4.3 Nested vectored interrupt controller (NVIC)

This section describes the Nested Vectored Interrupt Controller (NVIC) and the registers it uses. The NVIC supports:

- Up to 81 interrupts (interrupt number depends on the STM32 device type; refer to the datasheets)
- A programmable priority level of 0-15 for each interrupt. A higher level corresponds to a lower priority, so level 0 is the highest interrupt priority
- Level and pulse detection of interrupt signals
- Dynamic reprioritization of interrupts
- Grouping of priority values into group priority and subpriority fields
- Interrupt tail-chaining
- An external *Non-maskable interrupt* (NMI)

The processor automatically stacks its state on exception entry and unstacks this state on exception exit, with no instruction overhead. This provides low latency exception handling. The hardware implementation of the NVIC registers is:

Table 44. NVIC register summary

Address	Name	Type	Required privilege	Reset value	Description
0xE000E100-0xE000E10B	NVIC_ISER0-NVIC_ISER2	RW	Privileged	0x00000000	*Table 4.3.2: Interrupt set-enable registers (NVIC_ISERx) on page 209*
0XE000E180-0xE000E18B	NVIC_ICER0-NVIC_ICER2	RW	Privileged	0x00000000	*Table 4.3.3: Interrupt clear-enable registers (NVIC_ICERx) on page 210*
0XE000E200-0xE000E20B	NVIC_ISPR0-NVIC_ISPR2	RW	Privileged	0x00000000	*Table 4.3.4: Interrupt set-pending registers (NVIC_ISPRx) on page 211*
0XE000E280-0xE000E29C	NVIC_ICPR0-NVIC_ICPR2	RW	Privileged	0x00000000	*Table 4.3.5: Interrupt clear-pending registers (NVIC_ICPRx) on page 212*
0xE000E300-0xE000E31C	NVIC_IABR0-NVIC_IABR2	RW	Privileged	0x00000000	*Table 4.3.6: Interrupt active bit registers (NVIC_IABRx) on page 213*
0xE000E400-0xE000E503	NVIC_IPR0-NVIC_IPR20	RW	Privileged	0x00000000	*Table 4.3.7: Interrupt priority registers (NVIC_IPRx) on page 214*
0xE000EF00	STIR	WO	Configurable	0x00000000	*Table 4.3.8: Software trigger interrupt register (NVIC_STIR) on page 215*

4.3.1 Accessing the Cortex-M4 NVIC registers using CMSIS

CMSIS functions enable software portability between different Cortex-M profile processors. To access the NVIC registers when using CMSIS, use the following functions:

Table 45. CMSIS access NVIC functions

CMSIS function[1]	Description
void NVIC_EnableIRQ(IRQn_Type IRQn)	Enables an interrupt or exception.
void NVIC_DisableIRQ(IRQn_Type IRQn)	Disables an interrupt or exception.
void NVIC_SetPendingIRQ(IRQn_Type IRQn)	Sets the pending status of interrupt or exception to 1.
void NVIC_ClearPendingIRQ(IRQn_Type IRQn)	Clears the pending status of interrupt or exception to 0.
uint32_t NVIC_GetPendingIRQ(IRQn_Type IRQn)	Reads the pending status of interrupt or exception. This function returns non-zero value if the pending status is set to 1.
void NVIC_SetPriority(IRQn_Type IRQn, uint32_t priority)	Sets the priority of an interrupt or exception with configurable priority level to 1.
uint32_t NVIC_GetPriority(IRQn_Type IRQn)	Reads the priority of an interrupt or exception with configurable priority level. This function return the current priority level.

1. The input parameter IRQn is the IRQ number,

4.3.2 Interrupt set-enable registers (NVIC_ISERx)

Address offset: 0x00 - 0x0B

Reset value: 0x0000 0000

Required privilege: Privileged

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SETENA[31:16]															
rs	rs	rs	rs	rs	rs	rs	rs	rs	rs	rs	rs	rs	rs	rs	rs
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SETENA[15:0]															
rs	rs	rs	rs	rs	rs	rs	rs	rs	rs	rs	rs	rs	rs	rs	rs

Bits 31:0 **SETENA**: Interrupt set-enable bits.

Write:

0: No effect

1: Enable interrupt

Read:

0: Interrupt disabled

1: Interrupt enabled.

If a pending interrupt is enabled, the NVIC activates the interrupt based on its priority. If an interrupt is not enabled, asserting its interrupt signal changes the interrupt state to pending, but the NVIC never activates the interrupt, regardless of its priority.

4.3.3 Interrupt clear-enable registers (NVIC_ICERx)

Address offset: 0x00 - 0x0B

Reset value: 0x0000 0000

Required privilege: Privileged

The ICER0-ICER2 registers disable interrupts, and show which interrupts are enabled.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CLRENA[31:16]															
rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLRENA[15:0]															
rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1

Bits 31:0 **CLRENA**: Interrupt clear-enable bits.

Write:

0: No effect

1: Disable interrupt

Read:

0: Interrupt disabled

1: Interrupt enabled.

4.3.4 Interrupt set-pending registers (NVIC_ISPRx)

Address offset: 0x00 - 0x0B

Reset value: 0x0000 0000

Required privilege: Privileged

The ISPR0-ISPR2 registers force interrupts into the pending state, and show which interrupts are pending.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SETPEND[31:16]															
rs	rs	rs	rs	rs	rs	rs	rs	rs	rs	rs	rs	rs	rs	rs	rs
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SETPEND[15:0]															
rs	rs	rs	rs	rs	rs	rs	rs	rs	rs	rs	rs	rs	rs	rs	rs

Bits 31:0 **SETPEND**: Interrupt set-pending bits

Write:

0: No effect

1: Changes interrupt state to pending

Read:

0: Interrupt is not pending

1: Interrupt is pending

Writing 1 to the ISPR bit corresponding to an interrupt that is pending:

– has no effect.

Writing 1 to the ISPR bit corresponding to a disabled interrupt:

– sets the state of that interrupt to pending.

4.3.5 Interrupt clear-pending registers (NVIC_ICPRx)

Address offset: 0x00 - 0x0B

Reset value: 0x0000 0000

Required privilege: Privileged

The ICPR0-ICPR2 registers remove the pending state from interrupts, and show which interrupts are pending.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CLRPEND[31:16]															
rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLRPEND[15:0]															
rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1	rc_w1

Bits 31:0 **CLRPEND**: Interrupt clear-pending bits

Write:

0: No effect

1: Removes the pending state of an interrupt

Read:

0: Interrupt is not pending

1: Interrupt is pending

Writing 1 to an ICPR bit does not affect the active state of the corresponding interrupt.

4.3.6 Interrupt active bit registers (NVIC_IABRx)

Address offset: 0x00- 0x0B

Reset value: 0x0000 0000

Required privilege: Privileged

The IABR0-IABR2 registers indicate which interrupts are active.

The bit assignments are:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ACTIVE[31:16]															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ACTIVE[15:0]															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Bits 31:0 **ACTIVE**: Interrupt active flags

0: Interrupt not active

1: Interrupt active

A bit reads as 1 if the status of the corresponding interrupt is active or active and pending.

4.3.7 Interrupt priority registers (NVIC_IPRx)

Address offset: 0x00- 0x0B

Reset value: 0x0000 0000

Required privilege: Privileged

The NVIC_IPR0-IPR80 registers provide an 8-bit priority field for each interrupt. These registers are byte-accessible. Each register holds four priority fields, that map to four elements in the CMSIS interrupt priority array IP[0] to IP[67], as shown in *Figure 19*.

Figure 19. NVIC_IPRx register mapping

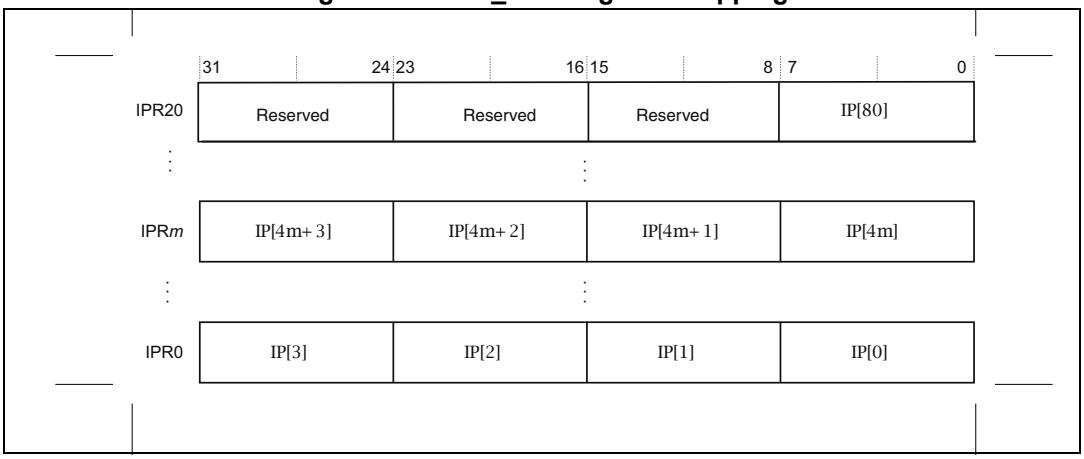

Table 46. IPR bit assignments

Bits	Name	Function
[31:24]	Priority, byte offset 3	Each priority field holds a priority value, 0-255. The lower the value, the greater the priority of the corresponding interrupt. The processor implements only bits[7:4] of each field, bits[3:0] read as zero and ignore writes.
[23:16]	Priority, byte offset 2	
[15:8]	Priority, byte offset 1	
[7:0]	Priority, byte offset 0	

See *Interrupt set-enable registers (NVIC_ISERx) on page 209* for more information about the interrupt priority array, that provides the software view of the interrupt priorities.

Find the IPR number and byte offset for interrupt *N* as follows:

- The corresponding IPR number, *M*, is given by *M* = *N* DIV 4
- The byte offset of the required Priority field in this register is *N* MOD 4, where:
  - byte offset 0 refers to register bits[7:0]
  - byte offset 1 refers to register bits[15:8]
  - byte offset 2 refers to register bits[23:16]
  - byte offset 3 refers to register bits[31:24].

4.3.8 Software trigger interrupt register (NVIC_STIR)

Address offset: 0xE00

Reset value: 0x0000 0000

Required privilege: When the USERSETMPEND bit in the SCR is set to 1, unprivileged software can access the STIR, see *Section 4.4.6: System control register (SCR)*. Only privileged software can enable unprivileged access to the STIR.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							INTID[8:0]								
							w	w	w	w	w	w	w	w	w

Bits 31:9 Reserved, must be kept cleared.

Bits 8:0 **INTID** Software generated interrupt ID

Write to the STIR to generate a Software Generated Interrupt (SGI). The value to be written is the Interrupt ID of the required SGI, in the range 0-239. For example, a value of 0x03 specifies interrupt IRQ3.

4.3.9 Level-sensitive and pulse interrupts

STM32 interrupts are both level-sensitive and pulse-sensitive. Pulse interrupts are also described as edge-triggered interrupts.

A level-sensitive interrupt is held asserted until the peripheral deasserts the interrupt signal. Typically this happens because the ISR accesses the peripheral, causing it to clear the interrupt request. A pulse interrupt is an interrupt signal sampled synchronously on the rising edge of the processor clock. To ensure the NVIC detects the interrupt, the peripheral must assert the interrupt signal for at least one clock cycle, during which the NVIC detects the pulse and latches the interrupt.

When the processor enters the ISR, it automatically removes the pending state from the interrupt, see *Hardware and software control of interrupts*. For a level-sensitive interrupt, if the signal is not deasserted before the processor returns from the ISR, the interrupt becomes pending again, and the processor must execute its ISR again. This means that the peripheral can hold the interrupt signal asserted until it no longer needs servicing.

Hardware and software control of interrupts

The Cortex-M4 latches all interrupts. A peripheral interrupt becomes pending for one of the following reasons:

- The NVIC detects that the interrupt signal is HIGH and the interrupt is not active
- The NVIC detects a rising edge on the interrupt signal
- Software writes to the corresponding interrupt set-pending register bit, see *Section 4.3.4: Interrupt set-pending registers (NVIC_ISPRx)*, or to the STIR to make an SGI pending, see *Section 4.3.8: Software trigger interrupt register (NVIC_STIR)*.

A pending interrupt remains pending until one of the following:

- The processor enters the ISR for the interrupt. This changes the state of the interrupt from pending to active. Then:
  - For a level-sensitive interrupt, when the processor returns from the ISR, the NVIC samples the interrupt signal. If the signal is asserted, the state of the interrupt changes to pending, which might cause the processor to immediately re-enter the ISR. Otherwise, the state of the interrupt changes to inactive.
  - For a pulse interrupt, the NVIC continues to monitor the interrupt signal, and if this is pulsed the state of the interrupt changes to pending and active. In this case, when the processor returns from the ISR the state of the interrupt changes to pending, which might cause the processor to immediately re-enter the ISR. If the interrupt signal is not pulsed while the processor is in the ISR, when the processor returns from the ISR the state of the interrupt changes to inactive.
- Software writes to the corresponding interrupt clear-pending register bit.

  For a level-sensitive interrupt, if the interrupt signal is still asserted, the state of the interrupt does not change. Otherwise, the state of the interrupt changes to inactive.

  For a pulse interrupt, state of the interrupt changes to:
  - Inactive, if the state was pending
  - Active, if the state was active and pending.

4.3.10 NVIC design hints and tips

Ensure software uses correctly aligned register accesses. The processor does not support unaligned accesses to NVIC registers. See the individual register descriptions for the supported access sizes.

An interrupt can enter pending state even it is disabled. Disabling an interrupt only prevents the processor from taking that interrupt.

Before programming VTOR to relocate the vector table, ensure the vector table entries of the new vector table are setup for fault handlers, NMI and all enabled exception like interrupts. For more information see *Section 4.4.4: Vector table offset register (VTOR) on page 226*.

NVIC programming hints

Software uses the CPSIE I and CPSID I instructions to enable and disable interrupts. The CMSIS provides the following intrinsic functions for these instructions:

```
void __disable_irq(void) // Disable Interrupts
void __enable_irq(void) // Enable Interrupts
```

In addition, the CMSIS provides a number of functions for NVIC control, including:

Table 47. CMSIS functions for NVIC control

CMSIS interrupt control function	Description
void NVIC_SetPriorityGrouping(uint32_t priority_grouping)	Set the priority grouping
void NVIC_EnableIRQ(IRQn_t IRQn)	Enable IRQn
void NVIC_DisableIRQ(IRQn_t IRQn)	Disable IRQn
uint32_t NVIC_GetPendingIRQ (IRQn_t IRQn)	Return true (IRQ-Number) if IRQn is pending
void NVIC_SetPendingIRQ (IRQn_t IRQn)	Set IRQn pending
void NVIC_ClearPendingIRQ (IRQn_t IRQn)	Clear IRQn pending status
uint32_t NVIC_GetActive (IRQn_t IRQn)	Return the IRQ number of the active interrupt
void NVIC_SetPriority (IRQn_t IRQn, uint32_t priority)	Set priority for IRQn
uint32_t NVIC_GetPriority (IRQn_t IRQn)	Read priority of IRQn
void NVIC_SystemReset (void)	Reset the system

The input parameter IRQn is the IRQ number, see *Table 16: Properties of the different exception types on page 37*. For more information about these functions see the CMSIS documentation.

4.3.11 NVIC register map

This table shows the NVIC register map and reset values. The base address of the main NVIC register block is 0xE000E100. The NVIC_STIR register is located in a separate block at 0xE000EF00.

Table 48. NVIC register map and reset values

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x000	NVIC_ISER0	SETENA[31:0]																															
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x004	NVIC_ISER1	SETENA[63:32]																															
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x008	NVIC_ISER2	Reserved															SETENA [80:64]															
	Reset Value																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x080	NVIC_ICER0	CLRENA[31:0]																															
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x084	NVIC_ICER1	CLRENA[63:32]																															
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x088	NVIC_ICER2	Reserved															CLRENA [80:64]															
	Reset Value																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x100	NVIC_ISPR0	SETPEND[31:0]																															
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x104	NVIC_ISPR1	SETPEND[63:32]																															
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x108	NVIC_ISPR2	Reserved															SETPEND [80:64]															
	Reset Value																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x180	NVIC_ICPR0	CLRPEND[31:0]																															
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x184	NVIC_ICPR1	CLRPEND[63:32]																															
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x188	NVIC_ICPR2	Reserved															CLRPEND [80:64]															
	Reset Value																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x200	NVIC_IABR0	ACTIVE[31:0]																															
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x204	NVIC_IABR1	ACTIVE[63:32]																															
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x208	NVIC_IABR2	Reserved															ACTIVE [80:64]															
	Reset Value																0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 48. NVIC register map and reset values (continued)

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x300	NVIC_IPR0	IP[3]								IP[2]								IP[1]								IP[0]							
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
:	:	:																															
0x320	NVIC_IPR20	Reserved																								IP[80]							
	Reset Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
SCB registers																																	
Reserved																																	
0xE00	NVIC_STIR	Reserved																								INTID[8:0]							
	Reset Value																									0	0	0	0	0	0	0	0