

**FDAS CLD Module (Convolution Load) Design Specification for Intel Agilex Implementation**

FDAS\_CLD\_DS Revision 1 Draft D

Classification: UNRESTRICTED

Document type: DTE

Date: 2022-08-18

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 |  | 18/08/2022 |
| 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 D | 18/08/2022 | Unmodified Design from the Arria 10 Implementation |

**Table of Contents**

1 Introduction 4

2 Place in the System 5

3 Functional Specification 6

3.1 DDR Data Bus 7

3.2 DDR Read Address Generator: DDR\_RAG Sub-Module 10

3.3 FIFO Sub-Module 11

3.4 FIFO Write Address Generator: FIFO\_WAG Sub-Module 13

3.5 FIFO Read Address Generator: FIFO\_RAG Sub-Module 13

4 Design Requirement Tags 15

5 Interface Specification 16

6 MCI Memory Mapped Interface 18

7 Design Parameterisation 19

# Introduction

This document captures the requirements for the CLD Module used in the FDAS FPGA.

The CLD Module is required to read the 64-bit freq-bin samples from external DDR memory and load them into an internal RAM so that they can be used to supply CONV module to perform a convolution of the samples with a number of filters.

Each 64-bit sample is comprised of a 32-bit real value and 32-bit imaginary value using IEEE 754 floating point nomenclature. However the CLD Module does not need to be aware of the content or structure of the 64-bit sample.

The CONV Module uses the “Overlap-Save” method which requires an overlap of the samples fed into the convolution process. This overlap of samples shall be performed by the CLD module.

The CLD Module has Generic Parameterisation so that the following can be scaled:-

* The number of DDR SDRAM instances.
* The number of samples in the FFT.

The CLD Module supports a maximum of 222 = 4194304 samples

# Place in the System

The CLD 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 – CLD Location in FDAS

# Functional Specification

The CLD Module architecture is shown in the figure below:-

Figure 3‑1 : CLD Architecture Block Diagram

CTRL Module

DDRIF2

Module

DDR\_ADDR

[31:0]

DDR\_DATA

[512\*ddr\_g-1:0]

DDR\_RAG

Read Address Generator

DDR\_EN

pulsed

when

FIFO

can take more samples

CLD\_TRIGGER

CLD\_PAGE[31:0]

**CLD**

**FIFO** (Distributed RAM).

Each RAM is 512-bit wide, and the number of RAMs set by ddr\_g. The depth of the RAMs is set by fifo\_waddr\_width\_g so that 4096 samples can be stored.

Number of RAMs set by ddr\_g

Depth of RAM write interface set by fifo\_waddr\_width\_g RAM read interface is 64-bit, reading all equivalent locations of each RAM in turn before moving on to the next location

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

CONV\_DATA63:0]

CONV\_REQ

FFT\_SAMPLE

[9:0]

CLD\_DONE

DATA\_VALID

**ddr\_g , fft\_ddr\_addr\_num\_g**

**fft\_ddr\_addr\_num\_width\_g**

**fop\_ddr\_addr\_max\_width\_g,**

**fft\_g , fft\_count\_width\_g**

**sample\_count\_width\_g**

**fifo\_waddr\_width\_g,**

**fifo\_raddr\_width\_g**

**FIFO\_RAG**

Counts samples and use the sample count to generate the FIFO read Address.

Only requests Convolution when there are enough samples (based on the sample number from FIFO\_WAG)

OVERLAP\_SIZE [9:0]

CLD\_ENABLE

READY

**FIFO\_WAG**

Counts samples and use the sample count to generate the FIFO write Address.

Only requests new samples when there is space (based on the sample number from FIFO\_RAG)

Completed

read\_sample

\_count for completed samples

write\_sample

\_count to enable calculation of how many sample are in the FIFO to determine if convolution can start

fifo\_waddr

fifo\_raddr

DDR\_READ

CLK\_SYS

RST\_SYS\_N

WAIT\_REQUEST

DDR\_

DONE

SOF

EOF

VALID

FOP\_SAMPLE\_NUM[22:0]

OVERLAP\_REM [4:0]

OVERLAP\_INT [4:0]

## DDR Data Bus

The Convolution Load Data (CLD) module shall control the reading of the observed data (FFT freq-bins in the frequency domain) from the external DDR memory via the DDRIF2 module and make it available to the convolution algorithm:-

A generic “ddr\_g” shall define the number of external DDR memory interfaces (1, 2 or 3).

The organisation or the freq-bin samples shall be as shown below for the different numbers of external DDR memory interfaces, assuming 4,194,304 samples.

Address bits [31:6] are shown in the tables below to support a 4 Gibi-byte DDR SDRAM. Address bits [5:0] are always 0, since the internal 512-bit data bus contains the data for 64 bytes. The tables show the address with CLD\_PAGE[31:0] set to 0. For non-zero values of CLD\_PAGE add this to the DDR\_ADDR shown in the tables below to obtain the actual DDR4 SDRAM address.

* ddr\_g = 1

DDR SDRAM #1

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| DDR\_  ADDR  [31:6]  /Hex | Observed Data Freq-bins “b”  64-bit per freq-bin: 32-bit real, 32-bit Imaginary  Bit 511 Bit 0 | | | | | | | |
| 0x0000000 | b[8] | b[7] | b[6] | b[5] | b[4] | b[3] | b[2] | b[1] |
| 0x0000001 | b[16] | b[15] | b[14] | b[13] | b[12] | b[11] | b[10] | b[9] |
| : | : | : | : | : | : | : | : | : |
| 0x007FFFE | b[4,194,296] | b[4,194,295] | b[4,194,294] | b[4,194,293] | b[4,194,292] | b[4,194,291] | b[4,194,290] | b[4,194,289] |
| 0x007FFFF | b[4,194,304] | b[4,194,303] | b[4,194,302] | b[4,194,301] | b[4,194,300] | b[4,194,299] | b[4,194,298] | b[4,194,297] |

Table 3‑1 : Internal data bus to CLD with ddr\_g = 1

* ddr\_g = 2

DDR SDRAM #1

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| DDR\_  ADDR  [31:6]  /Hex | Observed Data Freq-bins “b”  64-bit per freq-bin: 32-bit real, 32-bit Imaginary  Bit 511 Bit 0 | | | | | | | |
| 0x0000000 | b[8] | b[7] | b[6] | b[5] | b[4] | b[3] | b[2] | b[1] |
| 0x0000001 | b[24] | b[23] | b[22] | b[21] | b[20] | b[19] | b[18] | b[17] |
| : | : | : | : | : | : | : | : | : |
| 0x003FFFE | b[4,194,280] | b[4,194,279] | b[4,194,278] | b[4,194,277] | b[4,194,276] | b[4,194,275] | b[4,194,274] | b[4,194,273] |
| 0x003FFFF | b[4,194,296] | b[4,194,295] | b[4,194,294] | b[4,194,293] | b[4,194,292] | b[4,194,291] | b[4,194,290] | b[4,194,289] |

DDR SDRAM #2

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| DDR\_  ADDR  [31:6]  /Hex | Observed Data Freq-bins “b”  64-bit per freq-bin: 32-bit real, 32-bit Imaginary  Bit 511 Bit 0 | | | | | | | |
| 0x0000000 | b[16] | b[15] | b[14] | b[13] | b[12] | b[11] | b[10] | b[9] |
| 0x0000001 | b[32] | b[31] | b[30] | b[29] | b[28] | b[27] | b[26] | b[25] |
| : | : | : | : | : | : | : | : | : |
| 0x003FFFE | b[4,194,288] | b[4,194,287] | b[4,194,286] | b[4,194,285] | b[4,194,284] | b[4,194,283] | b[4,194,282] | b[4,194,281] |
| 0x003FFFF | b[4,194,304] | b[4,194,303] | b[4,194,302] | b[4,194,301] | b[4,194,300] | b[4,194,299] | b[4,194,298] | b[4,194,297] |

Table 3‑2 : Internal data bus to CLD with ddr\_g = 2

* ddr\_g = 3

DDR SDRAM #1

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| DDR\_  ADDR  [31:6]  /Hex | Observed Data Freq-bins “b”  64-bit per freq-bin: 32-bit real, 32-bit Imaginary  Bit 511 Bit 0 | | | | | | | |
| 0x0000000 | b[8] | b[7] | b[6] | b[5] | b[4] | b[3] | b[2] | b[1] |
| 0x0000001 | b[32] | b[31] | b[30] | b[29] | b[28] | b[27] | b[26] | b[25] |
| : | : | : | : | : | : | : | : | : |
| 0x002AAA9 | b[4,194,272] | b[4,194,271] | b[4,194,270] | b[4,194,269] | b[4,194,268] | b[4,194,267] | b[4,194,266] | b[4,194,265] |
| 0x002AAAA | b[4,194,296] | b[4,194,295] | b[4,194,294] | b[4,194,293] | b[4,194,292] | b[4,194,291] | b[4,194,290] | b[4,194,289] |

DDR SDRAM #2

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| DDR\_  ADDR  [31:6]  /Hex | Observed Data Freq-bins “b”  64-bit per freq-bin: 32-bit real, 32-bit Imaginary  Bit 511 Bit 0 | | | | | | | |
| 0x0000000 | b[16] | b[15] | b[14] | b[13] | b[12] | b[11] | b[10] | b[9] |
| 0x0000001 | b[40] | b[39] | b[38] | b[37] | b[36] | b[35] | b[34] | b[33] |
| : | : | : | : | : | : | : | : | : |
| 0x002AAA9 | b[4,194,280] | b[4,194,279] | b[4,194,278] | b[4,194,277] | b[4,194,276] | b[4,194,275] | b[4,194,274] | b[4,194,273] |
| 0x002AAAA | b[4,194,304] | b[4,194,303] | b[4,194,302] | b[4,194,301] | b[4,194,300] | b[4,194,299] | b[4,194,298] | b[4,194,297] |

DDR SDRAM #3

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| DDR\_  ADDR  [31:6]  /Hex | Observed Data Freq-bins “b”  64-bit per freq-bin: 32-bit real, 32-bit Imaginary  Bit 511 Bit 0 | | | | | | | |
| 0x0000000 | b[24] | b[23] | b[22] | b[21] | b[20] | b[19] | b[18] | b[17] |
| 0x0000001 | b[48] | b[47] | b[46] | b[45] | b[44] | b[43] | b[42] | b[41] |
| : | : | : | : | : | : | : | : | : |
| 0x002AAA9 | b[4,194,288] | b[4,194,287] | b[4,194,286] | b[4,194,285] | b[4,194,284] | b[4,194,283] | b[4,194,272] | b[4,194,281] |
| 0x002AAAA |  |  |  |  |  |  |  |  |

Table 3‑3: Internal data bus to CLD with ddr\_g = 3

## DDR Read Address Generator: DDR\_RAG Sub-Module

The DDR\_RAG sub-module shall generate the address to read the freq-bin samples from the external DDR memory. The DDR\_RAG sub-module shall operate at the CLK\_SYS frequency of 200MHz.

The actual base Address value for the external DDR memory for the samples shall be indicated by the CLD\_PAGE[31:0] signal from the MC Interface via the CTRL module.

The DDR\_RAG sub-module shall only operate if the CLD\_ENABLE signal from the MC Interface via the CTRL module is set to 1.

A rising edge of the CLD\_TRIGGER signal from MC Interface via the CTRL module shall cause the internal address counter to be set to 0, and the DDR\_ADDR signal shall be set to the base address indicated by the CLD\_PAGE[31:0] signal.

Assertion of the pulsed DDR\_EN signal from the FIFO Write Address Generator (FIFO\_WAG) sub-module shall stimulate the DDR\_ADDR and DDR\_READ signals to read “fft\_ddr\_addr\_num\_g” locations of the external DDR memory (such that “fft\_g” samples can be read).

The DDR\_ADDR signal shall be a byte address. Since the expected DDR SDRAM size is 4 Gibi-Bytes (4,294,967,296 bytes), the DDR\_ADDR signal shall need to be 32 bits (i.e [31:0]). However since the internal data bus is 512-bits (64 bytes) the lower 6 bits (i.e [5:0]) of DDR\_ADDR shall always be “000000” and DDR\_ADDR generated by the CLD module shall therefore increment in steps of 64.

The DDR\_RAG sub-module shall monitor the DATA\_VALID signal to indicate to the FIFO\_WAG sub-module via the DDR\_DONE signal when the request number of reads has occurred. The DDR\_EN signal from the FIFO\_WAG sub-module shall need to be pulsed again to request more sample.

If the WAIT\_REQUEST signal is asserted by the DDRIF2 module the DDR\_RAG sub-module shall pause its operation until the WAIT\_REQUEST signal is de-asserted.

The number of bits for the internal counter to support the number of DDR read requests for an FFT shall be defined by the “fft\_ddr\_addr\_num\_width\_g” generic.

The number of bits for the internal counter to support the number of DDR read requests for the FOP shall be defined by the “fop\_ddr\_addr\_max\_width\_g” generic.

## FIFO Sub-Module

The FIFO sub-module shall be formed from block RAM. The FIFO sub-module shall be comprised of dual-port RAM with write port to store data from the external DDR memory and a read port to provide data to the CONV module. The read and write port address values shall be independent and the write and read data widths shall be different.

Each external DDR SDRAM memory Interface from the DDRIF2 module shall have an associated RAM in the CLD module. The number of RAMs in the FIFO sub-module is thus set by the “ddr\_g” generic. The internal width of each RAM shall 512-bits, and hence the overall write interface data width receiving data from the DDRIF2 module is ddr\_g x 512-bits.

The “fifo\_waddr\_width\_g” generic shall set the number of locations of each RAM in the FIFO sub-module, with each location storing 512 bits on the write side. The intention is such that the RAM depth is set such that at least 4 x “fft\_g” samples each of 64-bit can be stored in the FIFO sub-module.

The read side of the FIFO sub-module shall be 64 bits to provide the 64-bit freq-bin samples to the CONV module.

As an example, assuming ddr\_g is set to three DDR interfaces, with 512-bit data interfaces, the write and read address progressions are as shown in the figure below:-

Figure 3‑2 : FIFO Address Progression

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  | : |  |  |  |
|  |  |  | : |  |  |  |
|  |  |  | : |  |  |  |
|  |  |  | : |  |  |  |
|  |  |  | : |  |  |  |
|  |  |  | : |  |  |  |
|  |  |  | : |  |  |  |
|  |  |  | : |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  | : |  |  |  |
|  |  |  | : |  |  |  |
|  |  |  | : |  |  |  |
|  |  |  | : |  |  |  |
|  |  |  | : |  |  |  |
|  |  |  | : |  |  |  |
|  |  |  | : |  |  |  |
|  |  |  | : |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  | : |  |  |  |
|  |  |  | : |  |  |  |
|  |  |  | : |  |  |  |
|  |  |  | : |  |  |  |
|  |  |  | : |  |  |  |
|  |  |  | : |  |  |  |
|  |  |  | : |  |  |  |
|  |  |  | : |  |  |  |

Bit 511

Bit 511

512-bit data

From DDR SDRAM #3 via DDRIF2 to FIFO RAM #3

RAM Write Location 0

Bit 511

Bit 0

Bit 0

Bit 0

512-bit data

From DDR SDRAM #2 via DDRIF2 to FIFO RAM #2

512-bit data

From DDR SDRAM #1 via DDRIF2 to FIFO RAM #1

Write Address Progression

Read Address

Progression

64-bit data to CONV

**RAM #1**

**RAM #2**

**RAM #3**

**FIFO**

RAM Read

Location 0

## FIFO Write Address Generator: FIFO\_WAG Sub-Module

The FIFO\_WAG sub-module shall generate the FIFO write address for the data from the external DDR memory via the DDRIF2 module that is written into the FIFO sub-module. The FIFO\_WAG sub-module shall operate at the CLK\_SYS frequency of 200MHz.

The FIFO\_WAG sub-module shall only operate if the CLD\_ENABLE signal from the CTRL module is set to 1.

On the rising edge of the CLD\_TRIGGER signal the write address shall be set to 0, and a “write\_sample\_counter” shall also be set to 0.

The FIFO\_WAG sub-module shall sense the value of the “read\_sample\_count” for the completed samples from the FIFO\_RAG sub-module. If the difference between the completed samples from the FIFO\_RAG sub-module and the current value of the sample count in the FIFO\_WAG sub-module is less than 2 x “fft\_g” the DDR\_EN to the DDR\_RAG sub-module shall be pulsed to request “fft\_g” more samples. The completed sample value from the FIFO\_RAG sub-module is only sensed again when the “fft\_g” samples from the external DDR memory via the DDRIF2 module have been delivered.

When samples arrive from the external DDR memory via the DDRIF2 module (as indicated by the DATA\_VALID signal from the DDRIF2 module) the sample counter shall be incremented by the number of samples received and the FIFO write address shall increment by 1. The number of samples received shall depend on the number of DDR memory devices (set by “ddr\_g”).

Number to increment sample counter by = ddr\_g x 512 / 64.

The number of bits in the FIFO Write address shall be set by the “fifo\_waddr\_width\_g” generic and the number of bits in the sample counter shall be set by the “sample\_count\_width\_g” generic.

## FIFO Read Address Generator: FIFO\_RAG Sub-Module

The FIFO\_RAG sub-module shall generate the FIFO read address for the data that is to be passed to the CONV module. The FIFO\_RAG sub-module shall operate at the CLK\_SYS frequency of 200MHz.

The FIFO\_RAG sub-module shall only operate if the CLD\_ENABLE signal from the CTRL module is set to 1.

On the rising edge of the CLD\_TRIGGER signal the read address shall be set to 0, and a “read\_sample\_counter” and “completed\_sample\_count” shall also be set to 0.

The FIFO\_RAG sub-module shall sense the value of the “write\_sample\_count” for the stored samples from the DDR\_RAG sub-module. If the “write\_sample\_count” is “fft\_g” more than the “completed\_sample\_count”, the CONV\_REQ signal shall be asserted to the CONV module to indicate there is data available for convolution.

If the READY signal is then asserted by the CONV module the FIFO\_RAG sub-module shall increment the Read address to the FIFO and increment the “read\_sample\_counter” in sympathy each clock cycle until “fft\_g” samples have been passed to the CONV module.

Assertion of the READY signal shall clear down the CONV\_REQ signal.

A VALID flag shall be asserted towards the CONV module for each valid 64-bit sample that is passed to the CONV module.

An FFT\_SAMPLE[9:0] signal shall indicate the sample number that is currently presented to the CONV module.

A Start of FFT flag “SOF” shall indicate to the CONV module the first sample for the FFT. An End of FFT flag “EOF” shall indicate to the CONV module the last sample for the FFT.

Once the samples have been provided the “read\_sample\_counter” and FIFO read address to the FIFO are reduced by the MCI configured overlap value OVERLAP\_SIZE[9:0] and the “completed\_sample\_count” shall be set to this new “read\_sample\_counter” value. In the case when ddr\_g = 3 the rollover of internal counter for the read address to the FIFO is not natural, and hence OVERLAP\_INT[4:0] and OVERLAP\_REM[4:0] configuration is additionally needed. When ddr\_g = 3 each location of the RAM in the FIFO sub-module contains 24 samples, and OVERLAP\_INT[4:0] indicates the number of complete RAM locations for the overlap and OVERLAP\_REM[4:0] indicates the number of samples within a RAM location for the overlap.

When the FIFO\_RAG sub-module has supplied all samples to the CONV module as configured via the FOP\_SAMPLE\_NUM[22:0] signal from the MC Interface via the CTRL module it shall pulse the CLD\_DONE signal to the CTRL module.

The number of bits in the FIFO Read address shall be set by the “fifo\_raddr\_width\_g” generic and the number of bits in the “read\_sample\_counter” and “completed\_sample\_counter” shall also be set by the “sample\_count\_width\_g” generic.

The number of bits in a counter for the number of samples in an FFT shall be set by the “fft\_count\_width\_g” generic.

# Design Requirement Tags

The relevant design requirement tags from the FDAS Design Specification Document for the CLD module are listed below.

| **Design Requirement Tag** | **Description** | **Comment** |
| --- | --- | --- |
| FDAS.DATAIN:010/A | The data presented to FDAS shall be the result of an FFT with 222 (4,194,304) freq-bins. | This shall actually be set by a MC Configuration “FOP\_SAMPLE\_NUM” with expected values of 4,194,303 or 2,097,151 |
| FDAS.DATAIN:020/A | Each freq-bin shall contain a complex number with real and imaginary value | Each sample is therefore a 64-bit value that CLD shall process |
| FDAS.DATAIN:030/A | Each element of the complex number shall be represented in IEEE 754 single precision format. Hence the complex number consists of two IEEE 754 values. |
| FDAS.DATAIN:040/A | The FFT sequence shall be stored by FDAS in external DDR SDRAM memory, with all values static for a known period of time. | CLD shall read the samples from external DDR memory via the DDRIF2 module. CLD shall indicate when it has finished processing via the CLD\_DONE signal. This will actually determine when new samples can be loaded via the PCIe |
| FDAS.DATAIN:050/A | The input data to FDAS shall be statically available for 111ms. |
| FDAS.DATAIN:060/A | The external memory containing the input data shall have two pages, to allow one page to be written to whilst the other page is being accessed by the FDAS core processing. | This is supported by the CLD\_PAGE [31:0] signal from the CTRL module to determine which page base address. |
| FDAS.DATAIN:070/A | For a given page the allocation of freq-bins to memory locations shall be known and fixed, thus the memory address can be used to identify the freq-bin number and the real and imaginary value of each freq-bin. | The address progression of DDR\_RAG is thus known and fixed. |
| FDAS.DATAIN:080/A | The input data to FDAS (i.e. the observed data) shall be stored in external DDR SDRAM memory with the information shared equally across the available SDRAM devices. The **internal** FDAS data bus width shall be dependent on the number of external DDR SDRAM memory interfaces. A generic “ddr\_g” shall indicate the number of external DDR interfaces. | The sample assignment is opposite to that shown in the requirement since this makes it easier to store in the FIFO (i.e. sample 1 is in bits [63:0]. See Sec 3.1 of this document for the actual sample assignment of the data from the DDR SDRAM. |

Table 4‑1 : CLD Design Requirement Tags

# Interface Specification

| Signal | Direction | Clock Domain | Description |
| --- | --- | --- | --- |
| **Interface to CTRL Module** |  |  |  |
| CLD\_TRIGGER | IN | CLK\_SYS | A 0 > 1 transition triggers the CLD module to commence processing a DM |
| CLD\_ENABLE | IN | CLK\_SYS | Enable the CLD processing. The CLD module only processes when CLD\_ENABLE = 1. If CLD\_ENABLE is set to 0 the processing in CLD halts and only re-commences from its current position when CLD\_ENABLE is set to 1. |
| CLD\_PAGE [31:0] | IN | CLK\_SYS | Indication of the base address for the external DDR SDRAM to read the freq-bin samples from. |
| CLD\_DONE | OUT | CLK\_SYS | Indication that the CLD module has completed processing the DM. CLD\_DONE shall pulse for a single cycle when the processing has completed. |
| OVERLAP\_SIZE[9:0] | IN | CLK\_SYS | MC Configuration value from the CTRL module indicating the size of the overlap in the Overlap Save convolution method. Size is the number of samples. |
| OVERLAP\_INT[4:0] | IN | CLK\_SYS | MC Configuration value from the CTRL module.  Only used when ddrg\_3 (can be grounded if not used).  Indicates the number of complete RAM locations in the FIFO for the overlap. Each RAM location contains 24 samples. |
| OVERLAP\_REM[9:0] | IN | CLK\_SYS | MC Configuration value from the CTRL module.  Only used when ddrg\_3 (can be grounded if not used).  Indicates the number of samples within a RAM location in the FIFO for the overlap. Each RAM location contains 24 samples. |
| FOP\_SAMPLE\_NUM[22:0] | IN | CLK\_SYS | MC Configuration from the CTRL module indicating the number of samples for the FOP. The bus width is one larger than expected to support maths roll over CLD. |
|  |  |  |  |
| **Interface to DDRIF2 Module** |  |  |  |
| DDR\_ADDR[31:0] | OUT | CLK\_SYS | Address to the DDRIF2 module (Byte granularity)  [31] = msb |
| DDR\_READ | OUT | CLK\_SYS | Read Enable to the DDRIF2 module to request data.  ‘1’ = Read |
| DDR\_DATA[512\*ddr\_g -1:0] | IN | CLK\_SYS | Freq-bin sample data from the external DDR SDRAM via the DDRIF2 module.  . |
| DATA\_VALID | IN | CLK\_SYS | Indication that the data from the DDRIF2 module is valid.  ‘1’ = valid |
| WAIT\_REQUEST | IN | CLK\_SYS | Indication from DDRIF2 to pause the Address generation to the DDRIF2 module.  ‘1’ = Wait. |
| **Interface to CONV Module** |  |  |  |
| CONV\_REQ | OUT | CLK\_SYS | Request from the CLD module to the CONV module to indicate that the CLD module has freq-bin samples available for convolution.  ‘1’ = Request Convolution.  This signal is cleared down by the READY signal. |
| READY | IN | CLK\_SYS | Signal from the CONV module in response to the asserted CONV\_REQ signal to indicate that the CONV module is ready to accept freq-bin samples for convolution. |
| FFT\_SAMPLE[9:0] | OUT | CLK\_SYS | Sample number from 0 to fft\_g-1 that the CLD module is currently supplying to the CONV module. |
| VALID | OUT | CLK\_SYS | Indication that the 64-bit sample to CONV is valid.  ‘1’ = Valid |
| SOF | OUT | CLK\_SYS | Start of FFT  Flag coincident with the first sample for the FFT  ‘1’ = First sample for the FFT. |
| EOF | OUT | CLK\_SYS | End of FFT  Flag coincident with the last sample for the FFT  ‘1’ = Last sample for the FFT. |
| CONV\_DATA[63:0] | OUT | CLK\_SYS | 64-bit wide freq-bin sample data that the CLD module is supplying to the CONV module. |
|  |  |  |  |
| **Global Clock/Resets** |  |  |  |
| CLK\_SYS | IN | - | 200MHz Core System Clock |
| RST\_SYS\_N | IN | - | Asynchronous Logic reset  ‘0’=Reset |

Table 5‑1 : CLD Pinlist

# MCI Memory Mapped Interface

The CLD module does not have a memory mapped interface.

# Design Parameterisation

The CLD module has the following generic parameters:

CLD Top Level Generics

| **Generic** | **Description** |
| --- | --- |
| ddr\_g | The number of external DDR SDRAM Interfaces.  Valid values:  1 = 1 DDR SDRAM Interface  2 = 2 DDR SDRAM Interface  3 = 3 DDR SDRAM Interface |
| fft\_g | The number of points in the Fast Fourier Transform for the convolution process in the CONV module.  Valid values:  128 = 128-point Fast Fourier Transform  256 = 256-point Fast Fourier Transform  512 = 512-point Fast Fourier Transform  1024 = 1,024-point Fast Fourier Transform |
| fft\_count\_width\_g | The number of bits in a counter for the number of samples in an FFT.  FFT sample count [fft\_count\_width\_g-1:0]  This should be set to ROUNDUP(log2(fft\_g)) |
| fop\_ddr\_addr\_max\_width\_g | The number of bits for the internal counter to support the number of DDR read requests for the FOP shall be defined by the “fop\_ddr\_addr\_max\_width\_g” generic.  Read request counter [fop\_ddr\_addr\_max\_width\_g-1:0]  This should be set to ROUNDUP(log2(ddr\_addr\_max)) as described in the table below:-   |  |  |  | | --- | --- | --- | | **ddr\_g** | **FOP Samples** | **ddr\_addr\_max** | | 1 | 2097152 | 262144 | | 1 | 4194304 | 524288 | |  |  |  | | 2 | 2097152 | 131072 | | 2 | 4194304 | 262144 | |  |  |  | | 3 | 2097152 | 87381.33333 | | 3 | 4194304 | 174762.6667 | |
| fft\_ddr\_addr\_num\_g | The number of external DDR SDRAM locations to read to obtain fft\_g samples.  Valid values are based on the ddr\_g and fft\_g generics:  fft\_ddr\_addr\_num\_g = ROUNDUP (fft\_g / [ddr\_g x (512/64)])   |  |  |  |  | | --- | --- | --- | --- | | **ddr\_g** | **fft\_g** | **Number of reads** | **fft\_ddr\_addr\_num\_g** | | 1 | 1024 | 128 | 128 | | 512 | 64 | 64 | | 256 | 32 | 32 | | 128 | 16 | 16 | |  |  |  |  | | 2 | 1024 | 64 | 64 | | 512 | 32 | 32 | | 256 | 16 | 16 | | 128 | 8 | 8 | |  |  |  |  | | 3 | 1024 | 43 | 43 | | 512 | 22 | 22 | | 256 | 11 | 11 | | 128 | 6 | 6 | |
| fft\_ddr\_addr\_num \_width\_g | The number of bits for the internal counter to support the number of DDR read requests for an FFT shall be defined by the “fft\_ddr\_addr\_num\_width\_g” generic.  FFT read request counter [fft\_ddr\_addr\_num \_width\_g-1:0]  This should be set to ROUNDUP(log2(fft\_ddr\_addr\_num \_g)) |
| fifo\_waddr\_width\_g | The number of bits in the FIFO Write address to ensure the desired number memory locations can be supported.  FIFO write address  fifo\_waddr [fifo\_waddr\_width\_g-1:0]  To store 4 FFTs worth of data this should be set to ROUNDDOWNP(log2(4\*fft\_g/(ddr\_g\*512/64))) |
| fifo\_raddr\_width\_g | The number of bits in the FIFO Read address to ensure the desired number memory locations can be supported.  FIFO Read address  fifo\_raddr [fifo\_raddr\_width\_g-1:0]  To store 4 FFTs worth of data this should be set to ROUNDUP(log2(4\*fft\_g)) |
| sample\_count\_width\_g | The number of bits in the sample counter to support the required number of samples in the FOP.  This should be set to ROUNDUP(log2(FOP Samples)) + 1 to ensure there are no rollover issues when internal additions occur |

Table 7‑1: CLD Generic Parameters

The generic values are type natural (i.e. +ve integers).

Settings for one DDR Interface, with 2,097,152 samples to analyse using1024 point FFT:-

| **Generic** | **Value** |
| --- | --- |
| ddr\_g | 1 |
| fft\_g | 1024 |
| fft\_count\_width\_g | 10 |
| fop\_ddr\_addr\_max\_width\_g | 18 |
| fft\_ddr\_addr\_num\_g | 128 |
| fft\_ddr\_addr\_num \_width\_g | 7 |
| fifo\_waddr\_width\_g | 9 |
| fifo\_raddr\_width\_g | 12 |
| sample\_count\_width\_g | 22 |

Table 7‑2 : Example of settings for Generic Parameters