32-Bit RISC Microcontroller

TMPM4L Group(1)

Reference Manual
Exception
(EXCEPT-M4L(1))

Revision 1.1

2019-06

TOSHIBA ELECTRONIC DEVICES & STORAGE CORPORATION

Contents

List of Figures

List of Tables

Preface

Related document

Document name
Power supply and reset operation
Oscillation Frequency Detector
Clock Selective Watchdog Timer
Voltage Detection Circuit
Clock Control and Operation Mode
Arm® Cortex®-M4 Processor Technical Reference Manual

Conventions

- Numeric formats follow the rules as shown below:
  Hexadecimal: 0xABC
  Decimal: 123 or 0d123 – Only when it needs to be explicitly shown that they are decimal numbers.
  Binary: 0b111 – It is possible to omit the "0b" when the number of bit can be distinctly understood from a sentence.
- "_N" is added to the end of signal names to indicate low active signals.
- It is called "assert" that a signal moves to its active level, "deassert" to its inactive level.
- When two or more signal names are referred, they are described like as [m: n].
  Example: S[3: 0] shows four signal names S3, S2, S1 and S0 together.
- The characters surrounded by ***[ ]*** defines the register.
  Example: ***[ABCD]***
- "n" substitutes suffix number of two or more same kind of registers, fields, and bit names.
  Example: ***[XYZ1]***, ***[XYZ2]***, ***[XYZ3]*** → ***[XYZn]***
- "x" substitutes suffix number or character of units and channels in the Register List.
  In case of unit, "x" means A, B, and C ...
  Example: ***[ADACR0]***, ***[ADBCR0]***, ***[ADCCR0]*** → ***[ADxCR0]***
  In case of channel, "x" means 0, 1, and 2 ...
  Example: ***[T32A0RUNA]***, ***[T32A1RUNA]***, ***[T32A2RUNA]*** → ***[T32AxRUNA]***
- The bit range of a register is written like as [m: n].
  Example: Bit[3: 0] expresses the range of bit 3 to 0.
- The configuration value of a register is expressed by either the hexadecimal number or the binary number.
  Example: ***[ABCD]***<EFG> =0x01 (hexadecimal), ***[XYZn]***<VW> =1 (binary)
- Word and Byte represent the following bit length.
  Byte: 8 bits
  Half word: 16 bits
  Word: 32 bits
  Double word: 64 bits
- Properties of each bit in a register are expressed as follows:
  R: Read only
  W: Write only
  R/W: Read and Write are possible
- Unless otherwise specified, register access supports only word access.
- The register defined as reserved must not be rewritten. Moreover, do not use the read value.
- The value read from the bit having default value of "—" is unknown.
- When a register containing both of writable bits and read-only bits is written, read-only bits should be written with their default value, in the cases that default is "—", follow the definition of each register.
- Reserved bits of the Write-only register should be written with their default value.
  In the cases that default is "—", follow the definition of each register.
- Do not use read-modified-write processing to the register of a definition which is different by writing and read out.

The Flash memory uses the Super Flash® technology under license from Silicon Storage Technology, Inc.
Super Flash® is registered trademark of Silicon Storage Technology, Inc.

All other company names, product names, and service names mentioned herein may be trademarks of their respective companies.

Terms and Abbreviations

Some of abbreviations used in this document are as follows:

ADC	Analog to Digital Converter
A-ENC32	Advanced Encoder input Circuit (32-bit)
A-VE	Advanced Vector Engine
IA	Interrupt control register A
IB	Interrupt control register B
IMCxx	Interrupt Mode Control xx
IMNFLGNMI	Interrupt Monitor Flag NMI
IMNFLGx	Interrupt Monitor Flag x
INT	Interrupt
INTIF	Interrupt Interface Logic
ISR	Interrupt Service Routine
LVD	Voltage Detection Circuit
NICxx	Non-Maskable Interrupt Control xx
NVIC	Nested Vectored Interrupt Controller
OFD	Oscillation Frequency Detector
PMD+	Programmable Motor Control Circuit Plus
POR	Power On Reset Circuit
RAMP	RAM parity
RLMRSTFLGx	RLM Reset Flag x
SIWDT	Clock Selective Watchdog Timer
TSPI	Toshiba Serial Peripheral Interface
T32A	32-bit Timer Event Counter
UART	Universal Asynchronous Receiver Transmitter

Exceptions have close relation to the CPU core. Refer to "Arm® Cortex®-M4 Processor Technical Reference Manua" if needed.

1. Outlines

Exceptions require CPU to suspend the currently executing process, and to start another process.

There are two types of exceptions: those that are generated when some error condition occurs or when an instruction to generate an exception is executed; and those that are generated by hardware, such as an interrupt request signal from an external pin or peripheral function.

All exceptions are handled by the Nested Vectored Interrupt Controller (NVIC) in the CPU according to the respective priority levels. When an exception occurs, the CPU stores the current state to the stack and branches to the corresponding interrupt service routine (ISR). Upon completion of the ISR, the information stored to the stack is automatically restored.

1.1. Exception Types

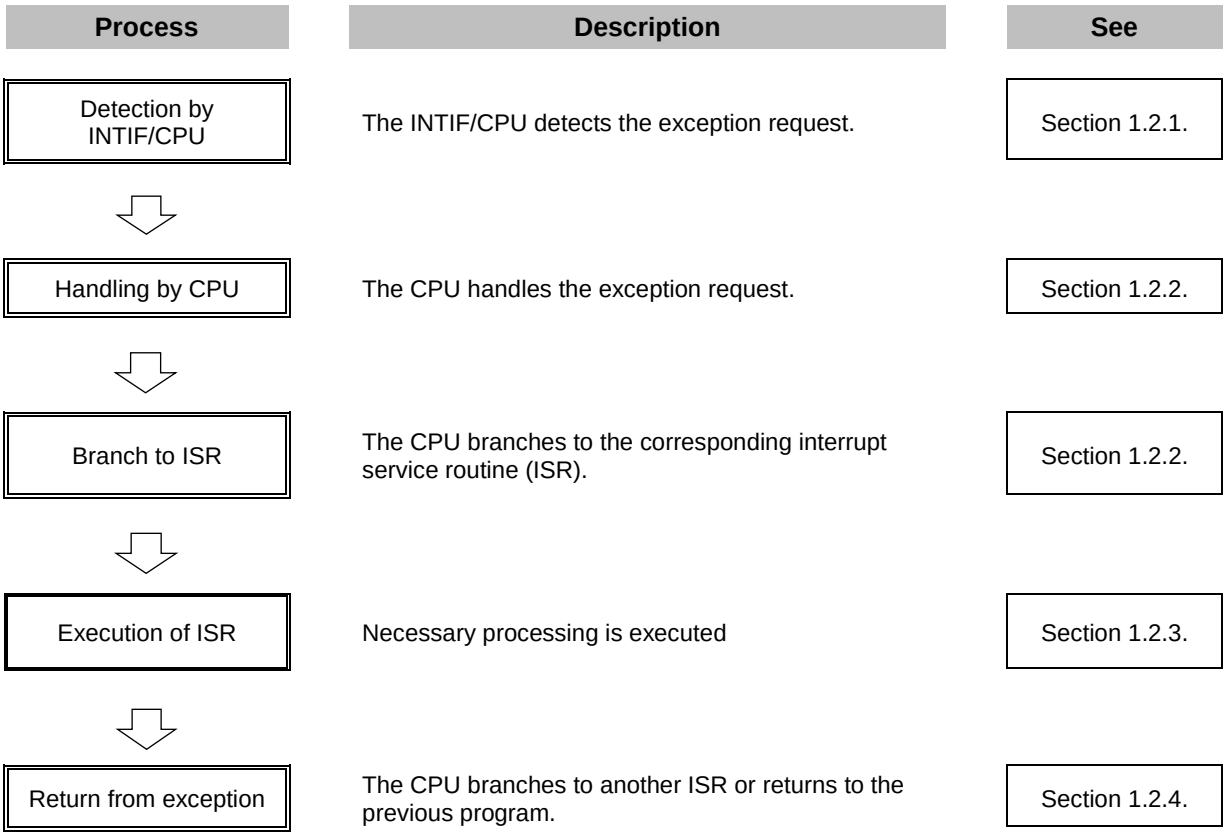
The following types of exceptions exist in this product.
For detailed descriptions on each exception, refer to "Arm Cortex-M4 Processor Technical Reference Manual".

- Reset
- Non-Maskable Interrupt(NMI)
- Hard Fault
- Memory Management
- Bus Fault
- Usage Fault
- SVCall (Supervisor Call)
- Debug Monitor
- PendSV
- SysTick
- External Interrupt

1.2. Exception Handling Flowchart

The following shows how an exception/interrupt is handled. In the following descriptions, exception handling by hardware and that by software are explained.

Each step is described later in this reference manual.



1.2.1. Exception Request and Detection

(1) Exception Occurrence

Exception sources include instruction execution by the CPU, memory accesses, and interrupt requests from external interrupt pins or peripheral functions.

An exception by the instruction execution occurs when the CPU executes an instruction that causes an exception or when an error condition occurs during instruction execution.

An exception also occurs by an instruction fetch from the Execute Never region or an access violation to the Fault region.

The request of the exception by the external interrupt pin or the peripheral function occurs by each functional factor. Regarding to interrupt which connected via INTIF, the setup of the interrupt control register is also needed. For details, refer to the chapter, "4.Interrupts".

(2) Exception Detection

If multiple exceptions occur simultaneously, the CPU takes the exception with the highest priority.

Table 1.1 shows the priority of exceptions. "Configurable" means that you can assign a priority level to that exception. Memory Management, Bus Fault and Usage Fault exceptions can be enabled or disabled. If a disabled exception occurs, it is handled as Hard Fault.

Table 1.1 Exception Types and Priority

Exception Type	Priority	Description	Offset
Reset	-3(highest)	Reset pin, POR reset, OFD reset, SIWDT reset, LVD reset, SYSRESETREQ reset, LOCKUP reset	0x00
Non-Maskable Interrupt	-2	SIWDT, LVD	0x08
Hard Fault	-1	Fault that cannot activate because a higher-priority fault is being handled or it is disabled	0x0C
Memory Management	Configurable	Exception from the Memory Protection Unit (MPU) Instruction fetch from the Execute Never (XN) region	0x10
Bus Fault	Configurable	Access violation to the Hard Fault region of the memory map	0x14
Usage Fault	Configurable	Undefined instruction execution or other faults related to instruction execution	0x18
Reserved	—	—	0x1C - 0x28
SVCall	Configurable	System service call with SVC instruction	0x2C
Debug Monitor	Configurable	Debug monitor when the CPU is not faulting	0x30
Reserved	—	—	0x34
PendSV	Configurable	Pending system service request	0x38
SysTick	Configurable	Notification from system timer	0x3C
External Interrupt	Configurable	External interrupt pin or peripheral function (Note)	0x40

Note: External interrupts have different sources and numbers in each product. For details, see "4.4.List of Interrupt Sources".

(3) Priority Setting

- Priority Level

  The external interrupt priority is set to the Interrupt Priority Register and other exceptions are set to <PRI_n> bit in the System Handler Priority Register.
  The configuration <PRI_n> can be changed, and the number of bits required for setting the priority varies from 3 bits to 8 bits depending on products. Thus, the range of priority values you can specify is different depending on products.
  In the case of 8-bit configuration, the priority can be configured in the range from 0 to 255. The highest priority is "0". If multiple elements with the same priority exist, the smaller the number, the higher the priority becomes.

  <PRI_n[7:0]> bit is defined as the upper 4-bit configuration with TMPM4L Group(1) products. The priority can be configured in the range from 0 to 15.

- Priority Grouping

  The priority group can be split into groups. By setting the <PRIGROUP> of the Application Interrupt and Reset Control Register, <PRI_n> can be divided into the pre-emption priority and the sub priority. A priority is compared with the pre-emption priority. If the priority is the same as the pre-emption priority, then it is compared with the sub priority. If the sub priority is the same as the priority, the

smaller the exception number, the higher the priority.
The Table 1.2 shows the priority group setting. The pre-emption priority and the sub priority in the table are the number in the case that <PRI_n> is defined as an 8-bit configuration.

Table 1.2 Priority grouping setting

<PRIGROUP[2:0]> setting	<PRI_n[7:0]>		Number of pre-emption priorities	Number of sub priorities
	Pre-emption field	Sub priority field		
000	[7:1]	[0]	128	2
001	[7:2]	[1:0]	64	4
010	[7:3]	[2:0]	32	8
011	[7:4]	[3:0]	16	16
100	[7:5]	[4:0]	8	32
101	[7:6]	[5:0]	4	64
110	[7]	[6:0]	2	128
111	None	[7:0]	1	256

Note: If the configuration of <PRI_n> is less than 8 bits, the lower bit is "0". For the example in the case of 4-bit configuration, the priority is set as <PRI_n[7:4]> and <PRI_n[3:0]> is "0000".

1.2.2. Exception Handling and Branch to Interrupt Service Routine (Pre-emption)

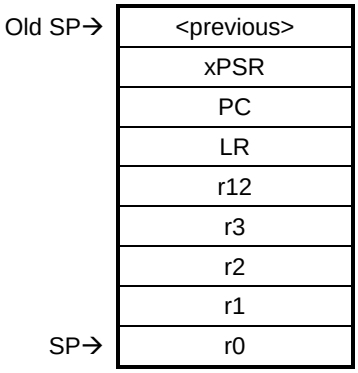
When an exception occurs, the CPU suspends the currently executing process and branches to the interrupt service routine. This is called "pre-emption".

(1) Stacking

When the CPU detects an exception, it pushes the contents of the following eight registers to the stack in the following order:

1. Program Counter (PC)
2. Program Status Register (xPSR)
3. r0 to r3
4. r12
5. Link Register (LR)

The SP is decremented by eight words by the completion of the stack push. The following shows the state of the stack after the register contents have been pushed.



(2) Fetching an ISR

The CPU performs the evacuation of the register. In addition, the CPU performs instruction fetch of the interrupt service routine at the same time.

Prepare a vector table containing the top addresses of ISRs for each exception. After reset, the vector table is located at address 0x00000000 in the Code area. By setting the Vector Table Offset Register, you can place the vector table at any address in the Code or SRAM space.

The vector table should also contain the initial value of the main stack.

(3) Late-arriving

If the CPU detects a higher priority exception before executing the ISR for a previous exception, the CPU handles the higher priority exception first. This is called "late-arriving".

A late-arriving exception causes the CPU to fetch a new vector address for branching to the corresponding ISR, but the CPU does not newly push the register contents to the stack.

(4) Vector Table

The vector table is configured as shown below.

You must always set the first four words (stack top address, reset ISR address, NMI ISR address, and Hard Fault ISR address). Set ISR addresses for other exceptions if necessary.

For other exceptions, you may prepare the ISR addresses if necessary.

Offset	Exception	Contents	Setting
0x00	Reset	Initial value of the main stack	Required
0x04	Reset	ISR address	Required
0x08	Non-Maskable Interrupt	ISR address	Required
0x0C	Hard Fault	ISR address	Required
0x10	Memory Management	ISR address	Optional
0x14	Bus Fault	ISR address	Optional
0x18	Usage Fault	ISR address	Optional
0x1C to 0x28	Reserved	—	—
0x2C	SVCall	ISR address	Optional
0x30	Debug Monitor	ISR address	Optional
0x34	Reserved	—	—
0x38	PendSV	ISR address	Optional
0x3C	SysTick	ISR address	Optional
0x40	External Interrupt	ISR address	Optional

1.2.3. Executing an ISR

An ISR performs necessary processing for the corresponding exception. ISRs must be prepared by the user.

An ISR may need to include code for clearing the interrupt request so that the same interrupt will not occur again upon return to normal program execution.
For details about interrupt handling, see "4.Interrupts".

If a higher priority exception occurs during ISR execution for the current exception, the CPU abandons the currently executing ISR and services the newly detected exception.

1.2.4. Exception Exit

(1) Execution after Returning from ISR

When returning from an ISR, the CPU takes one of the following actions:

- Tail-chaining
  If a pending exception exists and there are no stacked exceptions or the pending exception has higher priority than all stacked exceptions, the CPU returns to the ISR of the pending exception. In this case, the CPU skips the pop of eight registers and push of eight registers when exiting one ISR and entering another. This is called "tail-chaining".

- Returning to the last stacked ISR
  If there are no pending exceptions or if the highest priority stacked exception is of higher priority than the highest priority pending exception, the CPU returns to the last stacked ISR.

- Returning to the previous program
  If there are no pending or stacked exceptions, the CPU returns to the previous program.

(2) Exception Exit Sequence

When returning from an ISR, the CPU performs the following operations:

- Pop eight registers
  Pop eight registers (PC, xPSR, r0 to r3, r12, and LR) from the stack and adjust the SP.

- Load current active interrupt number
  Loads the current active interrupt number from the stacked xPSR. The CPU uses this to track which interrupt to return to.

- Select SP
  If returning to an exception (Handler Mode), SP is SP_main. If returning to Thread Mode, SP can be SP_main or SP_process.

2. Reset Exception

Reset exceptions are generated from the following seven sources.
Use the ***[RLMRSTFLGn]*** of the Reset Flag Register to identify the source of a reset.

- Reset exception by external reset pin
  A reset exception occurs when an external reset pin changes from "Low" level to "High" level.

- Reset exception by POR
  A reset exception occurs by POR. For details, refer to reference manual "Power supply and Reset operation"

- Reset exception by OFD
  A reset exception occurs by OFD. For details, refer to reference manual "Oscillation Frequency Detector".

- Reset exception by SIWDT
  The watchdog timer (SIWDT) has a reset generating feature. For details, refer to reference manual "Clock Selective Watchdog Timer".

- Reset exception by LVD
  The LVD has a reset generating feature. For details, refer to reference manual "Voltage Detector Circuit".

- Reset exception by <SYSRESETREQ>
  A reset can be generated by setting the <SYSRESETREQ> bit in the NVIC's Application Interrupt and Reset Control Register.

- Reset exception by LOCKUP signal
  A reset can be generated by the LOCKUP signal which can be output from the CPU when the un-recoverable interrupt occurs. For details on the LOCKUP signal, please refer to "Arm Cortex-M4 Processor Technical Reference Manual".

3. SysTick

SysTick provides interrupt features using the CPU's system timer.

When you set a value in the SysTick Reload Value Register and enable the SysTick features in the SysTick Control and Status Register, the counter loads with the value set in the Reload Value Register and begins counting down. When the counter reaches "0", a SysTick exception occurs. You may be pending exceptions and use a flag to know when the timer reaches "0".

4. Interrupts

This chapter explains the route from which a factor and an interrupt request are transmitted, and a required setup.

4.1. Non-Maskable Interrupt (NMI)

Non-maskable interrupts are generated from the following two sources.

- Non-maskable interrupt by SIWDT
  The watchdog timer (SIWDT) has a non-maskable interrupt generating feature. For details of SIWDT, refer to reference manual "Clock Selective Watchdog Timer".

- Non-maskable interrupt by LVD
  The LVD has a non-maskable interrupt generating feature. For details of LVD, refer to reference manual "Voltage Detector Circuit".

4.2. Maskable Interrupt

Please refer to interrupt control register A / interrupt control register B of the "4.4.List of Interrupt Sources" for the factor of maskable interrupt.

4.3. Interrupt Request

The CPU is notified of interrupt requests by the interrupt signal from each interrupt source. It sets priority on interrupts and handles an interrupt request with the highest priority.

4.3.1. Interrupt Route

The interrupt is available for the cancellation from a low power consumption mode, and a route varies according to a factor.

Figure 4.1 shows the interrupt transfer route diagram and Table 4.1 shows the explanation of each interrupt transfer route.

- The interrupt that is releasable from IDLE, STOP1 mode
  It has two interrupt transfer routes via INTIF that can release IDLE and STOP 1 mode.

  (1) It is controlled by interrupt control register A in INTIF, and notified to CPU. (Route A, B, C)
  (2) It is controlled by interrupt control register B in INTIF, and notified to CPU. (Route D, E, F)

- The interrupt that is releasable from IDLE mode
  Although some factors of interrupt which can be released of IDLE mode are controlled by the interrupt control register B via INTIF (Route G), other factors are notified to CPU directly (Route H) not passing through INTIF.

When the interrupt factor that went by way of an interrupt regardless of low power consumption mode cancellation is used, setting of interrupt control register A or B is necessary.
Please refer to the chapter of "Release sources of a Low Power Consumption mode" of a reference manual "Clock

Control and Operation Mode" for the details of a low power consumption mode release factor.

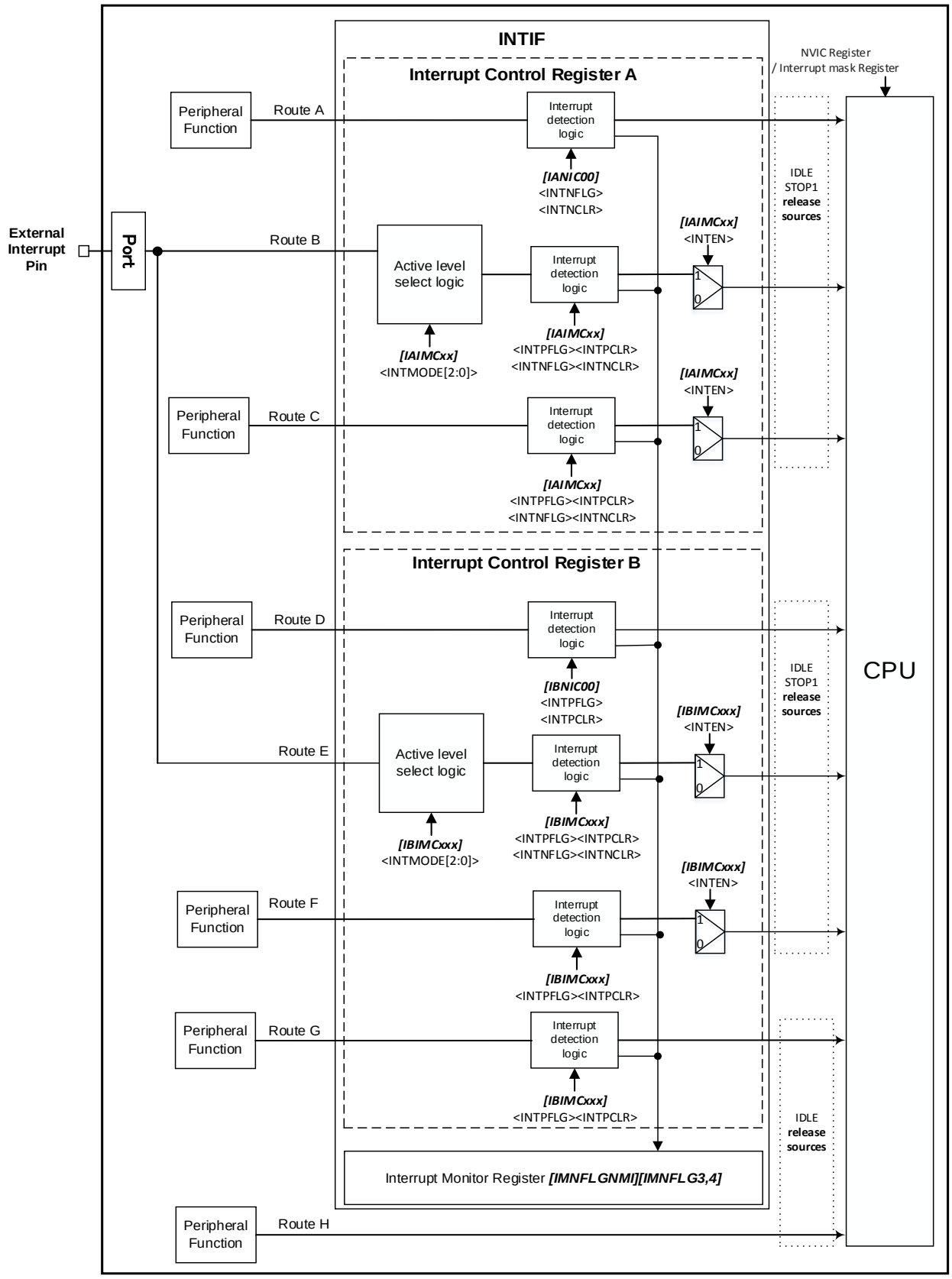


Figure 4.1 Interrupt transfer route Diagram

Table 4.1 Explanation of each interrupt transfer route

Route	Interrupt No.	Interrupt Request	Route Description
A	—	LVD interrupt	This route is NMI interrupt. It is a route inputted into CPU via INTIF. An interrupt release setup is carried out by the interrupt control register A (***[IANIC00]***).
B	—	—	The interrupt request of a port is a route inputted into CPU via INTIF. Permission/prohibition of selection of an Interrupt detection level, interrupt release, and an interrupt request are set up by the interrupt control register A (***[IAIMCxx]***) for every factor.
C	—	—	It is a route inputted into CPU via INTIF. Permission/prohibition of interrupt release and an interrupt request are set up by the interrupt control register A (***[IAIMCxx]***) for every factor.
D	—	WDT interrupt	It is mask impossible interrupt. It is a route inputted into CPU via INTIF. An interrupt release setup is carried out by the interrupt control register B (***[IBNIC00]***).
E	0 to 7	External interrupts (00 to 07)	The interrupt request of a port is a route inputted into CPU via INTIF. Permission/prohibition of selection of an Interrupt detection level, interrupt release, and an interrupt request are set up by the interrupt control register B (***[IBIMCxxx]***) for every factor.
F	8 to 23	T32A timer match, overflow, underflow interrupt T32A capture interrupt	It is a route inputted into CPU via INTIF. Permission/prohibition of interrupt release and an interrupt request are set up by the interrupt control register B (***[IBIMCxxx]***) for every factor.
G	—	—	It is a route inputted into CPU via INTIF. An interrupt release setup is carried out by the interrupt control register B (***[IBIMCxxx]***) for every factor.
H	24 to 58	Other interrupts (Note)	It is a route as which an interrupt request is directly inputted into CPU not passing through INTIF.

Note: For the details of other interrupts, refer to "4.4.List of Interrupt Sources".

4.3.2. Interrupt Request Generation

An interrupt request is generated from an external interrupt pin or peripheral function which are assigned as interrupt request sources, or by setting the relevant bit of NVIC's Interrupt Set-Pending Register for interrupt request source.

- Interrupt from external interrupt pin
  Set the port control register so that the external pin can perform as an interrupt function pin.

- Interrupt from peripheral function
  Set the peripheral function to make it possible to output interrupt requests.
  See the chapter of each peripheral function for details.

- By setting Interrupt Set-Pending Register (forced pending)
  An interrupt request can be forced to be generated by setting the relevant bit of the Interrupt Set-Pending Register of NVIC.

CPU will recognize the "High" level of the interrupt request as an interrupt.

4.3.3. Monitor of the Interrupt Request

INTIF has the interrupt monitor flags. It can know that the interrupt request has occurred by monitoring the flag. If one request source is representing several interrupt requests, Interrupt Monitor Register can be used to identify the actual interrupt request source.

For detail, please refer to "4.4.List of Interrupt Sources".

4.3.4. Transmission of Interrupt Request

An interrupt request which is not passing through the Interrupt Control Register will be directly input to the CPU. The interrupts connected to the CPU through INTIF, which are used as interrupt request sources for releasing the low power consumption mode, will need proper setting of the Interrupt Control Register in INTIF. A "High" level interrupt signal will be sent to the CPU, when the interrupt is used to release the low power consumption mode.

Please setup an interrupt detection level and interrupt detection enable/disable by INTIF.

By the way, please be cautious about external interrupt pin as in the next section.

4.3.5. Precautions When Using External Interrupt Pins

When you use external interrupt, please care about the following points so that an unexpected interrupt does not occur.

If input is disabled (***[PxIE]***<PxmIE>=0), inputs from external interrupt pins are "Low" level. When the <INTMODE> bit of Interrupt Control Register A (***[IAIMCxx]***) is "Low" level, then input signals from the external interrupt pins are sent to the CPU as is. Since the CPU recognizes "Low" level input as an interrupt, interrupts occur if corresponding interrupts are enabled by the CPU as inputs are being disabled.

To use the external interrupt without setting it as a trigger to release the low power consumption mode, set the interrupt pin input as "High" level and enable it. Then, enable interrupts on the CPU.

4.4. List of Interrupt Sources

Table 4.2 shows the list of interrupt sources of non-maskable interrupts. The setting for clearing the NMI sources can be done by Interrupt Control Registers A and B.

Table 4.2 List of Interrupt Sources (Non-Maskable Interrupt)

Interrupt Source	Interrupt Request	Interrupt Control Register	Interrupt Monitor Register
INTLVD	LVD interrupt	***[IANIC00]***	***[IMNFLGNMI]*** <INT000FLG>
INTWDT0	WDT interrupt	***[IBNIC00]***	***[IMNFLGNMI]*** <INT016FLG>

Table 4.3 shows the list of interrupt sources of Interrupt Control Register B. These interrupt sources can be the sources for releasing the low power consumption mode. The Interrupt Control Register B will perform several setting for detecting the release of the low power consumption mode, and interrupt detection enable/disable.

Table 4.3 List of Interrupt Sources (Interrupt Control Register B(1))

Interrupt No.	Interrupt Source	Interrupt Request	Interrupt Control Register	Interrupt Monitor Register
0	INT00	External interrupt 00	***[IBIMC000]***	***[IMNFLG3]*** <INT096FLG>
1	INT01	External interrupt 01	***[IBIMC001]***	***[IMNFLG3]*** <INT097FLG>
2	INT02	External interrupt 02	***[IBIMC002]***	***[IMNFLG3]*** <INT098FLG>
3	INT03	External interrupt 03	***[IBIMC003]***	***[IMNFLG3]*** <INT099FLG>
4	INT04	External interrupt 04	***[IBIMC004]***	***[IMNFLG3]*** <INT100FLG>
5	INT05	External interrupt 05	***[IBIMC005]***	***[IMNFLG3]*** <INT101FLG>
6	INT06	External interrupt 06	***[IBIMC006]***	***[IMNFLG3]*** <INT102FLG>
7	INT07	External interrupt 07	***[IBIMC007]***	***[IMNFLG3]*** <INT103FLG>

The factor list of the interrupt control registers B is shown in Table 4.4 to Table 4.6. A part of interrupt sets up interrupt detection enable/disable by the interrupt control register B.

Table 4.4 List of Interrupt Sources (Interrupt Control Register B(2)) (1/3)

Interrupt No.	Interrupt Source	Interrupt Request	Interrupt Control Register	Interrupt Monitor Register
8	INTT32A00_AT_CT	T32A ch0 timer A match,overflow, and underflow	***[IBIMC008]***	***[IMNFLG3]*** <INT104FLG>
		T32A ch0 timer C match,overflow, and underflow	***[IBIMC009]***	***[IMNFLG3]*** <INT105FLG>
9	INTT32A00_A01_C0	T32A ch0 timer A capture 0	***[IBIMC010]***	***[IMNFLG3]*** <INT106FLG>
		T32A ch0 timer A capture 1	***[IBIMC011]***	***[IMNFLG3]*** <INT107FLG>
		T32A ch0 timer C capture 0	***[IBIMC012]***	***[IMNFLG3]*** <INT108FLG>
10	INTT32A00_BT_C1	T32A ch0 timer B match,overflow, and underflow	***[IBIMC013]***	***[IMNFLG3]*** <INT109FLG>
		T32A ch0 timer C capture 1	***[IBIMC014]***	***[IMNFLG3]*** <INT110FLG>
11	INTT32A00_B01	T32A ch0 timer B capture 0	***[IBIMC015]***	***[IMNFLG3]*** <INT111FLG>
		T32A ch0 timer B capture 1	***[IBIMC016]***	***[IMNFLG3]*** <INT112FLG>
12	INTT32A01_AT_CT	T32A ch1 timer A match,overflow, and underflow	***[IBIMC017]***	***[IMNFLG3]*** <INT113FLG>
		T32A ch1 timer C match,overflow, and underflow	***[IBIMC018]***	***[IMNFLG3]*** <INT114FLG>
13	INTT32A01_A01_C0	T32A ch1 timer A capture 0	***[IBIMC019]***	***[IMNFLG3]*** <INT115FLG>
		T32A ch1 timer A capture 1	***[IBIMC020]***	***[IMNFLG3]*** <INT116FLG>
		T32A ch1 timer C capture 0	***[IBIMC021]***	***[IMNFLG3]*** <INT117FLG>
14	INTT32A01_BT_C1	T32A ch1 timer B match,overflow, and underflow	***[IBIMC022]***	***[IMNFLG3]*** <INT118FLG>
		T32A ch1 timer C capture 1	***[IBIMC023]***	***[IMNFLG3]*** <INT119FLG>
15	INTT32A01_B01	T32A ch1 timer B capture 0	***[IBIMC024]***	***[IMNFLG3]*** <INT120FLG>
		T32A ch1 timer B capture 1	***[IBIMC025]***	***[IMNFLG3]*** <INT121FLG>
16	INTT32A02_AT_CT	T32A ch2 timer A match,overflow, and underflow	***[IBIMC026]***	***[IMNFLG3]*** <INT122FLG>
		T32A ch2 timer C match,overflow, and underflow	***[IBIMC027]***	***[IMNFLG3]*** <INT123FLG>
17	INTT32A02_A01_C0	T32A ch2 timer A capture 0	***[IBIMC028]***	***[IMNFLG3]*** <INT124FLG>
		T32A ch2 timer A capture 1	***[IBIMC029]***	***[IMNFLG3]*** <INT125FLG>
		T32A ch2 timer C capture 0	***[IBIMC030]***	***[IMNFLG3]*** <INT126FLG>
18	INTT32A02_BT_C1	T32A ch2 timer B match,overflow, and underflow	***[IBIMC031]***	***[IMNFLG3]*** <INT127FLG>
		T32A ch2 timer C capture 1	***[IBIMC032]***	***[IMNFLG4]*** <INT128FLG>
19	INTT32A02_B01	T32A ch2 timer B capture 0	***[IBIMC033]***	***[IMNFLG4]*** <INT129FLG>
		T32A ch2 timer B capture 1	***[IBIMC034]***	***[IMNFLG4]*** <INT130FLG>

Table 4.5 List of Interrupt Sources (Interrupt Control Register B(2)) (2/3)

Interrupt No.	Interrupt Source	Interrupt Request	Interrupt Control Register	Interrupt Monitor Register
20	INTT32A03_AT_CT	T32A ch3 timer A match,overflow, and underflow	***[IBIMC035]***	***[IMNFLG4]*** <INT131FLG>
		T32A ch3 timer C match,overflow, and underflow	***[IBIMC036]***	***[IMNFLG4]*** <INT132FLG>
21	INTT32A03_A01_C0	T32A ch3 timer A capture 0	***[IBIMC037]***	***[IMNFLG4]*** <INT133FLG>
		T32A ch3 timer A capture 1	***[IBIMC038]***	***[IMNFLG4]*** <INT134FLG>
		T32A ch3 timer C capture 0	***[IBIMC039]***	***[IMNFLG4]*** <INT135FLG>
22	INTT32A03_BT_C1	T32A ch3 timer B match,overflow, and underflow	***[IBIMC040]***	***[IMNFLG4]*** <INT136FLG>
		T32A ch3 timer C capture 1	***[IBIMC041]***	***[IMNFLG4]*** <INT137FLG>
23	INTT32A03_B01	T32A ch3 timer B capture 0	***[IBIMC042]***	***[IMNFLG4]*** <INT138FLG>
		T32A ch3 timer B capture 1	***[IBIMC043]***	***[IMNFLG4]*** <INT139FLG>
24	INTVCN0	A-VE ch0 Schedule end interrupt		
25	INTVCT0	A-VE ch0 Task end interrupt		
26	INTEMG0	PMD+ ch0 EMG interrupt		
27	INTOVV0	PMD+ ch0 OVV interrupt		
28	INTPWM0	PMD+ ch0 PWM interrupt		
29	INTENC00	A-ENC32 ch0 Encoder input interrupt 0		
30	INTENC01	A-ENC32 ch0 Encoder input interrupt 1		
31	—	—		
32	INTADAPDA	ADC unitA PMD trigger interrupt A		
33	INTADAPDB	ADC unitA PMD trigger interrupt B		
34	INTADACP0	ADC unitA Monitor function 0 interrupt		
35	INTADACP1	ADC unitA Monitor function 1 interrupt		
36	INTADATRG	ADC unitA General purpose trigger interrupt		
37	INTADASGL	ADC unitA Single conversion interrupt		
38	INTADACNT	ADC unitA Continuous conversion interrupt		
39	INTT0RX	TSPI ch0 Receive interrupt		
40	INTT0TX	TSPI ch0 Transmit interrupt		
41	INTT0ERR	TSPI ch0 Error interrupt		
42	INTT1RX	TSPI ch1 Receive interrupt		
43	INTT1TX	TSPI ch1 Transmit interrupt		
44	INTT1ERR	TSPI ch1 Error interrupt		

Table 4.6 List of Interrupt Sources (Interrupt Control Register B(2)) (3/3)

Interrupt No.	Interrupt Source	Interrupt Request	Interrupt Control Register	Interrupt Monitor Register
45	INTT2RX	TSPI ch2 Receive interrupt		
46	INTT2TX	TSPI ch2 Transmit interrupt		
47	INTT2ERR	TSPI ch2 Error interrupt		
48	INTUART0RX	UART ch0 Reception interrupt		
49	INTUART0TX	UART ch0 Transmission interrupt		
50	INTUART0ERR	UART ch0 Error interrupt		
51	INTUART1RX	UART ch1 Reception interrupt		
52	INTUART1TX	UART ch1 Transmission interrupt		
53	INTUART1ERR	UART ch1 Error interrupt		
54	INTUART2RX	UART ch2 Reception interrupt		
55	INTUART2TX	UART ch2 Transmission interrupt		
56	INTUART2ERR	UART ch2 Error interrupt		
57	INTPARI0	RAMP ch0 RAM parity interrupt		
58	INTFLCRDY	Code FLASH Ready interrupt		

4.5. Interrupt detection level

When using interrupt via INTIF, interrupt detection level ("Low" level / "High" level / Rising edge / Falling edge) can be selected by interrupt control register A or B. The detected interrupt is output to the CPU with a "High" level signal.

The interrupt signals which are directly transmitted from the various peripheral functions to the CPU, a "High" pulse is output to the CPU as an interrupt request.

The CPU detects the interrupt signal "High" level to be an interrupt factor.

4.5.1. Precautions When Releasing the Low Power Consumption Mode

Following setting should be done when releasing STOP1 mode

- The setup of the interrupt control register B. (***[IBIMCxxx]***)
  Interrupt detection level, Interrupt detection enable/disable

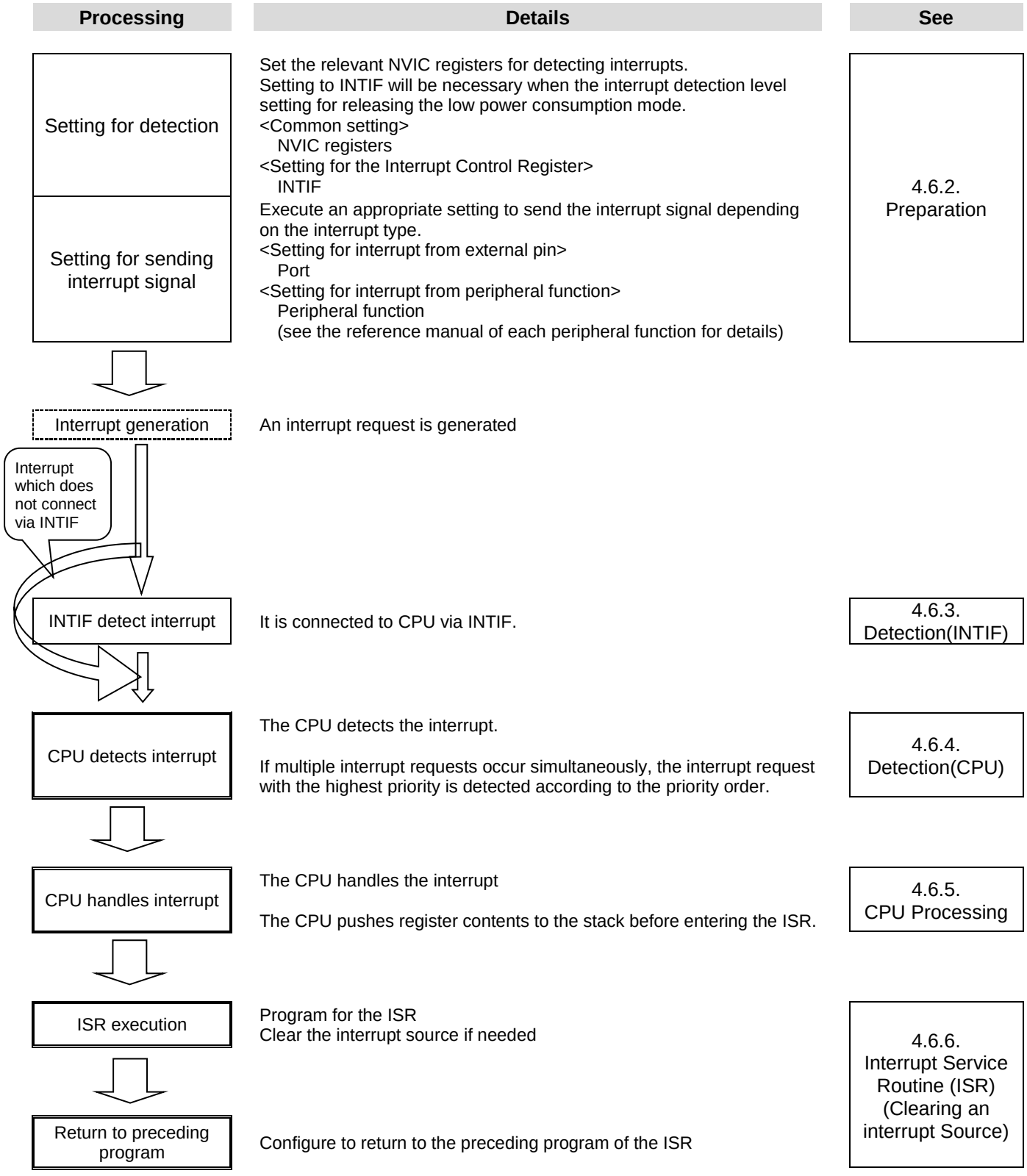
- The setup of the NVIC interrupt enable set register
  enable/disable setup

In order to return to NORMAL mode from STOP1 mode, resume suspended instruction by jumping into interrupt after high speed clock oscillation.

4.6. Interrupt Handling

4.6.1. Flowchart

The following shows how an interrupt is handled.
The flowchart below explains the interrupt handling process by hardware and software.



4.6.2. Preparation

When preparing for an interrupt, you need to pay attention to the order of configuration to avoid any unexpected interrupt on the way.

Initiating an interrupt or changing its configuration must be implemented in the following order basically. First, disable the interrupt by the CPU. Then, configure from the farthest route from the CPU. Finally, enable the interrupt by the CPU.

To configure the INTIF, you must follow the order indicated here not to cause any unexpected interrupt. First, configure the precondition. Secondly, clear the data related to the interrupt in the INTIF and then enable the interrupt.

The following sections are listed in the order of interrupt handling and describe how to configure them.

1. Disabling interrupt by CPU
2. CPU registers setting
3. Preconfiguration (1) (Interrupt from external pin)
4. Preconfiguration (2) (Interrupt from peripheral function)
5. Preconfiguration (3) (Interrupt Set-Pending Register)
6. Configuring the INTIF
7. Enabling interrupt by CPU

(1) Disabling Interrupt by CPU

To make the CPU for not accepting any interrupt, write "1" to the corresponding bit of the ***[PRIMASK]*** Register. All interrupts and exceptions other than non-maskable interrupts and hard faults can be masked.

Use "MSR" instruction to set this register.

Interrupt Mask Register		
[PRIMASK]	←	"1"(interrupt disabled)

Note 1: ***[PRIMASK]*** Register cannot be modified in the user access level

Note 2: If a fault causes when "1" is set to the ***[PRIMASK]*** Register, it is treated as a hard fault.

(2) CPU Registers Setting

You can assign a priority level by writing to <PRI_n> field in an Interrupt Priority Register in the NVIC.

Each interrupt source is provided with eight bits for assigning a priority level from "0" to "255", but the number of bits actually used varies with each product. Priority level "0" is the highest priority level. If multiple sources have the same priority, the smallest-numbered interrupt source has the highest priority.

You can assign grouping priority by using the <PRIGROUP> in the Application Interrupt and Reset Control Register.

NVIC Register		
<PRI_n>	←	"Priority"
<PRIGROUP>	←	"group priority" (This is configurable if required)

Note: "n" indicates the number of the corresponding exceptions/interrupts.
This product uses four bits for assigning a priority level.

(3) Preconfiguration (1) (Interrupt from external pin)

In order to use external interrupt pin, it is necessary to do proper setting to the port function register of the corresponding pin. Setting "1" to ***[PxIE]***<PxmIE> allows the pin to be used as the function pin and the input port.

Port Register		
[PxIE]<PxmIE>	←	"1"

Note: x: port number, m: corresponding bit of function register number. Be careful not to enable interrupts that are not used when performing interrupt setting. Also be aware of the description of "4.3.5.Precautions When Using External Interrupt Pins

(4) Preconfiguration (2) (Interrupt from peripheral function)

The setting varies depending on the peripheral function to be used. See the reference manual of each peripheral function for details

(5) Preconfiguration (3) (Interrupt from Set-Pending Register)

To generate an interrupt by using the Interrupt Set-Pending Register, set "1" to the corresponding bit of this register.

NVIC Register		
<SETPEND>	←	"1"

Note: <SETPEND>: corresponding bit

(6) Configuring the INTIF

The interrupt by way of INTIF sets the interrupt detection enable in interrupt control registers.

The ***[IANIC00]***/***[IBNIC00]***/***[IBIMCxxx]*** registers are capable of configuring each interrupt source. Before enabling an interrupt detection, clear the interrupt request having active level in order to avoid unexpected interrupt.

For details of the interrupt control register, refer to the following.

Interrupt Control Register		
[IBIMCxxx]<INTMODE>	←	Value corresponding to the interrupt to be used (Only for the interrupt having interrupt detection level)
[IANIC00]<INTNCLR> ***[IBNIC00]***<INTPCLR> ***[IBIMCxxx]***<INTPCLR><INTNCLR>	←	Interrupt request clear to use
[IBIMCxxx]<INTEN>	←	"1" (Interrupt detection enabled)

Note: xxx: number specific to the interrupt request

(7) Enabling Interrupt by CPU

Enable the interrupt by the CPU as shown below.

Clear the suspended interrupt in the Interrupt Clear-Pending Register. Enable the intended interrupt with the Interrupt Set-Enable Register. Each bit of the register is assigned to a single interrupt source.

Writing "1" to the corresponding bit of the Interrupt Clear-Pending Register clears the suspended interrupt. Writing "1" to the corresponding bit of the Interrupt Set-Enable Register enables the intended interrupt.

To generate interrupts in the Interrupt Set-Pending Register setting, factors to trigger interrupts are lost if pending interrupts are cleared. Thus, this operation is not necessary.

At the end, ***[PRIMASK]*** register is zero cleared.

NVIC Register		
<CLRPEND>	←	"1"
<SETENA>	←	"1"
Interrupt Mask Register		
[PRIMASK]	←	"0"

Note 1: <CLRPEND>,<SETENA>: corresponding bit

Note 2: ***[PRIMASK]*** Register cannot be modified by the user access level;

4.6.3. Detection(INTIF)

When the INTIF detects an interrupt request, it sends the interrupt signal in "High" level to the CPU.

INTIF has the functions of the interrupt detection level selection logic, the functions of interrupt detection logic, and the function of the interrupt detection enable/disable. Each function of INTIF is set up the Interrupt Control Register A or B.

It keeps sending the interrupt signal in "High" level to the CPU until the <Detection flag> is cleared in the Interrupt Control Register. If the ISR is exited without clearing the flag, the same interrupt will be detected again when normal operation is resumed. Thus, be sure to clear each <Detection flag> in the ISR. At the same time, the corresponding interrupt monitor register is also cleared.

4.6.4. Detection(CPU)

The CPU detects an interrupt request with the highest priority.

4.6.5. CPU Processing

On detecting an interrupt, the CPU pushes the contents of xPSR, PC, LR, r12, and r3–r0 to the stack then enter the ISR.

4.6.6. Interrupt Service Routine (ISR) (Clearing an interrupt Source)

An ISR requires specific programming according to the application to be used. This section describes what is recommended at the service routine programming and how the source is cleared.

(1) Process in the Interrupt Service Routine

An ISR normally pushes register contents to the stack and handles an interrupt as required.

The Cortex-M4 processor with FPU automatically pushes the contents of xPSR, PC, LR, r12, and r3–r0 to the stack. No extra programming is required for them.

Push the contents of other registers if needed.

Interrupt requests with higher priority and exceptions such as NMI are accepted even when an ISR is being executed. We recommend you to push the contents of general-purpose registers that might be rewritten.

(2) Clearing an Interrupt Source

Some interrupt requests have to be cleared with the Interrupt Control Register.

If an interrupt detection level is set as level-sensitive, an interrupt request continues to exist until it is cleared at its source. Therefore, the interrupt source must be cleared. If a factor is withdrawn in level detection, the interrupt request signal from INTIF will be withdrawn automatically.

A factor is withdrawn by clearing the <Detection flag> of the Interrupt Control Register of INTIF in the case of edge detection. When an effective edge occurs again, it is anew recognized as a factor.

Note: After clearing the <Detection flag> of the Interrupt Control Register, please be sure to read the flag which was cleared.

5. Exception/ Interrupt-Related Registers

5.1. Register List

Control Registers and their addresses are as follows;

Interrupt Control Registers A

Peripheral function		Channel/Unit	Base address
Interrupt control register A	IA	—	0x4003E000

Register name		Address (+Base)
Non-Maskable Interrupt A Control Register 00	***[IANIC00]***	0x0000

Note: Byte access is needed for ***[IANIC00]***.

Interrupt Control Registers B

Peripheral function		Channel/Unit	Base address
Interrupt control register B	IB	—	0x40083200

Register name		Address (+Base)
Non-Maskable Interrupt B Control Register 00	***[IBNIC00]***	0x0010
Interrupt B Mode Control Register 000	***[IBIMC000]***	0x0060
Interrupt B Mode Control Register 001	***[IBIMC001]***	0x0061
Interrupt B Mode Control Register 002	***[IBIMC002]***	0x0062
Interrupt B Mode Control Register 003	***[IBIMC003]***	0x0063
Interrupt B Mode Control Register 004	***[IBIMC004]***	0x0064
Interrupt B Mode Control Register 005	***[IBIMC005]***	0x0065
Interrupt B Mode Control Register 006	***[IBIMC006]***	0x0066
Interrupt B Mode Control Register 007	***[IBIMC007]***	0x0067
Interrupt B Mode Control Register 008	***[IBIMC008]***	0x0068
Interrupt B Mode Control Register 009	***[IBIMC009]***	0x0069
Interrupt B Mode Control Register 010	***[IBIMC010]***	0x006A
Interrupt B Mode Control Register 011	***[IBIMC011]***	0x006B
Interrupt B Mode Control Register 012	***[IBIMC012]***	0x006C
Interrupt B Mode Control Register 013	***[IBIMC013]***	0x006D
Interrupt B Mode Control Register 014	***[IBIMC014]***	0x006E
Interrupt B Mode Control Register 015	***[IBIMC015]***	0x006F
Interrupt B Mode Control Register 016	***[IBIMC016]***	0x0070
Interrupt B Mode Control Register 017	***[IBIMC017]***	0x0071
Interrupt B Mode Control Register 018	***[IBIMC018]***	0x0072
Interrupt B Mode Control Register 019	***[IBIMC019]***	0x0073
Interrupt B Mode Control Register 020	***[IBIMC020]***	0x0074
Interrupt B Mode Control Register 021	***[IBIMC021]***	0x0075
Interrupt B Mode Control Register 022	***[IBIMC022]***	0x0076
Interrupt B Mode Control Register 023	***[IBIMC023]***	0x0077
Interrupt B Mode Control Register 024	***[IBIMC024]***	0x0078
Interrupt B Mode Control Register 025	***[IBIMC025]***	0x0079
Interrupt B Mode Control Register 026	***[IBIMC026]***	0x007A
Interrupt B Mode Control Register 027	***[IBIMC027]***	0x007B
Interrupt B Mode Control Register 028	***[IBIMC028]***	0x007C
Interrupt B Mode Control Register 029	***[IBIMC029]***	0x007D
Interrupt B Mode Control Register 030	***[IBIMC030]***	0x007E
Interrupt B Mode Control Register 031	***[IBIMC031]***	0x007F
Interrupt B Mode Control Register 032	***[IBIMC032]***	0x0080
Interrupt B Mode Control Register 033	***[IBIMC033]***	0x0081
Interrupt B Mode Control Register 034	***[IBIMC034]***	0x0082
Interrupt B Mode Control Register 035	***[IBIMC035]***	0x0083
Interrupt B Mode Control Register 036	***[IBIMC036]***	0x0084
Interrupt B Mode Control Register 037	***[IBIMC037]***	0x0085
Interrupt B Mode Control Register 038	***[IBIMC038]***	0x0086
Interrupt B Mode Control Register 039	***[IBIMC039]***	0x0087
Interrupt B Mode Control Register 040	***[IBIMC040]***	0x0088
Interrupt B Mode Control Register 041	***[IBIMC041]***	0x0089
Interrupt B Mode Control Register 042	***[IBIMC042]***	0x008A
Interrupt B Mode Control Register 043	***[IBIMC043]***	0x008B

Note: Byte access is needed for ***[IBNIC00]*** and ***[IBIMCxxx]*** Registers.

Reset Flag Registers

Peripheral function		Channel/Unit	Base address
Reset flag	RLM	—	0x4003E400

Register name		Address (+Base)
Reset Flag Register 0	***[RLMRSTFLG0]***	0x0002
Reset Flag Register 1	***[RLMRSTFLG1]***	0x0003

Note: Byte access is needed for Reset Flag Register

Interrupt Monitor Registers

Peripheral function		Channel/Unit	Base address
Interrupt Monitor	IMN	—	0x40083300

Register name		Address (+Base)
Non-Maskable Interrupt Monitor Flag Register	***[IMNFLGNMI]***	0x0000
Interrupt Monitor Flag Register 3	***[IMNFLG3]***	0x000C
Interrupt Monitor Flag Register 4	***[IMNFLG4]***	0x0010

NVIC Registers

Peripheral function		Channel/Unit	Base address
NVIC Register	—	—	0xE000E000

Register name	Address (+Base)
SysTick Control and Status Register	0x0010
SysTick Reload Value Register	0x0014
SysTick Current Value Register	0x0018
SysTick Calibration Value Register	0x001C
Interrupt Set-Enable Register 0	0x0100
Interrupt Set-Enable Register 1	0x0104
Interrupt Clear-Enable Register 0	0x0180
Interrupt Clear-Enable Register 1	0x0184
Interrupt Set-Pending Register 0	0x0200
Interrupt Set-Pending Register 1	0x0204
Interrupt Clear-Pending Register 0	0x0280
Interrupt Clear-Pending Register 1	0x0284
Interrupt Priority Register	0x0400 to 0x043A
Vector Table Offset Register	0x0D08
Application Interrupt and Reset Control Register	0x0D0C
System Handler Priority Register	0x0D18, 0x0D1C, 0x0D20
System Handler Control and State Register	0x0D24

5.2. Interrupt Control Registers A

5.2.1. *[IANIC00]* (Non-Maskable Interrupt A Control Register 00)

Bit	Bit Symbol	After Reset	Type	Function
7	INTNCLR	0	W	Detection flag clear control 0: — 1: Clear Reading the bit returns "0"
6	—	0	R	Read as "0"
5	INTNFLG	0	R	Detection flag 0: Not detected 1: Detected
4:0	—	00101	R	Read as "00101"

5.3. Interrupt Control Registers B

5.3.1. *[IBNIC00]* (Non-Maskable Interrupt B Control Register 00)

Bit	Bit Symbol	After Reset	Type	Function
7	—	0	R	Read as "0"
6	INTPCLR	0	W	Detection flag clear control 0: — 1: Clear Reading the bit returns "0"
5	—	0	R	Read as "0"
4	INTPFLG	0	R	Detection flag 0: Not detected 1: Detected
3:0	—	0111	R	Read as "0111"

5.3.2. *[IBIMC000 to 043]* (Interrupt B Mode Control Registers n)

(1) *[IBIMC000 to 007]* Registers

Bit	Bit Symbol	After Reset	Type	Function
7	INTNCLR	0	W	Falling edge detection flag clear control 0: — 1: Clear Reading the bit returns "0"
6	INTPCLR	0	W	Rising edge detection flag clear control 0: — 1: Clear Reading the bit returns "0"
5	INTNFLG	0	R	Falling edge detection flag 0: Not detected 1: Detected
4	INTPFLG	0	R	Rising edge detection flag 0: Not detected 1: Detected
3:1	INTMODE[2:0]	000	R/W	Interrupt detection level selection 000: "Low" level 001: "High" level 010: Falling edge 011: Rising edge 100: Both edge 101: Reserved 110: Reserved 111: Reserved
0	INTEN	0	R/W	Interrupt control 0: Interrupt detection disabled 1: Interrupt detection enabled

(2) *[IBIMC008 to 043]* Register

Bit	Bit Symbol	After Reset	Type	Function
7	—	0	R	Read as "0"
6	INTPCLR	0	W	Detection flag clear control 0: — 1: Clear Reading the bit returns "0"
5	—	0	R	Read as "0"
4	INTPFLG	0	R	Detection flag 0: Not detected 1: Detected
3:1	—	011	R	Read as "011"
0	INTEN	0	R/W	Interrupt control 0: Interrupt detection disabled 1: Interrupt detection enabled

5.4. Reset Flag Registers

5.4.1. *[RLMRSTFLG0]* (Reset Flag Register 0)

Bit	Bit Symbol	After power on reset	Type	Function
7:6	—	Undefined	R	Read as an undefined valule.
5	LVDRSTF	Undefined	R	LVD reset flag 0: — 1: Reset from LVD
			W	LVD reset flag 0: Clear 1: Don’t care
4	—	Undefined	R	Read as an undefined valule.
			W	Write as "0"
3	PINRSTF	Undefined	R	Reset pin flag 0: — 1: Reset from reset pin
			W	Reset pin flag 0: Clear 1: Don’t care
2:1	—	Undefined	R	Read as an undefined valule.
			W	Write as "00"
0	PORSTF	1	R	Power on reset flag 0: — 1: Reset from by power on reset
			W	Power on reset flag 0: Clear 1: Don’t care

Note: Reset flags except <PORSTF> become undefined after power on reset release. When release of power on reset is detected, please write in to "0" to all the reset flags for initialize.

5.4.2. *[RLMRSTFLG1]* (Reset Flag Register 1)

Bit	Bit Symbol	After power on reset	Type	Function
7:4	—	0	R	Read as "0"
3	OFDRSTF	0	R	OFD reset flag 0: — 1: Reset from OFD
			W	OFD reset flag 0: Clear 1: Don't care
2	WDTRSTF	0	R	SIWDT reset flag 0: — 1: Reset from SIWDT
			W	SIWDT reset flag 0: Clear 1: Don't care
1	LOCKRSTF	0	R	LOCKUP reset flag 0: — 1: Reset from LOCKUP
			W	LOCKUP reset flag 0: Clear 1: Don't care
0	SYSRSTF	0	R	<SYSRESETREQ> reset flag 0: — 1: Reset from <SYSRESETREQ>
			W	<SYSRESETREQ> reset flag 0: Clear 1: Don't care

5.5. Interrupt Monitor Registers

5.5.1. *[IMNFLGNMI]* (Non-Maskable Interrupt Monitor Flag Register)

Bit	Bit Symbol	After Reset	Type	Function
31:17	—	0	R	Read as "0"
16	INT016FLG	0	R	INTWDT0 Interrupt detection flag 0: Interrupt not detected 1: Interrupt detected
15:1	—	0	R	Read as "0"
0	INT000FLG	0	R	INTLVD Interrupt detection flag 0: Interrupt not detected 1: Interrupt detected

5.5.2. *[IMNFLG3]* (Interrupt Monitor Flag Register 3)

Bit	Bit Symbol	After Reset	Type	Function
31	INT127FLG	0	R	INTT32A02_BT_C1 (T32A ch2 Timer B) Interrupt detection flag 0: Interrupt not detected 1: Interrupt detected
30	INT126FLG	0	R	INTT32A02_A01_C0 (T32A ch2 Timer C Capture 0) Interrupt detection flag 0: Interrupt not detected 1: Interrupt detected
29	INT125FLG	0	R	INTT32A02_A01_C0 (T32A ch2 Timer A Capture 1) Interrupt detection flag 0: Interrupt not detected 1: Interrupt detected
28	INT124FLG	0	R	INTT32A02_A01_C0 (T32A ch2 Timer A Capture 0) Interrupt detection flag 0: Interrupt not detected 1: Interrupt detected
27	INT123FLG	0	R	INTT32A02_AT_CT (T32A ch2 Timer C) Interrupt detection flag 0: Interrupt not detected 1: Interrupt detected
26	INT122FLG	0	R	INTT32A02_AT_CT (T32A ch2 Timer A) Interrupt detection flag 0: Interrupt not detected 1: Interrupt detected
25	INT121FLG	0	R	INTT32A01_B01 (T32A ch1 Timer B Capture 1) Interrupt detection flag 0: Interrupt not detected 1: Interrupt detected
24	INT120FLG	0	R	INTT32A01_B01 (T32A ch1 Timer B Capture 0) Interrupt detection flag 0: Interrupt not detected 1: Interrupt detected
23	INT119FLG	0	R	INTT32A01_BT_C1 (T32A ch1 Timer C Capture 1) Interrupt detection flag 0: Interrupt not detected 1: Interrupt detected

Bit	Bit Symbol	After Reset	Type	Function
22	INT118FLG	0	R	INTT32A01_BT_C1 (T32A ch1 Timer B) Interrupt detection flag 0: Interrupt not detected 1: Interrupt detected
21	INT117FLG	0	R	INTT32A01_A01_C0(T32A ch1 Timer C Capture 0) Interrupt detection flag 0: Interrupt not detected 1: Interrupt detected
20	INT116FLG	0	R	INTT32A01_A01_C0(T32A ch1 Timer A Capture 1) Interrupt detection flag 0: Interrupt not detected 1: Interrupt detected
19	INT115FLG	0	R	INTT32A01_A01_C0(T32A ch1 Timer A Capture 0) Interrupt detection flag 0: Interrupt not detected 1: Interrupt detected
18	INT114FLG	0	R	INTT32A01_AT_CT(T32A ch1 Timer C) Interrupt detection flag 0: Interrupt not detected 1: Interrupt detected
17	INT113FLG	0	R	INTT32A01_AT_CT(T32A ch1 Timer A) Interrupt detection flag 0: Interrupt not detected 1: Interrupt detected
16	INT112FLG	0	R	INTT32A00_B01 (T32A ch0 Timer B Capture 1) Interrupt detection flag 0: Interrupt not detected 1: Interrupt detected
15	INT111FLG	0	R	INTT32A00_B01 (T32A ch0 Timer B Capture 0) Interrupt detection flag 0: Interrupt not detected 1: Interrupt detected
14	INT110FLG	0	R	INTT32A00_BT_C1 (T32A ch0 Timer C Capture 1) Interrupt detection flag 0: Interrupt not detected 1: Interrupt detected
13	INT109FLG	0	R	INTT32A00_BT_C1 (T32A ch0 Timer B) Interrupt detection flag 0: Interrupt not detected 1: Interrupt detected
12	INT108FLG	0	R	INTT32A00_A01_C0(T32A ch0 Timer C Capture 0) Interrupt detection flag 0: Interrupt not detected 1: Interrupt detected
11	INT107FLG	0	R	INTT32A00_A01_C0(T32A ch0 Timer A Capture 1) Interrupt detection flag 0: Interrupt not detected 1: Interrupt detected
10	INT106FLG	0	R	INTT32A00_A01_C0(T32A ch0 Timer A Capture 0) Interrupt detection flag 0: Interrupt not detected 1: Interrupt detected
9	INT105FLG	0	R	INTT32A00_AT_CT(T32A ch0 Timer C) Interrupt detection flag 0: Interrupt not detected 1: Interrupt detected
8	INT104FLG	0	R	INTT32A00_AT_CT(T32A ch0 Timer A) Interrupt detection flag 0: Interrupt not detected 1: Interrupt detected
7	INT103FLG	0	R	INT07 Interrupt detection flag 0: Interrupt not detected 1: Interrupt detected

Bit	Bit Symbol	After Reset	Type	Function
6	INT102FLG	0	R	INT06 Interrupt detection flag 0: Interrupt not detected 1: Interrupt detected
5	INT101FLG	0	R	INT05 Interrupt detection flag 0: Interrupt not detected 1: Interrupt detected
4	INT100FLG	0	R	INT04 Interrupt detection flag 0: Interrupt not detected 1: Interrupt detected
3	INT099FLG	0	R	INT03 Interrupt detection flag 0: Interrupt not detected 1: Interrupt detected
2	INT098FLG	0	R	INT02 Interrupt detection flag 0: Interrupt not detected 1: Interrupt detected
1	INT097FLG	0	R	INT01 Interrupt detection flag 0: Interrupt not detected 1: Interrupt detected
0	INT096FLG	0	R	INT00 Interrupt detection flag 0: Interrupt not detected 1: Interrupt detected

5.5.3. *[IMNFLG4]* (Interrupt Monitor Flag Register 4)

Bit	Bit Symbol	After Reset	Type	Function
31:12	—	0	R	Read as "0"
11	INT139FLG	0	R	INTT32A03_B01 (T32A ch0 Timer B Capture 1) Interrupt detection flag 0: Interrupt not detected 1: Interrupt detected
10	INT138FLG	0	R	INTT32A03_B01 (T32A ch0 Timer B Capture 0) Interrupt detection flag 0: Interrupt not detected 1: Interrupt detected
9	INT137FLG	0	R	INTT32A03_BT_C1 (T32A ch0 Timer C Capture 1) Interrupt detection flag 0: Interrupt not detected 1: Interrupt detected
8	INT136FLG	0	R	INTT32A03_BT_C1 (T32A ch0 Timer B) Interrupt detection flag 0: Interrupt not detected 1: Interrupt detected
7	INT135FLG	0	R	INTT32A03_A01_C0(T32A ch0 Timer C Capture 0) Interrupt detection flag 0: Interrupt not detected 1: Interrupt detected
6	INT134FLG	0	R	INTT32A03_A01_C0(T32A ch0 Timer A Capture 1) Interrupt detection flag 0: Interrupt not detected 1: Interrupt detected
5	INT133FLG	0	R	INTT32A03_A01_C0(T32A ch0 Timer A Capture 0) Interrupt detection flag 0: Interrupt not detected 1: Interrupt detected
4	INT132FLG	0	R	INTT32A03_AT_CT(T32A ch0 Timer C) Interrupt detection flag 0: Interrupt not detected 1: Interrupt detected
3	INT131FLG	0	R	INTT32A03_AT_CT(T32A ch0 Timer A) Interrupt detection flag 0: Interrupt not detected 1: Interrupt detected
2	INT130FLG	0	R	INTT32A02_B01(T32A ch2 Timer B Capture 1) Interrupt detection flag 0: Interrupt not detected 1: Interrupt detected
1	INT129FLG	0	R	INTT32A02_B01(T32A ch2 Timer B Capture 0) Interrupt detection flag 0: Interrupt not detected 1: Interrupt detected
0	INT128FLG	0	R	INTT32A02_BT_C1(T32A ch2 Timer C Capture 1) Interrupt detection flag 0: Interrupt not detected 1: Interrupt detected

5.6. NVIC Registers

5.6.1. SysTick Control and Status Register

Bit	Bit Symbol	After Reset	Type	Function
31:17	—	0	R	Read as "0"
16	COUNTFLAG	0	R/W	0: Timer not counted to 0 1: Timer counted to 0 Returns "1" if timer counted to "0" since last time this was read. Clears on read of any part of the SysTick Control and Status register.
15:3	—	0	R	Read as "0"
2	CLKSOURCE	0	R/W	0: External reference clock (fosc/64) 1: CPU clock(fsys)
1	TICKINT	0	R/W	0: Do not pend SysTick 1: Pend SysTick
0	ENABLE	0	R/W	0: Disable 1: Enable If "1" is set, it re-load with the value of the Reload Value register and starts operation.

5.6.2. SysTick Reload Value Register

Bit	Bit Symbol	After Reset	Type	Function
31:24	—	0	R	Read as "0"
23:0	RELOAD[23:0]	Undefined	R/W	Reload value Set the value to load into the SysTick Current Value Register when the timer reaches "0".

5.6.3. SysTick Current Value Register

Bit	Bit Symbol	After Reset	Type	Function
31:24	—	0	R	Read as "0"
23:0	CURRENT[23:0]	Undefined	R	Current SysTick timer value
			W	Clear Writing to this register with any value clears it to 0. Clearing this register also clears the <COUNTFLAG> bit of the SysTick Control and Status Register.

5.6.4. SysTick Calibration Value Register

Bit	Bit Symbol	After Reset	Type	Function
31	NOREF	0	R	0: Reference clock provided 1: No reference clock
30	SKEW	1	R	0: Calibration value is 10 ms. 1: Calibration value is not 10 ms.
29:24	—	0	R	Read as "0"
23:0	TENMS	0x000000	R	Calibration value (Note)

Note: This product does not prepare the calibration value.

5.6.5. Interrupt Control Registers

Following four registers will be used to control each interrupt source; Interrupt Set-Enable Register, Interrupt Clear-Enable Register, Interrupt Set-Pending Register, and Interrupt Clear-Pending Register.

Each bit corresponds to specified interrupt.

5.6.5.1. Interrupt Set-Enable Register

Each bit corresponds to the specified number of interrupts. It can enable interrupts and check if interrupts are enabled.

Writing "1" to a bit in this register enables the corresponding interrupt.

Writing "0" has no effect.

Reading the bits can see the enable/disable condition of the corresponding interrupts. Writing "1" to a corresponding bit in the Interrupt Clear-Enable Register clears the bit in this register.

(a) Interrupt Set-Enable Register 0

Bit	Bit Symbol	After Reset	Type	Function
31	—	0	R/W	Write as “0”
30	SETENA (Interrupt 30)	0	R/W	[Write] 1: Enable interrupt [Read] 0: Interrupt is disabled 1: Interrupt is enabled
29	SETENA (Interrupt 29)	0		
28	SETENA (Interrupt 28)	0		
27	SETENA (Interrupt 27)	0		
26	SETENA (Interrupt 26)	0		
25	SETENA (Interrupt 25)	0		
24	SETENA (Interrupt 24)	0		
23	SETENA (Interrupt 23)	0		
22	SETENA (Interrupt 22)	0		
21	SETENA (Interrupt 21)	0		
20	SETENA (Interrupt 20)	0		
19	SETENA (Interrupt 19)	0		
18	SETENA (Interrupt 18)	0		
17	SETENA (Interrupt 17)	0		
16	SETENA (Interrupt 16)	0		
15	SETENA (Interrupt 15)	0		
14	SETENA (Interrupt 14)	0		
13	SETENA (Interrupt 13)	0		
12	SETENA (Interrupt 12)	0		
11	SETENA (Interrupt 11)	0		
10	SETENA (Interrupt 10)	0		
9	SETENA (Interrupt 9)	0		
8	SETENA (Interrupt 8)	0		
7	SETENA (Interrupt 7)	0		
6	SETENA (Interrupt 6)	0		
5	SETENA (Interrupt 5)	0		
4	SETENA (Interrupt 4)	0		
3	SETENA (Interrupt 3)	0		
2	SETENA (Interrupt 2)	0		
1	SETENA (Interrupt 1)	0		
0	SETENA (Interrupt 0)	0		

(b) Interrupt Set-Enable Register 1

Bit	Bit Symbol	After Reset	Type	Function
31:28	—	0	R	Read as “0”
27	—	0	R/W	Write as “0”
26	SETENA (Interrupt 58)	0	R/W	[Write] 1: Enable interrupt [Read] 0: Interrupt is disabled 1: Interrupt is enabled
25	SETENA (Interrupt 57)	0		
24	SETENA (Interrupt 56)	0		
23	SETENA (Interrupt 55)	0		
22	SETENA (Interrupt 54)	0		
21	SETENA (Interrupt 53)	0		
20	SETENA (Interrupt 52)	0		
19	SETENA (Interrupt 51)	0		
18	SETENA (Interrupt 50)	0		
17	SETENA (Interrupt 49)	0		
16	SETENA (Interrupt 48)	0		
15	SETENA (Interrupt 47)	0		
14	SETENA (Interrupt 46)	0		
13	SETENA (Interrupt 45)	0		
12	SETENA (Interrupt 44)	0		
11	SETENA (Interrupt 43)	0		
10	SETENA (Interrupt 42)	0		
9	SETENA (Interrupt 41)	0		
8	SETENA (Interrupt 40)	0		
7	SETENA (Interrupt 39)	0		
6	SETENA (Interrupt 38)	0		
5	SETENA (Interrupt 37)	0		
4	SETENA (Interrupt 36)	0		
3	SETENA (Interrupt 35)	0		
2	SETENA (Interrupt 34)	0		
1	SETENA (Interrupt 33)	0		
0	SETENA (Interrupt 32)	0		

5.6.5.2. Interrupt Clear-Enable Register

Each bit corresponds to the specified number of interrupts. It can disable interrupts and check if interrupts are disabled.

Writing "1" to a bit in this register disables the corresponding interrupt.

Writing "0" has no effect.

Reading the bits can see the enable/disable condition of the corresponding interrupts.

(a) Interrupt Clear-Enable Register 0

Bit	Bit Symbol	After Reset	Type	Function
31	—	0	R/W	Write as “0”
30	CLRENA (Interrupt 30)	0	R/W	[Write] 1: Disable Interrupt [Read] 0: Interrupt is disabled 1: Interrupt is enabled
29	CLRENA (Interrupt 29)	0		
28	CLRENA (Interrupt 28)	0		
27	CLRENA (Interrupt 27)	0		
26	CLRENA (Interrupt 26)	0		
25	CLRENA (Interrupt 25)	0		
24	CLRENA (Interrupt 24)	0		
23	CLRENA (Interrupt 23)	0		
22	CLRENA (Interrupt 22)	0		
21	CLRENA (Interrupt 21)	0		
20	CLRENA (Interrupt 20)	0		
19	CLRENA (Interrupt 19)	0		
18	CLRENA (Interrupt 18)	0		
17	CLRENA (Interrupt 17)	0		
16	CLRENA (Interrupt 16)	0		
15	CLRENA (Interrupt 15)	0		
14	CLRENA (Interrupt 14)	0		
13	CLRENA (Interrupt 13)	0		
12	CLRENA (Interrupt 12)	0		
11	CLRENA (Interrupt 11)	0		
10	CLRENA (Interrupt 10)	0		
9	CLRENA (Interrupt 9)	0		
8	CLRENA (Interrupt 8)	0		
7	CLRENA (Interrupt 7)	0		
6	CLRENA (Interrupt 6)	0		
5	CLRENA (Interrupt 5)	0		
4	CLRENA (Interrupt 4)	0		
3	CLRENA (Interrupt 3)	0		
2	CLRENA (Interrupt 2)	0		
1	CLRENA (Interrupt 1)	0		
0	CLRENA (Interrupt 0)	0		

(b) Interrupt Clear-Enable Register 1

Bit	Bit Symbol	After Reset	Type	Function
31:28	—	0	R	Read as “0”
27	—	0	R/W	Write as “0”
26	CLRENA (Interrupt 58)	0	R/W	[Write] 1: Disable Interrupt [Read] 0: Interrupt is disabled 1: Interrupt is enabled
25	CLRENA (Interrupt 57)	0		
24	CLRENA (Interrupt 56)	0		
23	CLRENA (Interrupt 55)	0		
22	CLRENA (Interrupt 54)	0		
21	CLRENA (Interrupt 53)	0		
20	CLRENA (Interrupt 52)	0		
19	CLRENA (Interrupt 51)	0		
18	CLRENA (Interrupt 50)	0		
17	CLRENA (Interrupt 49)	0		
16	CLRENA (Interrupt 48)	0		
15	CLRENA (Interrupt 47)	0		
14	CLRENA (Interrupt 46)	0		
13	CLRENA (Interrupt 45)	0		
12	CLRENA (Interrupt 44)	0		
11	CLRENA (Interrupt 43)	0		
10	CLRENA (Interrupt 42)	0		
9	CLRENA (Interrupt 41)	0		
8	CLRENA (Interrupt 40)	0		
7	CLRENA (Interrupt 39)	0		
6	CLRENA (Interrupt 38)	0		
5	CLRENA (Interrupt 37)	0		
4	CLRENA (Interrupt 36)	0		
3	CLRENA (Interrupt 35)	0		
2	CLRENA (Interrupt 34)	0		
1	CLRENA (Interrupt 33)	0		
0	CLRENA (Interrupt 32)	0		

5.6.5.3. Interrupt Set-Pending Register

Each bit corresponds to the specified number of interrupts. It can force interrupts into the pending state and determines which interrupts are currently pending.

Writing "1" to a bit in this register pends the corresponding interrupt. However, writing "1" has no effect on an interrupt that is already pending or is disabled.

Writing "0" has no effect.

Reading the bit returns the current state of the corresponding interrupts.

Writing "1" to a corresponding bit in the Interrupt Clear-Pending Register clears the bit in this register.

(a) Interrupt Set-Pending Register 0

Bit	Bit Symbol	After Reset	Type	Function
31	—	0	R/W	Write as “0”
30	SETPEND (Interrupt 30)	Undefined	R/W	[Write] 1: Pend interrupt [Read] 0: Not pending 1: Pending
29	SETPEND (Interrupt 29)	Undefined		
28	SETPEND (Interrupt 28)	Undefined		
27	SETPEND (Interrupt 27)	Undefined		
26	SETPEND (Interrupt 26)	Undefined		
25	SETPEND (Interrupt 25)	Undefined		
24	SETPEND (Interrupt 24)	Undefined		
23	SETPEND (Interrupt 23)	Undefined		
22	SETPEND (Interrupt 22)	Undefined		
21	SETPEND (Interrupt 21)	Undefined		
20	SETPEND (Interrupt 20)	Undefined		
19	SETPEND (Interrupt 19)	Undefined		
18	SETPEND (Interrupt 18)	Undefined		
17	SETPEND (Interrupt 17)	Undefined		
16	SETPEND (Interrupt 16)	Undefined		
15	SETPEND (Interrupt 15)	Undefined		
14	SETPEND (Interrupt 14)	Undefined		
13	SETPEND (Interrupt 13)	Undefined		
12	SETPEND (Interrupt 12)	Undefined		
11	SETPEND (Interrupt 11)	Undefined		
10	SETPEND (Interrupt 10)	Undefined		
9	SETPEND (Interrupt 9)	Undefined		
8	SETPEND (Interrupt 8)	Undefined		
7	SETPEND (Interrupt 7)	Undefined		
6	SETPEND (Interrupt 6)	Undefined		
5	SETPEND (Interrupt 5)	Undefined		
4	SETPEND (Interrupt 4)	Undefined		
3	SETPEND (Interrupt 3)	Undefined		
2	SETPEND (Interrupt 2)	Undefined		
1	SETPEND (Interrupt 1)	Undefined		
0	SETPEND (Interrupt 0)	Undefined		

(b) Interrupt Set-Pending Register 1

Bit	Bit Symbol	After Reset	Type	Function
31:28	—	0	R	Read as “0”
27	—	0	R/W	Write as “0”
26	SETPEND (Interrupt 58)	Undefined	R/W	[Write] 1: Pend interrupt [Read] 0: Not pending 1: Pending
25	SETPEND (Interrupt 57)	Undefined		
24	SETPEND (Interrupt 56)	Undefined		
23	SETPEND (Interrupt 55)	Undefined		
22	SETPEND (Interrupt 54)	Undefined		
21	SETPEND (Interrupt 53)	Undefined		
20	SETPEND (Interrupt 52)	Undefined		
19	SETPEND (Interrupt 51)	Undefined		
18	SETPEND (Interrupt 50)	Undefined		
17	SETPEND (Interrupt 49)	Undefined		
16	SETPEND (Interrupt 48)	Undefined		
15	SETPEND (Interrupt 47)	Undefined		
14	SETPEND (Interrupt 46)	Undefined		
13	SETPEND (Interrupt 45)	Undefined		
12	SETPEND (Interrupt 44)	Undefined		
11	SETPEND (Interrupt 43)	Undefined		
10	SETPEND (Interrupt 42)	Undefined		
9	SETPEND (Interrupt 41)	Undefined		
8	SETPEND (Interrupt 40)	Undefined		
7	SETPEND (Interrupt 39)	Undefined		
6	SETPEND (Interrupt 38)	Undefined		
5	SETPEND (Interrupt 37)	Undefined		
4	SETPEND (Interrupt 36)	Undefined		
3	SETPEND (Interrupt 35)	Undefined		
2	SETPEND (Interrupt 34)	Undefined		
1	SETPEND (Interrupt 33)	Undefined		
0	SETPEND (Interrupt 32)	Undefined		

5.6.5.4. Interrupt Clear-Pending Register

Each bit corresponds to the specified number of interrupt. It can clear pending interrupts and determines which interrupts are currently pending.

Writing "1" to a bit in this register clears the corresponding pending interrupt. However, writing "1" has no effect on an interrupt that is already being serviced. Writing "0" has no effect.

Reading the bit returns the current state of the corresponding interrupts.

(a) Interrupt Clear-Pending Register 0

Bit	Bit Symbol	After Reset	Type	Function
31	—	0	R/W	Write as “1”
30	CLRPEND (Interrupt 30)	Undefined	R/W	[Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending
29	CLRPEND (Interrupt 29)	Undefined		
28	CLRPEND (Interrupt 28)	Undefined		
27	CLRPEND (Interrupt 27)	Undefined		
26	CLRPEND (Interrupt 26)	Undefined		
25	CLRPEND (Interrupt 25)	Undefined		
24	CLRPEND (Interrupt 24)	Undefined		
23	CLRPEND (Interrupt 23)	Undefined		
22	CLRPEND (Interrupt 22)	Undefined		
21	CLRPEND (Interrupt 21)	Undefined		
20	CLRPEND (Interrupt 20)	Undefined		
19	CLRPEND (Interrupt 19)	Undefined		
18	CLRPEND (Interrupt 18)	Undefined		
17	CLRPEND (Interrupt 17)	Undefined		
16	CLRPEND (Interrupt 16)	Undefined		
15	CLRPEND (Interrupt 15)	Undefined		
14	CLRPEND (Interrupt 14)	Undefined		
13	CLRPEND (Interrupt 13)	Undefined		
12	CLRPEND (Interrupt 12)	Undefined		
11	CLRPEND (Interrupt 11)	Undefined		
10	CLRPEND (Interrupt 10)	Undefined		
9	CLRPEND (Interrupt 9)	Undefined		
8	CLRPEND (Interrupt 8)	Undefined		
7	CLRPEND (Interrupt 7)	Undefined		
6	CLRPEND (Interrupt 6)	Undefined		
5	CLRPEND (Interrupt 5)	Undefined		
4	CLRPEND (Interrupt 4)	Undefined		
3	CLRPEND (Interrupt 3)	Undefined		
2	CLRPEND (Interrupt 2)	Undefined		
1	CLRPEND (Interrupt 1)	Undefined		
0	CLRPEND (Interrupt 0)	Undefined		

(b) Interrupt Clear-Pending Register 1

Bit	Bit Symbol	After Reset	Type	Function
31:28	—	0	R	Read as “0”
27	—	0	R/W	Write as “1”
26	CLRPEND (Interrupt 58)	Undefined	R/W	[Write] 1: Clear pending interrupt [Read] 0: Not pending 1: Pending
25	CLRPEND (Interrupt 57)	Undefined		
24	CLRPEND (Interrupt 56)	Undefined		
23	CLRPEND (Interrupt 55)	Undefined		
22	CLRPEND (Interrupt 54)	Undefined		
21	CLRPEND (Interrupt 53)	Undefined		
20	CLRPEND (Interrupt 52)	Undefined		
19	CLRPEND (Interrupt 51)	Undefined		
18	CLRPEND (Interrupt 50)	Undefined		
17	CLRPEND (Interrupt 49)	Undefined		
16	CLRPEND (Interrupt 48)	Undefined		
15	CLRPEND (Interrupt 47)	Undefined		
14	CLRPEND (Interrupt 46)	Undefined		
13	CLRPEND (Interrupt 45)	Undefined		
12	CLRPEND (Interrupt 44)	Undefined		
11	CLRPEND (Interrupt 43)	Undefined		
10	CLRPEND (Interrupt 42)	Undefined		
9	CLRPEND (Interrupt 41)	Undefined		
8	CLRPEND (Interrupt 40)	Undefined		
7	CLRPEND (Interrupt 39)	Undefined		
6	CLRPEND (Interrupt 38)	Undefined		
5	CLRPEND (Interrupt 37)	Undefined		
4	CLRPEND (Interrupt 36)	Undefined		
3	CLRPEND (Interrupt 35)	Undefined		
2	CLRPEND (Interrupt 34)	Undefined		
1	CLRPEND (Interrupt 33)	Undefined		
0	CLRPEND (Interrupt 32)	Undefined		

5.6.6. Interrupt Priority Register

Each interrupt is provided with eight bits of an Interrupt Priority Register.
The following shows the addresses of the Interrupt Priority Registers corresponding to interrupt numbers.

Address	31 24	23 16	15 8	7 0
0xE000E400	PRI_3	PRI_2	PRI_1	PRI_0
0xE000E404	PRI_7	PRI_6	PRI_5	PRI_4
0xE000E408	PRI_11	PRI_10	PRI_9	PRI_8
0xE000E40C	PRI_15	PRI_14	PRI_13	PRI_12
0xE000E410	PRI_19	PRI_18	PRI_17	PRI_16
0xE000E414	PRI_23	PRI_22	PRI_21	PRI_20
0xE000E418	PRI_27	PRI_26	PRI_25	PRI_24
0xE000E41C	—	PRI_30	PRI_29	PRI_28
0xE000E420	PRI_35	PRI_34	PRI_33	PRI_32
0xE000E424	PRI_39	PRI_38	PRI_37	PRI_36
0xE000E428	PRI_43	PRI_42	PRI_41	PRI_40
0xE000E42C	PRI_47	PRI_46	PRI_45	PRI_44
0xE000E430	PRI_51	PRI_50	PRI_49	PRI_48
0xE000E434	PRI_55	PRI_54	PRI_53	PRI_52
0xE000E438	—	PRI_58	PRI_57	PRI_56

The number of bits to be used for assigning a priority varies with each product. This product uses four bits for assigning a priority.
The following shows the fields of the Interrupt Priority Registers for interrupt numbers 0 to 3. The Interrupt Priority Registers for all other interrupt numbers have the identical fields. Unused bits return "0" when read, and writing to unused bits has no effect.

Bit	Bit Symbol	After Reset	Type	Function
31:28	PRI_3[3:0]	0000	R/W	Priority of interrupt number 3
27:24	—	0	R	Read as "0"
23:20	PRI_2[3:0]	0000	R/W	Priority of interrupt number 2
19:16	—	0	R	Read as "0"
15:12	PRI_1[3:0]	0000	R/W	Priority of interrupt number 1
11:8	—	0	R	Read as "0"
7:4	PRI_0[3:0]	0000	R/W	Priority of interrupt number 0
3:0	—	0	R	Read as "0"

5.6.7. Vector Table Offset Register

Bit	Bit Symbol	After Reset	Type	Function
31:7	TBLOFF[24:0]	0x0000000	R/W	Offset value Set the offset value from the address of "0x00000000". The offset must be aligned based on the number of exceptions in the table. This means that the minimum alignment is 32 words that you can use for up to 16 interrupts. For more interrupts, you must adjust the alignment by rounding up to the next power of two.
6:0	—	0	R	Read as "0"

5.6.8. Application Interrupt and Reset Control Register

Bit	Bit Symbol	After Reset	Type	Function
31:16	VECTKEY/ VECTKEYSTAT[15:0]	Undefined	W	Register key Writing to this register requires “0x05FA” in the <VECTKEY> field.
			R	Register key Read as “0xFA05”
15	ENDIANESS	0	R/W	Endianness bit: (Note 1) 1: Big endian 0: Little endian
14:11	—	0	R	Read as "0"
10:8	PRIGROUP[2:0]	000	R/W	Interrupt priority grouping 000: seven bits of pre-emption priority, one bit of sub priority 001: six bits of pre-emption priority, two bits of sub priority 010: five bits of pre-emption priority, three bits of sub priority 011: four bits of pre-emption priority, four bits of sub priority 100: three bits of pre-emption priority, five bits of sub priority 101: two bits of pre-emption priority, six bits of sub priority 110: one bit of pre-emption priority, seven bits of sub priority 111: no pre-emption priority, eight bits of sub priority The bit configuration to split the interrupt priority register <PRI_n> into pre-emption priority and sub priority.
7:3	—	0	R	Read as "0"
2	SYSRESETREQ	0	R/W	System Reset Request 1=CPU outputs a SYSRESETREQ signal. (Note 2)
1	VECTCLRACTIVE	0	R/W	Clear active vector bit 1: clear all state information for active NMI, fault, and interrupts. 0: do not clear. This bit self-clears. It is the responsibility of the application to reinitialize the stack.
0	VECTRESET	0	R/W	System Reset bit 1: reset system. 0: do not reset system. Resets the system, with the exception of debug components (FPB, DWT and ITM) by setting "1" and this bit is also zero cleared

Note 1: Little-endian is the default memory format for this product.

Note 2: When SYSRESETREQ is output, warm reset is performed on this product. <SYSRESETREQ> is cleared by warm reset.

5.6.9. System Handler Priority Register

Each exception is provided with eight bits of a System Handler Priority Register.
The following shows the addresses of the System Handler Priority Registers corresponding to each exception.

Address	31 24	23 16	15 8	7 0
0xE000ED18	PRI_7	PRI_6 (Usage Fault)	PRI_5 (Bus Fault)	PRI_4 (Memory Management)
0xE000ED1C	PRI_11 (SVCall)	PRI_10	PRI_9	PRI_8
0xE000ED20	PRI_15 (SysTick)	PRI_14 (PendSV)	PRI_13	PRI_12 (Debug Monitor)

The number of bits to be used for assigning a priority varies with each product. This product uses four bits for assigning a priority.

The following shows the fields of the System Handler Priority Registers for Usage Fault, Bus Fault, and Memory Management. Unused bits return "0" when read, and writing to unused bits has no effect.

Bit	Bit Symbol	After Reset	Type	Function
31:28	PRI_7[3:0]	0000	R/W	Reserved
27:24	—	0	R	Read as "0"
23:20	PRI_6[3:0]	0000	R/W	Priority of Usage Fault
19:16	—	0	R	Read as "0"
15:12	PRI_5[3:0]	0000	R/W	Priority of Bus Fault
11:8	—	0	R	Read as "0"
7:4	PRI_4[3:0]	0000	R/W	Priority of Memory Management
3:0	—	0	R	Read as "0"

5.6.10. System Handler Control and State Register

Bit	Bit Symbol	After Reset	Type	Function
31:19	—	0	R	Read as "0"
18	USGFAULT ENA	0	R/W	Usage Fault 0: Disabled 1: Enabled
17	BUSFAULT ENA	0	R/W	Bus Fault 0: Disabled 1: Enabled
16	MEMFAULT ENA	0	R/W	Memory Management 0: Disabled 1: Enabled
15	SVCALL PENDED	0	R/W	SVCall 0: Not pended 1: Pended
14	BUSFAULT PENDED	0	R/W	Bus Fault 0: Not pended 1: Pended
13	MEMFAULT PENDED	0	R/W	Memory Management 0: Not pended 1: Pended
12	USGFAULT PENDED	0	R/W	Usage fault 0: Not pended 1: Pended
11	SYSTICKACT	0	R/W	SysTick 0: Inactive 1: Active
10	PENDSVACT	0	R/W	PendSV 0: Inactive 1: Active
9	—	0	R	Read as "0"
8	MONITOR ACT	0	R/W	Debug Monitor 0: Inactive 1: Active
7	SVCALLACT	0	R/W	SVCall 0: Inactive 1: Active
6:4	—	0	R	Read as "0"
3	USGFAULT ACT	0	R/W	Usage Fault 0: Inactive 1: Active
2	—	0	R	Read as "0"
1	BUSFAULT ACT	0	R/W	Bus Fault 0: Inactive 1: Active
0	MEMFAULT ACT	0	R/W	Memory Management 0: Inactive 1: Active

Note: You must clear or set the active bits with extreme caution because clearing and setting these bits does not repair stack contents.

6. List of Interrupt Sources for Each Product

6.1. TMPM4L2/TMPM4L1

M4L2	M4L1	Interrupt No	Interrupt Source	Interrupt Request	Interrupt Control Register	Interrupt Monitor Register
✓	✓	NMI	INTLVD	LVD interrupt	***[IANIC00]***	***[IMNFLGNMI]*** <INT000FLG>
			INTWDT0	WDT interrupt	***[IBNIC00]***	***[IMNFLGNMI]*** <INT016FLG>
✓	✓	0	INT00	External interrupt 00	***[IBIMC000]***	***[IMNFLG3]*** <INT096FLG>
✓	✓	1	INT01	External interrupt 01	***[IBIMC001]***	***[IMNFLG3]*** <INT097FLG>
✓	✓	2	INT02	External interrupt 02	***[IBIMC002]***	***[IMNFLG3]*** <INT098FLG>
✓	✓	3	INT03	External interrupt 03	***[IBIMC003]***	***[IMNFLG3]*** <INT099FLG>
✓	✓	4	INT04	External interrupt 04	***[IBIMC004]***	***[IMNFLG3]*** <INT100FLG>
✓	✓	5	INT05	External interrupt 05	***[IBIMC005]***	***[IMNFLG3]*** <INT101FLG>
✓	✓	6	INT06	External interrupt 06	***[IBIMC006]***	***[IMNFLG3]*** <INT102FLG>
✓	—	7	INT07	External interrupt 07	***[IBIMC007]***	***[IMNFLG3]*** <INT103FLG>
✓	✓	8	INTT32A00_AT_CT	T32A ch0 timer A match, overflow and underflow	***[IBIMC008]***	***[IMNFLG3]*** <INT104FLG>
✓	✓			T32A ch0 timer C match, overflow and underflow	***[IBIMC009]***	***[IMNFLG3]*** <INT105FLG>
✓	✓	9	INTT32A00_A01_C0	T32A ch0 timer A capture 0	***[IBIMC010]***	***[IMNFLG3]*** <INT106FLG>
✓	✓			T32A ch0 timer A capture 1	***[IBIMC011]***	***[IMNFLG3]*** <INT107FLG>
✓	✓			T32A ch0 timer C capture 0	***[IBIMC012]***	***[IMNFLG3]*** <INT108FLG>
✓	✓	10	INTT32A00_BT_C1	T32A ch0 timer B match, overflow and underflow	***[IBIMC013]***	***[IMNFLG3]*** <INT109FLG>
✓	✓			T32A ch0 timer C capture 1	***[IBIMC014]***	***[IMNFLG3]*** <INT110FLG>
✓	✓	11	INTT32A00_B01	T32A ch0 timer B capture 0	***[IBIMC015]***	***[IMNFLG3]*** <INT111FLG>
✓	✓			T32A ch0 timer B capture 1	***[IBIMC016]***	***[IMNFLG3]*** <INT112FLG>
✓	✓	12	INTT32A01_AT_CT	T32A ch1 timer A match, overflow and underflow	***[IBIMC017]***	***[IMNFLG3]*** <INT113FLG>
✓	✓			T32A ch1 timer C match, overflow and underflow	***[IBIMC018]***	***[IMNFLG3]*** <INT114FLG>
✓	✓	13	INTT32A01_A01_C0	T32A ch1 timer A capture 0	***[IBIMC019]***	***[IMNFLG3]*** <INT115FLG>
✓	✓			T32A ch1 timer A capture 1	***[IBIMC020]***	***[IMNFLG3]*** <INT116FLG>
✓	✓			T32A ch1 timer C capture 0	***[IBIMC021]***	***[IMNFLG3]*** <INT117FLG>
✓	✓	14	INTT32A01_BT_C1	T32A ch1 timer B match, overflow and underflow	***[IBIMC022]***	***[IMNFLG3]*** <INT118FLG>
✓	✓			T32A ch1 timer C capture 1	***[IBIMC023]***	***[IMNFLG3]*** <INT119FLG>
✓	✓	15	INTT32A01_B01	T32A ch1 timer B capture 0	***[IBIMC024]***	***[IMNFLG3]*** <INT120FLG>
✓	✓			T32A ch1 timer B capture 1	***[IBIMC025]***	***[IMNFLG3]*** <INT121FLG>
✓	✓	16	INTT32A02_AT_CT	T32A ch2 timer A match, overflow and underflow	***[IBIMC026]***	***[IMNFLG3]*** <INT122FLG>
✓	✓			T32A ch2 timer C match, overflow and underflow	***[IBIMC027]***	***[IMNFLG3]*** <INT123FLG>
✓	✓	17	INTT32A02_A01_C0	T32A ch2 timer A capture 0	***[IBIMC028]***	***[IMNFLG3]*** <INT124FLG>

M4L2	M4L1	Interrupt No	Interrupt Source	Interrupt Request	Interrupt Control Register	Interrupt Monitor Register
✓	✓			T32A ch2 timer A capture 1	***[IBIMC029]***	***[IMNFLG3]*** <INT125FLG>
✓	✓			T32A ch2 timer C capture 0	***[IBIMC030]***	***[IMNFLG3]*** <INT126FLG>
✓	✓	18	INTT32A02_BT_C1	T32A ch2 timer B match overflow and underflow	***[IBIMC031]***	***[IMNFLG3]*** <INT127FLG>
✓	✓			T32A ch2 timer C capture 1	***[IBIMC032]***	***[IMNFLG4]*** <INT128FLG>
✓	✓	19	INTT32A02_B01	T32A ch2 timer B capture 0	***[IBIMC033]***	***[IMNFLG4]*** <INT129FLG>
✓	✓			T32A ch2 timer B capture 1	***[IBIMC034]***	***[IMNFLG4]*** <INT130FLG>
✓	✓	20	INTT32A03_AT_CT	T32A ch3 timer A match, overflow and underflow	***[IBIMC035]***	***[IMNFLG4]*** <INT131FLG>
✓	✓			T32A ch3 timer C match, overflow and underflow	***[IBIMC036]***	***[IMNFLG4]*** <INT132FLG>
✓	✓	21	INTT32A03_A01_C0	T32A ch3 timer A capture 0	***[IBIMC037]***	***[IMNFLG4]*** <INT133FLG>
✓	✓			T32A ch3 timer A capture 1	***[IBIMC038]***	***[IMNFLG4]*** <INT134FLG>
✓	✓			T32A ch3 timer C capture 0	***[IBIMC039]***	***[IMNFLG4]*** <INT135FLG>
✓	✓	22	INTT32A03_BT_C1	T32A ch3 timer B match, overflow and underflow	***[IBIMC040]***	***[IMNFLG4]*** <INT136FLG>
✓	✓			T32A ch3 timer C capture 1	***[IBIMC041]***	***[IMNFLG4]*** <INT137FLG>
✓	✓	23	INTT32A03_B01	T32A ch3 timer B capture 0	***[IBIMC042]***	***[IMNFLG4]*** <INT138FLG>
✓	✓			T32A ch3 timer B capture 1	***[IBIMC043]***	***[IMNFLG4]*** <INT139FLG>
✓	✓	24	INTVCN0	A-VE ch0 Schedule end interrupt		
✓	✓	25	INTVCT0	A-VE ch0 Task end interrupt		
✓	✓	26	INTEMG0	PMD+ ch0 EMG interrupt		
✓	✓	27	INTOVV0	PMD+ ch0 OVV interrupt		
✓	✓	28	INTPWM0	PMD+ ch0 PWM interrupt		
✓	✓	29	INTENC00	A-ENC32 ch0 Encoder input interrupt 0		
✓	✓	30	INTENC01	A-ENC32 ch0 Encoder input interrupt 1		
—	—	31	—	—		
✓	✓	32	INTADAPDA	ADC unitA PMD trigger interrupt A		
✓	✓	33	INTADAPDB	ADC unitA PMD trigger interrupt B		
✓	✓	34	INTADACP0	ADC unitA Monitor function 0 interrupt		
✓	✓	35	INTADACP1	ADC unitA Monitor function 1 interrupt		
✓	✓	36	INTADATRG	ADC unitA General purpose trigger interrupt		
✓	✓	37	INTADASGL	ADC unitA Single conversion interrupt		
✓	✓	38	INTADACNT	ADC unitA Continuous conversion interrupt		
✓	✓	39	INTT0RX	TSPI ch0 Receive interrupt		
✓	✓	40	INTT0TX	TSPI ch0 Transmit interrupt		
✓	✓	41	INTT0ERR	TSPI ch0 Error interrupt		
✓	✓	42	INTT1RX	TSPI ch1 Receive interrupt		
✓	✓	43	INTT1TX	TSPI ch1 Transmit interrupt		
✓	✓	44	INTT1ERR	TSPI ch1 Error interrupt		
✓	✓	45	INTT2RX	TSPI ch2 Receive interrupt		
✓	✓	46	INTT2TX	TSPI ch2 Transmit interrupt		
✓	✓	47	INTT2ERR	TSPI ch2 Error interrupt		
✓	✓	48	INTUART0RX	UART ch0 Reception interrupt		
✓	✓	49	INTUART0TX	UART ch0 Transmission interrupt		

M4L2	M4L1	Interrupt No	Interrupt Source	Interrupt Request	Interrupt Control Register	Interrupt Monitor Register
✓	✓	50	INTUART0ERR	UART ch0 Error interrupt		
✓	✓	51	INTUART1RX	UART ch1 Reception interrupt		
✓	✓	52	INTUART1TX	UART ch1 Transmission interrupt		
✓	✓	53	INTUART1ERR	UART ch1 Error interrupt		
✓	✓	54	INTUART2RX	UART ch2 Reception interrupt		
✓	✓	55	INTUART2TX	UART ch2 Transmission interrupt		
✓	✓	56	INTUART2ERR	UART ch2 Error interrupt		
✓	✓	57	INTPARI0	RAMP ch0 RAM parity interrupt		
✓	✓	58	INTFLCRDY	Code FLASH Ready interrupt		

✓: Available, —: N/A

7. Revision History

Table 7.1 Revision History

Revision	Date	Description
1.0	2018-10-05	- First release
1.1	2019-06-10	- Overall : Corrected "interruption" → "interrupt" Modified "High", "High" level and High level to "High" level Modified "Low", "Low" level and Low level to "Low" level - "Conventions" Revised the SST register trademark - "Terms and Abbreviations" Added NVIC and RAMP Deleted DNF,TRGSEL and TRM - "1.2.1.Exception Request and Detection" (1) Exception Occurrence: "external interruption terminal"→ "external interrupt pin" "is needed." → "is also needed." - "4.1.Non-Maskable Interrupt (NMI)" 1st term: Added " For details of SIWDT, ... "Clock Selective Watchdog Timer"." 2nd term: Added " For details of LVD, ... "Voltage Detector Circuit"." - "4.2.Maskable Interrupt" "mask interuption" → "maskable interrupt" - "4.3.1.Interrupt Route" "two interrupt routes" → "tow interrupt transfer routes" Table4.1 F / Route Description: "interruption" → "interrupt release and an interrupt request" H / Interrupt Request: Added "(Note)" Added "Note: For the details of other interrupts, refer to "4.4.List ... "." - "4.3.4.Transmission of Interrupt Request" "interruption enable/" → "interrupt detection enable/" - "4.4. List of Interrupt Sources" Table4.2 Interrupt Request column: "Voltage detector circuit" → "LVD" "Watchdog timer" → "WDT" Above Table 4.3: "interrupt enable/disable" →"Interrupt detection enable/disable" Above Table 4.4: "interruption permission/prohibition" → "Interrupt detection enable/disable" Table4.5 Interrupt Request column: "Vector engine" → "Schedule end" - "4.6.1.Fowchart" Setting for detection / Details: "releasing the low power consumption mode" → "the Interrupt Control Register" - "4.6.2." (6): "permission of the interrupt" → "interrupt detection enable" "enabling an interrupt" → "enabling an interrupt detection" - "4.6.3.Detection(INTIF)" "interrupt enable/" → "interrupt detection enable/" "interrupt detection flag" → "<Detection flag>" "interrupt reuest" → "flag" "cleard" → "cleared" - "4.6.6." (2): "interruption flag" → "<Detection flag>" - "5.1. Register List" Table of Interrupt Monitor Registers: "Base address(+BASE)" → "Address(+Base)" - "5.5.2." Bit Symbol column: <INT96FLG> → <INT096FLG>, <INT97FLG> → <INT097FLG>, <INT98FLG> → <INT098FLG>, <INT99FLG> → <INT099FLG> - "5.6.5.Interrupt Control Registers" Added "Each bit corresponds to specified interrupt." - "6.1. MPM4L2/TMPM4L1" Interrupt No column: "NMI interrupt" → "NMI" Interrupt Request column: "Vector engine" → "Schedule end" "A-ENC32" → "A-ENC32 ch0" Added "✓: Available, —: N/A"

RESTRICTIONS ON PRODUCT USE

Toshiba Corporation and its subsidiaries and affiliates are collectively referred to as “TOSHIBA”.
Hardware, software and systems described in this document are collectively referred to as “Product”.

- TOSHIBA reserves the right to make changes to the information in this document and related Product without notice.
- This document and any information herein may not be reproduced without prior written permission from TOSHIBA. Even with TOSHIBA's written permission, reproduction is permissible only if reproduction is without alteration/omission.
- Though TOSHIBA works continually to improve Product's quality and reliability, Product can malfunction or fail. Customers are responsible for complying with safety standards and for providing adequate designs and safeguards for their hardware, software and systems which minimize risk and avoid situations in which a malfunction or failure of Product could cause loss of human life, bodily injury or damage to property, including data loss or corruption. Before customers use the Product, create designs including the Product, or incorporate the Product into their own applications, customers must also refer to and comply with (a) the latest versions of all relevant TOSHIBA information, including without limitation, this document, the specifications, the data sheets and application notes for Product and the precautions and conditions set forth in the "TOSHIBA Semiconductor Reliability Handbook" and (b) the instructions for the application with which the Product will be used with or for. Customers are solely responsible for all aspects of their own product design or applications, including but not limited to (a) determining the appropriateness of the use of this Product in such design or applications; (b) evaluating and determining the applicability of any information contained in this document, or in charts, diagrams, programs, algorithms, sample application circuits, or any other referenced documents; and (c) validating all operating parameters for such designs and applications. **TOSHIBA ASSUMES NO LIABILITY FOR CUSTOMERS' PRODUCT DESIGN OR APPLICATIONS.**
- **PRODUCT IS NEITHER INTENDED NOR WARRANTED FOR USE IN EQUIPMENTS OR SYSTEMS THAT REQUIRE EXTRAORDINARILY HIGH LEVELS OF QUALITY AND/OR RELIABILITY, AND/OR A MALFUNCTION OR FAILURE OF WHICH MAY CAUSE LOSS OF HUMAN LIFE, BODILY INJURY, SERIOUS PROPERTY DAMAGE AND/OR SERIOUS PUBLIC IMPACT ("UNINTENDED USE").** Except for specific applications as expressly stated in this document, Unintended Use includes, without limitation, equipment used in nuclear facilities, equipment used in the aerospace industry, lifesaving and/or life supporting medical equipment, equipment used for automobiles, trains, ships and other transportation, traffic signaling equipment, equipment used to control combustions or explosions, safety devices, elevators and escalators, and devices related to power plant. **IF YOU USE PRODUCT FOR UNINTENDED USE, TOSHIBA ASSUMES NO LIABILITY FOR PRODUCT.** For details, please contact your TOSHIBA sales representative or contact us via our website.
- Do not disassemble, analyze, reverse-engineer, alter, modify, translate or copy Product, whether in whole or in part.
- Product shall not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable laws or regulations.
- The information contained herein is presented only as guidance for Product use. No responsibility is assumed by TOSHIBA for any infringement of patents or any other intellectual property rights of third parties that may result from the use of Product. No license to any intellectual property right is granted by this document, whether express or implied, by estoppel or otherwise.
- **ABSENT A WRITTEN SIGNED AGREEMENT, EXCEPT AS PROVIDED IN THE RELEVANT TERMS AND CONDITIONS OF SALE FOR PRODUCT, AND TO THE MAXIMUM EXTENT ALLOWABLE BY LAW, TOSHIBA (1) ASSUMES NO LIABILITY WHATSOEVER, INCLUDING WITHOUT LIMITATION, INDIRECT, CONSEQUENTIAL, SPECIAL, OR INCIDENTAL DAMAGES OR LOSS, INCLUDING WITHOUT LIMITATION, LOSS OF PROFITS, LOSS OF OPPORTUNITIES, BUSINESS INTERRUPTION AND LOSS OF DATA, AND (2) DISCLAIMS ANY AND ALL EXPRESS OR IMPLIED WARRANTIES AND CONDITIONS RELATED TO SALE, USE OF PRODUCT, OR INFORMATION, INCLUDING WARRANTIES OR CONDITIONS OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, ACCURACY OF INFORMATION, OR NONINFRINGEMENT.**
- Do not use or otherwise make available Product or related software or technology for any military purposes, including without limitation, for the design, development, use, stockpiling or manufacturing of nuclear, chemical, or biological weapons or missile technology products (mass destruction weapons). Product and related software and technology may be controlled under the applicable export laws and regulations including, without limitation, the Japanese Foreign Exchange and Foreign Trade Law and the U.S. Export Administration Regulations. Export and re-export of Product or related software or technology are strictly prohibited except in compliance with all applicable export laws and regulations.
- Please contact your TOSHIBA sales representative for details as to environmental matters such as the RoHS compatibility of Product. Please use Product in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. **TOSHIBA ASSUMES NO LIABILITY FOR DAMAGES OR LOSSES OCCURRING AS A RESULT OF NONCOMPLIANCE WITH APPLICABLE LAWS AND REGULATIONS.**