

**FDAS MSIX Module (Extended Message Signalling Interrupts) Design Specification for Intel Agilex Implementation**

FDAS\_MSIX\_DS Revision 1 Draft B

Classification: UNRESTRICTED

Document type: DTE

Date: 2022-04-28

Status: DRAFT

© Copyright 2022 SKA Observatory.

  This work is licensed under a [Creative Commons Attribution 4.0 International License](http://creativecommons.org/licenses/by/4.0/)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *Role* | *Name* | *Designation* | *Affiliation* | *Signature* | *Date* |
| Author | Martin Droog | FPGA Design Engineer | SKAO |  | 28/04/2022 |
| Owner | Ben Stappers | Head of SKA PSS Team | SKAO |  |  |
| Approver | Ben Stappers | Head of SKA PSS Team | SKAO |  |  |
| Released by | Lina Levin Preston | SKA PSS Team Scrum Master | SKAO |  |  |

|  |  |  |
| --- | --- | --- |
| Document History | | |
| Issue | Date | Comments |
| Issue 1 Draft A | 19/04/2022 | First Issue |
| Issue 1 Draft B | 28/04/2022 | * Intel Agilex PCIe User Guide reference added |

**Table of Contents**

1 Introduction 4

2 Place in the System 5

3 Functional Specification 6

3.1 MSIX Module Architecture 8

3.2 Done Pulse Detection: PULSEDET Sub-Module 8

3.3 Micro Interface Configuration: MSIXMCI Sub-Module 9

3.3.1 Write to MSIXMCI 9

3.3.2 Read from MSIXMCI 10

3.4 TXINT Write Generation: TXINT Sub-Module 12

4 Interface Specification 14

5 MCI Memory Mapped Interface 16

5.1 CLD MSI-X Interrupt Configuration 16

5.2 CONV MSI-X Interrupt Configuration 16

5.3 HSUM MSI-X Interrupt Configuration 17

6 Design Parameterisation 18

7 References 19

8 Abbreviations and Acronyms 20

# Introduction

This document captures the requirements for the MSIX module used in the Intel Agilex family implementation of the FDAS FPGA.

The MSIX module is required to receive the ‘DONE’ signals from the core processing modules and generate MSI-X (Extended Message Signalled Interrupt) interrupts to the host computer/PC when they are activated.

The MSI-X interrupt is generated by sending an “Interrupt Vector” to the computer/PC.

A table containing the various Interrupt Vector values resides in the Intel Agilex PCIe Hard IP Macro [Ref: Agilex PCIe User Guide] and is configured during PCIe enumeration by the PC/Computer. The MSI-X module when in receipt of an activated “DONE” signal from the FDAS core processing modules (i.e. CLD\_DONE, CONV\_DONE & HSUM\_DONE) sends an appropriate 16-bit value to the PCIe Hard IP Macro to address the desired location of the Interrupt Vector Table, thus causing the Interrupt Vector value stored at that table location to be sent to the PC/Computer over the PCIe interface.

To provide flexibility the 16-bit value sent by the MSIX module shall be configurable (via the Micro Configuration interface) for each for each of the “DONE” signals from the FDAS core processing modules.

The System Clock “CLK\_SYS” on which domain the “DONE” signals are generated and the “CLK\_PCIE” on which the PCIe Hard IP Macro operates shall both be 350MHz although no phase relationship should be assumed between these two clocks.

The Micro Configuration Clock “CLK\_MC” shall operate at up to 350MHz and no phase relationship should be assumed between the CLK\_MC clock and the CLK\_SYS and CLK\_PCIE clocks.

# Place in the System

The MSI Module’s place in the system is shown highlighted in the figure below:-

PCIe

**CLD**

**CTRL**

FDAS Control

CLD\_DONE

**CONV**

data[63:0]

ready

**HSUM**

**FDAS**

CONV\_DONE CONV\_FFT\_REDAY

valid

**PCIE HARD IP MACRO (INTEL IP)**

**MCI\_TOP**

CLD\_DM\_TRIGGER

CLD\_ENABLE

CLD\_PAGE[31:0]

CONV\_DM\_TRIGGER

CONV\_ENABLE

CONV\_PAGE[31:0]

IFFT\_LOOP\_NUM[5:0]

HSUM\_DM\_TRIGGER

HSUM\_ENABLE

HSUM\_PAGE[31:0]

HSUM\_DONE

**PCIF**

ADDR

DATA[511:0]

**DDRIF2 #1**

**DDRIF2 #2**

ADDR

DATA[511:0]

DATA[511:0]

ADDR

ADDR

DATA511:0]

*Note: The PCIe Hard Macro can read and write to both External DDR memories for diagnostic purposes. Not shown in this figure for clarity.*

*Note: CLD, CONV and HSUM are designed with generically sized data width interfaces to DDRIF2 for a future implementation, but in this implementation the DDRIF2 data width is fixed. CLD, CONV and HSUM are also designed with paging of the DDR memory for a future implementation.*

**FDAS\_DDR\_CONTROLLER\_**

**CALIBRATION (INTEL IP) #1**

**FDAS\_DDR\_CONTROLLER\_**

**CALIBRATION (INTEL IP) #2**

DMA Transfer

**External 4 Gibi-Byte**

**DDR4 SDRAM Memory #1**

**External 4 Gibi-Byte**

**DDR4 SDRAM Memory #2**

MC

Interface

Via

PCIe

*Note: In this implementation CLD\_PAGE, CONV\_PAGE and HSUM\_PAGE are fixed (static) in this implementation. However with a future implementation with more DDR Interfaces to CONV/HSUM a Paging technique shall enable increased processing performance.*

MC Bus

OVERLAP\_SIZE[9:0]

FOP\_SAMPLE\_NUM[22:0]

**MSIX**

MSI-X

Figure 2‑1 : MSIX Location in FDAS

# Functional Specification

h2ddmWrite\_o

h2ddm\_address\_o[63:0]

h2ddm\_writedata511:0]

h2ddm\_burstcount\_o[3:0]

h2dmm\_byteenable\_o[31:0]]

h2dmm\_waitrequest\_i

d2hdm\_read\_o

d2hdm\_address\_o[63:0]

d2hdm\_byteenable\_o[31:0]

d2hdm\_burstcount\_o[3:0]

d2mdm\_readdata\_i[511:0]

d2hdm\_waitrequest\_i

d2hdm\_readdatavalid\_i

d2hdm\_response\_i{1:0]

rx\_pio\_read\_o

rx\_pio\_write\_o

rx\_pio\_address\_o[m:0]

rx\_pio\_burstcount\_o[3:0]

rx\_io\_byteenable\_o[7:0]

rx\_pio\_writedata\_o[63:0]

rx\_pio\_readdata\_i[63:0]

rx\_pio\_readdatavalid\_i

rx\_pio\_waitrequest\_i

rx\_pio\_response\_i[1:0]

rx\_pio\_writeresponsevalid\_i

usr\_event\_msix\_ready\_o

usr\_even\_msix\_valid\_i

usr\_event\_msix\_data\_i[15:0]

**Agilex**

**PCIe Hard IP with**

**DMA Controller and Avalon MM Bridge**

app\_clk

refclk0

refclk1

app\_rst\_n

ninit\_done

pin\_perst

p0\_pld\_link\_req\_rst\_o

p0\_pld\_warm\_rst\_rdy\_i

rx\_out0[15:0:0]

rx\_in0[15:0:0]

Resets

Serial

Data

(PCIe)

**DDRIF2 #1**

**MCI\_TOP**

**PCIF**

**FDAS**

Avalon MM Rx Master

Module Interface

Host to Device Avalon

MM Master Interface

(512-bit data)

Device to Host

DMA Avalon

MM Master Interface

(512 -bit data)

**DDRIF2 #2**

refclk0

**TAB**

**TAB**

**TAB**

**TAB**

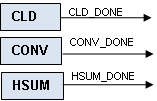
Notes:-

* TAB = Transparent Avalon MM Bridge to allow address decodes to select the appropriate DDRIF2 module
* For Clarity the Clock/Reset Bridges are not shown

**PCIE\_HIP\_FDAS**

refclk1

**MSIX**



Config

User MSI-X

Interface

Figure 3‑1: MSIX Top Level Connectivity in FDAS

The above figure shows the main connectivity of the MSIX module within the FDAS FPGA.

The following interfaces of the Intel Agilex PCIe Hard IP macro connect to the MSIX Module:-

* **User MSI-X Interface**: This interface supports the sending of MSI-X (Extended Message Signalled Interrupts) by the FDAS FPGA to the Host PC via the PCIe Interface.

The PCIe Hard IP Macro contains the MSI-X “Interrupt Vector Table” that is used by both the DMA engine and the User Interrupts to signal to the PC/Computer that an event (i.e. DMA or User) has occurred. The table has four entries per DMA channel (also called Queue), with a maximum of 512 DMA channels, with each entry containing an Interrupt vector configured by the PC/Computer. For each DMA channel (Queue) two of the entries are needed for DMA operation Interrupts and two are available for User Interrupts. The lower two bits of the address to the Interrupt vector table select the Interrupt:-

**Address[1:0] Interrupt**

“00” Host to Device DMA Interrupt

“01” Host to Device User MSI-X

“10” Device to Host DMA Interrupt

“11” Device to Host User MSI-X

The User MSI-X interface allows the desired entry to be selected via a 16-bit value “user\_event\_msix\_data\_i[15:0]” signal from the MSIX module:-:

* + user\_event\_msix\_data\_i[15:12] = rsvd[3:0],
  + user\_event\_msix\_data\_i[11] = msix\_queue\_dir
  + user\_event\_msix\_data\_i[10:0] = msix\_queue\_num\_i[10:0]

msix\_queue\_dir = 1 for Host to Device and msix\_queue\_dir = 0 for Device to Host. msix\_queue\_dir is effectively selecting the bottom two bits of the Interrupt Vector Table addressing (i.e. “01” or “11”) and msix\_queue\_num\_i[10:0] is selecting the DMA channel.

The “usr\_event\_msix\_ready\_o” signal to the MSIX module indicates when the PCIe Hard IP Macro is ready to accept an Interrupt request and the “usr\_event\_msix\_valid\_i” signal from the MSIX module indicates when the “user\_event\_msix\_data\_i[15:0]” signal information is valid to allow capture by the PCIe Hard IP Macro.

## MSIX Module Architecture

The figure below shows the top-level architecture of the MSIX Module:-

CLD\_DONE

**MSIX**

CLK\_PCIE

RST\_PCIE\_N

CONV\_DONE

HSUM\_DONE

RST\_SYS\_N

CLK\_SYS

MCADDR[3:0]

MCDATAIN31:0]

CLD\_MSIX\_DATA[15:0]

USR\_EVENT\_MSIX\_READY

USR\_EVENT\_MSIX\_DATA[15:0]

USR\_EVENT\_MSIX\_VALID

TXINT

PULSEDET

CLD\_TOGGLE

CONV\_TOGGLE

HSUM\_TOGGLE

MSIXMCI

CLK\_MC

RST\_MC\_N

MCDATAOUT[31:0]

MCRWN

MCMS

CONV\_MSIX\_DATA[15:0]

HSUM\_MSIX\_DATA[15:0]

CLD\_MSIX\_EN

CONV\_MSIX\_EN

HSUM\_MSIX\_EN

Figure 3‑2 : MSIX Architecture Block Diagram

## Done Pulse Detection: PULSEDET Sub-Module

The purpose of this module is to allow a pulse on the ‘xxx\_DONE’ inputs, which are on the CLK\_SYS clock domain, to be detected in the CLK\_PCIe clock domain. This shall be achieved by detecting a rising-edge of the ‘xxx\_DONE’ signal that shall then cause the respective ‘xxx\_TOGGLE’ output to change state.

## Micro Interface Configuration: MSIXMCI Sub-Module

The purpose of this module is to allow the user to configure the data for each Interrupt source (i.e. xxx\_DONE signal) which is ultimately used to address the Interrupt Vector Table within the Intel Agilex PCIe Hard IP Macro. There is also configuration to allow each Interrupt source to enable MSIX Interrupts.

### Write to MSIXMCI

When MCMS =1 and MCRWN = 0, if MCADDR[3:0] is a valid address of the memory map the value of MCDATAIN[31:0] shall be written. This operation shall be synchronous to CLK\_MC, with the D-type storage identified by MCADDR[3:0] being loaded with MCDATAIN[31:0] each clock cycle at which retimed MCMS =1 and MCRWN = 0.

All internal stored values in memory shall be set to 0 if the asynchronous reset “RST\_MC\_N” is asserted (active low).

The figure below shows a write access with the minimum access time that is required to store the data.

Figure 3‑3 : MSIXMCI Write Access Timing Diagram showing the minimum access time

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  | A |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  | D |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  | D |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

MCDATAIN[31:0]

MCADDR[3:0]

MCMS

Internal Storage at Location “A”

CLK\_MC

**DATAIN “D” stored at MCADDR “A” here, hence the access can complete on this cycle**

MCRWN

### Read from MSIXMCI

When MCMS =1, if MCADDR[3:0] is a valid address of the memory map the value of MCDATAOUT[31;0] shall be set to the value of the targeted location. This operation shall be synchronous to CLK\_MC, with the targeted location identified by MCADDR[3:0] driving MCDATAOUT[31:0] each clock cycle at which retime MCMS =1.

If MCMS = 0 or MCADDR[3:0] is not a valid address of the memory map MCDATAOUT[31:0] shall be set to 0x0000,0000

The figure below shows a Read access with the minimum access time that is required to read the data.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  | |  |  | |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  | |  |  | |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  | |  |  | |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  | |  |  | |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  | Don’t care for Read Access | | | | | | | | |  |  |  |  |  |  |  |
|  |  |  |  | |  |  | |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  | A | |  |  | |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  | |  |  | |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  | D |  |  |  |  |  |  |  |  |
|  |  |  |  | |  |  | |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  | |  |  | |  |  |  |  |  |  |  |  |  |  |  |

MCDATAOUT[31:0]

MCADDR[3:0]

MCMS

CLK\_MC

**MCDATAOUT “D” from MCADDR “A” available here to be sampled.**

MCRWN

Figure 3‑4 : MSIXMCI Read Access Timing Diagram showing the minimum access time

## TXINT Write Generation: TXINT Sub-Module

The purpose of this module is to pass the configured data for the activated Interrupt source (i.e. xxx\_DONE signal) to the PCIe Hard IP Macro. The data is only passed whenever any of the ‘xxx\_TOGGLE’ inputs change state and the associated interrupt source is enabled by the user via MC configuration.

The ‘xxx\_TOGGLE’ inputs shall be meta-stabilized onto the PCIe clock ‘CLK\_PCIE’ and then edge-detected.

When more than one toggle input transitions simultaneously, or close together, the appropriate user configured data for each input interrupt source that has transitioned shall still be passed to the PCIe Hard IP Macro.. The order of data transfer to the PCIe Hard IP Macro in these “simultaneous” situations is not important. It may be assumed, however, that a particular input interrupt source will not transition again before the data for that interrupt source has been passed to the PCIe Hard IP Macro.

The timing diagram below shows a transfer of usr\_event\_msix\_data[15:0] from the MSIX module to the Intel Agilex PCIe Hard IP Macro.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

USR\_EVENT\_MSIX\_

READY

USR\_EVENT\_MSIX\_

DATA[15:0]

USR\_EVENT\_MSIX\_

VALID

CLK\_PCIE

**For each transfer The MSIX module should de-assert the USR\_EVENT\_MSIX\_VALID signal when the USR\_EVENT\_MSIX\_READY signal and the USR\_EVENT\_MSIX\_VALID are both asserted.**

**For “back-to-back” transfers the USR\_EVENT\_MSIX\_VALID**

**shall be de-asserted for one cycle between the two transfers.**

Figure 3‑5 : USR\_EVENT\_MSIX Interface Timing Diagram

# Interface Specification

| Signal | Direction | Clock Domain | Description |
| --- | --- | --- | --- |
| **User MSI-X Interface to Intel Agilex PCIe Hard IP Macro** | | | |
| USR\_EVENT\_MSIX\_VALID | OUT | CLK\_PCIE | Signal to indicate the USR\_EVENT\_MSIX\_DATA[15:0] information from the MSIX Module is valid  ‘1’ = Valid |
| USR\_EVENT\_MSIX\_DATA[15:0] | OUT | CLK\_PCIE | User Configured Data value for the activated Interrupt:-   * + rsvd[3:0],   + msix\_queue\_dir   + msix\_queue\_num\_i[10:0]   This information is used to address the Interrupt Vector Table within the Intel Agilex PCIe Hard IP Macro to cause the appropriate Interrupt Vector to be sent over the PCIe Interface to the PC/Computer. |
| USR\_EVENT\_MSIX\_REDAY | IN | CLK\_PCIE | Signal to indicate that the Intel Agilex PCIe Hard IP Macro is ready to accept a new USR\_EVENT\_MSIX\_DATA[15:0] value. |
| **Interface to MCI\_TOP** | | | |
| MCMS | IN | CLK\_MC | Module Select. When MCMS =1 the MSIX module is selected by the MCI\_TOP module. |
| MCADDR[3:0] | IN | CLK\_MC | Micro Configuration Address.  [3] = msb |
| MCRWN | IN | CLK\_MC | Micro Configuration Read Not Write.  ‘0’ = Write to MSIX module.  A read access from the MSIX module does not care about the state of MCRWN. The MCI\_TOP module generates MCRWN and controls the direction of the data bus to the PCIe Hard IP Macro. |
| MCDATAIN[31:0] | IN | CLK\_MC | Micro Configuration Data In from MCI\_TOP. This data will be written to address location MCADDR if MCCSN =0 and MCRWN = 0.  [31] = msb |
| MCDATAOUT[31:0] | OUT | CLK\_MC | Micro Configuration Data Out to MCI\_TOP. This data will be valid from a valid MCADDR if MCMS =1  If MCMS = 0 or MCADDR is invalid then MCDATAOUT will be 0x00000000  [31] = msb |
| **Done Signals from FDAS Processing Modules** | | | |
| CLD\_DONE | IN | CLK\_SYS | Pulse indicating CLD processing is complete. ‘1’ = done. |
| CONV\_DONE | IN | CLK\_SYS | Pulse indicating CONV processing is complete. ‘1’ = done. |
| HSUM\_DONE | IN | CLK\_SYS | Pulse indicating HSUM processing is complete. ‘1’ = done. |
| **Global Clock/Resets** | | | |
| CLK\_PCIE | IN | - | 350MHz PCIe application clock from the PCIe Hard IP macro. |
| RST\_PCIE\_N | IN | - | Asynchronous Logic reset for the CLK\_PCIE domain.  ‘0’=Reset |
| CLK\_SYS | IN | - | 350 MHz system clock. |
| RST\_SYS\_N | IN | - | Asynchronous Logic reset for the CLK\_SYS domain.  ‘0’=Reset |
| CLK\_MC | IN | - | Up to 350 MHz Micro Configuration Clock |
| RST\_MC\_N | IN | - | Asynchronous Logic reset for the CLK\_MC domain.  ‘0’=Reset |

Table 4‑1 : MSIX Pinlist

# MCI Memory Mapped Interface

Note that the addresses are byte based

## CLD MSI-X Interrupt Configuration

CLD MSI-X Data and Enable

Register: CLD\_MSIX\_DATA (default: 0x00000000)

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Address | Mode | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| 0x00 | RW |  |  |  |  |  |  |  | CLD\_MSIX\_EN |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| CLD\_MSIX\_DATA[15:8] | | | | | | | |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CLD\_MSIX\_DATA[7:0] | | | | | | | |

CLD\_MSIX\_DATA[15:0]: Data for the CLD MSI-X Interrupt Vector Table Access:-  
Bits[15:12] : rsvd[3:0]  
Bit[11] : msix\_queue\_dir (1 = Host to Device, 0 = Device to Host)  
Bits[10:0]: msix\_queue\_num[10:0] (DMA Channel number or "queue number) Default: 0.

CLD\_MSIX\_EN: 1 = CLD MSI-X Interrupts Enabled Default: 0.

## CONV MSI-X Interrupt Configuration

CONV MSI-X Data and Enable

Register: CONV\_MSIX\_DATA (default: 0x00000000)

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Address | Mode | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| 0x10 | RW |  |  |  |  |  |  |  | CONV\_MSIX\_EN |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| CONV\_MSIX\_DATA[15:8] | | | | | | | |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CONV\_MSIX\_DATA[7:0] | | | | | | | |

CONV\_MSIX\_DATA[15:0]: Data for the CONV MSI-X Interrupt Vector Table Access:-  
Bits[15:12] : rsvd[3:0]  
Bit[11] : msix\_queue\_dir (1 = Host to Device, 0 = Device to Host)  
Bits[10:0]: msix\_queue\_num[10:0] (DMA Channel number or "queue number) Default: 0.

CONV\_MSIX\_EN: 1 = CONV MSI-X Interrupts Enabled Default: 0.

## HSUM MSI-X Interrupt Configuration

HSUM MSI-X Data and Enable

Register: HSUM\_MSIX\_DATA (default: 0x00000000)

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Address | Mode | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| 0x20 | RW |  |  |  |  |  |  |  | HSUM\_MSIX\_EN |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| HSUM\_MSIX\_DATA[15:8] | | | | | | | |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| HSUM\_MSIX\_DATA[7:0] | | | | | | | |

HSUM\_MSIX\_DATA[15:0]: Data for the HSUM MSI-X Interrupt Vector Table Access:-  
Bits[15:12] : rsvd[3:0]  
Bit[11] : msix\_queue\_dir (1 = Host to Device, 0 = Device to Host)  
Bits[10:0]: msix\_queue\_num[10:0] (DMA Channel number or "queue number) Default: 0.

HSUM\_MSIX\_EN: 1 =HSUM MSI-X Interrupts Enabled Default: 0.

# Design Parameterisation

The MSI module does not have any generic parameterisation.

# References

|  |  |  |
| --- | --- | --- |
| **Bookmark** | **Reference** | **Description** |
| [Ref: Agilex PCIe User Guide] | Multi Channel DMA Intel FPGA IP for PCI Express User Guide UG-20297 | 2021.10.29 | Intel PCIe interface with DMA User Guide for the Stratix 10 and Agilex FPGA families using the Intel Quartus Prime software version 21.3 |

# Abbreviations and Acronyms

|  |  |
| --- | --- |
| DMA | Direct Memory Access |
| DDR | Double Data Rate |
| FDAS | Fourier Domain Acceleration Search |
| FPGA | Field Programmable Gate Array |
| IEEE | Institute of Electrical Engineers |
| IP | Intellectual Property |
| MC | Micro Controller Interface |
| MSI-X | Extended Message Signalled Interrupt |
| PCIe | Peripheral Component Interconnect Express |
| SDRAM | Synchronous Dynamic RAM |