mmap Specification

Matrix Mul Add Processor

Revision 0.2

30 July 2025

English ver.

1. Overview

1.1. Introduction

- The Matrix Mul Add Processor (hereafter referred to as mmap) is a processor capable of performing matrix operations with arbitrary kernel sizes using the im2col function. It is particularly effective in convolution operations involving multiple input/output channels in deep learning applications.
- It offers excellent functional and performance scalability, as it allows easy expansion of the computation pipeline, enhancement of computational precision, and finer pipeline pitch segmentation. This version utilizes both a 16-bit half-precision floating-point arithmetic unit and an 8-bit floating-point arithmetic unit.
- SRAM is required to repeatedly perform reference, computation, and storage operations. It also serves as an intermediary for data exchange with external memory. In this case, connections to DMA or other processors via SRAM are also necessary.

1.2. Main Parameters

- **4x4 Matrix Mul** × 2^{INR+1} (INR is determined at implementation time), equivalent to 64 × 2^{INR+1} multiply-accumulate units
- **4x4 Matrix Add** × 2^{INR+1} (INR is determined at implementation time), equivalent to 16 × 2^{INR+1} adders
- **Throughput**: Up to 144 × 2^{INR+1} operations per cycle
- **Clock**: Undefined (depends on the implementation process)

1.3. Implementation Parameters

Parameter Name	Description	Default Value
LNR	• Radix of logical SRAM depth number	11
BKR	• Radix of SRAM bank bumber	3
SLR	• Scalar Length Radix • SLR<=16-PNR	10

PNR	• Processor Number Radix	4
MNR	• Matrix Number Radix	4
IMR	• Implement Multiple Radix	2
CLR	• Radix of pss channel number	6
INR	• MNR - 2	2
PN	• 1<<PNR	32
MN	• 1<<MNR	32
IM	• 1<<IMR	4

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
cntlWrStrb	I	+	clk	• Write storobe signal • Evaluate cntlWrAck
cntlWrAck	O	+	clk	• Writ acknowledge signal
cntlWrData[31:0]	I	+	clk	• Write data signal • Evaluate cntlWrAck
cntlRdStrb	I	+	clk	• Read strobe signal • Evaluate cntlRdAck
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
iCID[CLR-1:0]	I	+	clk	• Logocal channel number
iEnd[3:0]	I	+	clk	• Information of end of indexes
iAddr[31:0]	I	+	clk	• Address to fetch context data • Evaluate iVld & !iStall • iAddr[31:8] and iAddr[1:0] indicates program start address • iAddr[7:4] indicates offset of logical register bank • iAddr[3] indicates select of volume whether iDelta or TR register
iDelta[15:0]	I	+	clk	• Transfer volume • Evaluate iVld & !iStall
iIndex[64:0]	I	+	clk	• Five coordinates 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 (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
meFlush	O	+	clk	• Read flush signal
meData[63:0]	I	+	clk	• Read data signal

2.4. SRAM Interface (Write Q)

Signal Name	IO	Pol	Source	Description
ramEQ[16*MN-1:0]	O	+	clk	• Write enable signal (MN port)
ramBQ[BKR-1:0]	O	+	clk	• Write bank signal
ramAQ[15:MNR]	O	+	clk	• Write address signal
ramDQ[16*MN*32-1:0]	O	+	clk	• Write data signal (MN port)

2.5. SRAM Interface (Read A/B)

Signal Name	IO	Pol	Source	Description
ramEC*	O	+	clk	• Read enable signal
ramBC*[BKR-1:0]	O	+	clk	• Read bank signal
ramAC*[15:MNR]	O	+	clk	• Read address signal
ramDC*[16*MN*32-1:0]	I	+	clk	• Read data signal (MN port)

*: A/B

2.6. SRAM Interface (Read Write P)

Signal Name	IO	Pol	Source	Description
ramEP[3:0]	O	+	clk	• Read enable signal (4 port)
ramBP[4BKR-1:0]	O	+	clk	• Read bank signal (4 port)
ramAP[4*(16-MNR)-1:0]	O	+	clk	• Read address signal (4 port)
ramMP[3:0]	O	+	clk	• Read modify signal (4 port)
ramDP[16*MN*32-1:0]	I	+	clk	• Read data signal (MN port)
ramGP	O	+	clk	•

2.7. Scalar Register Interface (A/B)

Signal Name	IO	Pol	Source	Description
scEC*, scECD	O	+	clk	• Read enable signal

scAC*[SLR+PNR-1:0]	O	+	clk	• Read address signal
scDC*[PN*64-1:0]	I	+	clk	• Read data signal

*: A/B

2.8. Utillity

Signal Name	IO	Pol	Source	Description
rstReq	O	+	clk	• Internal reset signal to reset the external system
rstAck	I	+	clk	• Acknowledge of rstReq
reg_swap	O	+	clk	• Swap information for mermoy interface
fReq	I	+	clk	• 1 clock early request against the miReq signal • Use to generate gate signal (for memory controller)
pReq	O	+	clk	• 1 clock early request against the meReq signal • Use to generate gate signal (for memory controller)
gate[IM*4+1:0]	O	+	clk	• Gated clock control signal signifying condition of each internal block(expansion case is gate)
gclk[IM*4+1:0]	I	+	clk	• Gated clock(expansion case is gclk)
clk	I	+	clk	• Clock
reset	I	+	clk	• Synchronous reset signal
reset_n	I	-	clk	• Asynchronous reset

3. Configuration and Operation Description

3.1. Notation

- This document uses the following notations and abbreviations for explanation.

Symbol	Description
[:]	Denotes a bit range A[X:Y] represents data A with MSB at position X and LSB at position Y

$R_b[n]$	Refers to the 32-bit register number n in bank b of the Vector Register (b = 0 to 2^{BKR} - 1, n = 0 to 15). (The b designation may be omitted in some cases.)
SR[n]	Refers to the n-th 32-bit Scalar Register (n = 0 to 255)
Infinity	Represents infinity in IEEE 754 floating-point format
NaN	Represents "Not a Number" in IEEE 754 floating-point format

3.2. System Overview

- As shown in Figure 1, *mmap* is a processor in which 2^{INR+1} processor elements sequentially scan and process the register file (SRAM).

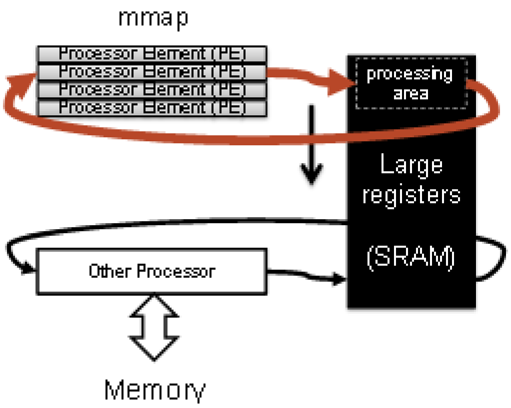


Figure 1 ***mmap*** **system**

- It is not directly connected to memory; instead, data exchange is performed through the register file. Another processor (for example, our company's processors such as *dmap* or *pp*) supplies and retrieves data from the register file.
- The figure represents the overall block structure. The processor is driven by *pss* and continues computation until the program finishes. The architecture is designed such that increasing the number of pipeline stages does not affect performance, and the register file is assumed to be implemented in SRAM

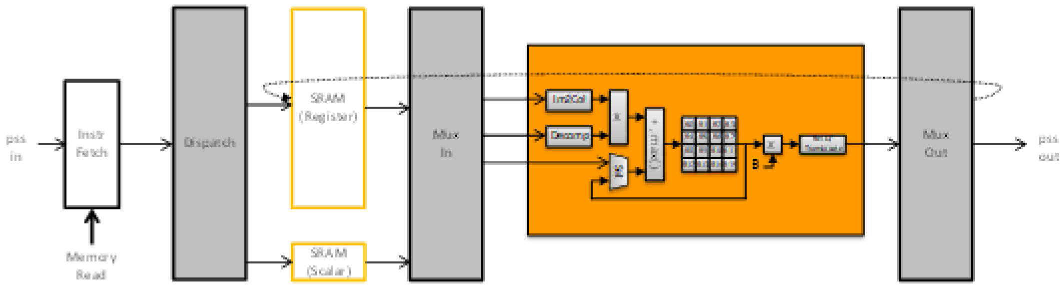


Figure 3 Block Diagram

- The SRAM (register file) consists of 16 16-bit vector registers per unit, organized into 2^{LNR+1} sets × 2^{BKR} banks. The value 2^{LNR+1} corresponds to the maximum number of logical processors. When another processor accesses it in 32-bit mode, even-numbered logical processor IDs map to the lower 16 bits, and odd-numbered ones to the upper 16 bits.

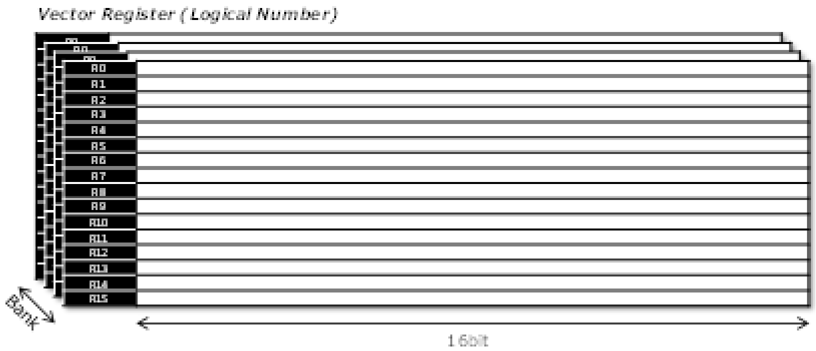


Figure 4 Register File

- Generally, **R[0] to R[15]** in the register file store **4×4** matrix data, and the results of matrix multiplication and addition between banks are stored into the designated bank. If the banks are configured exclusively, the original data can be preserved.
- By utilizing the **im2col** function and organizing the register file data in a specialized layout, the processor supports multiplication, accumulation, and **min/max** operations on arbitrary matrices up to **256×256** in size.
- Using **im2col**, arbitrary numbers of input and output channels can be defined. Channel data is expanded into the register file according to the **step** values specified in the command list.
- Accumulation can be performed using either **16-bit** half-precision floating-point or **8-bit** floating-point formats. Addition is always performed using **16-bit** half-precision floating-point. At the final stage, transposition and application of an activation function for **deep learning** are supported. The activation function can be defined using up to three-piece linear segments, allowing implementation of **ReLU** or **Leaky ReLU**.
- In place of the register file, each logical processor can also reference shared **scalar**

registers. These are used for common coefficients or constants. **Scalar registers** are also shared with other processors and can be accessed bidirectionally.

3.3. Drive Interface (Initiator)

- The *pss* sends XYZW indices to the Initiator of the P-Cube (PX, PY, PZ, PW). The configuration for *pss* (such as processing units) is pre-arranged in memory. The *pss* manages multiple configurations (N in number, depending on implementation) using time division, and after scheduling, it drives the *mmap*.

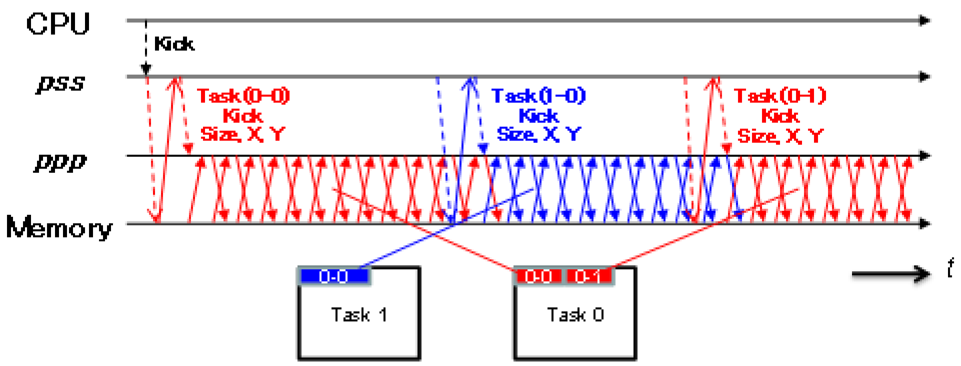


Figure 5 Resource Sharing using *pss*

- The Initiator reads from memory using the start address of the command sent by *pss* and sets up the pipeline. It repeats processing based on the amount indicated by iDelta in *pss*.
  If iAddr[4] is '0', the processing amount is iDelta (1D mode).
  If iAddr[4] is '1', it interprets iDelta as two dimensions: ΔX = iDelta[7:0], ΔY = iDelta[15:8], and performs 2D processing. In the latter case, it does not use the index indicated by *pss*, making it effective for processing in small units.

3.4. Fragmentation Considerations

- In general, fragmentation poses no issues; however, when using the *im2col* function, fragmentation is not allowed.

3.5. Instruction Start and End

- The *mmap* performs processing for the amount indicated by iDelta from *pss* using a single instruction.
  To use it repeatedly, *pss* settings must be configured accordingly.

3.6. Floating-Point Operations

- Multiply-accumulate floating-point operations support both **IEEE 754** half-precision (**16-bit**) and **8-bit** formats.
- Addition floating-point operations are all performed using **IEEE 754** half-precision (**16-bit**) format.
- Depending on the computation result, **NaN** or **Infinity** may be generated. No exception interrupts are triggered. However, the overflow flag may change, and if necessary, the system can assert an interrupt accordingly.

A + B

B \ A	0	A	Infinity	NaN
0	0	A	Infinity	NaN
B	B, -B	A + B, A - B	Infinity	NaN
Infinity	Infinity	Infinity	Infinity, NaN	NaN
NaN	NaN	NaN	NaN	NaN

A * B

B \ A	0	A	Infinity	NaN
0	0	0	NaN	NaN
B	0	A * B	Infinity	NaN
Infinity	NaN	Infinity	Infinity	NaN
NaN	NaN	NaN	NaN	NaN

3.7. im2col Matrix Operations

- By arranging the data in the register file in a specialized layout and setting **TypeP[0]** in the command list to "1", it becomes possible to perform arbitrary matrix multiplication, accumulation, and **min/max** operations using the *im2col* function, supporting matrix sizes up to **256×256**.
- In *im2col* matrix operations, arbitrary numbers of input channels (C) and output channels (**M**) can be defined. Channel data is expanded into the register file according to **Step** parameters specified in the command list.

3.8. Standard Matrix Operations

- Supports execution of 4×4 matrix AxP + B operations. Set **TypeP[0]** to **"0"**.
- The number of 4×4 matrices to be processed is defined by *pss* size (W0).
- For FP16 data format, up to 8 simultaneous 4×4 matrix operations are executed. For FP8, up to 16 simultaneous 4×4 matrix operations are executed.

3.9. Hadamard Matrix Operations

- Supports execution of 4×4 Hadamard matrix operations. Set **TypeP[1]** to **"1"**.
- The number of 4×4 matrices to be processed is defined by *pss* size (W0).
- For FP16 data format, up to 8 simultaneous 4×4 Hadamard operations are executed. For FP8, up to 16 simultaneous 4×4 Hadamard operations are executed.

3.10. Min/Max Function

- By configuring **Min/Max** settings in *im2col* matrix operations, it is possible to implement **MinPooling/MaxPooling** functionalities.

3.11. Register File Structure

- The **SRAM** (register file) is composed of 16 **16-bit vector registers** per unit, organized into 2^{LNR+1} sets × 2^{BKR} banks.

3.12. Parallel Circuits

- The multiply-accumulate units used for matrix operations are structured in parallel as follows:
  - 2 parallel units for FP16 support
  - 4 parallel units for FP8 support

3.13. Computational Error

This section is under preparation.

4. Control Register Description

4.1. Overview

- Control registers are accessed via the control bus. Unlike R[n], these registers are shared settings across the processor, and include scalar registers.
- 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 with Clear on write
- Do not access registers marked as Reserved. When writing to reserved fields, set the value to ‘0’.
- In address and data notations, ‘x’ indicates don’t care.

4.2. Definition

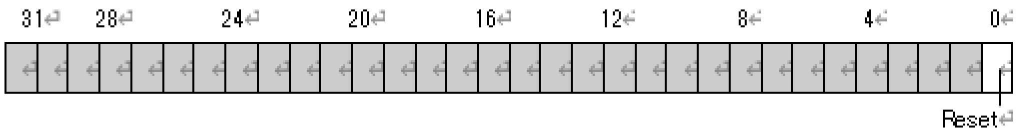
Address	Register Name	Description
0000_0000	Reset	Reset control
0000_0004	System	Gated Clock Control、Data Bus Control
0000_0008	–	Reserved
0000_000c	–	Reserved
0000_0010	Control	Master control
0000_0014	Val	Coefficient Setting
0000_0018	ConstAB	Coefficient Setting
0000_001c	—	Reserved
0000_0020	Select	
0000_0024	–	Reserved
0000_0028	–	Reserved
0000_002c	–	Reserved
0000_0030	–	Reserved
0000_0034	–	Reserved
0000_0038	–	Reserved
0000_003c	–	Reserved

0000_0040	MonitorXY	Active Index XY
0000_0044	MonitorZW	Active Index ZW
0000_0048	MonitorST	Active Index ST
0000_004c	MonitorID	Active Index ID
0000_0050	BreakXY	Break index XY
0000_0054	BreakZW	Break index ZW
0000_0058	BreakST	Break index ST
0000_005c	BreakID	Break index ID
0000_0060	BCondition0	
0000_0064	BCondition1	
0000_0068	–	Reserved
0000_006c	–	Reserved

4.3. Details

4.3.1.1. Reset Register

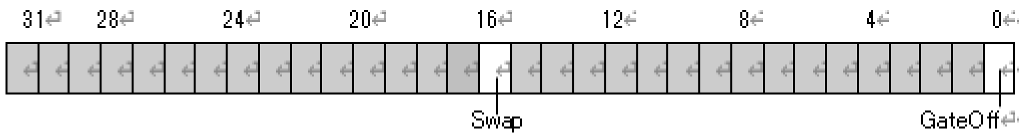
[Address: 0x0000_0000]



Name	Type	Default	Description
Reset	R/W	0	Synchronous Reset After being set to ‘1’, the internal reset state is triggered and then automatically cleared. Unlike the reset_n signal, the contents of other registers are preserved.

4.3.1.2. System Register

[Address: 0x0000_0004]



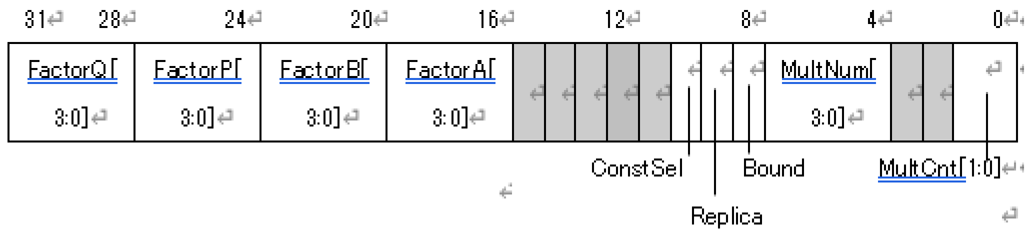
Name	Type	Default	Description
Swap	R/W	0	Configures the final byte swap with memory.

The behavior follows the same specification as instruction-based swapping (refer to the 2D access mode of memory instructions).
For write operations, the swap specified in the instruction is applied after this swap.
For read operations, the swap specified in the instruction is applied after this swap.

Name	Type	Default	Description
GateOff	R/W	0	Gated Clock Off Mode When set to ‘1’, all bits of the **gate** signal are forcibly fixed to ‘1’.

4.3.1.3. Control Register

[Address: 0x0000_0010]



Name	Type	Default	Description
FactorQ[3:0]	R/W	0	
FactorP[3:0]	R/W	0	
FactorB[3:0]	R/W	0	
FactorA[3:0]	R/W	0	
ConstSel	R/W	0	

ConstSel	Use
0	StepPM, StepQC
1	reg_constAB /reg_slope, reg_thresh

Name	Type	Default	Description
Replica	R/W	0	

Bound	R/W	0	
MultNum[3:0]	R/W	0	
MultCnt[1:0]	R/W	0	・MAX represents the maximum parallelism of Channel M. Automatically selected based on TypeP

MultCntl	M並列度
0	Auto (1,2,4,,, MAX)
1	Auto (1 or MAX)
2	Fixed (MultNum)
3	Reserved

4.3.1.4. Value Register

[Address: 0x0000_0014]

31	28	24	20	16	12	8	4	0
Th[15:0]					Slope[15:0]			

Name	Type	Default	Description
Th	R/W	0	・Threshold (Th) applied to the LUT used for the final output value
Slope	R/W	0	・Slope applied to the LUT used for the final output value

4.3.1.5. ConstAB Register

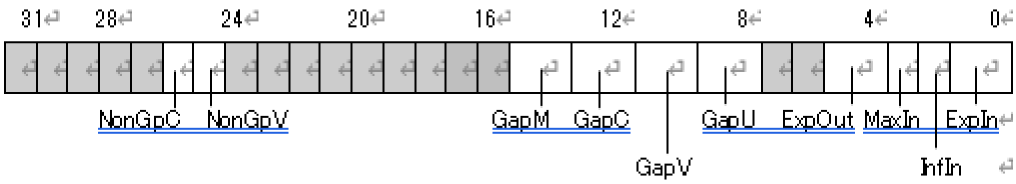
[Address: 0x0000_0018]

31	28	24	20	16	12	8	4	0
ConstB[15:0]					ConstA[15:0]			

Name	Type	Default	Description
ConstB	R/W	0	・Fixed value to be set for B
ConstA	R/W	0	・Fixed value to be set for A

4.3.1.6. Select Register

[Address: 0x0000_0020]



Name	Type	Default	Description
NonGpC	R/W	0	・

NonGp	SRAM Group
0	if Step=0 force to 0
1	none

Name	Type	Default	Description
NonGpV	R/W	0	・
GapM	R/W	0	・
GapC	R/W	0	・
GapV	R/W	0	・
GapU	R/W	0	・
ExpOut	R/W	0	・Format selection for FP8 output

ExpOut	fp8 Output
0	1-5-2
1	1-4-3
2	1-3-4
3	Reserved

Name	Type	Default	Description
MaxIn	R/W	0	・

MaxIn	Exp Max
0	1f
1	1e

Name	Type	Default	Description
InfIn	R/W	0	・

InfIn	Max(7fff/ffff) x Zero
0	7ffff/ffff
1	0

Name	Type	Default	Description
ExpIn	R/W	0	・Format selection for FP8 output

ExpIn	fp8 Input
0	1-5-2
1	1-4-3
2	1-3-4
3	Reserved

4.3.1.7. Monitor Register

[Address: 0x0000_0040]

31	28	24	20	16	12	8	4	0
IndexY[15:0]					IndexX[15:0]			

Name	Type	Default	Description
IndexY	R	0	Indicates the currently executing index Y
IndexX	R	0	Indicates the currently executing index X

4.3.1.8. MonitorZW Register

[Address: 0x0000_0044]

31	28	24	20	16	12	8	4	0
IndexW[15:0]					IndexZ[15:0]			

Name	Type	Default	Description
IndexW	R	0	Indicates the currently executing index W
IndexZ	R	0	Indicates the currently executing index Z

4.3.1.9. MonitorST Register

[Address: 0x0000_0048]

31	28	24	20	16	12	8	4	0
IndexT[15:0]					IndexS[15:0]			

Name	Type	Default	Description
IndexT	R	0	Indicates the currently executing index T
IndexS	R	0	Indicates the currently executing index S

4.3.1.10. MonitorID Register

[Address: 0x0000_004c]

31	28	24	20	16	12	8	4	0
							ID[5:0]	

Name	Type	Default	Description
ID	R	0	Indicates the currently executing index ID

4.3.1.11. BreakXY Register

[Address: 0x0000_0050]

31	28	24	20	16	12	8	4	0
IndexY[15:0]				IndexX[15:0]				

Name	Type	Default	Description
Y	R/W	0	Specifies the break index Y. Processing will halt before this index is executed. Setting the value to ‘0’ cancels the break.
X	R/W	0	Specifies the break index X. Performs the same operation as index Y.

4.3.1.12. BreakZW Register

[Address: 0x0000_0054]

31	28	24	20	16	12	8	4	0
IndexW[15:0]				IndexZ[15:0]				

Name	Type	Default	Description
W	R/W	0	Specifies the break index W. Performs the same operation as index Y.
Z	R/W	0	Specifies the break index Z. Performs the same operation as index Y.

4.3.1.13. BreakST Register

[Address: 0x0000_0058]

31	28	24	20	16	12	8	4	0
IndexT[15:0]				IndexS[15:0]				

Name	Type	Default	Description
IndexT	R/W	0	Specifies the break index T.Performs the same operation as index Y.
IndexS	R/W	0	Specifies the break index S. Performs the same operation as index Y.

4.3.1.14. BreakID Register

[Address: 0x0000_005c]

31	28	24	20	16	12	8	4	0
							ID[5:0]	

Name	Type	Default	Description
ID	R/W	0	Specifies the break ID. Performs the same operation as index X.

4.3.1.15. BCondition0 Register

[Address: 0x0000_0060]

31	28	24	20	16	12	8	4	0
BreakCondition[31:0]								

Name	Type	Default	Description
BreakCondition	R/W	0	

4.3.1.16. BCondition1 Register

[Address: 0x0000_0064]

31	28	24	20	16	12	8	4	0
BreakCondition[63:32]								

Name	Type	Default	Description
BreakCondition	R/W	0	

5. Command List Description

5.1. Overview

- The **command list** has its starting address specified by the address value output from **pss**. After **MMAP** is activated, it retrieves the command list and stores it into internal registers.
- The command list is **independently managed at each stage** within the pipeline. As a result, even during pipeline operation, it is possible to hold different command lists and drive the executable stages accordingly. Therefore, synchronization commands are **not required**.
- For **reserved registers and fields**, set the value to **‘0’**.
- The addresses shown are **relative to the address value output from pss**.

5.2. Definition

Address	Command Name	Description
00	Cntl	Control Command
04	Format	Data Format Setting
08	Offset	Coordinate Offset / Vector Register Bank Configuration
0C	Gap	Gap Configuration per Channel
10	Size	Size Configuration per Channel
14	Size	Size Configuration per Channel
18	StepO	Step Configuration per Channel
1C	StepO	Step Configuration per Channel
20	StepA	Step Configuration per Channel
24	StepA	Step Configuration per Channel
28	StepB	Step Configuration per Channel
2C	StepB	Step Configuration per Channel
30	StepP	Step Configuration per Channel
34	StepP	Step Configuration per Channel
38	StepQ	Step Configuration per Channel
3c	StepQ	Step Configuration per Channel

5.3. Details

5.3.1.1 Cntl Command

The command details are described below.

[Address: 0x0000_0000]

31 28 24 20 16 12 8 4 0

Util	OL(Output Lut)	KV	KU

Name	Size	Description
Util	4	・[0]: U, [1]: V Sampling Point Configuration

Util[0][1]	Sampling Point U/V
0	+0
1	+1

Util[2]	Gap
0	Zero Write
1	Mask Write

・Parallel Circuit Usage Configuration

Setting to “1” disables the use of parallel circuits.

Util[3]	Multi Force One
0	Auto Set
1	Force 1

・Next address when the bank address reaches the end of the bank

Util[4]-[7]	Bank Bound [A][B][P][Q]
0	Carry
1	Ring

OL 8 LUT selection applied to the final output value

・For each of the four regions into which the output value is divided Select from the four output processing options shown in the table below.

・The boundaries of the regions are configured using Th in 4.3.1.4 Value Register

OL[7:6] −∞ ～ −Th

OL[5:4] −Th ～ 0

OL[3:2] +Th ～ +∞

OL[1:0] 0 ～ +Th

Op	Output
0	f(x)
1	f(x)*slope
2	0
3	(sign ? −thresh : thresh) *(1 or slope)

KV 8 Set the kernel size (vertical direction) to the value of **size 1**.

KU 8 Set the kernel size (horizontal direction) to the value of **size 1**.

5.3.1.2 Format Command

[Address: 0x0000_0004]

31 28	24	20	16	12	8	4	0
FormQ	FormP	FormB	FormA	TypeQ	TypeP	TypeB	TypeA

Name	Size	Description

FormQ 4

・Output Q value format setting

FormQ[3][2]	Compress U/V
0	-
1	On

Form*[1:0]	Data Type
0	fp16
1	fp8
2	
3	

FormP 4

・Input P value format setting

Form*[1:0]	Data Type
0	fp16
1	fp8
2	
3	

FormB 4

・Input B value format setting

Form*[1:0]	Data Type
0	fp16
1	fp8
2	
3	

・Coefficient Expansion Direction Setting

Automatically set to "1" during **im2col**. Automatic behavior can be disabled by setting **reg_autoRepOff = 1**.

FormB[3:2]	Coef Arrange on im2col
0	U Replica
1	M Replica
2	M+ Replica
3	Reserved

FormA 4

Input A value format setting

Form*[1:0]	Data Type
0	fp16
1	fp8
2	fp4
3	fp2

・Coefficient Expansion Direction Setting

FormA[3:2]	Coef Arrange on im2col
0	U Replica
1	M Replica
2	Reserved
3	Reserved

TypeQ 4

・Calculation Setting for Output Q

TypeQ[0]	Translocate
0	-
1	On

TypeQ[1]	Flat (no twiddle)
0	-
1	On

TypeQ[3:2]	new	not new
0	B	B
1	Acc	B
2	B	Acc
3	Acc	Acc

TypeP 4 ・Calculation Setting for Input P

TypeP[0]	Property
0	Normal
1	Im2Col

TypeP[1]	Product
0	Normal
1	Hadamard

TypeP[3:2]	Merge
0	Normal
1	-
2	Min
3	Max

TypeB 4 ・Configuration of storage location or fixed value for input B

TypeAB[2:0]	Operand
0	RegisterVec(addr)
1	RegisterSel(addr)
2	ScalarVec(addr)
3	ScalarSel(addr)
4	Zero
5	One
6	Identity
7	Const

TypeAB[3]	Address Shrink
0	-
1	On

TypeA 4 ・Input B Value: Storage Location or Fixed Value Configuration

Same as TypeB

5.3.1.3 Offset Command

[Address: 0x0000_0008]

31 28	24	20	16	12	8	4	0
OffM	OffC	OffV	OffU	BankQ	BankP	BankB	BankA

Name	Size	Description
OffM	4	Processing Start Coordinate Offset Setting

		Negative values are represented in two's complement. The valid range is −8 to 7.
OffC	4	Processing Start Coordinate Offset Setting Negative values are represented in two's complement format. The valid range is −8 to 7.
OffV	4	Processing Start Coordinate Offset Setting Negative values are represented in two's complement format. The valid range is −8 to 7.
OffU	4	Processing Start Coordinate Offset Setting Negative values are represented in two's complement format. The valid range is −8 to 7.
BankQ	4	Register File Bank Configuration
BankP	4	Register File Bank Configuration
BankB	4	Register File Bank Configuration
BankA	4	Register File Bank Configuration

5.3.1.4. Gap Command

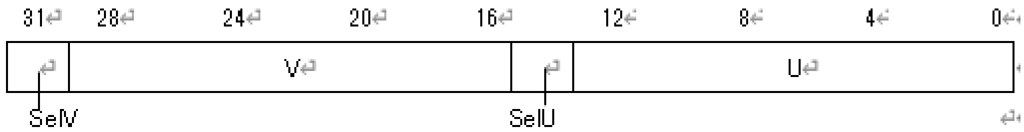
[Address: 0x0000_000c]

31 28 24	20 16	12 8	4 0
M	C	V	U

Name	Size	Description
M	8	・Gap value set for the output channel
C	8	・Gap value set for the intput channel
V	8	・Gap value set in the vertical (V) direction
U	8	・Gap value set in the vertical (U) direction

5.3.1.5. Size Command

[Address: 0x0000_0010]



Name	Size	Description
SelV	2	Handling when exceeding the valid data range

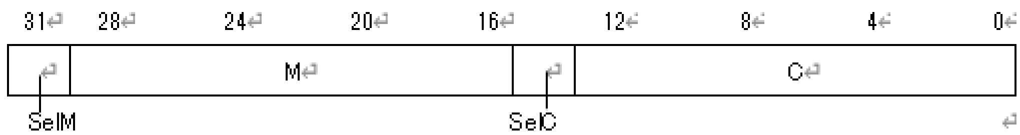
Size.Sel		TypeP[3:2]		
		Norm	Min	Max
0	-	-	-	-
1	under border	0	+PM	-
2	over border	0	-	-PM
3	out of border	0	+PM	-PM

PM: Possible Max (0x7bff)

Name	Size	Description
V	14	Matrix size of input data (vertical direction) Set the value of size n minus 1.
SelU	2	Same as SelV
U	14	Matrix size of input data (Horizontal direction) Set the value of size n minus 1.

5.3.1.6. Size Command

[Address: 0x0000_0014]

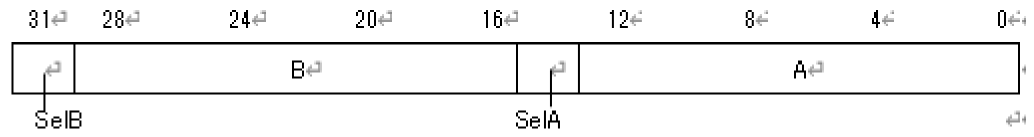


Name	Size	Description
SelM	2	Same as SelV
M	14	Output channel size Set the value of size n minus 1.。
SelC	2	Same as SelV

C	14	Input channel size Set the value of size n minus 1.

5.3.1.7. StepO Command

[Address: 0x0000_0018]



Name	Size	Description
SelB	2	・Selection of Step Setting Application Method

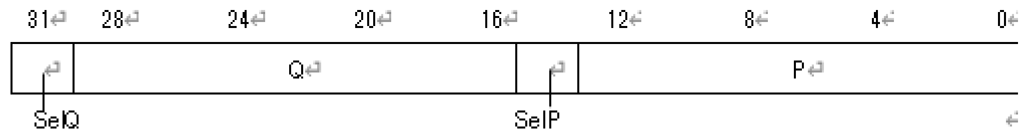
Step*[14]	Gap Mask/Zero
0	No Gap
1	Gap*[*[2:0]]

Step*[15]	Step * Index Operation
0	Step[13:0]*Index
1	Step[11:0]*Index >> (Step[13:12]+1) & (-1 << (Step[13:12]+1))

B	14	・Base address of input B within the bank
SelA	2	same as SelB
A	14	・Base address of input A within the bank

5.3.1.8. StepO Command

[Address: 0x0000_001c]

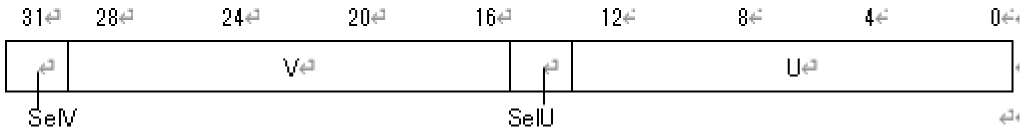


Name	Size	Description
SelQ	2	Same as SelB in StepO
Q	14	・Base address of output Q within the bank
SelP	2	Same as SelB in StepO

P	14	・Base address of input P within the bank

5.3.1.9. StepA Command

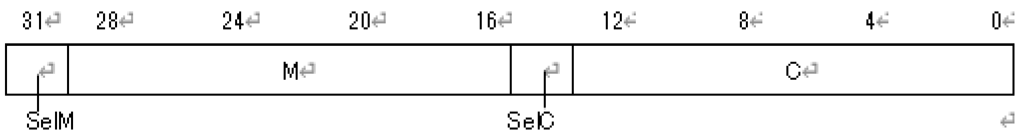
[Address: 0x0000_0020]



Name	Size	Description
SelV	2	Same as SelB in StepO. [14] Gap function is disabled.
V	14	Change in the storage position of input A when coordinate **V** changes by “1” during computation. Specify the value in terms of the number of data elements.
SelU	2	Same as SelB in StepO. [14] Gap function is disabled.
U	14	Change in the storage position of input A when coordinate **U** changes by “1” during computation. Specify the value in terms of the number of data elements.

5.3.1.10. StepA Command

[Address: 0x0000_0024]

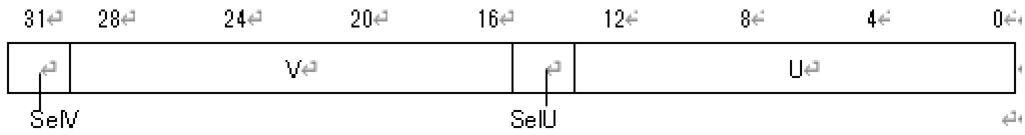


Name	Size	Description
SelM	2	Same as SelB in StepO. [14] Gap function is disabled.
M	14	・Change in the storage position of input A when output channel **M** changes by “1” during computation. Specify the value in terms of the number of data elements.
SelC	2	Same as SelB in StepO. [14] Gap function is disabled.

C	14	･Change in the storage position of input A when input channel **C** changes by “1” during computation. Specify the value in terms of the number of data elements.

5.3.1.11. StepB Command

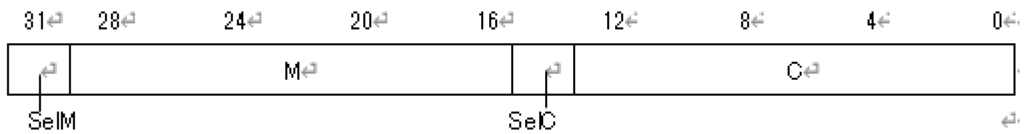
[Address: 0x0000_0028]



Name	Size	Description
SelV	2	Same as SelB in StepO. [14] Gap function is disabled.
V	14	Change in the storage position of input B when coordinate **V** changes by “1” during computation. Specify the value in terms of the number of data elements.
SelU	2	Same as SelB in StepO. [14] Gap function is disabled.
U	14	Change in the storage position of input B when coordinate **U** changes by “1” during computation. Specify the value in terms of the number of data elements.

5.3.1.12. StepB Command

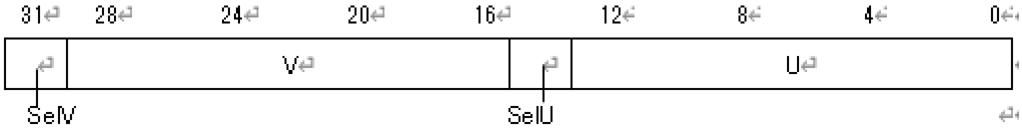
[Address: 0x0000_002c]



Name	Size	Description
SelM	2	Same as SelB in StepO. [14] Gap function is disabled.
M	14	Change in the storage position of input A when output channel M changes by “1” during computation. Specify the value in terms of the number of data elements.
SelC	2	Same as SelB in StepO. [14] Gap function is disabled.
C	14	Change in the storage position of input A when input channel C changes by “1” during computation. Specify the value in terms of the number of data elements.

5.3.1.13. StepP Command

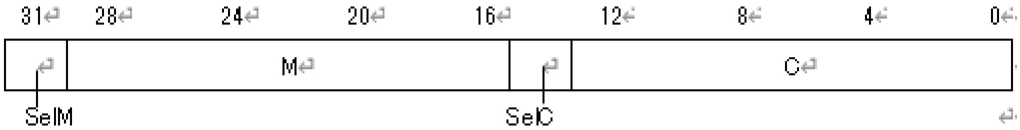
[Address: 0x0000_0030]



Name	Size	Description
SelV	2	Same as SelB in StepO. [14] Gap function is disabled.
V	14	・Change in the storage position of input **P** when coordinate **V** changes by "1" during computation. Specify the value in terms of the number of data elements.
SelU	2	Same as SelB in StepO. [14] Gap function is disabled.
U	14	Change in the storage position of input **P** when coordinate **U** changes by "1" during computation. Specify the value in terms of the number of data elements.

5.3.1.14. StepP Command

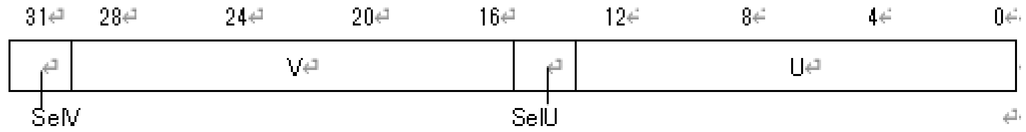
[Address: 0x0000_0034]



Name	Size	Description
SelM	2	Same as SelB in StepO. [14] Gap function is disabled.
M	14	Change in the storage position of input **P** when output channel **M** changes by "1" during computation. Specify the value in terms of the number of data elements. Normally set to **0**.
SelC	2	Same as SelB in StepO. [14] Gap function is disabled.
C	14	Change in the storage position of input **P** when input channel **C** changes by "1" during computation. Specify the value in terms of the number of data elements.

5.3.1.15. StepQ Command

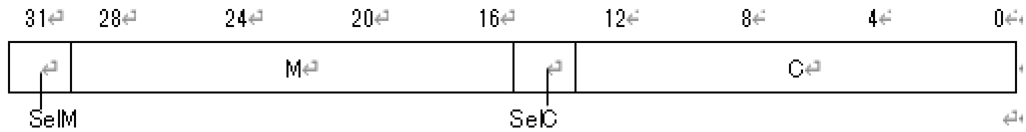
[Address: 0x0000_0038]



Name	Size	Description
SelV	2	Same as SelB in StepO. [14] Gap function is disabled.
V	14	Change in the storage position of output **Q** when coordinate **V** changes by "1" during computation. Specify the value in terms of the number of data elements.
SelU	2	Same as SelB in StepO. [14] Gap function is disabled.
U	14	Change in the storage position of output **Q** when coordinate **U** changes by "1" during computation. Specify the value in terms of the number of data elements.

5.3.1.16. StepQ Command

[Address: 0x0000_003c]



Name	Size	Description
SelM	2	Same as SelB in StepO. [14] Gap function is disabled
M	14	Change in the storage position of output **Q** when output channel **M** changes by "1" during computation. Specify the value in terms of the number of data elements.
SelC	2	Same as SelB in StepO. [14] Gap function is disabled
C	14	Change in the storage position of output **Q** when input channel **C** changes by "1" during computation. Specify the value in terms of the number of data elements. Normally set to **0**.

6. Application Note

6.1. Example Implementation

6.1.1. conv2d (im2col)

- This is an example **command list** (**CL**) used for **conv2D(im2col)** processing.
    - Input data matrix size: **4×4**
    - Number of input channels: **4**
    - Number of output channels: **3**

カーネルサイズ　Type 設定　レジスタファイルバンク設定

```
64'h00000202_00008131, 64'h00002100_00000000,
```

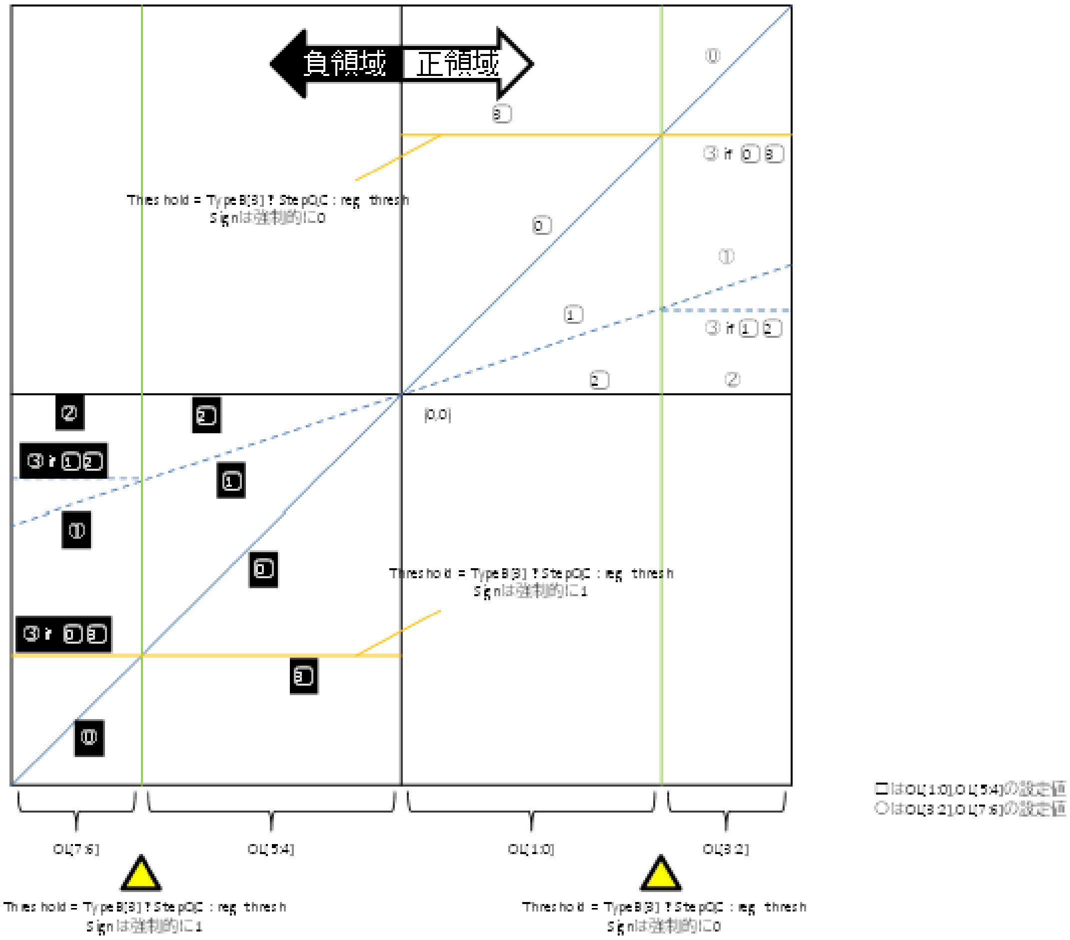
入力マトリクスサイズ　入出力チャネル数

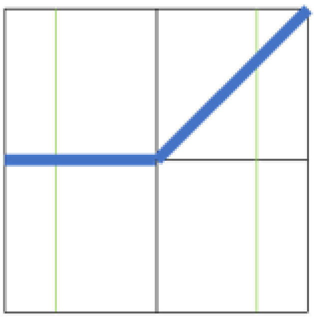
```
64'h00030003_00030003, 64'h00000000_00000000,

64'h00000000_00090001, 64'h00000000_00100000,

64'h00200001_00000020, 64'h00200001_00200000);
```

Step 設定

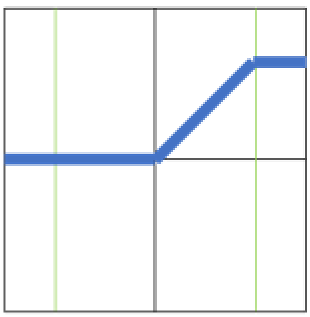
6.1.2. LUT Applied to the Final Output Value

- The relationship between each setting value and the output is shown in the diagram below.

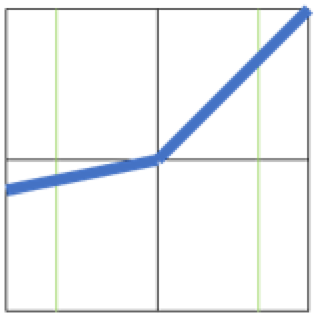




LeRU: OL = 01010101
(傾き=1なら01010000でもよい)



LeRUx: OL = 10101101



Leaky LeRU: OL = 00000101