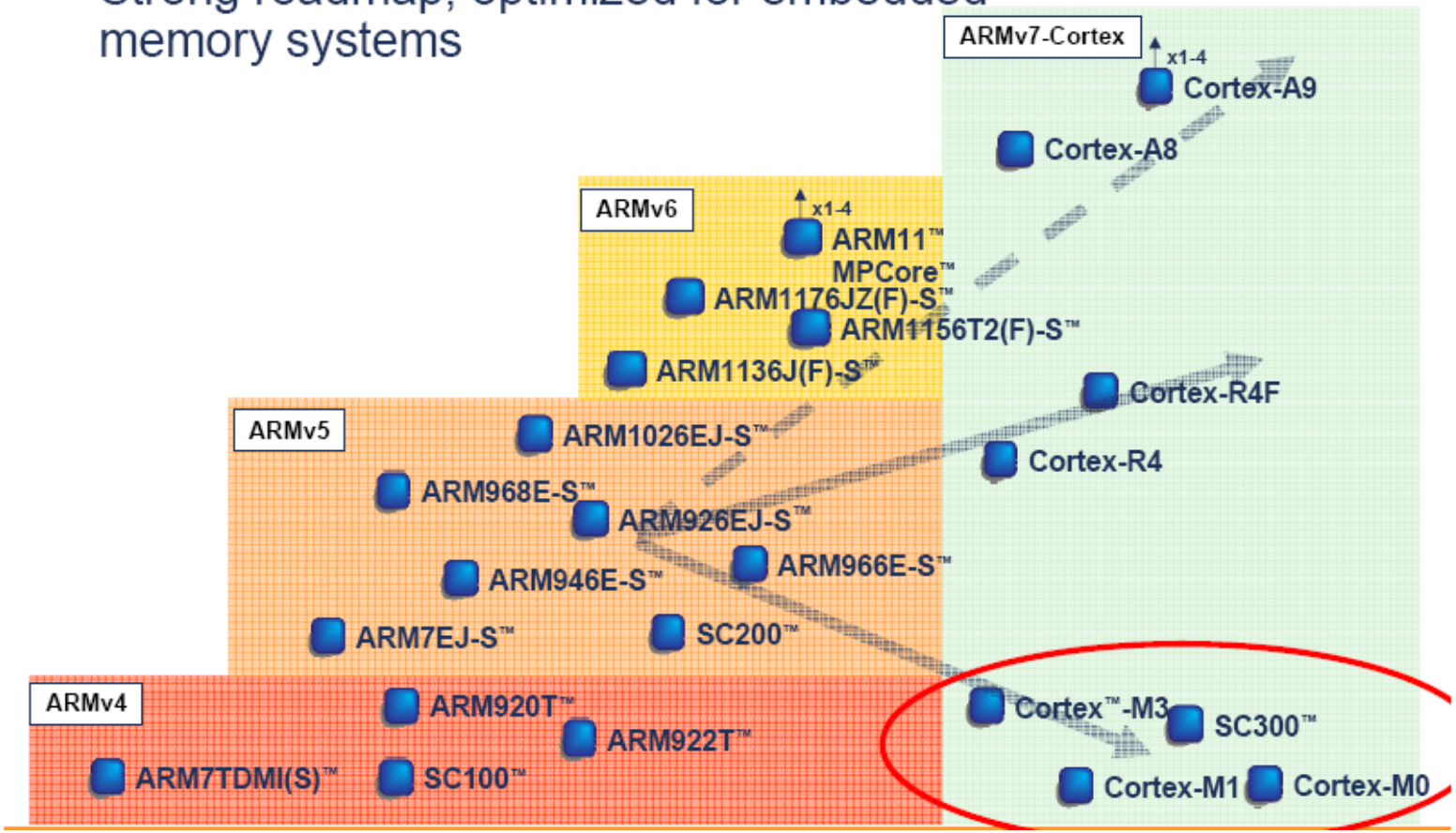
2013/1/5

Agenda

- ARM Processors roadmap
- ARM Cortex-M0 roadmap
- Cortex-M0 performance
- Cortex-M0 functional blocks
- Cortex-M0 architecture overview
- Memory Model
- Exception Model
- Cortex-M0 address map
- System Control Space
- System Timer
- NVIC

ARM Processors roadmap

- Strong roadmap, optimized for embedded memory systems

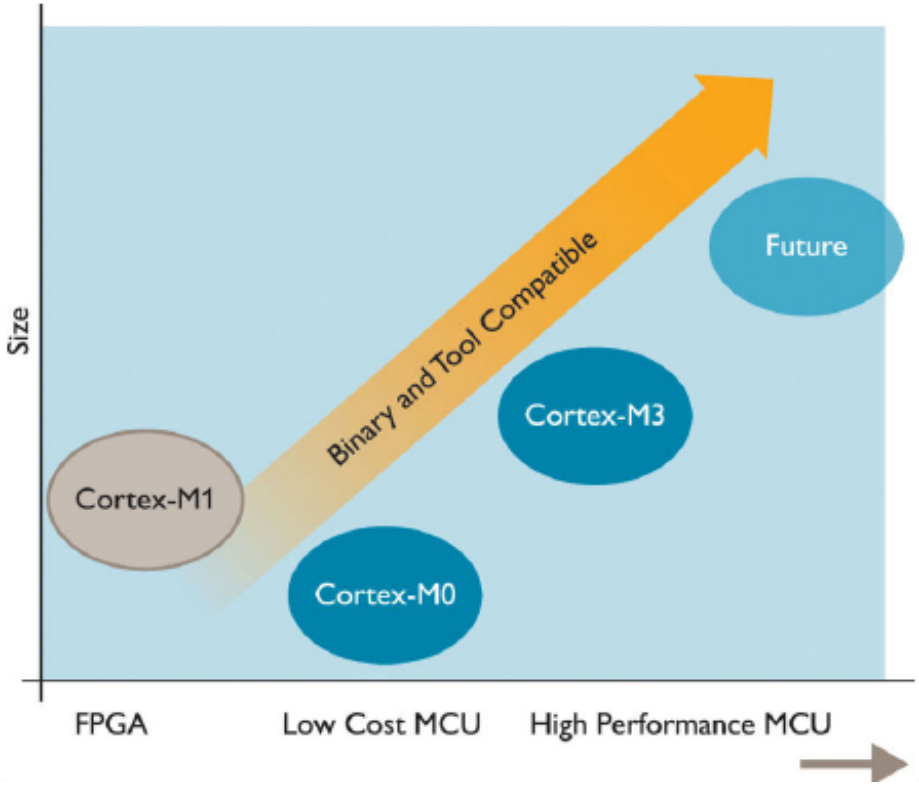


Reference from ARM

ARM cortex-M0

ARM Cortex-M Series roadmap

- **Designed for microcontroller applications**
  - **Broad software development tools and RTOS support**



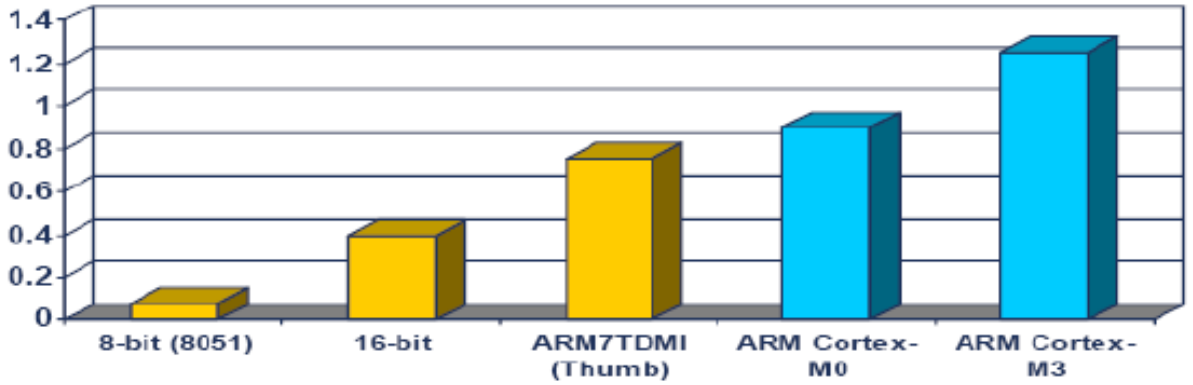
ARM Cortex-A Series:
Applications processors for feature-rich OS and user applications

ARM Cortex-R Series:
Embedded processors for real-time signal processing and control applications

ARM Cortex-M Series:
Deeply embedded processors optimized for microcontroller and low-power applications

Reference from ARM

Cortex-M0 performance

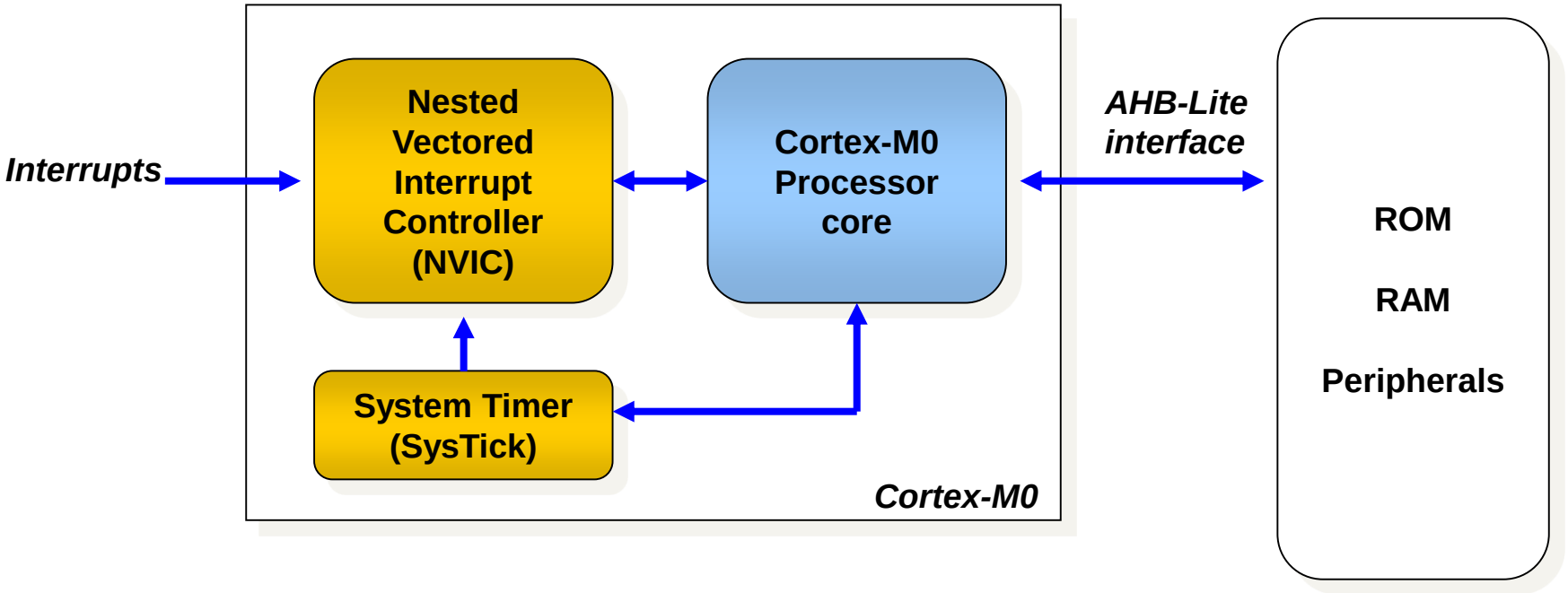


	ARM7TDMI	Cortex-M0	Cortex-M3
DMIPS/MHz	0.75 (Thumb) 0.95 (ARM)	0.90	1.25
Gate count	45k	12k	43k
Number interrupts	2	1-32 + NMI	1-240 + NMI
Interrupt priorities	None	4	8-256
Breakpoints, Watchpoints	2/0	4/2, 2/1	8/4, 2/1
MPU, integrated trace option	No	No	Yes
Hardware Divide	No	No	Yes

Reference from ARM

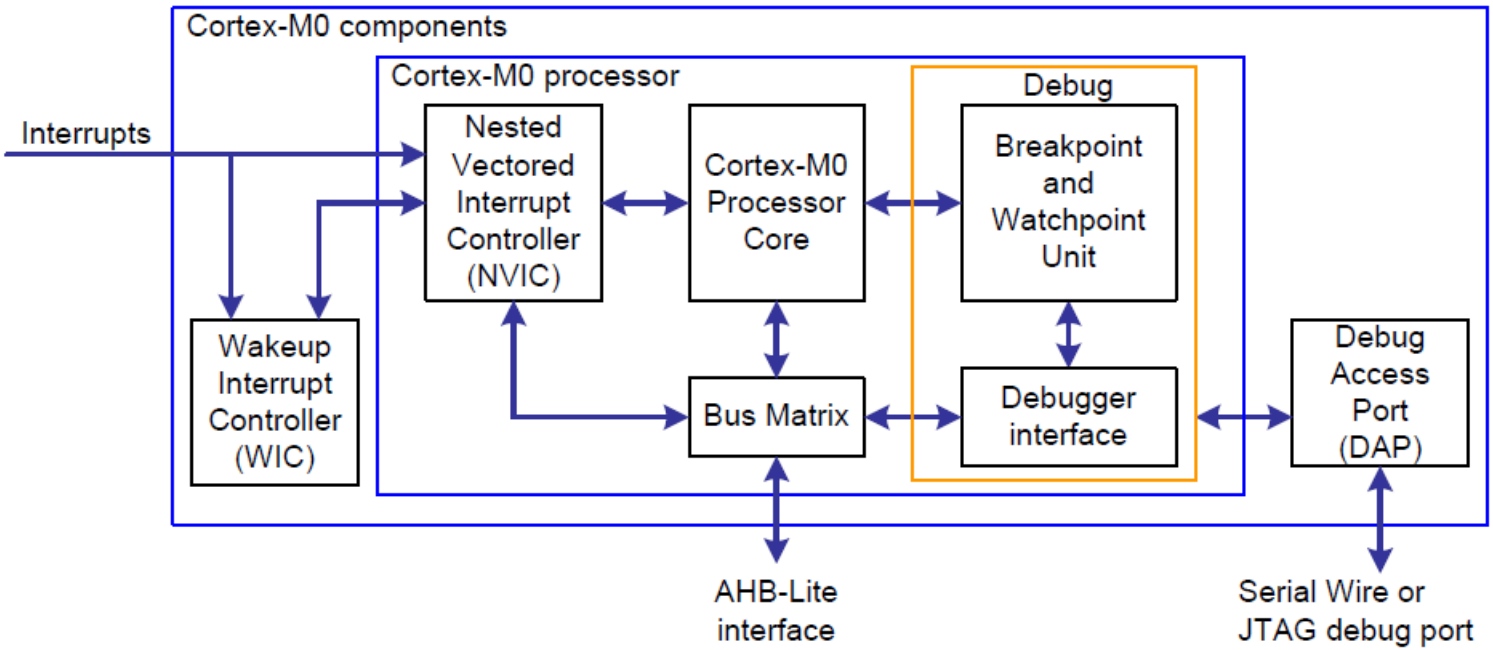
Cortex-M0 Functional Controller

- Cortex-M0 processor includes
  - Cortex-M0 processor core
  - Nested Vectored Interrupt Controller (NVIC)
  - System Timer (SysTick)



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



Cortex-M0 Architecture Overview

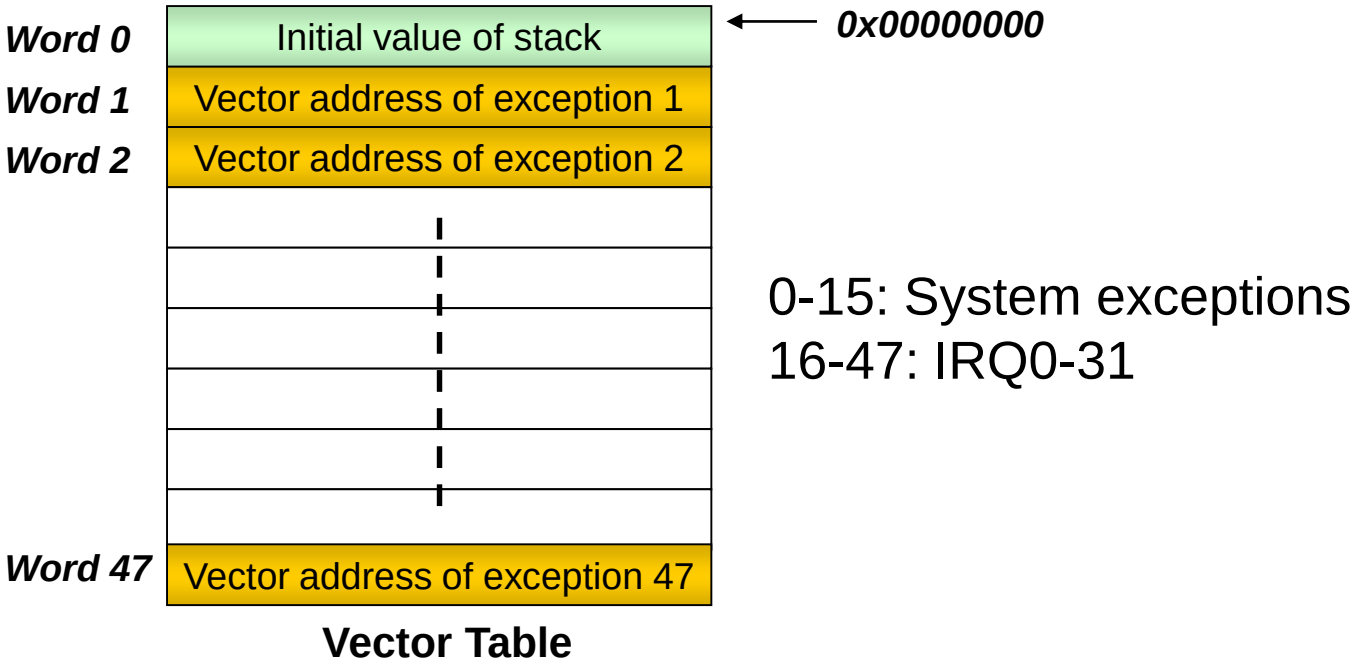
- ARMv6-M (which is a subset of ARMv7-M, upward compatible)
  - It supports only the Thumb instruction set. No Interworking code is required.
  - Total are 56 instructions. 6 are 32-bit length, the others are 16-bit length
    - 32-bit instructions : BL, DMB, DSB, ISB, MRS, MSR
- Supports byte (8-bit), halfword (16-bit) and word (32-bit) data types, each must be accessed with natural alignment.
- Processor modes are Thread and Handler.
  - Always in privilege
  - Supports hardware key context state saving and restoring (Need not Assembly coding for ISR).
- Built-in timer and interrupt controller
  - NVIC, Systick
- Memory mapped I/O
  - Use same instructions to access memory and registers
  - Ex : LDR, STR

Memory Model

- 32-bit address space
- Virtual memory is not supported in ARMv6-M.
- Instruction fetches are always halfword-aligned
- Data accesses are always naturally aligned
  - Ex : word at address A : A, A+1, A+2 and A+3.

Exceptional Model

- An exception may be an interrupt or a hardware error
- Each exception has exception number, priority number and vector address
  - Vector address
  - Vector table base address is fixed at *0x00000000*



Exception Model – cont.

Exceptions defined in Cortex-M0

Exception number	Exception	Priority level
1	Reset	-3 (1st優先權)
2	NMI	-2 (2nd優先權)
3	HardFault	-1 (3th優先權)
11	SVCall	configured by register SHPR2 (4st優先權:0-3)
14	PendSV	configured by register SHPR3 (4st優先權:0-3)
15	SysTick	configured by register SHPR3 (4st優先權:0-3)
16	External Interrupt (0)	configured by register NVIC_IPRx (4st優先權:0-3)
…	…	…
47	External Interrupt (31)	configured by register NVIC_IPRx (4st優先權:0-3)

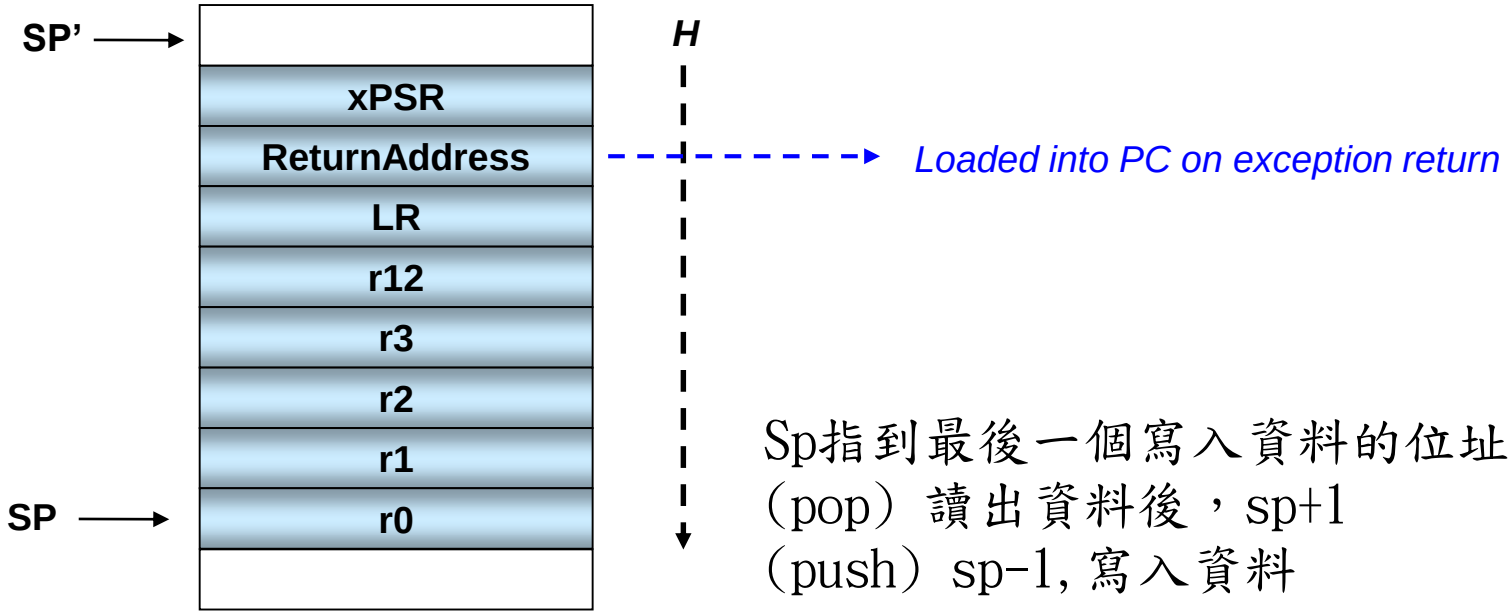
0-15: System exceptions
16-47: IRQ0-31

Exception Model – cont.

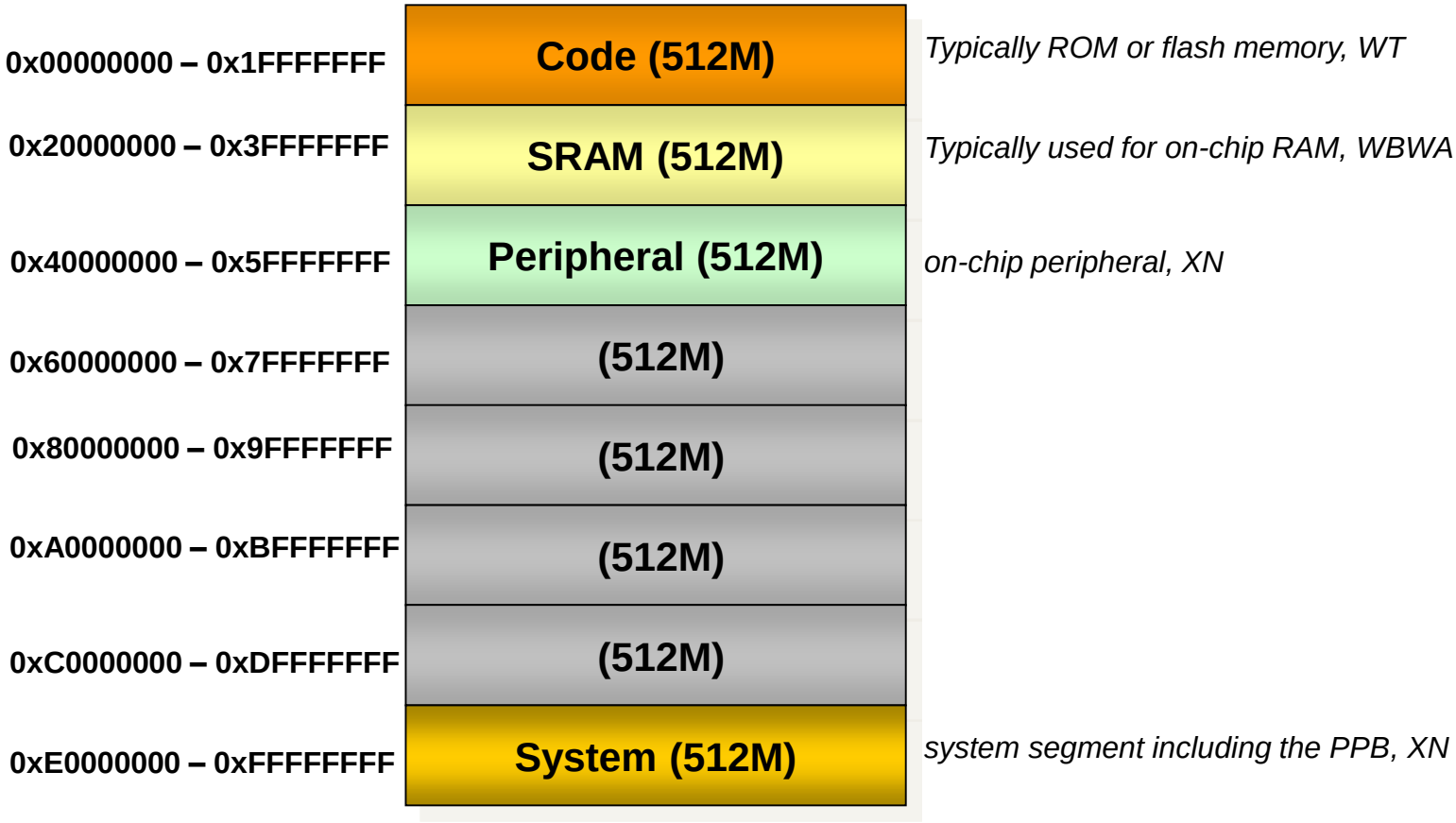
- Exception priorities and preemption
  - Lower numbers take higher precedence
  - If multiple exceptions have the same priority number, the pending exception with the lowest exception number takes precedence.
  - Only exceptions with a higher priority (lower priority number) can preempt an active exception.
- Priority number: 0:高優先權，3:低優先權
- Same Priority number: exception number較小者，先執行

Context Saving & Restoring

- Key context saving and restoring
  - Using full-descending stack format (decremented immediately before storing, incremented after reading)
  - *xPSR, ReturnAddress(), LR (R14), R12, R3, R2, R1, and R0*

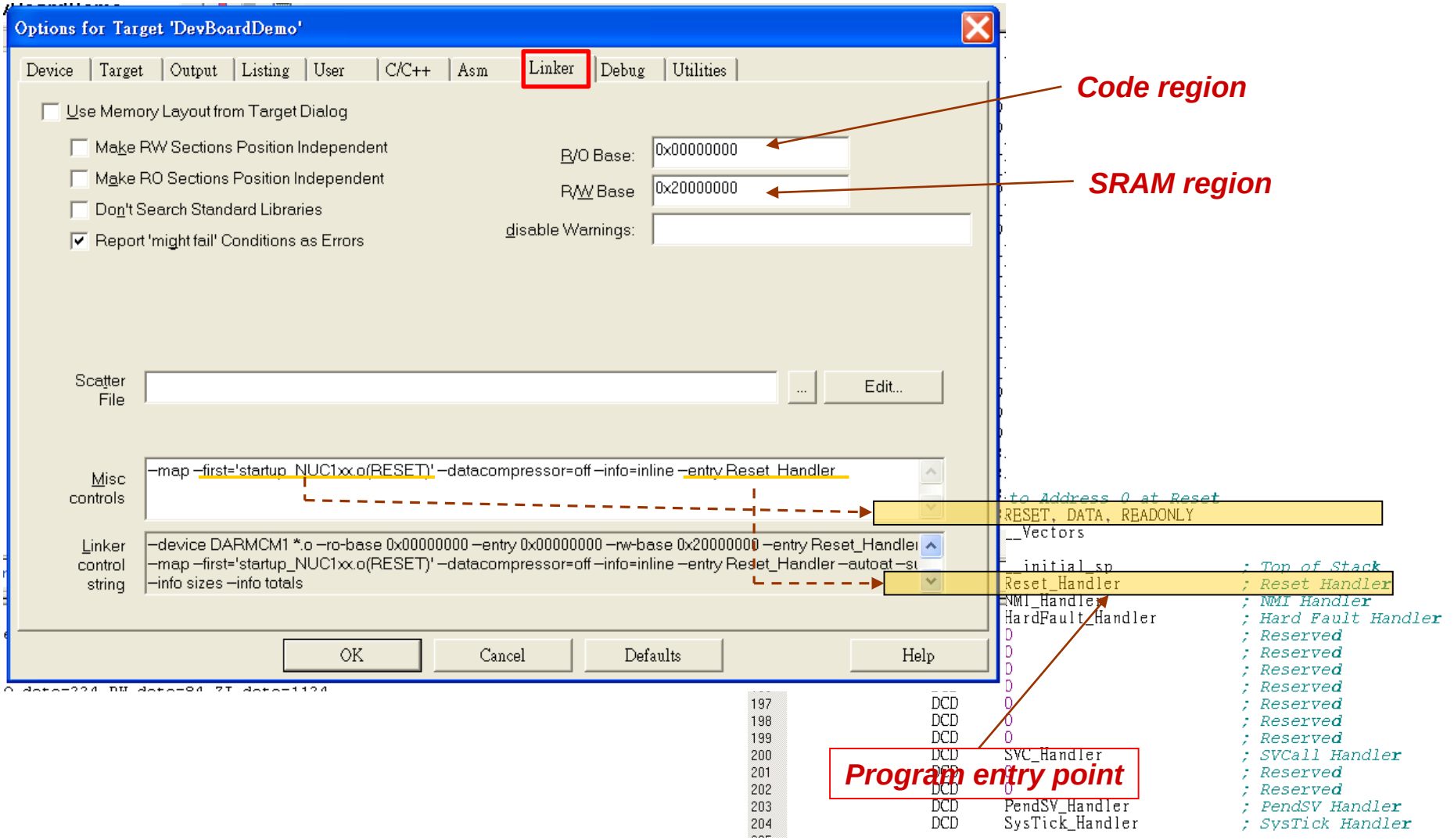


Cortex-M0 address map



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

Setting in Keil



Setting in Keil

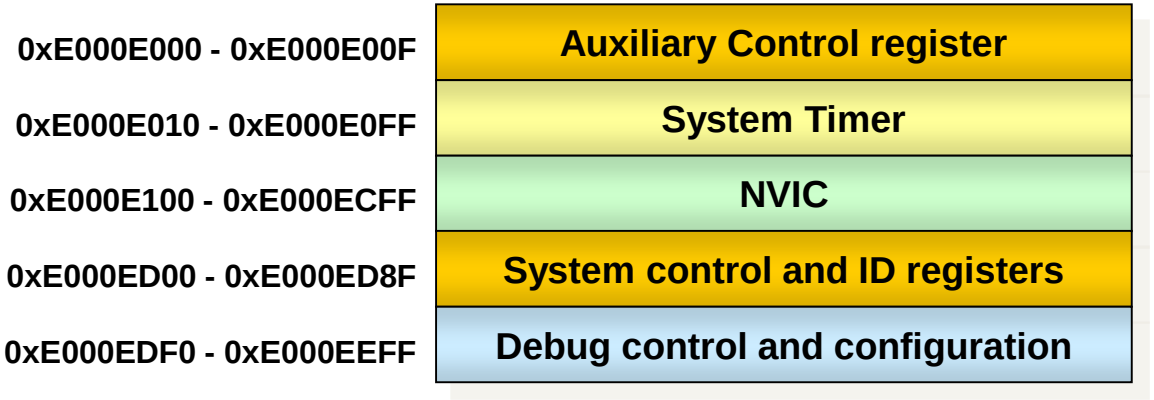
```
; Vector Table Mapped to Address 0 at Reset
                AREA    RESET, DATA, READONLY
                EXPORT  __Vectors

__Vectors       DCD     __initial_sp              ; Top of Stack
                DCD     Reset_Handler             ; Reset Handler
                DCD     NMI_Handler               ; NMI Handler
                DCD     HardFault_Handler         ; Hard Fault Handler
```

Program entry point

System Control Space (SCS)

- Consists of the following groups
  - CPUID space.
  - System control, configuration and status.
  - SysTick system timer
  - Nested Vectored Interrupt Controller (NVIC)



System Control Space – cont.

System control register

- Cortex-M0 status and operating mode
- Including CPUID, Cortex-M0 interrupt priority and Cortex-M0power management

Address	Name	Function	Type	Reset Value
0xE000ED00	CPUID	CPUID Register	RO	IMPLEMENTATION DEFINED
0xE000ED04	ICSR	Interrupt Control State Register	R/W	0x00000000
0xE000ED08	VTOR	Vector Table Offset Register	RO	0x00000000
0xE000ED0C	AIRCR	Application Interrupt/Reset Control Register	R/W	bits [10:8] = '000'
0xE000ED10	SCR	System Control Register (optional)	R/W	bits [4,2,1] = '000'
0xE000ED1C	SHPR2	System Handler Priority Register 2	R/W	SBZ
0xE000ED20	SHPR3	System Handler Priority Register 3	R/W	SBZ

System Control Space – cont.

CPUID Register (CPUID)

Bits	Name	Function
[31:24]	IMPLEMENTER	Implementer code assigned by ARM. (ARM = 0x41)
[19:16]	PART	Reads as 0xC for ARMv6-M parts
[115:4]	PARTNO	Reads as 0xC20.
[3:0]	REVISION	Reads as 0x0

Application Interrupt and Reset Control Register (AIRCR)

Bits	Name	Function
[31:16]	VECTORKEY	write 0x05FA only, otherwise it will be unpredictable.
[2]	SYSRESETREQ	Writing 1, will cause a reset signal.
[1]	VECTCLRACTIVE	Set 1, will clears all active state information for fixed and configurable exceptions. (write only when the core is halted.

System Control Space – cont.

Interrupt Control State Register (ICSR)

Bits	Name	Function
[31]	NMIPENDSET	NMI set-pending bit Write: 0 = no effect 1 = changes NMI exception state to pending. Read: 0 = NMI exception is not pending 1 = NMI exception is pending.
[28]	PENDSVSET	PendSV set-pending bit. Write: 0 = no effect 1 = changes PendSV exception state to pending. Read: 0 = PendSV exception is not pending 1 = PendSV exception is pending.

System Control Space – cont.

Interrupt Control State Register (ICSR)

Bits	Name	Function
[27]	PENDSVCLR	PendSV clear-pending bit. Write: 0 = no effect 1 = removes the pending state from the PendSV exception.
[26]	PENDSTSET	SysTick exception set-pending bit. Write: 0 = no effect 1 = changes SysTick exception state to pending. Read: 0 = SysTick exception is not pending 1 = SysTick exception is pending.
[25]	PENDSTCLR	SysTick exception clear-pending bit. Write: 0 = no effect 1 = removes the pending state from the SysTick exception.

System Control Space – cont.

Interrupt Control State Register (ICSR)

Bits	Name	Function
[23]	ISRPREEMPT	If set, a pending exception will be serviced on exit from the debug halt state. (read only)
[22]	ISRPENDING	Interrupt pending flag, excluding NMI and Faults: (read only) 0 = interrupt not pending 1 = interrupt pending.
[17:12]	VECTPENDING	Indicates the highest priority exception number : 0 = no pending exceptions Nonzero = the highest priority exception number.
[5:0]	VECTACTIVE	Contains the active exception number 0 = Thread mode Nonzero = The currently active exception number.

System Control Space – cont.

System Control Register (SCR)

Bits	Name	Function
[4]	SEVONPEND	Send Event on Pending bit: 0 = only enabled interrupts or events can wakeup the processor, disabled interrupts are excluded 1 = enabled events and all interrupts, including disabled interrupts, can wakeup the processor.
[2]	SLEEPDEEP	Controls whether the processor uses sleep or deep sleep as its low power mode: 0 = sleep 1 = deep sleep
[1]	SLEEPONEXIT	Indicates sleep-on-exit when returning from Handler mode to Thread mode: 0 = do not sleep when returning to Thread mode. 1 = enter sleep, or deep sleep, on return from an ISR to Thread mode.

System Control Space – cont.

System Handler Priority Register 2 (SHPR2)

Bits	Name	Function
[31:30]	IMPLEMENTER	**Priority of system handler 11 – SVCall** 0= the highest priority, 3=the lowest priority

System Handler Priority Register 3 (SHPR3)

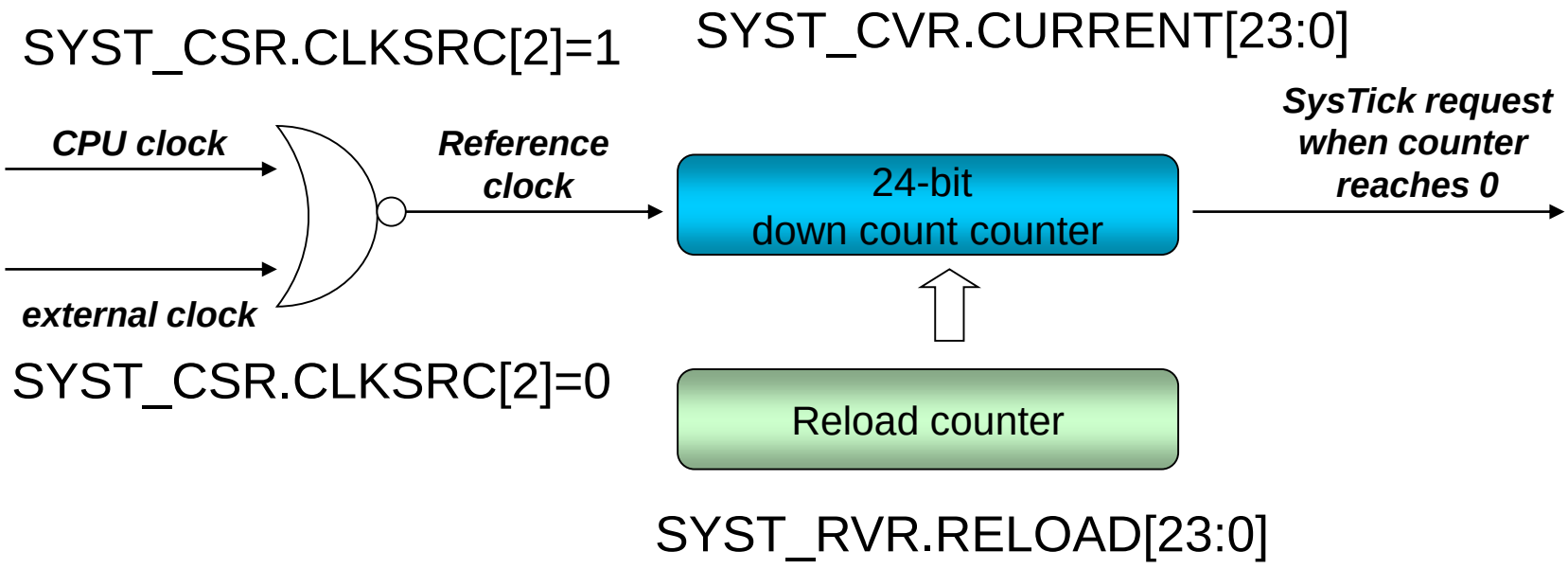
Bits	Name	Function
[31:30]	VECTORKEY	**Priority of system handler 15 – SysTick** 0= the highest priority, 3=the lowest priority
[23:22]	SYSRESETREQ	**Priority of system handler 14 – PendSV** 0= the highest priority, 3=the lowest priority

System Timer - SysTick

- SysTick: 24-bit clear-on-write, decrementing, wrap-on-zero counter.
- be used as a Real Time Operating System (RTOS) tick timer or as a simple counter.
- When enabled, count down from SysTick Current Value Register (SYST_CVR) to zero, and reload SysTick Reload Value Register (SYST_RVR), then continue decrement.
- When count down to zero, COUNTFLAG=1. COUNTFLAG=0, on reads.
- SYST_CVR value is UNKNOWN on reset.
- SYST_RVR =0, timer=0. (disable timer even if timer enable)

System Timer - SysTick

- The reference clock can be the core clock or an external clock source.



System Timer – cont.

System timer register

Address	Name	Function	Type	Reset Value
0xE000E010	SYST_CSR	**SysTick Control and Status** Enables counting and interrupt Selects reference clock source	R/W	0x00000000
0xE000E014	SYST_RVR	**SysTick Reload value** Copied to current value register when counter reaches 0	R/W	UNKNOWN
0xE000E018	SYST_CVR	**SysTick Current value** Keeps current counter value	R/W	UNKNOWN
0xE000E01C	SYST_CALIB	**SysTick Calibration value** Defined by implementation	RO	IMP DEF

System Timer – cont.

SysTick Control and Status (SYST_CSR)

Bits	Name	Function
[16]	COUNTFLAG	=1, if timer counted to 0. =0, read or write to the Current Value register.
[2]	CLKSRC	1 = Core clock used for SysTick. 0 = Clock source is external reference clock
[1]	TICKINT	1 = Counting down to 0 will cause the SysTick exception. 0 = does not cause the SysTick exception. (check COUNTFLAG)
[0]	ENABLE	1 = The counter will operate in a multi-shot manner 0 = The counter is disabled

System Timer – cont.

SysTick Reload Value Register (SYST_RVR)

Bits	Name	Function
[23:0]	RELOAD	Value to load into the Current Value register when the counter reaches 0.

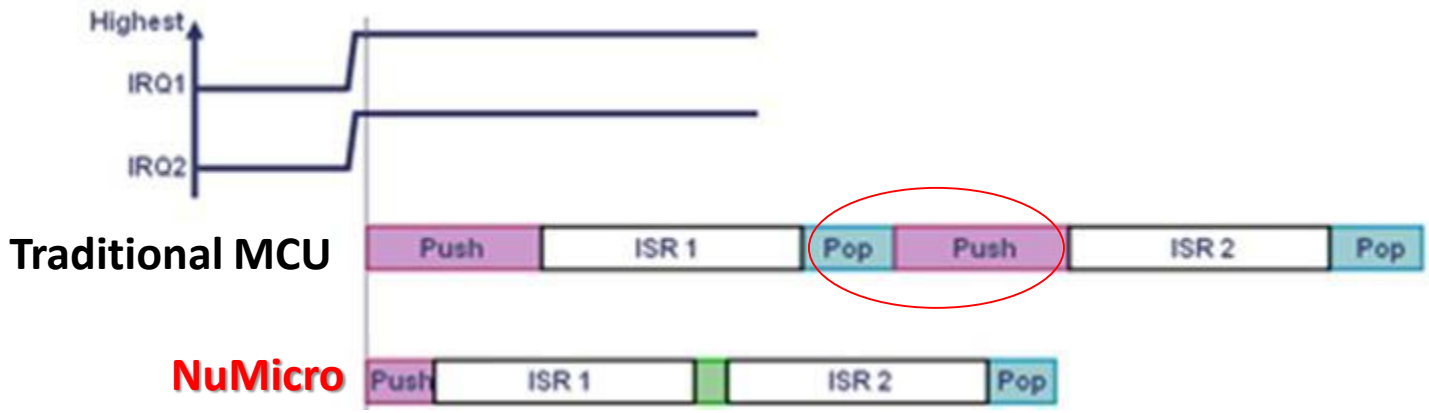
SysTick Current Value Register (SYST_CVR)

Bits	Name	Function
[23:0]	CURRENT	Current counter value.

NVIC (Nested Vectored Interrupt Controller)

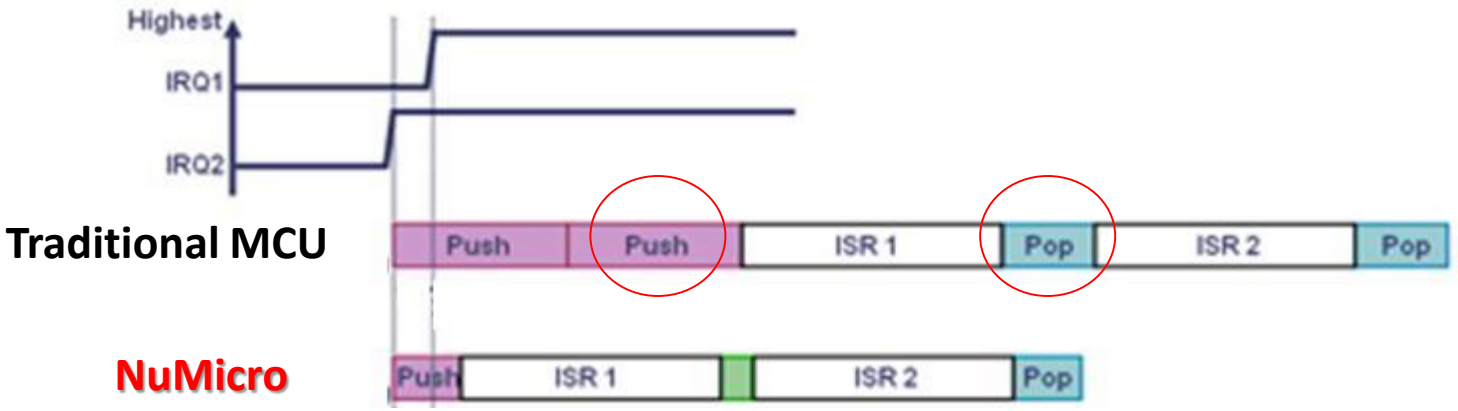
- Upon stack based exception model
  - Restore registers to resume to normal execution automatically
  - Remove redundant PUSH/POP operations needed by traditional C-based ISRs.
- Benefits over traditional systems
  - Enhancing performance in low MHz systems
  - Provide deterministic response for late arrival and preemption
  - Achieve lower latency without state restore and save

Tail Chaining Technology



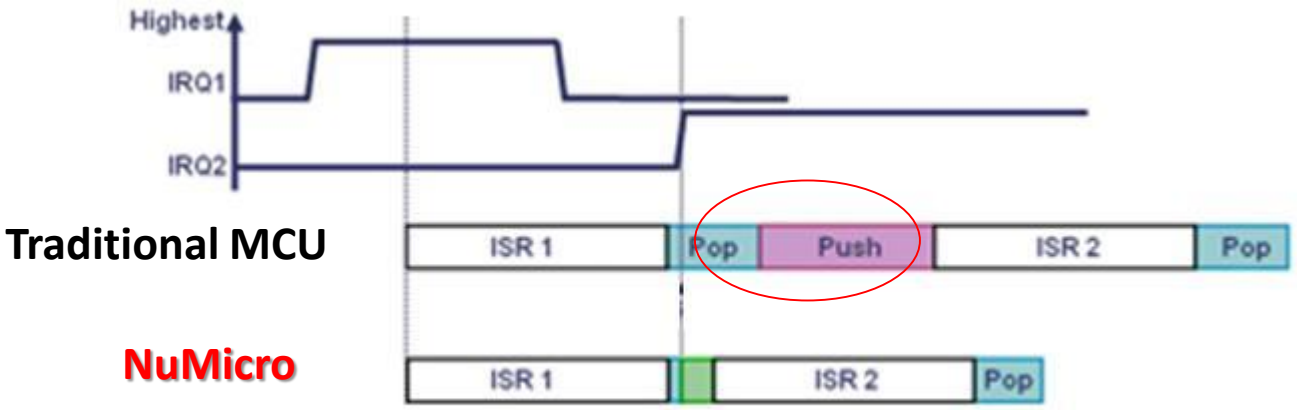
- The processor state is automatically saved on interrupt entry, and restored on interrupt exit
- Fewer cycles than a software implementation, significantly enhancing performance in low MHz systems

Late Arrival of Higher Priority Interrupts



- When a higher priority interrupt request occurs before the current ISR starts to execute (at the stage of state saving and starting address fetching), the NVIC will give priority to the higher one.
- NVIC immediately fetches a new vector address to service the pending interrupt

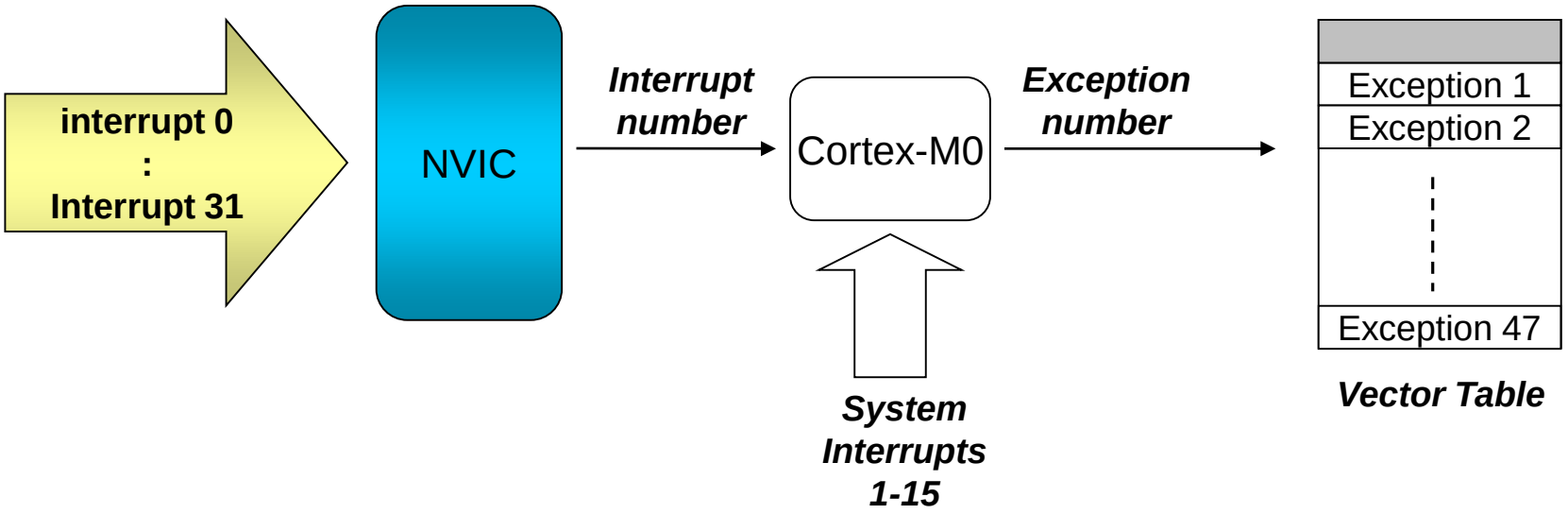
Stack Pop Preemption



- Abandon a stack Pop if an exception arrives and service the new interrupt immediately.
- Achieve lower latency in a deterministic manner by pre-empting and switching to the second interrupt without completing the state restore and save

Nested Vectored Interrupt Controller (NVIC)

- Supports up to 32 (IRQ[31:0]) 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



NVIC Exception Model

Exception Name	Vector Number	Priority
Reset	1	-3
NMI	2	-2
Hard Fault	3	-1
Reserved	4 ~ 10	Reserved
SVCall	11	Configurable
Reserved	12 ~ 13	Reserved
PendSV	14	Configurable
SysTick	15	Configurable
Interrupt (IRQ0 ~ IRQ31)	16 ~ 47	Configurable

System Interrupt Map

向量號	中斷號	中斷名稱	源 IP	中斷描述
16	0	BOD_OUT	Brown-Out	欠壓檢測中斷
17	1	WDT_INT	WDT	看門狗定時器中斷
18	2	EINT0	GPIO	PB.14 管腳上的外部信號中斷
19	3	EINT1	GPIO	PB.15 管腳上的外部信號中斷
20	4	GPAB_INT	GPIO	PA[15:0]/PB[13:0] 的外部信號中斷
21	5	GPCDE_INT	GPIO	PC[15:0]/PD[15:0]/PE[15:0] 的外部信號中斷
22	6	PWMA_INT	PWM0~3	PWM0, PWM1, PWM2 與 PWM3 中斷
23	7	PWMB_INT	PWM4~7	PWM4, PWM5, PWM6 與 PWM7 中斷

System Interrupt Map

向量號	中斷號	中斷名稱	源 IP	中斷描述
24	8	TMR0_INT	TMR0	Timer 0 中斷
25	9	TMR1_INT	TMR1	Timer 1 中斷
26	10	TMR2_INT	TMR2	Timer 2 中斷
27	11	TMR3_INT	TMR3	Timer 3 中斷
28	12	UART02_INT	UART0/2	UART0 與 UART2 中斷
29	13	UART1_INT	UART1	UART1 中斷
30	14	SPI0_INT	SPI0	SPI0 中斷
31	15	SPI1_INT	SPI1	SPI1 中斷

NVIC registers

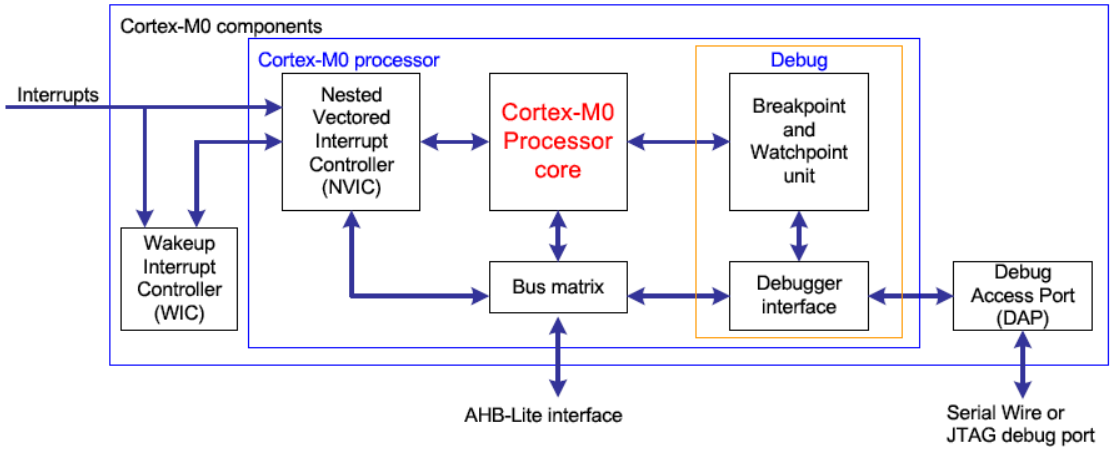
Address	Name	Function	Type	Reset Value
0xE000E100	NVIC_ISER	Irq 0 to 31 Set-Enable Register	R/W	0x00000000
0xE000E180	NVIC_ICER	Irq 0 to 31 Clear-Enable Register	R/W	0x00000000
0xE000E200	NVIC_ISPR	Irq 0 to 31 Set-Pending Register	R/W	0x00000000
0xE000E280	NVIC_ICPR	Irq 0 to 31 Clear-Pending Register	R/W	0x00000000
0xE000E400	NVIC_IPR0	Irq 0 to 3 Priority Register	R/W	0x00000000
0xE000E404	NVIC_IPR1	Irq 4 to 7 Priority Register	R/W	0x00000000
0xE000E408	NVIC_IPR2	Irq 8 to 11 Priority Register	R/W	0x00000000
0xE000E40C	NVIC_IPR3	Irq 12 to 15 Priority Register	R/W	0x00000000
0xE000E410	NVIC_IPR4	Irq 16 to 19 Priority Register	R/W	0x00000000
0xE000E414	NVIC_IPR5	Irq 20 to 23 Priority Register	R/W	0x00000000
0xE000E418	NVIC_IPR6	Irq 24 to 27 Priority Register	R/W	0x00000000
0xE000E41C	NVIC_IPR7	Irq 28 to 31 Priority Register	R/W	0x00000000

NVIC – cont.

- Interrupt Set-Enable and Clear-Enable Registers
  - NVIC_ISER : write 1 to enable interrupt
  - NVIC_ICER : write 1 to disable interrupt
- Interrupt Set-Pending and Clear-Pending Registers
  - NVIC_ISPR : write 1 to generate interrupt request
  - NVIC_ICPR : write 1 to remove interrupt request
- Interrupt Priority Registers
  - NVIC_IPR0 ~ NVIC_IPR7
  - Each priority register can set up priority number of four interrupts

Cortex-M0 Summary

- Cortex-M0 is always in privileged mode and supports Thumb instruction only.
- Supports key context saving and restoring
  - xPSR, return address, r14, r12, r3, r2, r1, r0
- System address map is defined in advance.
  - Code, SRAM, Peripheral, System
- Contains SysTick and NVIC



General Disclaimer

The Lecture is strictly used for educational purpose.

MAKES NO GUARANTEE OF VALIDITY

- **The lecture cannot guarantee the validity of the information found here.** The lecture may recently have been changed, vandalized or altered by someone whose opinion does not correspond with the state of knowledge in the relevant fields. Note that most other encyclopedias and reference works also have similar disclaimers.

No formal peer review

- The lecture is not uniformly peer reviewed; while readers may correct errors or engage in casual peer review, they have no legal duty to do so and thus all information read here is without any implied warranty of fitness for any purpose or use whatsoever. Even articles that have been vetted by informal peer review or *featured article* processes may later have been edited inappropriately, just before you view them.

No contract; limited license

- Please make sure that you understand that the information provided here is being provided freely, and that no kind of agreement or contract is created between you and the owners or users of this site, the owners of the servers upon which it is housed, the individual Wikipedia contributors, any project administrators, sysops or anyone else who is in *any way connected* with this project or sister projects subject to your claims against them directly. You are being granted a limited license to copy anything from this site; it does not create or imply any contractual or extracontractual liability on the part of Wikipedia or any of its agents, members, organizers or other users.
- There is **no agreement or understanding between you and the content provider** regarding your use or modification of this information beyond the Creative Commons Attribution-Sharealike 3.0 Unported License (CC-BY-SA) and the GNU Free Documentation License (GFDL);

General Disclaimer

Trademarks

- Any of the trademarks, service marks, collective marks, design rights or similar rights that are mentioned, used or cited in the lectures are the property of their respective owners. Their use here does not imply that you may use them for any purpose other than for the same or a similar informational use as contemplated by the original authors under the CC-BY-SA and GFDL licensing schemes. Unless otherwise stated , we are neither endorsed by nor affiliated with any of the holders of any such rights and as such we cannot grant any rights to use any otherwise protected materials. Your use of any such or similar incorporeal property is at your own risk.

Personality rights

- The lecture may portray an identifiable person who is alive or deceased recently. The use of images of living or recently deceased individuals is, in some jurisdictions, restricted by laws pertaining to personality rights, independent from their copyright status. Before using these types of content, please ensure that you have the right to use it under the laws which apply in the circumstances of your intended use. *You are solely responsible for ensuring that you do not infringe someone else's personality rights.*