Tsi148™ Initialization Application Note

80A3020_AN001_03

October 14, 2009

6024 Silver Creek Valley Road San Jose, California 95138

Telephone: (408) 284-8200 • FAX: (408) 284-3572

Printed in U.S.A.

GENERAL DISCLAIMER

Integrated Device Technology, Inc. ("IDT") reserves the right to make changes to its products or specifications at any time, without notice, in order to improve design or performance. IDT does not assume responsibility for use of any circuitry described herein other than the circuitry embodied in an IDT product. Disclosure of the information herein does not convey a license or any other right, by implication or otherwise, in any patent, trademark, or other intellectual property right of IDT. IDT products may contain errata which can affect product performance to a minor or immaterial degree. Current characterized errata will be made available upon request. Items identified herein as "reserved" or "undefined" are reserved for future definition. IDT does not assume responsibility for conflicts or incompatibilities arising from the future definition of such items. IDT products have not been designed, tested, or manufactured for use in, and thus are not warranted for, applications where the failure, malfunction, or any inaccuracy in the application carries a risk of death, serious bodily injury, or damage to tangible property. Code examples provided herein by IDT are for illustrative purposes only and should not be relied upon for developing applications. Any use of such code examples shall be at the user's sole risk.

1. Introduction

The Tsi148 application note is intended to be used for software and hardware initialization. It details the major functions of the Tsi148 and how to use them. The focus of the document is to get a board design working quickly and efficiently.

1.1 Background

The reader should be familiar with the following documents: *PCI Local Bus Specification (Revision 2.2), PCI-X Addendum to the PCI Local Bus Specification (Revision 1.0b), American National Standard for VME64, VME64 Extensions* and *Vita 1.5-2003 Standard for 2eSST.* For information on connecting the Tsi148 to the VMEbus through the transceivers please see the *Tsi148 Schematic Review Checklist* as well as the *Tsi148 Bridge User Manual*.

2. Hardware Initialization

Knowledge of the system design is very important before hardware initialization of the Tsi148. The following key system design characteristics must be considered:

- What is the power-up sequence?
- What is the PCI/X bus mode and clock frequency?
- Is the PCI/X bus 32-bit or 64-bit?
- Is the PCI/X connection multi-point or point-to-point?
- How is the CR/CSR Base Address being configured?
- Who is the VMEbus System Controller?

2.1 Power-up Sequencing

Power-up sequencing is the order in which different devices on the board are supplied power. The Tsi148 requires two voltages and a ground for proper operation: Digital Ground (VSS), I/O Buffer Supply (VDD33 = 3.3V), and the Core Supply (VDD18 = 1.8V).

2.1.1 How to Sequence Power-up of Tsi148

There are no sequencing requirements with respect to the Core Supply and the I/O Supply voltages.

It is required that the Tsi148 I/O supply be completely powered up before any of its I/O PADs (aka signal I/O pads or pins). If I/O PAD is powered up before VDD33, current can be drawn from the source connected to PAD, through the Tsi148 and into VDD33. This current could impact the operation or reliability of the part. There is no issue with I/O PADs having pull-ups to I/O Supply (i.e I/O PADs and I/O Supply being powered up at the same time).

2.2 PCI/X Bus Mode and Clock Frequency

The Tsi148 has one input clock (PCLK) located on the PCI/X port. The input clock must be present prior to the Tsi148 coming out of reset in order to lock the Tsi148 internal PLL. The Tsi148 PLL requires a minimum clock input frequency of 33 MHz. The Tsi148 will stay in a reset state until the clock is presented and will not come out of reset until the PLL has locked.

2.2.1 Configuring the PCI/X Bus Mode and Clock Frequency

The clock frequency and bus mode are configured on the rising edge of the Local Reset (PCI) input LRSTI_L as shown in the following table.

Table 1: PCI/X Bus Configuration

PCI/X Bus Signal						PCI Bus Mode	PCI/X Clock Frequency (MHz)	
FRAME_L	IRDY_L	DEVSEL_L	STOP_L	TRDY_L	M66EN		Min	Max
1	1	1	1	1	0	PCI	33.3	33.3
1	1	1	1	1	1	PCI	50	66.6
1	1	1	1	0	X	PCI-X	50	66.6
1	1	1	0	1	X	PCI-X	66.6	100
1	1	1	0	0	X	PCI-X	100	133.3

The PCI Specification does not require the PCI/X bus configuration signals to be valid until 10 clocks before the negation of PCI reset. The Tsi148 has more stringent requirements. The Tsi148 expects the PCI/X bus configuration signals to be valid upon the PCI clock starting and remain valid until the PCI reset signal (LRSTI_L) is negated. This allows the internal PLL to lock to the PCI bus clock. Please see Chapter 3 of the *Tsi148 Bridge User Manual* for more information.

2.3 PCI/X Bus Data Width

The Tsi148 supports both 32-bit or 64-bit PCI/X Bus widths. The PCI/X bus width is configured during a PCI/X bus reset (LRSTI_L). If REQ64_L is high (pull-up) during the rising edge of LRSTI_L, the PCI/X bus is configured for 32-bit mode. If REQ64_L is low (pulldown) during the rising edge of LRSTI_L, the PCI/X bus is configured for 64-bit mode. When the Tsi148 is used on a 32-bit PCI/X bus, it drives CBE[7:4]_L, AD[63:32], PAR64 and ACK64_L at all times. These signals may be left unconnected when operating on a 32-bit PCI/X bus.

2.4 PCI/X Driver Mode Control

The Tsi148 supports a power-up option that allows the user to control the PCI/X bus buffers drive strength. The value of the PCIMC pin is latched at power-up. When this signal is asserted (pull-up), the PCI/X drivers are configured with a 40 ohm impedance for point-to-point operation. When this signal is negated (pulldown), the PCI/X drivers are configured with a 20 ohm impedance for multi-point operation.

2.5 VMEbus Power-up Options

The Tsi148's VMEbus interface supports a number of power-up options which must be configured upon bringing up the device. These power-up options are used to determine how the CR/CSR Base Address is configured, System Controller functionality as well as the devices ability to drive the VMEbus System Fail Output (SFAILO).

The Tsi148 supports both Auto Slot ID and Geographical Slot ID methods of determining the CR/CSR Base Address. The CR/CSR Base Address is determined at power-up, the Tsi148 supports power-up options to determine the method. Power-up options are latched on the Tsi148's VMEbus interface during the assertion of PURSTI_L.

During PURSTI_L the Tsi148 negates the External Transceiver Enable (DBOE_) signal, therefore putting the data bus VD[31:0] transceivers into a high impedance state. External pull-ups or pulldowns placed between the Tsi148 and the external transceivers bring these power-up option signals to their proper state while DBOE_ is negated. The following table shows the data bus signal and corresponding functionality at power-up.

Table 2: VMEbus Data Signal Power-up Options

Description	Power-up Option	VMEbus Data Signal
SFAILEN Control Bit Reset Value	SFAILEN_RV	VD[0]
SFAILAI Control Bit Auto Clear	SFAILAI_AC	VD[1]
Auto Slot ID Enable	ASIDEN	VD[2]
Geographical Slot ID Enable	GSIDEN	VD[3]

2.5.1 Assigning CR/CSR Base Address

The ASIDEN and GSIDEN power-up options are used to define the Tsi148's CR/CSR Base Address. The table below describes the interaction of these two functions.

Table 3: ASIDEN and GSIDEN Definition

ASIDEN	GSIDEN	Description
0	0	CR/CSR Disabled
0	1	Geographical Address

Table 3: ASIDEN and GSIDEN Definition

ASIDEN	GSIDEN	Description
1	0	Auto Slot ID
1	1	Geographical Address defaults to Auto Slot ID if GA[4:0] pins are all high.

2.5.2 Auto Slot ID Enable (ASIDEN)

The Tsi148's ASIDEN feature is controlled through a power-up option, it allows the CR/CSR Base address to be configured using the Auto Slot ID protocol. The Auto Slot ID method of defining the CR/CSR Base Address register is detailed in the *American National Standard for VME64*.

2.5.3 System Failure Auto Slot ID Auto Clear (SFAILAI_AC)

The Tsi148's SFAILAI_AC feature is controlled through a power-up option, this can be used in conjunction with the Auto Slot ID method of determining CR/CSR Base Address. The Auto Slot ID method requires a device to negate its System Fail Output signal in order for the system interrupt handler to perform an interrupt service routine. The SFAILAI bit in the Tsi148's VMEbus Control Register must be cleared in order for the Tsi148 to negate it's System Fail Output signal. The SFAILAI_AC feature can be configured at power-up to clear the SFAILAI bit, therefore the Tsi148's System Fail Output is automatically negated. Otherwise this is done through software by clearing the SFAILAI bit in the VCTRL register.

2.5.4 Geographical Slot ID Enable (GSIDEN)

The Tsi148's GSIDEN feature is controlled through a power-up option, it allows the CR/CSR Base Address to be configured using the Geographical Addressing protocol. The Geographical Slot ID Enable feature allows a board to come out of reset with the CR/CSR registers visible from the VMEbus and the CR/CSR Base Address is determined by the VMEbus GA signals. The Geographical Addressing method of defining the CR/CSR Base Address Register is detailed in the *American National Standard for VME64*. Please see the *Tsi148 Bridge User Manual* for additional information.

2.5.5 System Fail Enable Reset Value (SFAILEN_RV)

The SFAILEN_RV feature is controlled through a power-up option, it can be used to determine the initial value of the Tsi148's SFAILEN bit at power-up. The Tsi148's System Failure Enable (SFAILEN) bit is used in conjunction with the Board Fail (BDFAIL_) signal to control the assertion of the devices System Failure Output signal (SFAILO). Please see the *Tsi148 Bridge User Manual* for additional information.

The SFAILEN_RV power-up option must be cleared when using the Auto Slot ID method of determining the CR/CSR Base Address.

2.5.6 System Controller (SCON)

The Tsi148's System Controller functionality is also determined at power-up. The SCONEN_ and SCONDIS_ signals are used to control the SCON function. The values of these signals are controlled through pull-ups and pulldowns on the board and are sampled at the rising edge of PURSTI_L. If both SCONEN_ and SCONDIS_ are both high at power-up, the Auto System Controller feature is used. The following table describes the Tsi148 SCON configuration.

Table 4: Tsi148 System Controller Configuration

Function	Reset	Signals	State	Description
SCON	PURSTI_L	SCONEN_ SCONDIS_	0 1	SCON Enabled
		SCONEN_ SCONDIS_	1 0	SCON Disabled
		SCONEN_ SCONDIS_ BG3IN_	1 1 0	Auto System Controller SCON Enabled
		SCONEN_ SCONDIS_ BG3IN_	1 1 1	Auto System Controller SCON Disabled

3. Software Initialization

This section will detail the necessary registers accesses to the Tsi148's internal registers in order to get the part up and running. It is assumed that the correct hardware configuration has been chosen for the given design. The Tsi148 contains 4KBytes of register space for initialization and configuration. The complete Tsi148 programming model can be viewed in Chapter 6 of the *Tsi148 Bridge User Manual*.

In order to initialize the Tsi148, the relevant PCI and VME bus specifications, *Tsi148 Bridge User Manual* and the overall system configuration must be understood. The Tsi148 can be configured from either the PCI/X or VMEbus depending on the system architecture requirements.

The following section describes only major functions of the Tsi148 software initialization.

The following system characteristics must be determined during system design:

- What is the VMEbus memory map?
- What is the PCI/X bus memory map?
- What is the interrupt scheme?

Using software, the Tsi148 can be initialized from the PCI/X bus or the VMEbus. The Tsi148 internal registers can be accessed in various ways as described below.

3.1 Software initialization from PCI/X and VMEbus interfaces

This section will detail the necessary register accesses that the Tsi148 must receive to allow its internal registers to be visible from both the VME and PCI/X buses.

3.1.1 How to initialize from the PCI/X Bus

The following Tsi148 registers must be accessed through PCI/X configuration cycles in order to get access to the full 4KBytes of internal register space.

1. The MEMSP bit in the Command/Status Register at offset 0x04 to allow the Tsi148 to accept memory cycles as a PCI/X Target.
2. The base address field (BASEL) within the Memory Base Address Lower Register offset 0x10 has to be initialized for the Tsi148 to know what address range to decode for access to the Tsi148 registers. If 64-bit memory space is required the base address field (BASEU) within the Memory Base Address Upper Register offset 0x14 must be initialized as well. Once these fields are programmed the Tsi148's entire 4KBytes of register space is accessible via the PCI/X bus through memory cycles.

3.1.2 How to initialize from the VMEbus

The following Tsi148 registers must be accessed through the CR/CSR space using the special CR/CSR AM code to allow access to the devices internal register space. Once the CR/CSR Base Address has been defined through the Auto Slot ID or Geographical Slot ID methods the entire 4KBytes of internal register space is completely accessible.

1. Once the CR/CSR Base Address has been defined through the Auto Slot ID or Geographical Slot ID methods the Combined Register Group (CRG) base address can be programmed. The CRG image maps the entire 4KBytes of Tsi148 internal register space to the VMEbus. The CRG image can be programmed from the VMEbus using the special CR/CSR AM code or be programmed from the PCI/X bus. There are 2 to 3 registers that must be written to setup the CRG image. The first is the CRG Attribute Register at offset 0x414, the CRG image is enabled through this register and defines VMEbus address space and access types the CRG decoder will respond to. The second is the base address field (CBAL) within the CRG Base Address Lower Register offset 0x410, this has to be initialized for the Tsi148 to know what address range to decode for access to the Tsi148 registers. The third is optional, if A64 cycles are required the base address field (CBAU) within the CRG Base Address Upper Register offset 0x40C must be initialized as well. Once these fields are programmed the Tsi148's entire 4KBytes of register space is accessible via the VMEbus.

3.2 Initializing PCI/X Target Images

In order to communicate through the Tsi148 from the PCI/X bus to the VMEbus a PCI/X target image must be initialized with software. A PCI/X bus target image is an address range (in PCI/X address space) where the Tsi148 responds to a PCI/X bus initiator's request. If the PCI/X initiator's request falls within one of the Tsi148's PCI/X bus target image address ranges, the transaction (read or write) is forwarded to the VMEbus. The Tsi148 supports up to eight unique PCI/X Target Images. For more information about the Tsi148's PCI/X target interface see Chapter 2 of the *Tsi148 Bridge User Manual*.

3.2.1 How to initialize the PCI/X Target Images

The following registers are initialized using PCI/X memory write cycles from the PCI/X bus. These registers can also be initialized from the VMEbus as part of the CRG image. If address translation is not a requirement the translation offsets associated with each image are optional and do not have to be initialized. The table below describes all of the internal registers associated with the PCI/X Target Images.

Table 5: PCI/X Target Image Register Initialization

Register	Field	Description
OTSAUx	STAU	Start Address Upper. Initialized so the Tsi148 knows what upper starting address range to decode for the target image. The value of this field will be compared with AD63-AD32 of the PCI/X bus address. The PCI/X target address is decoded when the PCI/X address is greater than or equal to the start address and less than or equal to the end address.
OTSALx	STAL	Start Address Lower. Initialized so that the Tsi148 knows what lower starting address range to decode for the target image. The value of this field will be compared with AD31-AD16 of the PCI/X bus address. The PCI/X target address is decoded when the PCI/X address is greater than or equal to the start address and less than or equal to the end address.
OTEAUx	ENDU	End Address Upper. Initialized so that the Tsi148 knows what upper ending address range to decode for the target image. The value of this field will be compared with AD63-AD32 of the PCI/X bus address. The PCI/X target address is decoded when the PCI/X address is greater than or equal to the start address and less than or equal to the end address.
OTEALx	ENDL	End Address Lower. Initialized so that the Tsi148 knows what lower ending address range to decode for the target image. The value of this field will be compared with AD31-AD16 of the PCI/X bus address. The PCI/X target address is decoded when the PCI/X address is greater than or equal to the start address and less than or equal to the end address.
OTOFUx	OFFU	Translation Offset Upper. This field contains the offset that is added to PCI/X address lines AD63-AD32 to create the VMEbus address.
OTOFLx	OFFL	Translation Offset Lower. This field contains the offset that is added to PCI/X address lines AD31-AD16 to create the VMEbus address.
OTBSx	2eBS	2eSST Broadcast Select. This register is optional, it is used for 2eSST broadcast.
OTATx	EN	Translation attribute. This bit enables the PCI/X Target Image and should be set last. This register also contains all fields required to determine how the transaction is initiated on the VMEbus.
Tsi148 supports up to 8 PCI/X Target Images x=7:0. Please see the *Tsi148 Bridge User Manual* Chapter 6.		

3.3 Initializing the VMEbus Slave Images

In order to communicate through the Tsi148 from the VMEbus to the PCI/X bus a VMEbus slave image must be initialized with software. A VMEbus slave image is an address range (in VMEbus address space) where the Tsi148 responds to a VMEbus initiator's request. If the VMEbus initiator's request falls within one of the Tsi148's VMEbus bus slave image address ranges, the transaction (read or write) is forwarded to the PCI/X bus. The Tsi148 supports up to eight unique VMEbus Slave Images. For more information about the Tsi148's VMEbus slave interface see Chapter 2 of the *Tsi148 Bridge User Manual*.

3.3.1 How to Initialize the VMEbus Slave Images

VMEbus Slave images must be initialized in order for the Tsi148 to accept transactions targeted for the PCI/X bus. The VMEbus slave images can be initialized either from the PCI/X bus or from the VMEbus as part of the CRG image. If address translation is not a requirement the translation offsets associated with each image are optional and do not have to be initialized. The table below describes all of the internal registers associated with the VMEbus Slave Images.

Table 6: VMEbus Slave Image Register Initialization

Register	Field	Description
ITSAUx	STAU	Start Address Upper. Initialized so the Tsi148 knows what upper starting address range to decode for the slave image. If the VMEbus address is 64-bit, the value of this field will be compared with VMEbus address bits A63-A32. The VMEbus slave address is decoded when the VME address is greater than or equal to the start address and less than or equal to the end address.
ITSALx	STAL	Start Address Lower. Initialized so the Tsi148 knows what lower starting address range to decode for the slave image. If the VMEbus address is 64-bit or 32-bit, then the start address lower bits 31 to 16 are compared with VMEbus address bits A31-A16. If the VMEbus address is 24-bits, then the start address lower bits 23 to 12 are compared with VMEbus address bits A23-A12. If the VMEbus address is 16 bits, then the start address lower bits 15 to 4 are compared with VMEbus address bits A15-A4. The VMEbus slave address is decoded when the VME address is greater than or equal to the start address and less than or equal to the end address.
ITEAUx	ENDU	End Address Upper. Initialized so that the Tsi148 knows what upper ending address range to decode for the slave image. If the VMEbus address is 64-bit, the value of this field will be compared with VMEbus address bits A63-A32. The VMEbus slave address is decoded when the VME address is greater than or equal to the start address and less than or equal to the end address.
ITEALx	ENDL	End Address Lower. Initialized so the Tsi148 knows what lower ending address range to decode for the slave image. If the VMEbus address is 64-bit or 32-bit, then the end address lower bits 31 to 16 are compared with VMEbus address bits A31-A16. If the VMEbus address is 24-bits, then the end address lower bits 23 to 12 are compared with VMEbus address bits A23-A12. If the VMEbus address is 16 bits, then the end address lower bits 15 to 4 are compared with VMEbus address bits A15-A4. The VMEbus slave address is decoded when the VME address is greater than or equal to the start address and less than or equal to the end address.
Tsi148 supports up to 8 VMEbus Slave Images x=7:0. Please see the *Tsi148 Bridge User Manual* Chapter 6.		

Table 6: VMEbus Slave Image Register Initialization

Register	Field	Description
ITOFUx	OFFU	Translation Offset Upper. This field contains the offset that is added to VMEbus address bits A63-A32 to create the PCI/X bus address. If the VMEbus address is not 64-bit, then the internal VMEbus address bits 63 to 32 are zeroed before the offset is added.
ITOFLx	OFFL	Translation Offset Lower. This field contains the offset that is added to the lower VMEbus address bits to create the PCI/X bus address. If the VMEbus address is 24-bit, then the internal VMEbus address bits 31 to 24 are zeroed and then offset bits 31 to 12 are added. If the VMEbus address is 16-bit, then the internal VMEbus address bits 31 to 16 are zeroed and offset bits 31 to 4 are added.
ITATx	EN	Translation Attribute. This bit enables the VMEbus Slave Image and should be set last. This register also contains all fields required to determine what VMEbus cycles the slave image will respond to as well as the pre-fetching on the PCI/X bus.
Tsi148 supports up to 8 VMEbus Slave Images x=7:0. Please see the *Tsi148 Bridge User Manual* Chapter 6.		

3.4 Interrupts

The Tsi148 can be programmed to act as an interrupter and an interrupt handler in a VME system by accessing internal registers through software. The Tsi148 is designed so that the interrupt handling intelligence is expected to exist on the local PCI/X bus. Tsi148's local (PCI/X) interrupter provides a mechanism to control the interrupts generated by internal and external sources. The local interrupter receives interrupts from internal (Tsi148) and external (VMEbus, PCI/X bus) sources, these interrupts can be routed to one of four interrupt output lines. For a complete list of sources please see Chapter 1 of the *Tsi148 Bridge User Manual*.

Interrupt schemes are configured by mapping the proper interrupt to the proper interrupt pin, the Tsi148 supports 4 interrupt output pins on the local PCI/X bus: INTA_, INTB_, INTC_ and INTD_. The Tsi148 also capable of generating 7 software controlled interrupt outputs on the VMEbus IRQ[7:1]O. The status of a given interrupt can be ascertained by reading the interrupt status register. An interrupt can be cleared by writing the interrupt clear register.

3.4.1 How to generate VMEbus Interrupts

The Tsi148 generates interrupts on the VMEbus as follows:

1. The STID and IRQL fields must be set in the VMEbus Interrupt Control Register (VICR at offset 0x440). The IRQL field defines the level of the VMEbus interrupt output signals (IRQ[7:1]O). The STID field defines the VMEbus vector that will be returned on the interrupt acknowledge cycle. A VMEbus interrupt is generated when the IRQL field is written. Only one interrupt at a time can be generated.

3.4.2 How to generate VMEbus IACK cycles

The Tsi148 has seven VMEbus Interrupt Acknowledge Registers which, when read, generate an IACK Cycle on the VMEbus. There is one IACK register for each of the VMEbus IRQ[7:1]I_ inputs. The VMEbus IACK registers are located at offsets 0x204 - 0x21C. The interrupt handler has the following features:

- Supports 8, 16, and 32-bit IACK cycles
  - A word read of an IACK register causes a 32-bit VMEbus IACK cycle
  - A half-word read causes a 16-bit VMEbus IACK cycle
  - A byte read causes an 8-bit VMEbus IACK cycle
- Once the IACK cycle is generated the interrupter supplies its status/ID.

3.4.3 How to generate a Local (PCI/X) bus interrupt

As previously mentioned the Tsi148 expects the interrupt handling intelligence to exist on the Local bus. All internal and external sources of interrupts can be mapped to one of the four Local bus interrupt outputs (INTA_, INTB_, INTC_ and INTD_). The Tsi148 provides various ways of dealing with interrupts by utilizing the following interrupt registers.

Table 7: Tsi148 Interrupt enabling/routing

Register	Fields	Description
INTEN	DMA1EN, DMA0EN, LM3EN, LM2EN, LM1EN, LM0EN, MB3EN, MB2EN, MB1EN, MB0EN, PERREN, VERREN, VIEEN, IACKEN, SYSFLEN, ACFLEN, IRQ7EN, IRQ6EN, IRQ5EN, IRQ4EN, IRQ3EN, IRQ2EN, IRQ1EN	When any of these bits are high the corresponding interrupt is enabled.
INTEO	DMA1EO, DMA0EO, LM3EO, LM2EO, LM1EO, LM0EO, MB3EO, MB2EO, MB1EO, MB0EO, PERREO, VERREO, VIEEO, IACKEO, SYSFLEO, ACFLEO, IRQ7EO, IRQ6EO, IRQ5EO, IRQ4EO, IRQ3EO, IRQ2EO, IRQ1EO	When any of these bits are high the corresponding interrupt is enabled to one of the 4 PCI/X INTx outputs.
INTS	DMA1S, DMA0S, LM3S, LM2S, LM1S, LM0S, MB3S, MB2S, MB1S, MB0S, PERRS, VERRS, VIES, IACKS, SYSFLS, ACFLS, IRQ7S, IRQ6S, IRQ5S, IRQ4S, IRQ3S, IRQ2S, IRQ1S	When any of these status bits are high the interrupt is pending.

Table 7: Tsi148 Interrupt enabling/routing

Register	Fields	Description
INTC	DMA1C, DMA0C, LM3C, LM2C, LM1C, LM0C, MB3C, MB2C, MB1C, MB0C, PERRC, VERRC, VIEC, IACKC, SYSFLC, ACFLC	When these bits are written with a 1 the corresponding interrupt is cleared.
INTM1	DMA1M, DMA0M, LM3M, LM2M, LM1M, LM0M, MB3M, MB2M, MB1M, MB0M	These 2 bit fields indicate which PCI/X INTx output the interrupt is routed to.
INTM2	PERRM, VERRM, VIEM, IACKM, SYSFLM, ACFLM, IRQ7M, IRQ6M, IRQ5M, IRQ4M, IRQ3M, IRQ2M, IRQ1M	

IMPORTANT NOTICE AND DISCLAIMER

RENESAS ELECTRONICS CORPORATION AND ITS SUBSIDIARIES (“RENESAS”) PROVIDES TECHNICAL SPECIFICATIONS AND RELIABILITY DATA (INCLUDING DATASHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING, WITHOUT LIMITATION, ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT OF THIRD-PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for developers who are designing with Renesas products. You are solely responsible for (1) selecting the appropriate products for your application, (2) designing, validating, and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, or other requirements. These resources are subject to change without notice. Renesas grants you permission to use these resources only to develop an application that uses Renesas products. Other reproduction or use of these resources is strictly prohibited. No license is granted to any other Renesas intellectual property or to any third-party intellectual property. Renesas disclaims responsibility for, and you will fully indemnify Renesas and its representatives against, any claims, damages, costs, losses, or liabilities arising from your use of these resources. Renesas' products are provided only subject to Renesas' Terms and Conditions of Sale or other applicable terms agreed to in writing. No use of any Renesas resources expands or otherwise alters any applicable warranties or warranty disclaimers for these products.

(Disclaimer Rev.1.01)

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan
www.renesas.com

Contact Information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit www.renesas.com/contact-us/.

Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.