SigFFT Specification

Signal FFT Core

Revision 1.1

14 July 2025

English ver.

1. Overview

1.1. Introduction

- The Signal FFT Core (hereinafter referred to as ***sigFFT***) is an embedded core that performs high-speed Discrete Fourier Transform (FFT) from a source image to a destination image.

- It processes the destination image by repeatedly performing one-dimensional transformations while scanning the required portions of the source image on a line-by-line basis. Ultimately, both horizontal and vertical transformations are required.

- The line unit must be a power of 4. Time-division processing (i.e., splitting by line units) is not supported.

- Supported pixel formats include 8-bit (fixed-point), 16-bit (IEEE754 Binary16), and 32-bit (IEEE754). Internal processing is performed in single precision.

- A one-cycle-throughput butterfly processor (Radix-4) is used, and the entire pipeline—including memory accesses—is fully pipelined. For N pixels, the maximum performance is achieved in $N/4 \times \log_4 N$ cycles.

Note:

- The memory interface (I/F) must be customized according to the system.

- It is also possible to use the core as a transformation component without utilizing the memory interface.

1.2. Main Parameters

- **Memory Bus**
  Data Read/Write: 64-bit × 1
  Command List Read: 64-bit × 1

- **Throughput**

Radix-4 butterfly operation per cycle

- **Supported Formats**

  8-bit component (monochrome), 16-bit float, 32-bit float

- **Supported Size**

  2N-point FFT/IFFT

- **Clock**

  Undefined (depends on implementation process)

1.3. Implementation Parameters

Parameter Name	Description	Default Value
MRR	• Radix (2^n) of the maximum processing size *N* • Any value other than the specified one requires design modification, as it affects the capacity and structure of the SRAM	**10**
BLR	• Radix of burst length of externa(parameter) bus • Configurable burst unit for 64-bit memory access	**1** (4 or less)
BSR	• Radix of burst length of data bus • Configurable burst unit for 64-bit memory access	**1** (4 or less)

2. Signal Lines

2.1. Control Bus Interface

Signal Name	IO	Pol	Source	Description
cntlReq	I	+	clk	• Request signal • Evaluate cntlGnt
cntlGnt	O	+	clk	• Grant signal
cntlRxw	I	+	clk	• R/W signal • Evaluate cntlReq & cntlGnt 0: Write 1: Read
cntlAddr[31:0]	I	+	clk	• Address signal • Evaluate cntlReq & cntlGnt
cntlWrAck	O	+	clk	• Writ acknowledge signal
cntlWrData[31:0]	I	+	clk	• Write data signal • Evaluate cntlWrAck
cntlRdAck	O	+	clk	• Read acknowledge signal
cntlRdData[31:0]	O	+	clk	• Read data signal • Sync cntlRdAck
cntlIrq	O	+	clk	• Interrupt signal • Level hold type

2.2. PSS Interface

Signal Name	IO	Pol	Source	Description
iVld	I	+	clk	• Pipeline start valid signal
iStall	O	+	clk	• Pipeline start stall signal
iAddr[31:4]	I	+	clk	• Address to fetch context data • Evaluate iVld & !iStall
iDelta[15:0]	I	+	clk	• Transfer volume • Evaluate iVld & !iStall
iIndex[64:0]	I	+	clk	• Five coodinates to specify the processing • Evaluate iVld & !iStall
oVld	O	+	clk	• Pipeline end valid signal
oStall	I	+	clk	• Pipeline end stall signal

2.3. Memory Interface (Data R/W Use)

Signal Name	IO	Pol	Source	Description
miReq	O	+	clk	• Request signal
miGnt	I	+	clk	• Grant signal
miRxw	O	+	clk	• R/W signal
miAddr[31:0]	O	+	clk	• Address signal
miBurst[BSR-1:0]	O	+	clk	• Burst signal
miRdStrb	O	+	clk	• Read strobe
miRdAck	I	+	clk	• Read acknowledge signal
miRdData[63:0]	I	+	clk	• Read data signal
miWrStrb	O	+	clk	• Write strobe signal
miWrAck	I	+	clk	• Write acknowledge signal
miWrData[63:0]	O	+	clk	• Write data signal
miWrMask[7:0]	O	+	clk	• Write mask signal

2.4. Memory Interface (Parameter Read Use)

Signal Name	IO	Pol	Source	Description
meReq	O	+	clk	• Request signal
meGnt	I	+	clk	• Grant signal
meAddr[31:0]	O	+	clk	• Address signal
meStrb	O	+	clk	• Read strobe signal
meAck	I	+	clk	• Read acknowledge signal
meData[63:0]	I	+	clk	• Read data signal

2.5. Utillity

Signal Name	IO	Pol	Source	Description
rstReq	O	+	clk	• Internal reset signal to reset the external system
rstAck	I	+	clk	• Acknowledge of rstReq
fReq	I	+	clk	• 1 clock early request against the iVld signal • Use to generate gate signal (for **pss**)
pReq	O	+	clk	• 1 clock early request against the all memory access signal • Use to generate gate signal (for memory)
gate[6:0]	O	+	clk	• Gated clock control signal signifying condition of each internal block
gclk[6:0]	I	+	clk	• Gated clock
clk	I	┼	clk	• Clock
reset_n	I	-	-	• Asynchronous reset signal

3. Structure and Operation Description

3.1. Structural Overview

- The Pipeline Slice Scheduler (hereinafter referred to as ***pss***) retrieves the necessary context from memory, generates coordinate and related information, and initiates ***sigFFT***. For more details on ***pss***, please refer to its dedicated specification document. Since the connection interface is simple, using ***pss*** is not mandatory; in such cases, consider replacing the ***pss*** component with your own custom core.

- The ***sigFFT*** is structured as a pipeline as shown in Figure 1. The Initiator retrieves parameters from a command list stored in memory and manages overall control. Data is processed by repeatedly interacting with memory and the butterfly processor, centering around a banked SRAM architecture.

- The maximum processing size is proportional to the capacity of the onboard SRAM. The SRAM is composed of 8 banks in total, supporting 4 simultaneous accesses by the butterfly processor and 2 simultaneous memory accesses.

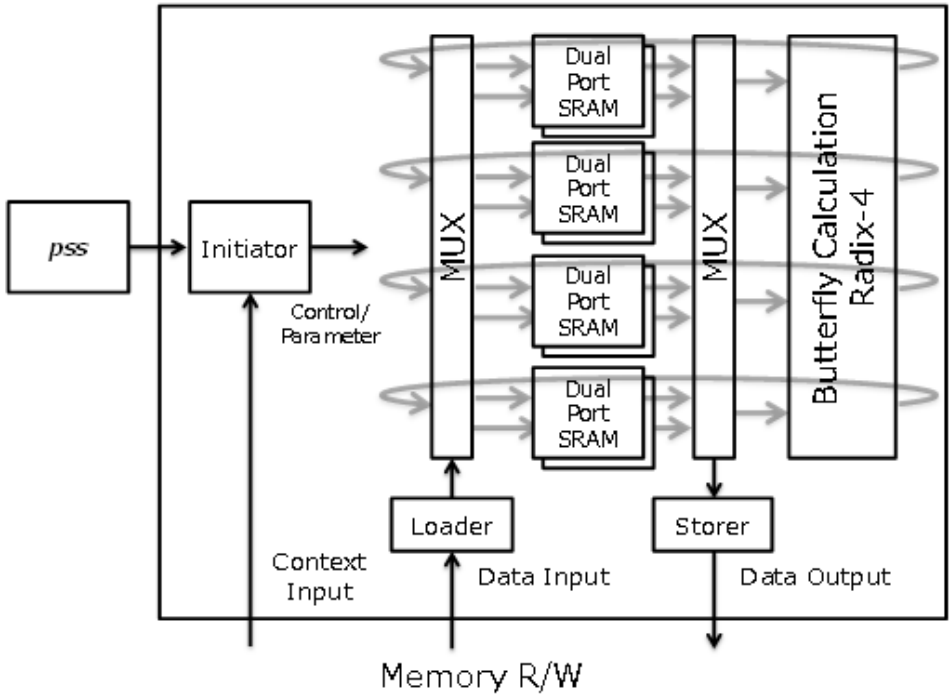


Figure 1 *frFFT* Block Diagram

3.2. Operational Overview (Processing Sequence)

- The ***pss*** scans the destination coordinates along an arbitrary axis direction and sends the results to the Initiator. Settings for ***pss*** (such as image information and processing units) must be preloaded into memory. The ***pss*** can manage up to 256 different configurations (depending on the implementation) in a time-division manner and schedules execution to drive ***sigFFT***.

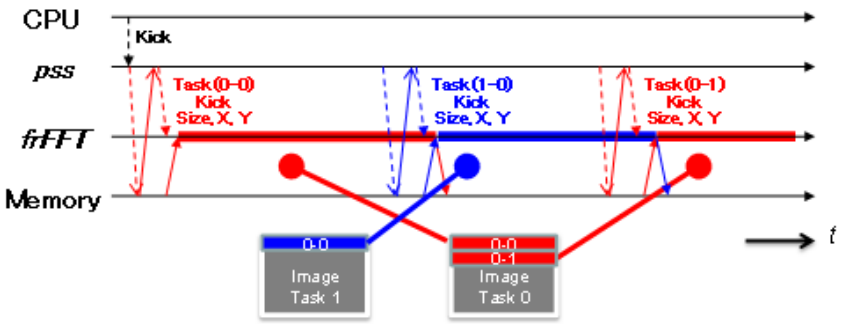


Figure 2 Resource sharing using *pss*

- The Initiator reads the context provided by the ***pss***, which includes image information, and performs the initial setup of the pipeline. The parameters extracted from the context are managed with double buffering, so performance degradation does not occur unless the processing unit designated by ***pss*** is extremely small.

- Once the Initiator is activated, the Loader begins transferring data from memory to SRAM. The Loader converts the input from its specified format to the internal processing format (single-precision floating-point), then performs FFT-specific address transformation before storing the data into SRAM.

- After the data transfer by the Loader is complete, the butterfly processor repeatedly processes the data in SRAM. The butterfly processor has a latency of 10 cycles and a throughput of 1 cycle, processing four complex data elements at a time (Radix-4). The number of cycles required to process N points is $N/4 \times \log_4 N$.

- Upon completion of the butterfly operation, the Storer begins transferring data from SRAM back to memory. The Storer converts the data from the internal SRAM format to the specified output format before storing it in memory.

- The Loader and Storer operate mutually exclusively, but they can function independently of the butterfly processor. As a result, it is possible to overlap butterfly processing with Loader/Storer operations, enabling efficient pipelined processing.

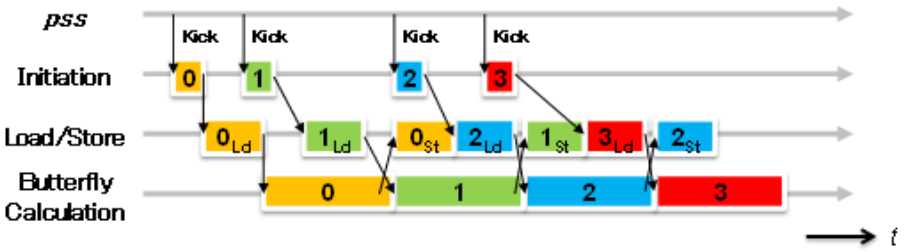


Figure 3 Pipeline scheduling

3.3. Input/Output Format

- The core supports the following data formats.
  Fixed-point format represents values divided by 256.
  Floating-point format handles a subset of the IEEE754 standard.
  A format that converts RGB format into grayscale values is also supported.

Bit/Word	Fixed Point	Float (IEEE754)	
		Binary 16	Binary 32
8	Yes	-	-
16	-	Yes	-
32	Yes	-	Yes

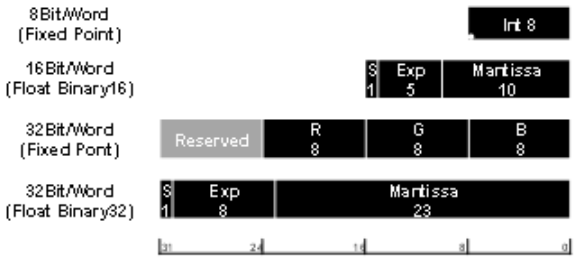


Figure 4 Data Format

- In memory mapping, data is generally stored in order from the Most Significant Byte (MSB) downward. To change the order, adjust the Byte Swap and Word Swap parameters in the

memory operation settings.

- Real (Re) and Imaginary (Im) components are stored in separate planes (see Figure 5). Each requires its own independent base address.

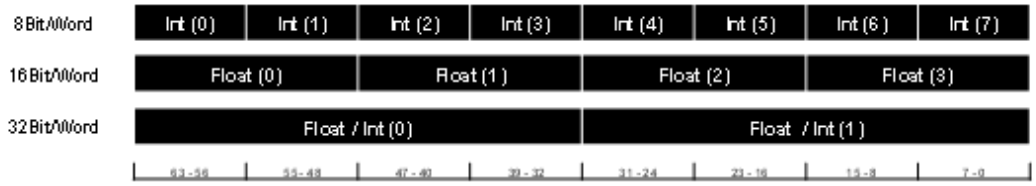


Figure 5 Memory mapping for planer

3.4. Internal Computation

- All computations are performed by the butterfly processor. All internal data representations, including coefficients, use single-precision floating-point format.

- Any precision beyond the internal representation in computation results is truncated. Extremely small values that cannot be represented are rounded to zero, while extremely large values cannot be guaranteed.

- When converting to a different format, values outside the representable range of the specified format are rounded to the maximum or minimum representable value.

- NaN and Infinity values propagate through computation; therefore, input data must avoid including such values.

3.5. Connection with pss

- Based on the address output by ***pss*** (iAddr), the command list is retrieved from memory. For details on the command list, refer to the corresponding command list section. If ***pss*** is not present, access the ***pss*** interface directly.

- Using the coordinate (iIndex) output by ***pss*** and the parameters in the command list, the starting addresses for the input and output image data are calculated. Zero-order

coordinates are not handled. The initial address is determined using the following calculation. Here, *Y* and *Z* are the first-order and second-order coordinates, respectively, and *StrideY* and *StrideZ* are the step values for coordinate changes (units vary depending on the format). The addressing mode defines whether *Y* and *Z* are referenced and how the strides are set.

$$\text{StartAddress} = \text{BaseAddress} + (\text{Stride}_Y \times \text{Y}) + (\text{Stride}_Z \times \text{Z})$$

- Normally, data is read or written by incrementing the address in memory word units. In contrast, for transposition, data is read or written by adding the stride value. As with the initial address calculation, these behaviors are defined by the addressing mode. In general, performance is better when transposition is not used.

- Swapping of first-order and second-order coordinates is handled by ***pss***.

- The transfer length output by ***pss*** (iDelta) must generally match the FFT length. If it does not match, the following processing will occur (details to follow).

Condition	Memory Access	Data Processing
iDelta > N	Read/Write only *N* pixels	Process *N* pixels
iDelta < N	Read/Write only *iDelta* pixels	Pad remaining values with zero up to *N* pixels

- Real (Re) and Imaginary (Im) components that are not required can be excluded from memory access and treated as zero during processing.

3.6. Performance

- From start to finish, the total number of cycles required includes both memory access cycles (Load/Store) and the butterfly computation time, which is $N/4 \times \log_4 N$ cycles. When executions are continuous and N is sufficiently large (e.g., N = 256), the butterfly computation becomes the dominant factor, effectively hiding the memory access time. In such cases, $N/4 \times \log_4 N$ cycles represent the maximum throughput.

- For reference, consider the case where memory performance is at the maximum of 64

bits per cycle, and the data being handled is 16 bits per word. Note that in practice, additional pipeline overhead (approximately 10 cycles) must be added to this.

	N=16	N=64	N=256	N=1024
Single-shot	24	112	512	2304
Continuous	16	64	256	1280

※The shaded area indicates where memory cycles dominate during continuous execution.

3.7. Additional Features

- A transformation map placed in memory can be used to convert input indices (X, Y) into coordinates (X, Y), allowing specification of the FFT/IFFT starting position using 2D addressing. This is controlled via MemCnt.SrcRemap, MapInfo, and MapBase. In image processing, it enables transformation from arbitrary XY positions. Note that this operation is executed each time the X index changes by one.

- A window function can be applied before executing the FFT/IFFT by setting SrcInfo.Exp to 3. Either the real or imaginary signal series is used as a one-to-one coefficient sequence for the window. This feature takes advantage of the case where either the real or imaginary part is zero, using that series as the coefficient. When processing an entire 2D region at once, ensure that the coefficient sequence does not use SrcSize or SrcOffset by setting SrcInfo.Div to 15.

- It is possible to apply FFT/IFFT over a combined 2D region. For example, for a 1024-point FFT with SrcInfo.Div set to 2 (for 1/4 division), scanning is performed in the X direction with 256 points and in the Y direction with 4 lines, executing the 1024-point FFT in aggregate. Since different signal sequences are involved, edge effects must be addressed using window functions or similar techniques.

- The maximum value and its corresponding position from the above processing can be written to memory. This is controlled using MemCnt.DstStack, StackInfo, and StackBase.

4. Register Description

4.1. Overview

- Each register is accessed via the control bus.
- Since some settings may affect pipeline operation or performance, care must be taken regarding the timing of configuration.
- In the detailed register descriptions, the following symbols are used to indicate access types:
  - **R**: Read Only (writes have no effect)
  - **R/W**: Read / Write
  - **R/WC**: Read / Write Clear
- Do not access registers marked as *Reserved*. When writing to reserved fields, always set the value to '0'.
- The character 'x' in address or data representations indicates "don't care".

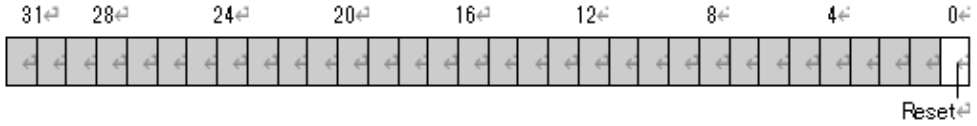
4.2. Definition

Address	Register Name	Description
0000_0000	Reset	Reset control
0000_0004	System	System control

4.3. Details

▪ 4.3.1.1. Reset Register

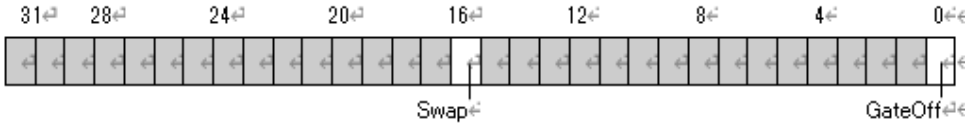
[Address: 0x0000_0000]



Name	Type	Default	Description
Reset	R/W	0	Synchronous reset. Requires setting to '1' followed by clearing to '0'. Unlike the reset_n signal, register contents are retained. After setting to '1', the rstReq signal is immediately asserted. This signal notifies external modules that ***sigFFT*** has entered a reset state and requests their response. Once the response is completed, the rstAck signal must be asserted (or kept asserted at all times if no response is necessary). After these procedures are completed, the Reset automatically returns to '0'.

▪ 4.3.1.2. System Register

[Address: 0x0000_0004]



Name	Type	Default	Description
Swap	R/W	0	**Word Swap setting.** When set to '1', the upper 32 bits and lower 32 bits of the 64-bit bus are swapped. Swapping within a 32-bit word is specified in the command list.
GateOff	R/W	0	**Gated Clock Off Mode.** When set to '1', all bits of the gate signal are fixed to '1'.

5. Command List Description

5.1. Overview

- The command list's starting address is specified by the address value output from ***pss***. After ***sigFFT*** is triggered, it retrieves the command list and stores it in internal registers.

- The command list is managed completely independently within each stage of the pipeline. This allows different stages to hold and operate on different command lists even while the pipeline is running. Therefore, no synchronization commands are necessary.

- Reserved registers and fields must be set to '0'.

- The addresses shown are relative to the address value output by ***pss***.

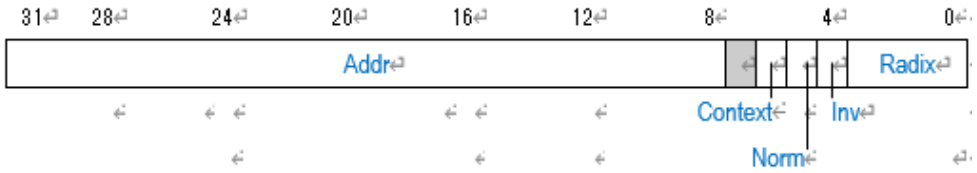
5.2. Definition

Address	Command Name	Description
00	MasterCntl	Master control
04	MemCntl	Memory control
08	SrcSize	Source size
0c	SrcOffset	Source offset
10	MapInfo	Map inforrmation
14	MapBase	Map address
18	StackInfo	Stack information
1c	StackBase	Stack address
20	SecInfo(Re)	Source Real Number Information
24	SrcBase(Re)	Source Real Number address
28	SrcInfo(Im)	Source Imaginary Number Information
2c	SrcBase(Im)	Source Imaginary Number address
30	DstInfo(Im)	Destination real number information
34	DstBase(Re)	Destination real number address
38	DstInfo(Im)	Destination imaginary number information
3c	DstBase(Im)	Destination imaginary number address

5.3. Deftails

▪ 5.3.1.1. MasterCntl Command

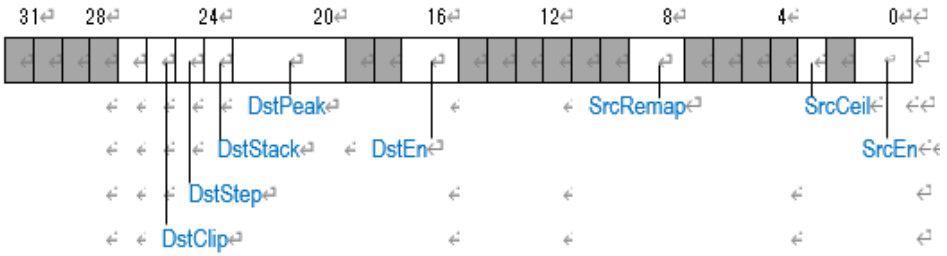
[Address: 0x00]



Name	Description
Addr[31:8]	Context address for reading the processing volume.
Context	Set to '1' when reading the processing volume.
Norm	Performs normalization. For a forward FFT, the output is scaled by 1/2^{Radix[3:1]}. For an inverse IFFT, the output is scaled by 1/2^{Radix[3:1] + Radix[0]}.
Inv	Set to '0' for FFT, and '1' for IFFT.
Radix[3:0]	Set **$\log_2 N$** as the processing size *N*. The value must be between 4 and 10 (inclusive).

▪ 5.3.1.2. MemCntl Command

[Address: 0x04]



Name	Description
DstClip	When the destination data output is shifted so that the peak value is centered, values that exceed the output width are clipped to zero.
DstStep	Each time the X index is updated, the destination data output address is incremented by the output width.
DstStack	The peak value and its index of the destination data output are written to the memory indicated by the Stack address.
DstPeak[3:0]	Set the conditions for calculating the peak value of the destination data output.

DstPeak[3:2]	Description
0	Search by absolute value
1	Reserved
2	Search for positive values
3	Search for negative values

DstPeak[1:0]	Description
0	Reserved
1	Reserved
2	Search for real number values
3	Search for imaginary number values

DstEn[1:0] Set the permission for outputting the destination data.

DstEn[1]	Description
0	Do not allow imaginary output
1	allow imaginary output

DstEn[0]	Description
0	Do not allow real number output
1	allow real number output

SrcRemap[1:0] Set the transformation method for the reference coordinate values of the source data.

The reference coordinates are obtained from the Map address indicated by the indices X and Y.

SrcRemap	Description
0	No transformation
1	Obtain new reference coordinates X and Y from memory based on indices X and Y, and perform the transformation.
2	Obtain offset coordinates X and Y from memory based on indices X and Y, then add them to the indices to convert to reference coordinates X and Y.
3	Obtain offset coordinates X and Y from memory based on indices X and Y, then subtract them from the indices to convert to reference coordinates X and Y.

SrcCeil When the source data is 8-bit fixed-point, 0xFF is considered as 1.0.

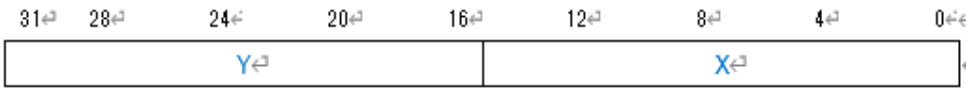
SrcEn[1:0] Set the permission for inputting the source data.

SrcEn[1]	Description
0	Do not allow imaginary input (treat as zero)
1	Allow imaginary input

SrcEn[0]	Description
0	Do not allow real input (treat as zero)
1	Allow real input

▪ 5.3.1.3. SrcSize Command

[Address: 0x08]

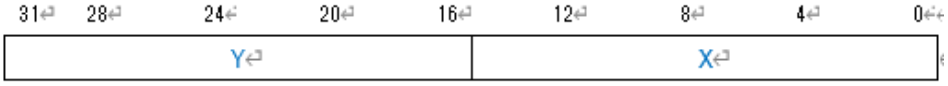


Name	Description
Y[15:0]	Set the input width size of the source in the X direction. Data at coordinates exceeding this width will be clipped to zero. If set to 0, the setting is disabled.

X[15:0]	Set the input width size of the source in the Y direction. Data at coordinates exceeding this height will be clipped to zero. If set to 0, the setting is disabled.

▪ 5.3.1.4. SrcOffset Command

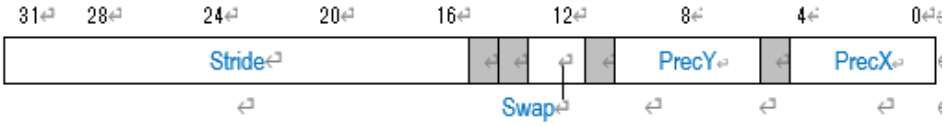
[Address: 0x0c]



Name	Description
Y[15:0]	Set the offset of the source in the X direction. Value is in two's complement format.
X[15:0]	Set the offset of the source in the Y direction. Value is in two's complement format.

▪ 5.3.1.5. MapInfo Command

[Address: 0x10]



Name	Description
Stride[15:0]	Set the address update stride minus 1 for Y-coordinate updates in the map data. Applicable when SrcRemap ≠ 0.
Swap[1:0]	Set the Half Word Swap for map data. Performs mapping from input data In[31:0] to output data Out[31:0]. Each half-word segment in Out[31:0] is controlled by the corresponding bit in Swap[1:0]. Each bit in Swap selects one half-word from In[31:0]. In principle, mapping is done one-to-one. If not, unmappable lanes or overlapping mapped lanes may occur.

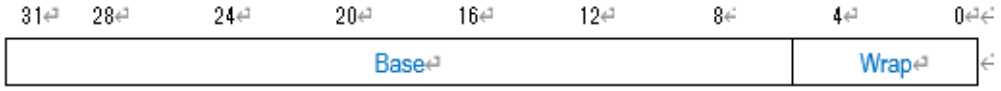
Value	Swap[1]	Swap[0]
	Out[31:16]	Out[15:0]
0	Pipe[31:16]	Pipe[15:0]

1	Pipe[15:0]	Pipe[31:16]

PrecX/PrecY[4:0] Set the precision for the X and Y values in the map data.
Coordinate values are represented in two's complement:
if the value is positive, it is right-shifted by that amount;
if negative, it is left-shifted by the magnitude of the value.

▪ 5.3.1.6. MapBase Command

[Address: 0x14]

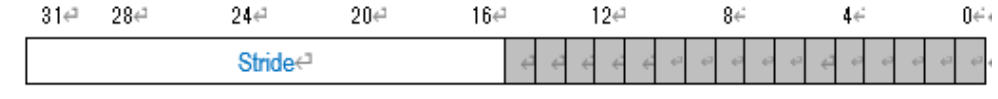


Name	Description
Base[31:6]	Set the base address of the map data. It must be aligned to a 64-byte boundary. Applicable when MemCntl.SrcRemap ≠ 0.
Wrap[5:0]	By setting the MSB to '1', the remaining 5 bits are used to enable the mask position. The mask applies from the MSB of the address down to the specified bit position (inclusive). For example, if the value is 37, the upper 6 bits of the generated address remain the same as those of the base address, and only the lower 26 bits are modified.

Wrap	Description
0	No mask
1-31	Reserved
32-62	Bits [62-n:0] of the generated address are valid.
63	The address value is always the same as the base.

▪ 5.3.1.7. StackInfo Command

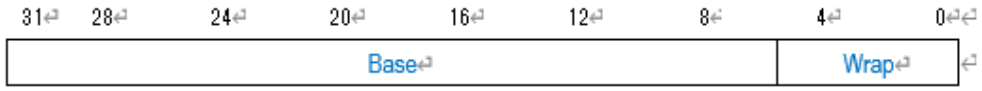
[Address: 0x18]



Name	Description
Stride[15:0]	Refer to MapInfo.Stride. Applicable when MemCntl.DstStack = 1.

▪ 5.3.1.8. StackBase Command

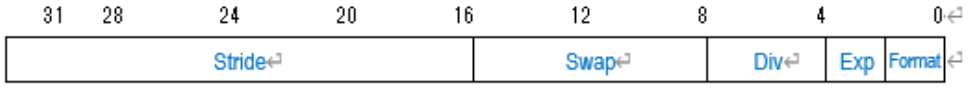
[Address: 0x1c]



Name	Description
Base[31:6]	Refer to MapBase.Base. Applicable when MemCntl.DstStack = 1.
Wrap[5:0]	Refer to MapBase.Wrap

▪ 5.3.1.9. SrcInfo(Re/Im) Command

[Address: 0x20/0x28]



Name	Description
Stride[15:0]	Refer to MapInfo.Stride
Swap[7:0]	Configure Byte Swap for the source data (real/imaginary). Performs byte-wise mapping from input data In[31:0] to internal data Pipe[31:0]. If the mapping is not one-to-one, undefined values or overlapping assignments may occur.

Value	Swap[7:6]	Swap[5:4]	Swap[3:2]	Swap[1:0]
	Pipe[31:24]	Pipe[23:16]	Pipe[15:8]	Pipe[7:0]
0	In[31:24]	In[23:16]	In[15:8]	In[7:0]
1	In[7:0]	In[31:24]	In[23:16]	In[15:8]
2	In[15:8]	In[7:0]	In[31:24]	In[23:16]
3	In[23:16]	In[15:8]	In[7:0]	In[31:24]

Div[3:0]	Perform 2D expansion of the source data (real/imaginary).

The data is divided into 2^{Div} segments along the X-axis for a total of 2^{MasterCntl.Radix} elements, and 2^{Div} processing steps are performed along the Y-axis.
If Div is 15, operations on X and Y are disabled.
This setting is used when applying different processing to paired data (e.g., using a window function on the real part while treating the imaginary part differently, or vice versa).

Exp[1:0] Configure the operation for the source data (real/imaginary).

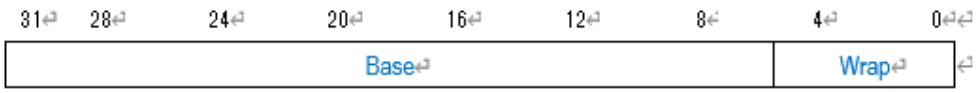
Exp	Description
0	No operation
1	Reserved
2	Zero value
3	Multiplication with the paired component (real if imaginary, imaginary if real)

Format[1:0] Set the format of the source data (real/imaginary).

Format	Description
0	8bpp (positive fixed-point values less than 1.0; if MemCntl.SrcCeil is set to 1, 0xFF is treated as 1.0)
1	16bpp (half-precision float)
2	32bpp (internally treated as integer; uses Gray = (Pipe[23:16] * 2 + Pipe[15:8] * 5 + Pipe[7:0]) / 8)
3	32bpp (single-precision float)

▪5.3.1.10. SrcBase(Re/Im) Command

[Address: 0x24/0x2c]



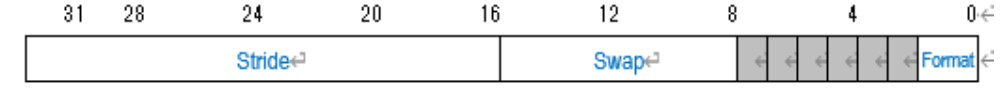
Name	Description
Base[31:6]	refer to MapBase.Base

Wrap[5:0] refer to MapBase.Wrap

Wrap	Description
0	No mask
1-31	Reserved
32-62	Bits [62-n:0] of the generated address are valid.
63	The address value is always the same as the base.

▪ 5.3.1.11. DstInfo(Re/Im) Command

[Address: 0x30/0x38]



Name	Description
Stride[15:0]	refer to SrcInfo.Stride
Swap[7:0]	Configure Byte Swap for the destination data. Performs byte-wise mapping from internal data Pipe[31:0] to output data Out[31:0]. If the mapping is not one-to-one, undefined values or overlapping assignments may occur.

Value	Swap[7:6]	Swap[5:4]	Swap[3:2]	Swap[1:0]
	Out[31:24]	Out[23:16]	Out[15:8]	Out[7:0]
0	Pipe[31:24]	Pipe[23:16]	Pipe[15:8]	Pipe[7:0]
1	Pipe[23:16]	Pipe[15:8]	Pipe[7:0]	Pipe[31:24]
2	Pipe[15:8]	Pipe[7:0]	Pipe[31:24]	Pipe[23:16]
3	Pipe[7:0]	Pipe[31:24]	Pipe[23:16]	Pipe[15:8]

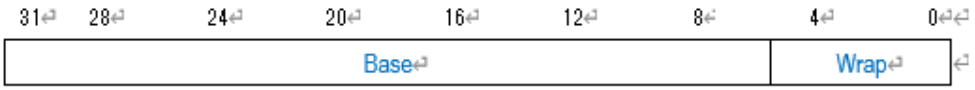
Format[1:0] Set the format of the destination data (real/imaginary).Destination

Format	Description
0	8bpp (positive fixed-point; negative values are treated as 0)
1	16bpp (half-precision float)

2	Reserved
3	32bpp (single-precision float)

▪ 5.3.1.12. DstBase(Re/Im) Command

[Address: 0x34/0x3c]



Name	Description
Base[31:6]	refer to MapBase.Base
Wrap[5:0]	refer to MapBase.Wrap