

**FDAS PCIF Module Design Specification for Intel Agilex F Implementation**

FDAS\_PCIF\_DS Revision 2 Draft B

Classification: UNRESTRICTED

Document type: DTE

Date: 2022-04-28

Status:

© 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 2 Draft A | 05/04/2022 | First Issue for review.  Comments from Prabu Thiagaraj and Atul Ghalame, and responses from Martin Droog:-  1) References to the PCIe documentation etc. may be added to the document..  **M. Droog Response: The Intel Agilex PCIe User Guide reference shall be added to this document.** |
| Issue 2 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 PCIF Module Architecture 8

3.2 Micro Configuration Interface: MCIF Sub-Module 9

3.2.1 Write to FDAS Memory Mapped Configuration 9

3.2.2 Read from FDAS Memory Mapped Configuration 12

3.2.3 RXM\_WAIT\_REQUEST During Reset 14

4 Interface Specification 15

5 MCI Memory Mapped Interface 17

6 Design Parameterisation 18

7 References 19

8 Abbreviations and Acronyms 20

# Introduction

This document captures the requirements for the PCIF module used in the FDAS FPGA implemented in the Intel Agilex FPGA family. Specifically the Agilex F series family.

The PCIF module is required to support the configuration of the FDAS FPGA via the PCIe interface, and to provide status information back to the PC/Computer via the PCIe interface.

The System Clock “CLK\_SYS” shall be 350MHz, however as a future proofing exercise the PCIF module shall also use a micro configuration clock “CLK\_MC” so that if necessary the micro configuration aspects of PCIF can be decoupled from CLK\_SYS and operate at a lower frequency.

The clock frequency for the PCIe internal interfaces “CLK\_PCIE” is dependent on the PCIe Generation and the number of PCIe lanes.

|  |  |  |
| --- | --- | --- |
| PCIe Lanes | PCIe Generation | CLK\_PCIE |
| x8 | Gen 4 | 250MHz |
| x16 | Gen 4 | 350MHz |

Table ‑ : PCIe Clock Frequency

For the FDAS implementation in Intel Agilex the PCIe interface shall be Gen 4 x 16 lanes and hence the CLK\_PCIE frequency shall be 350MHz.

The Intel Agilex PCIe Hard IP Macro to which the PCIF module connects is described in an Intel document [Ref: Agilex PCIe User Guide].

# Place in the System

The PCIF 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

DATA[511:0]

*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 : PCIF 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**

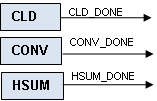
Notes:-

* 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: PCIF Top Level Connectivity in FDAS

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

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

* **Avalon MM Rx Master Module Interface**: This interface supports transfers initiated by the Computer/PC over the PCIe. This is used when the FDAS Micro Configuration register values are programmed, for example with the convolution filter values. The Computer/PC can initiate a write to the FDAS configuration registers or request a read of the FDAS configuration registers via this interface. The interface is compliant the Avalon MM protocol, with Address /Data buses and with the Hard PCIe macro performing the TLP encapsulation and de-encapsulation. Only the lower 32 bits of the data buses are used by the PCIF module.

The following interfaces of the FDAS Modules connect to the PCIF Module:-

* **FDAS MCI\_TOP Interface**: This interface supports Micro Configuration of the FDAS. It presents a standard micro-controller style interface to the PCIF module with Clock, Address, Data, Read/Write line and Chip Select. The MC read and write access time shall be fixed.

## PCIF Module Architecture

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

MCIF

RXM\_READ

RXM\_WRITE

RXM\_ADDRESS[21:0]

RXM\_BYTE\_ENABLE[3:0]

RXM\_WRITE\_DATA[31:0]

RXM\_READ\_DATA[31:0]

RXM\_READ\_DATA\_VALID

RXM\_WAIT\_REQUEST

MCRWN

MCADDR[21:0]

MCDATAIN[31:0]

MCCS

MCDATAOUT[31:0]

CLK\_MC

RST\_MC\_N

**PCIF**

CLK\_PCIE (from PCIe Hard IP macro

RST\_PCIE\_N (from PCIe Hard IP macro

RXM\_RESPONSE[1:0]

RXM\_WRITE\_RESPONSE\_VALID

Figure 3‑3 : PCIF Architecture Block Diagram

## Micro Configuration Interface: MCIF Sub-Module

The MCIF sub-module shall require;

* CLK\_MC clock operating at a frequency of up to 350MHz.
* CLK\_PCIE clocks operating at a frequency of 350MHz

No phase alignment between the two clocks shall be assumed. The interface to the PCIe Hard IP macro shall be on the CLK\_PCIE domain and the interface to the MCI\_TOP module shall be on the CLK\_MC domain.

The MCIF sub-module shall respond to activity on the Avalon MM Rx Master Module Interface from the PCIe Hard IP Macro.

The RXM\_WRITE and RXM\_READ signals shall indicate if the PC/Computer wishes to perform a write to FDAS memory mapped configuration or to perform a read from FDAS memory mapped configuration. Normally the MCIF sub-module shall detect the rising edge of the RXM\_WRITE and RXM\_READ signals on the CLK\_MC domain to determine when the PC/Computer wishes to perform an access. However for the CLK\_PCIE cycle after an access the MCIF sub-module shall detect the level of RXM\_WRITE or RXM\_READ, - essentially this allows back-to-back accesses where RXM\_WRITE or RXM\_READ do not go low between accesses. Any rising edge detection must take account of meta-stability due to the different clock domains of the MCIF sub-module and the PCIe Hard IP macro.

If RXM\_READ is also asserted when RXM\_WRITE is asserted the MCIF sub-module shall not respond to the requests.

The RXM\_ADDRESS[21:0] shall indicate the address in the FDAS memory mapped configuration space that the PC/Computer wishes to access.

### Write to FDAS Memory Mapped Configuration

In the case of a write to FDAS memory mapped configuration (indicated by RXM\_WRITE to RXM\_ADDRESS[21:0]) the RXM\_WRITE\_DATA[31:0] shall provide the data that is to be written along with RXM\_BYTE\_ENABLE[3:0] which indicates which of the 4 bytes in the 32-bit data word contain valid information. The table below defines the legal RXM\_BYTE\_ENABLE values:-

|  |  |  |
| --- | --- | --- |
| RXM\_BYTE\_ENABLE[3:0]  /binary | Valid RXM\_WRITE\_DATA[31:0]  bits | Comment |
| 0001 | [7:0] | Byte #1 valid |
| 0010 | [15:8] | Byte #2 valid |
| 0100 | [23:16] | Byte #3 valid |
| 1000 | [31:24] | Byte #4 valid |
| 0011 | [15:0] | Bytes #2 and #1 valid |
| 1100 | [31:16] | Bytes #4 and #3 valid |
| 1111 | [31:0] | All Bytes valid |

Table ‑ : RX\_BYTE\_ENABLE Mapping

**In the case of the PCIF module it shall be assumed that the RXM\_BYTE\_ENABLE shall always set to “1111” indicating that 32-bits of RXM\_WRITE\_DATA are valid (hence FDAS shall not support “read modified write” to allow only part of a configuration register to be modified).**

During the idle before the start of an access the MCIF sub-module shall assert the RXM\_WAIT\_REQUEST signal (active high) to the PCIe Hard IP macro. The PCIe Hard IP macro shall start the access but then hold the RXM\_WRITE, RXM\_ADDRESS, RXM\_WRITE\_DATA and RXM\_BYTE\_ENABLE steady until the RXM\_WAIT\_REQUEST signal is de-asserted.

The RXM\_WRITE signal shall be transferred from the CLK\_PCIE to the CLK\_MC clock domain and the rising edge shall be detected on the CLK\_MC domain. On this detected rising edge the MCIF sub-module shall translate the Avalon MM protocol signals from the Rx Master Module of the PCIe Hard IP macro to a standard microcontroller interface (MCADDR, MCDATAIN , MCRWN and MCCS).

The microcontroller interface shall be a 7 CLK\_MC cycle transfer to provide enough time for address decoding and latching of data in the FDAS modules.

The de-assertion of the MCCS signal (active high) shall be transferred from the CLK\_MC to the CLK\_PCIE domain and the rising edge shall be detected in the CLK\_PCIE domain. On this detected rising edge the MCIF sub-module shall de-assert the RXM\_WAIT\_REQUEST signal for one CLK\_PCIE cycle and on this same cycle assert the RXM\_WRITE\_RESPONSE\_VALID signal. The PCIe Hard IP macro shall detect this RXM\_WAIT\_REQUEST de-assertion and RXM\_WRITE\_RESPONSE\_VALID assertion and complete the write access. On the cycle when RXM\_WAIT\_REQUEST is de-asserted the MCIF sub-module shall effectively be level sensitive to RXM\_WRITE to allow back-to-back accesses.

An RXM\_RESPONSE[1:0] signal shall also be generated by the MCIF sub-module. This RXM\_RESPONSE[1:0] shall be fixed at the value “00” indicating the transaction was OK, as this is what is required by the current PCIe Hard IP Macro.

The timing diagram below shows a write to FDAS memory mapped configuration space:-

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

RXM\_WRITE\_DATA[31:0]

RXM\_ADDRESS[63:0]

RXM\_WRITE

CLK\_MC

rxm\_write\_rise

(stretch rising edge to

ensure safe sample on clk\_sys)

RXM\_BYTE\_ENABLE[3:0]

rxm\_write\_rise\_ret\_1

(meta)

rxm\_write\_rise\_ret\_2

MCADDR[63:0]

MCDATAIN[31:0]

MCRWN

MCCS

RXM\_WAIT\_REQUEST

CLK\_PCIE

mccs\_ret\_1

(meta)

mccs\_ret\_2

mccs\_ret\_3

**RXM\_WAIT\_REQUEST asserted in the idle before the start of the transfer**

**MCADDR, MCDATAIN, MCRWN & MCCS asserted when RXM\_WRITE assertion crosses to the CLK\_SYS clock domain**

**MCCS asserted for fixed time of 7 CLK\_SYS cycles**

**RXM\_WAIT\_REQUEST de-asserted for one CLK\_PCIE cycle when MCCS de-assertion crosses to the CLK\_PCIE** **clock domain**

rxm\_write\_rise\_ret\_3

**Level sensitive to RXM\_WRITE when RXM\_WAIT\_REQUEST = 0**

RXM\_WRITE\_

RESPONSE\_VALID

RXM\_RESPONSE[1:0]

**00**

Figure 3‑4 : Write Access to FDAS Memory Mapped Configuration

### Read from FDAS Memory Mapped Configuration

In the case of a read of FDAS memory mapped configuration (indicated by RXM\_READ from RXM\_ADDRESS[21:0]) The MCIF sub-module shall provide the read data via the RXM\_READ\_DATA[31:0] signal with the RXM\_READ\_DATA\_VALID signal asserted (active high) to indicate when the PCIe Hard IP macro can sample the data.

During the idle before the start of an access the MCIF sub-module shall assert the RXM\_WAIT\_REQUEST signal (active high) to the PCIe Hard IP macro. The PCIe Hard IP macro shall start the access but then hold the RXM\_READ and RXM\_ADDRESS steady until the RXM\_WAIT\_REQUEST signal is de-asserted.

The RXM\_READ signal shall be transferred from the CLK\_PCIE to the CLK\_MC clock domain and the rising edge shall be detected on the CLK\_MC domain. On this detected rising edge the MCIF sub-module shall translate the Avalon MM protocol signals from the Rx Master Module of the PCIe Hard IP macro to a standard microcontroller interface (MCADDR, MCDATAOUT , MCRWN and MCCS).

The microcontroller interface shall be a 7 CLK\_SYS cycle transfer to provide enough time for address decoding and returning of data in the FDAS modules on the MCDATAOUT[31:0] interface

The de-assertion of the MCCS signal (active high) shall be transferred from the CLK\_MC to the CLK\_PCIE domain and the rising edge shall be detected in the CLK\_PCIE domain. On this detected rising edge the MCIF sub-module shall de-assert the RXM\_WAIT\_REQUEST signal for one CLK\_PCIE cycle and ensure that the RXM\_READ\_DATA[31:0] and RXM\_READ\_DATA\_VALID are correctly set. The PCIe Hard IP macro shall detect the de-assertion of RXM\_WAIT\_REQUEST and on the following cycle sample RXM\_READ\_DATA[31:0] and RXM\_READ\_DATA\_VALID which must be correct at this point, thus completing the read access. On the cycle when RXM\_WAIT\_REQUEST is de-asserted the MCIF sub-module shall effectively be level sensitive to RXM\_READ to allow back-to-back accesses.

The timing diagram below shows a read from FDAS memory mapped configuration space:-

Figure 3‑5 : Read Access from FDAS Memory Mapped Configuration

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

RXM\_READ\_DATA[31:0]

RXM\_ADDRESS[63:0]

RXM\_READ

CLK\_PCIE

rxm\_read\_rise\_ret\_1

(meta)

RXM\_READ\_DATA\_VALID

rxm\_read\_rise\_ret\_2

rxm\_read\_rise\_ret\_3

MCADDR[63:0]

MCDATAOUT[31:0]

MCRWN

MCCS

RXM\_WAIT\_REQUEST

CLK\_MC

mccs\_ret\_1

mccs\_ret\_2

mccs\_ret\_3

**RXM\_WAIT\_REQUEST asserted in the idle before the start of the transfer**

**MCADDR, MCRWN & MCCS asserted when RXM\_READ assertion crosses to the CLK\_SYS clock domain**

**MCCS asserted for fixed time of 7 CLK\_SYS cycles**

**RXM\_WAIT\_REQUEST de-asserted for one CLK\_PCIE cycle when MCCS de-assertion crosses to the CLK\_PCIE** **clock domain**

mcdataout\_latched[31:0]

**Latched Data for this Access**

**Data**

rxm\_read\_rise

(stretch rising edge to

ensure safe sample on clk\_sys)

**Level sensitive to RXM\_READ when RXM\_WAIT\_REQUEST = 0**

### RXM\_WAIT\_REQUEST During Reset

In reset RXM\_WAIT\_REQUEST must be asserted (active high).

# Interface Specification

| Signal | Direction | Clock Domain | Description |
| --- | --- | --- | --- |
| **PCIF Interface to PCIE Hard IP Macro RX Master Module** |  |  |  |
| RXM\_READ | IN | CLK\_PCIE | Read Request (to read from FDAS) Indication Issued by the PC/Computer.  ‘1’ = Read Request. |
| RXM\_WRITE | IN | CLK\_PCIE | Write Request (to write to FDAS) Indication Issued by the PC/Computer.  ‘1’ = Write Request. |
| RXM\_ADDRESS[21:0] | IN | CLK\_PCIE | FDAS Address location that the PC/Computer wishes to read from or write to.  [21] = MSB (Most Significant Bit) |
| RXM\_BYTE\_ENABLE[3:0] | IN | CLK\_PCIE | Enables for the 32-bit word from the PC/Computer when a write to FDAS is requested. (see Sec 3.2.1) |
| RXM\_WRITE\_DATA[31:0] | IN | CLK\_PCIE | Data from the PC/Computer to be written to FDAS.  [31] = MSB (Most Significant Bit) |
| RXM\_READ\_DATA[31:0] | OUT | CLK\_PCIE | Data from the selected Address of FDAS to be sent to the PC/ Computer.  [31] = MSB (Most Significant Bit) |
| RXM\_READ\_DATA\_VALID | OUT | CLK\_PCIE | Indication that the Read data from the selected Address of FDAS is valid.  ‘1’ = Read Data is Valid |
| RXM\_WAIT\_REQUEST | OUT | CLK\_PCIE | Request by PCIF to indicate that it is not ready to respond to the Read/Write Request from the PC/Computer. |
| RXM\_WRITE\_RESPONE\_VALID | OUT | CLK\_PCIE | Indication that the Write access has completed and the RXM\_RESPONSE[1:0] is valid. |
| RXM\_RRSPONSE | OUT | CLK\_PCIE | The access Response code. Fixed at “00” for this implementation as this is the code required by the PCIe Hard IP Macro |
|  |  |  |  |
| **Interface to MCI\_TOP Module** |  |  |  |
| MCRWN | OUT | CLK\_MC | MC Read / Not Write  ‘1’ = Read, ‘0’ = Write |
| MCADDR[21:0] | OUT | CLK\_MC | MC Address  [21] = MSB (Most Significant Bit) |
| MCDATAIN[31:0] | OUT | CLK\_MC | MC Data in to FDAS  [31] = MSB (Most Significant Bit) |
| MCCS | OUT | CLK\_MC | MC Chip Select  ‘1’ = FDAS selected |
| MCDATAOUT[31:0] | IN | CLK\_MC | MC Data out of FDAS  [31] = MSB (Most Significant Bit) |
|  |  |  |  |
| **Global Clock/Resets** |  |  |  |
| CLK\_MC | IN | - | Micro configuration clock up to 350MHz |
| CLK\_PCIE | IN | - | 350MHz Reference clock for the PCIe Hard IP macro |
| RST\_MC\_N | IN | - | Asynchronous Logic reset for the CLK\_MC domain.  ‘0’=Reset |
| RST\_PCIE\_N | IN | - | Asynchronous Logic reset for the CLK\_PCIE domain.  ‘0’=Reset |

Table ‑ : PCIF Pinlist

# MCI Memory Mapped Interface

The PCIF module does not have a memory mapped interface.

# Design Parameterisation

The PCIF 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 |