

27. Convolutional Neural Network (CNN)

27.1 Overview

The CNN accelerator consists of 64 parallel processors with 512KB of SRAM-based storage. Each processor includes a pooling unit and a convolutional engine with dedicated weight memory. Four processors share one data memory. These are further organized into groups of 16 processors that share common controls. A group of 16 processors operates as a slave to another group or independently. Data is read from SRAM associated with each processor and written to any data memory located within the accelerator. Any given processor has visibility of its dedicated weight memory and to the data memory instance it shares with the three others.

Maxim provides a complete toolset for the CNN, including model training and synthesis on the Maxim Integrated AI GitHub repository. Refer to https://github.com/MaximIntegratedAI/MaximAI_Documentation for a full suite of tools and training available for the CNN.

The features of the CNN accelerator include:

- 512KB SRAM data storage:
 - ♦ Configured as 8K×8-bit integers x64 channels or 32K×8-bit integers x4 channels for input layers
 - ♦ Hardware load and unload assist.
- 64 parallel physical channel processors:
 - ♦ Organized as 4×16 processors.
 - ♦ 8-bit integer data path with an option for 32-bit integers on the output layer
 - ♦ Per-channel processor enable/disable.
 - ♦ Expandable to 1024 parallel logical channel processors.
- 1×1 or 3×3 2D kernel sizes
- Configurable 1D kernel size to 1×9
- Full resolution sum-of-products arithmetic for 1024 8-bit integer channels
- Operating frequency up to 50MHz.
- Nominal 1 output channel per clock, maximum 4 output channels per clock (pass through).
- Configurable input layer image size:
 - ♦ 32K pixels, 16 channels, non-streaming.
 - ♦ 8K pixels, 4 channels, non-streaming.
 - ♦ 1024×1024 pixels, 4 channels, streaming.
- Hidden layers:
 - ♦ Up to 8K 8-bit integer data per channel, x64 channels, non-streaming.
 - ♦ 8K bytes can be split equally across 1 to 16 logical channels, non-streaming.
 - ♦ 1M 8-bit integer data per channel, x64 channels, streaming.
 - ♦ 1M bytes can be split equally across eight layers, streaming.
- Optional interrupt on CNN completion.
- User-accessible BIST on all SRAM storage
- User-accessible zeroization of all SRAM storage
- Single-step operation with full data SRAM access for CNN operation debug.

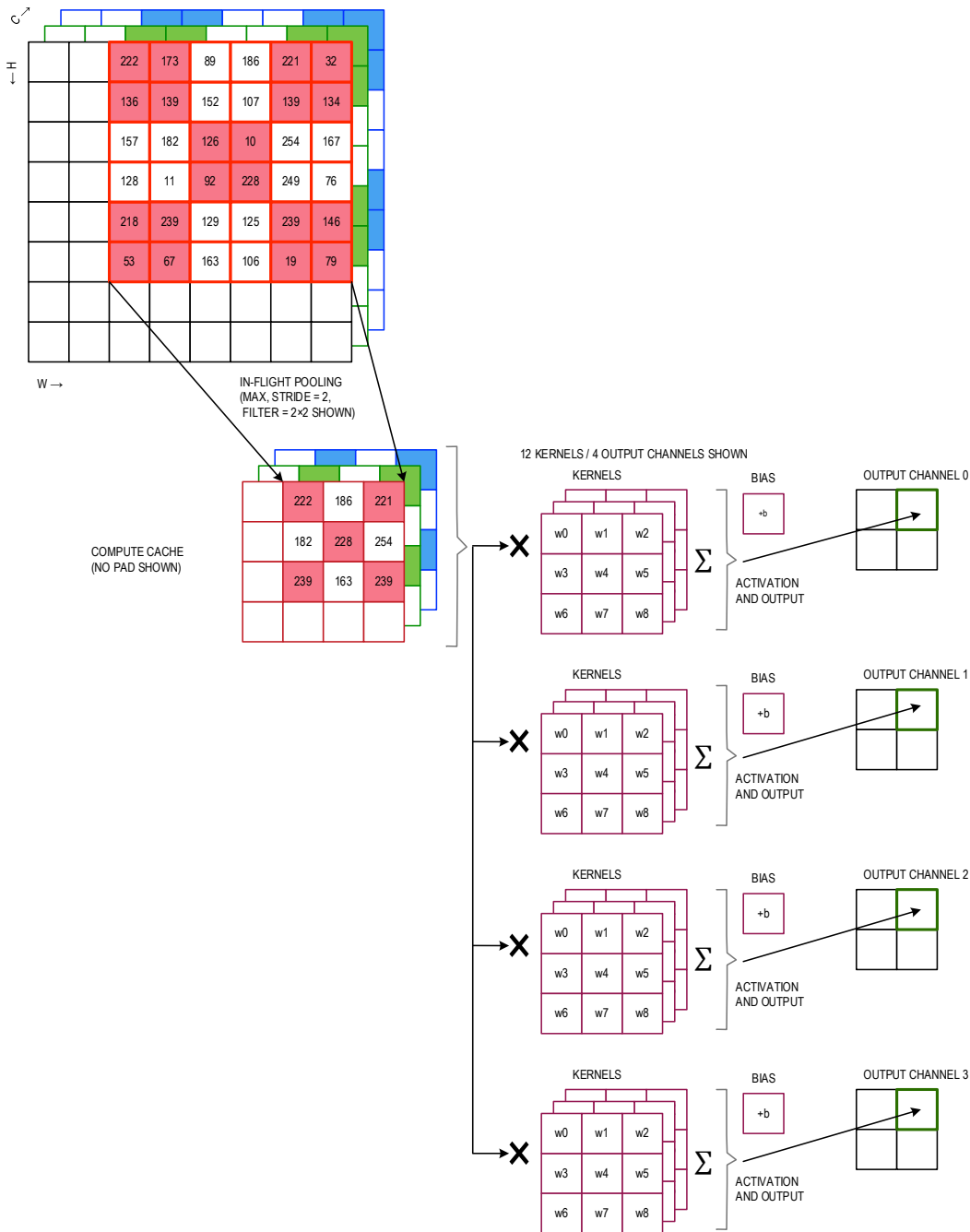
- Flexible power management:
 - ♦ Independent x16 processor supply enables.
 - ♦ Independent x16 processor mask retention enables.
 - ♦ Independent x16 data path clock enables.
 - ♦ Active Arm peripheral bus clock gating with per x16 processor override
 - ♦ CNN clock frequency scaling (divide by 2, 4, 8, 16).
 - ♦ Chip-level voltage control for performance power optimization.
- Configurable weight storage:
 - ♦ SRAM-based weight storage with selectable data retention
 - ♦ Configurable from 442,368 8-bit integer weights to 3.538M 1-bit logical weights:
 - Organized as 768×9×64 8-bit integer weights to 768×72×64 1-bit logical weights.
 - Can be configured on a per-layer basis.
 - ♦ Programmable per x16 processor weight RAM start address, start pointer, and mask count.
 - ♦ Optional weight load hardware assist for packed weight storage.
- 32 independently configurable layer groups:
 - ♦ Each group can contain element-wise, and/or pooling, and/or convolution operations for a minimum of 32 and a maximum of 96 layers.
 - ♦ Processor and mask enables (16 channels)
 - ♦ Input data format
 - ♦ Per-layer data streaming:
 - Stream start - relative to prior stream
 - Dual-stream processing delay counters - 1 column, 1 row delta counter
 - Data SRAM circular buffer size
 - ♦ Input data size (row, column)
 - ♦ Row and column padding 0 to 4 bytes
 - ♦ Number of input channels 1 to 1024
 - ♦ Kernel bit width size (1, 2, 4, 8)
 - ♦ Kernel SRAM start pointer and count
 - ♦ Inflight input image pooling:
 - Pool mode - none, maximum or average
 - Pool size - 1x1 to 16x16
 - ♦ Stride - 1 row, 1 column to 4 rows, 4 columns
 - ♦ Data SRAM read pointer base address
 - ♦ Data SRAM write pointer configuration:
 - Base address
 - Independent offsets for output channel storage in SRAM
 - Programmable stride increment offset

- ◆ Bias - 2048 8-bit integers
- ◆ Pre-activation output scaling from 0 to ± 15 -bits
- ◆ Output activation: none (implicit clipping), ReLU, absolute value
- ◆ Pass through: 8-bit or 32-bit integers
- ◆ Element-wise operations (add, subtract, xor, or) with optional convolution - up to 16 elements
- ◆ Deconvolution (upsampling)
- ◆ Flattening for MLP processing

A typical CNN operation consists of pooling followed by a convolution. While these are traditionally expressed as separate layers, pooling can be done "in-flight" on the MAX78000 for greater efficiency.

The accelerator is optimized for convolutions with in-flight pooling on a sequence of layers to minimize data movement. The MAX78000 also supports in-flight element-wise operations, pass-through layers, and 1-D convolutions without element-wise operations. [Figure 27-1](#) shows a high-level diagram of the MAX78000's convolutional neural network flow.

Figure 27-1: CNN Overview



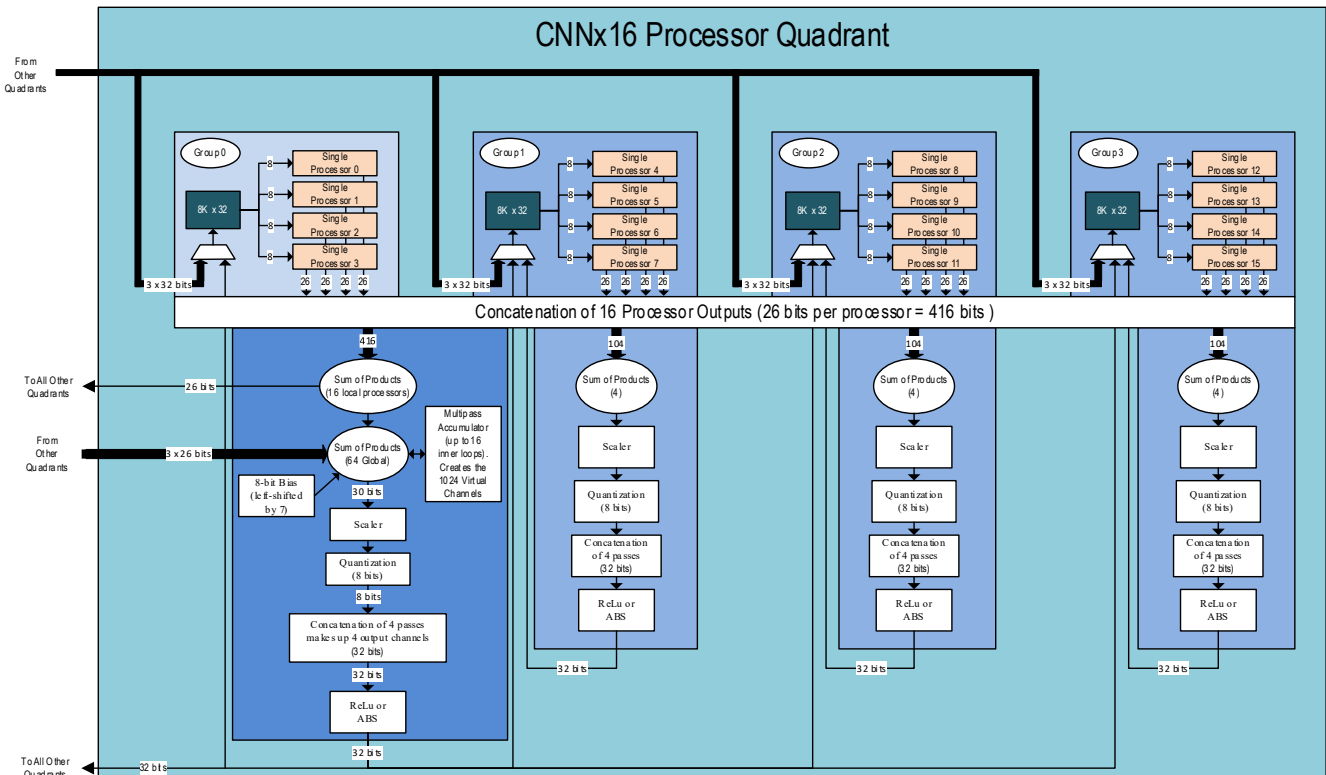
27.2 Instances

There is one instance of the CNN. The CNN contains four CNNx16 processor quadrants, generically referred to as CNNx16_n. Each CNNx16_n processor quadrant contains sixteen processors grouped in four groups of four. These groups are labeled group 0 to group 3 in the CNNx16_n processor quadrant block diagram and are referred to as CNNx16_n_q0 to CNNx16_n_q3 in the documentation.

27.2.1 Block Diagram

Figure 27-2 shows a block diagram of a single CNNx16 Processor Quadrant. The MAX78000 includes four CNNx16 processor quadrants. The processor groups are labeled in Figure 27-2 as Group 0 to Group 3.

Figure 27-2: CNNx16_n Processor Quadrant Block Diagram



27.3 Memory Configuration

The CNN includes four CNNx16 processor arrays, and each processor array includes 128KB of SRAM, 110KB of mask RAM (MRAM), 512B of bias RAM (BRAM), and 98KB of tornado RAM (TRAM). [Figure 27-3](#) shows the APB mapping for the CNN and the Quad CNNx16n processor arrays.

Figure 27-3: CNN Global and Quad CNNx16n Processor Array APB Memory Map

0x50C1_FFFF	APB Access CNNx16_0 Quadrant 2	128KB SRAM	0x5101_FFFF	APB Access CNNx16_0 Quadrant 3	128KB SRAM
0x50C0_0000		Reserved	0x5100_0000		Reserved
0x50BF_FFFF		110KB Mask RAM 16 Blocks of 16KB (256KB) allocated space. Only the lower 6,912B of each block has Mask RAM.	0x50FF_FFFF		110KB Mask RAM 16 Blocks of 16KB (256KB) allocated space. Only the lower 6,912B of each block has Mask RAM.
0x509C_0000		Reserved	0x50DC_0000		Reserved
0x509B_FFFF		98KB TRAM 16 Blocks of 16KB (256KB) allocated space. Only the lower 6,144B of each block has TRAM.	0x50DB_FFFF		98KB TRAM 16 Blocks of 16KB (256KB) allocated space. Only the lower 6,144B of each block has TRAM.
0x5098_0000		512B Bias RAM 32KB allocated space	0x50D8_0000		512B Bias RAM 32KB allocated space
0x5097_FFFF		Reserved	0x50D7_FFFF		Reserved
0x5095_0000		Reserved	0x50D5_0000		Reserved
0x5094_FFFF		Global Control and Status	0x50D4_FFFF		Global Control and Status
0x5091_0000		Reserved	0x50D1_0000		Reserved
0x5090_9FFF		Per Layer Control and Status	0x50D0_9FFF		Per Layer Control and Status
0x5090_8000		Reserved	0x50D0_8000		Reserved
0x5090_7FFF		Global Control and Status	0x50D0_7FFF		Global Control and Status
0x5090_5000		Reserved	0x50D0_5000		Reserved
0x5090_4FFF		Global Control and Status	0x50D0_4FFF		Global Control and Status
0x5090_1000		Reserved	0x50D0_1000		Reserved
0x5090_0FFF		Per Layer Control and Status	0x50D0_0FFF		Per Layer Control and Status
0x5090_0A90		Reserved	0x50D0_0A90		Reserved
0x5090_0A8F		Global Control and Status	0x50D0_0A8F		Global Control and Status
0x5090_0A10		Reserved	0x50D0_0A10		Reserved
0x5090_0A0F		Global Control and Status	0x50D0_0A0F		Global Control and Status
0x5090_0994		Reserved	0x50D0_0994		Reserved
0x5090_0993		Global Control and Status	0x50D0_0993		Global Control and Status
0x5090_0990		Reserved	0x50D0_0990		Reserved
0x5090_098F	Global Control and Status	0x50D0_098F	Global Control and Status		
0x5090_0930	Reserved	0x50D0_0930	Reserved		
0x5090_092F	Per Stream Control and Status	0x50D0_092F	Per Stream Control and Status		
0x5090_0810	Per Layer Control and Status	0x50D0_0810	Per Layer Control and Status		
0x5090_080F	Global Control and Status	0x50D0_080F	Global Control and Status		
0x5090_0010	Global Control and Status	0x50D0_0010	Global Control and Status		
0x5090_000F	Global Control and Status	0x50D0_000F	Global Control and Status		
0x5090_0000	Global Control and Status	0x50D0_0000	Global Control and Status		

0x5041_FFFF	APB Access CNNx16_0 Quadrant 0	128KB SRAM	0x5081_FFFF	APB Access CNNx16_0 Quadrant 1	128KB SRAM
0x5040_0000		Reserved	0x5080_0000		Reserved
0x503F_FFFF		110KB Mask RAM 16 Blocks of 16KB (256KB) allocated space. Only the lower 6,912B of each block has Mask RAM.	0x507F_FFFF		110KB Mask RAM 16 Blocks of 16KB (256KB) allocated space. Only the lower 6,912B of each block has Mask RAM.
0x501C_0000		Reserved	0x505C_0000		Reserved
0x501B_FFFF		98KB TRAM 16 Blocks of 16KB (256KB) allocated space. Only the lower 6,144B of each block has TRAM.	0x505B_FFFF		98KB TRAM 16 Blocks of 16KB (256KB) allocated space. Only the lower 6,144B of each block has TRAM.
0x5018_0000		512B Bias RAM 32KB allocated space	0x5058_0000		512B Bias RAM 32KB allocated space
0x5017_FFFF		Reserved	0x5057_FFFF		Reserved
0x5015_0000		Reserved	0x5055_0000		Reserved
0x5014_FFFF		Global Control and Status	0x5054_FFFF		Global Control and Status
0x5011_0000		Reserved	0x5051_0000		Reserved
0x5010_9FFF		Per Layer Control and Status	0x5050_9FFF		Per Layer Control and Status
0x5010_8000		Reserved	0x5050_8000		Reserved
0x5010_7FFF		Global Control and Status	0x5050_7FFF		Global Control and Status
0x5010_5000		Reserved	0x5050_5000		Reserved
0x5010_4FFF		Global Control and Status	0x5050_4FFF		Global Control and Status
0x5010_1000		Reserved	0x5050_1000		Reserved
0x5010_0FFF		Per Layer Control and Status	0x5050_0FFF		Per Layer Control and Status
0x5010_0A90		Reserved	0x5050_0A90		Reserved
0x5010_0A8F		Global Control and Status	0x5050_0A8F		Global Control and Status
0x5010_0A10		Reserved	0x5050_0A10		Reserved
0x5010_0A0F		Global Control and Status	0x5050_0A0F		Global Control and Status
0x5010_0994		Reserved	0x5050_0994		Reserved
0x5010_0993		Global Control and Status	0x5050_0993		Global Control and Status
0x5010_0990		Reserved	0x5050_0990		Reserved
0x5010_098F	Global Control and Status	0x5050_098F	Global Control and Status		
0x5010_0930	Reserved	0x5050_0930	Reserved		
0x5010_092F	Per Stream Control and Status	0x5050_092F	Per Stream Control and Status		
0x5010_0810	Per Layer Control and Status	0x5050_0810	Per Layer Control and Status		
0x5010_080F	Global Control and Status	0x5050_080F	Global Control and Status		
0x5010_0010	Global Control and Status	0x5050_0010	Global Control and Status		
0x5010_000F	Global Control and Status	0x5050_000F	Global Control and Status		
0x5010_0000	Global Control and Status	0x5050_0000	Global Control and Status		

0x5000_1000	AON and FIFO Control	AON Control
0x5000_0FFF		Reserved
0x5000_0017		Quadrant FIFO Write Data
0x5000_0018		Quadrant FIFO Control and Status
0x5000_0008		
0x5000_0007		
0x5000_0000		

27.3.1 CNNx16_n TRAM Details

Each CNNx16 processor array includes 98KB of Tornado RAM (TRAM); however, the memory allocated to this region spans 256KB addressable memory. The TRAM is arranged as 16 blocks of 16KB of addressable memory space; however, only 6,144B of each of these 16 blocks contains TRAM. The TRAM is organized as 3,072×16-bits and due to memory alignment it consumes 12KB space within the allocated 16KB.

Table 27-1, Table 27-2, Table 27-3, and Table 27-4 show each of the four CNNx16 Processor Array's TRAM start address and the valid TRAM end address for the 16 TRAM blocks. Memory addresses between the *Valid TRAM Block End Address* and the *Block End Address* are invalid memory addresses and are not used.

Table 27-1: CNNx16 Processor Array 0 TRAM Mapping Details (APB Accessible)

CNN Quadrant#	TRAM Block	Size	Allocated Size	Start Address	Valid TRAM Block End Address	Block End Address
CNNx16_0	0	6,144B	16KB	0x5011 0000	0x5011 2FFF	0x5011 3FFF
	1	6,144B	16KB	0x5011 4000	0x5011 6FFF	0x5011 7FFF
	2	6,144B	16KB	0x5011 8000	0x5011 AFFF	0x5011 BFFF
	3	6,144B	16KB	0x5011 C000	0x5011 EFFF	0x5011 FFFF
	4	6,144B	16KB	0x5012 0000	0x5012 2FFF	0x5012 3FFF
	5	6,144B	16KB	0x5012 4000	0x5012 6FFF	0x5012 7FFF
	6	6,144B	16KB	0x5012 8000	0x5012 AFFF	0x5012 BFFF
	7	6,144B	16KB	0x5012 C000	0x5012 EFFF	0x5012 FFFF
	8	6,144B	16KB	0x5013 0000	0x5013 2FFF	0x5013 3FFF
	9	6,144B	16KB	0x5013 4000	0x5013 6FFF	0x5013 7FFF
	10	6,144B	16KB	0x5013 8000	0x5013 AFFF	0x5013 BFFF
	11	6,144B	16KB	0x5013 C000	0x5013 EFFF	0x5013 FFFF
	12	6,144B	16KB	0x5014 0000	0x5014 2FFF	0x5014 3FFF
	13	6,144B	16KB	0x5014 4000	0x5014 6FFF	0x5014 7FFF
	14	6,144B	16KB	0x5014 8000	0x5014 AFFF	0x5014 BFFF
	15	6,144B	16KB	0x5014 C000	0x5014 EFFF	0x5014 FFFF

Table 27-2: CNNx16 Processor Array 1 TRAM Mapping Details (APB Accessible)

CNN Quadrant#	TRAM Block	Size	Allocated Size	Start Address	Valid TRAM Block End Address	Block End Address
CNNx16_1	0	6,144B	16KB	0x5051 0000	0x5051 2FFF	0x5051 3FFF
	1	6,144B	16KB	0x5051 4000	0x5051 6FFF	0x5051 7FFF
	2	6,144B	16KB	0x5051 8000	0x5051 AFFF	0x5051 BFFF
	3	6,144B	16KB	0x5051 C000	0x5051 EFFF	0x5051 FFFF
	4	6,144B	16KB	0x5052 0000	0x5052 2FFF	0x5052 3FFF
	5	6,144B	16KB	0x5052 4000	0x5052 6FFF	0x5052 7FFF
	6	6,144B	16KB	0x5052 8000	0x5052 AFFF	0x5052 BFFF
	7	6,144B	16KB	0x5052 C000	0x5052 EFFF	0x5052 FFFF
	8	6,144B	16KB	0x5053 0000	0x5053 2FFF	0x5053 3FFF
	9	6,144B	16KB	0x5053 4000	0x5053 6FFF	0x5053 7FFF
	10	6,144B	16KB	0x5053 8000	0x5053 AFFF	0x5053 BFFF

CNN Quadrant#	TRAM Block	Size	Allocated Size	Start Address	Valid TRAM Block End Address	Block End Address
	11	6,144B	16KB	0x5053 C000	0x5053 EFFF	0x5053 FFFF
	12	6,144B	16KB	0x5054 0000	0x5054 2FFF	0x5054 3FFF
	13	6,144B	16KB	0x5054 4000	0x5054 6FFF	0x5054 7FFF
	14	6,144B	16KB	0x5054 8000	0x5054 AFFF	0x5054 BFFF
	15	6,144B	16KB	0x5054 C000	0x5054 EFFF	0x5054 FFFF

Table 27-3: CNNx16 Processor Array 2 TRAM Mapping Details (APB Accessible)

CNN Quadrant#	TRAM Block	Size	Allocated Size	Start Address	Valid TRAM Block End Address	Block End Address
CNNx16_2	0	6,144B	16KB	0x5091 0000	0x5091 2FFF	0x5091 3FFF
	1	6,144B	16KB	0x5091 4000	0x5091 6FFF	0x5091 7FFF
	2	6,144B	16KB	0x5091 8000	0x5091 AFFF	0x5091 BFFF
	3	6,144B	16KB	0x5091 C000	0x5091 EFFF	0x5091 FFFF
	4	6,144B	16KB	0x5092 0000	0x5092 2FFF	0x5092 3FFF
	5	6,144B	16KB	0x5092 4000	0x5092 6FFF	0x5092 6FFF
	6	6,144B	16KB	0x5092 8000	0x5092 AFFF	0x5092 BFFF
	7	6,144B	16KB	0x5092 C000	0x5092 EFFF	0x5092 FFFF
	8	6,144B	16KB	0x5093 0000	0x5093 2FFF	0x5093 3FFF
	9	6,144B	16KB	0x5093 4000	0x5093 6FFF	0x5093 7FFF
	10	6,144B	16KB	0x5093 8000	0x5093 AFFF	0x5093 BFFF
	11	6,144B	16KB	0x5093 C000	0x5093 EFFF	0x5093 FFFF
	12	6,144B	16KB	0x5094 0000	0x5094 2FFF	0x5094 3FFF
	13	6,144B	16KB	0x5094 4000	0x5094 6FFF	0x5094 7FFF
	14	6,144B	16KB	0x5094 8000	0x5094 AFFF	0x5094 BFFF
	15	6,144B	16KB	0x5094 C000	0x5094 EFFF	0x5094 FFFF

Table 27-4: CNNx16 Processor Array 3 TRAM Mapping Details (APB Accessible)

CNN Quadrant#	TRAM Block	Size	Allocated Size	Start Address	Valid TRAM Block End Address	Block End Address
CNNx16_3	0	6,144B	16KB	0x50D1 0000	0x50D1 2FFF	0x50D1 3FFF
	1	6,144B	16KB	0x50D1 4000	0x50D1 6FFF	0x50D1 7FFF
	2	6,144B	16KB	0x50D1 8000	0x50D1 AFFF	0x50D1 BFFF
	3	6,144B	16KB	0x50D1 C000	0x50D1 EFFF	0x50D1 FFFF
	4	6,144B	16KB	0x50D2 0000	0x50D2 2FFF	0x50D2 3FFF
	5	6,144B	16KB	0x50D2 4000	0x50D2 6FFF	0x50D2 7FFF
	6	6,144B	16KB	0x50D2 8000	0x50D2 AFFF	0x50D2 BFFF
	7	6,144B	16KB	0x50D2 C000	0x50D2 EFFF	0x50D2 FFFF
	8	6,144B	16KB	0x50D3 0000	0x50D3 2FFF	0x50D3 3FFF
	9	6,144B	16KB	0x50D3 4000	0x50D3 6FFF	0x50D3 7FFF
	10	6,144B	16KB	0x50D3 8000	0x50D3 AFFF	0x50D3 BFFF
	11	6,144B	16KB	0x50D3 C000	0x50D3 EFFF	0x50D3 FFFF

CNN Quadrant#	TRAM Block	Size	Allocated Size	Start Address	Valid TRAM Block End Address	Block End Address
	12	6,144B	16KB	0x50D4 0000	0x50D4 2FFF	0x50D4 3FFF
	13	6,144B	16KB	0x50D4 4000	0x50D4 6FFF	0x50D4 7FFF
	14	6,144B	16KB	0x50D4 8000	0x50D4 AFFF	0x50D4 BFFF
	15	6,144B	16KB	0x50D4 C000	0x50D4 EFFF	0x50D4 FFFF

27.3.2 CNNx16_n MRAM Details

Each CNNx16 processor array includes 110KB of MRAM; however, the memory allocated to this region spans 256KB address space. The MRAM is arranged as 16 blocks of 16KB of addressable memory space; however, only the first 6,912B of each of these 16 blocks contains usable MRAM. Each MRAM instance is organized as 768×72-bits. The MRAM uses 128-bits of address space for each 72-bits of MRAM. Due to memory alignment it consumes 12KB space within the allocated 16KB.

The following tables below ([Table 27-5](#), [Table 27-6](#), [Table 27-7](#), and [Table 27-8](#)) show each of the four CNNx16 Processor Array's MRAM start address and the valid MRAM end address for the 16 MRAM blocks. Memory addresses between the *Valid MRAM Block End Address* and the *Block End Address* are invalid memory addresses and are not used.

Table 27-5: CNNx16 Processor Array 0 MRAM Mapping Details (APB Accessible)

CNN Quadrant#	MRAM Block	Size	Allocated Size	MRAM Block Start Address	Valid MRAM Block End Address	Block End Address
CNNx16_0	0	6,912B	16KB	0x5018 0000	0x5018 2FFF	0x5018 3FFF
	1	6,912B	16KB	0x5018 4000	0x5018 6FFF	0x5018 7FFF
	2	6,912B	16KB	0x5018 8000	0x5018 AFFF	0x5018 BFFF
	3	6,912B	16KB	0x5018 C000	0x5018 EFFF	0x5018 FFFF
	4	6,912B	16KB	0x5019 0000	0x5019 2FFF	0x5019 3FFF
	5	6,912B	16KB	0x5019 4000	0x5019 6FFF	0x5019 7FFF
	6	6,912B	16KB	0x5019 8000	0x5019 AFFF	0x5019 BFFF
	7	6,912B	16KB	0x5019 C000	0x5019 EFFF	0x5019 FFFF
	8	6,912B	16KB	0x501A 0000	0x501A 2FFF	0x501A 3FFF
	9	6,912B	16KB	0x501A 4000	0x501A 6FFF	0x501A 7FFF
	10	6,912B	16KB	0x501A 8000	0x501A AFFF	0x501A BFFF
	11	6,912B	16KB	0x501A C000	0x501A EFFF	0x501A FFFF
	12	6,912B	16KB	0x501B 0000	0x501B 2FFF	0x501B 3FFF
	13	6,912B	16KB	0x501B 4000	0x501B 6FFF	0x501B 7FFF
	14	6,912B	16KB	0x501B 8000	0x501B AFFF	0x501B BFFF
	15	6,912B	16KB	0x501B C000	0x501B EFFF	0x501B FFFF

Table 27-6: CNNx16 Processor Array 1 MRAM Mapping Details (APB Accessible)

CNN Quadrant#	MRAM Block	Size	Allocated Size	MRAM Block Start Address	Valid MRAM Block End Address	Block End Address
CNNx16_1	0	6,912B	16KB	0x5058 0000	0x5058 2FFF	0x5058 3FFF
	1	6,912B	16KB	0x5058 4000	0x5058 6FFF	0x5058 7FFF
	2	6,912B	16KB	0x5058 8000	0x5058 AFFF	0x5058 BFFF
	3	6,912B	16KB	0x5058 C000	0x5058 EFFF	0x5058 FFFF

CNN Quadrant#	MRAM Block	Size	Allocated Size	MRAM Block Start Address	Valid MRAM Block End Address	Block End Address
	4	6,912B	16KB	0x5059 0000	0x5059 2FFF	0x5059 3FFF
	5	6,912B	16KB	0x5059 4000	0x5059 6FFF	0x5059 7FFF
	6	6,912B	16KB	0x5059 8000	0x5059 AFFF	0x5059 BFFF
	7	6,912B	16KB	0x5059 C000	0x5059 EFFF	0x5059 FFFF
	8	6,912B	16KB	0x505A 0000	0x505A 2FFF	0x505A 3FFF
	9	6,912B	16KB	0x505A 4000	0x505A 6FFF	0x505A 7FFF
	10	6,912B	16KB	0x505A 8000	0x505A AFFF	0x505A BFFF
	11	6,912B	16KB	0x505A C000	0x505A EFFF	0x505A FFFF
	12	6,912B	16KB	0x505B 0000	0x505B 2FFF	0x505B 3FFF
	13	6,912B	16KB	0x505B 4000	0x505B 6FFF	0x505B 7FFF
	14	6,912B	16KB	0x505B 8000	0x505B AFFF	0x505B BFFF
	15	6,912B	16KB	0x505B C000	0x505B EFFF	0x505B FFFF

Table 27-7: CNNx16 Processor Array 2 MRAM Mapping Details (APB Accessible)

CNN Quadrant#	MRAM Block	Size	Allocated Size	Start Address	Valid MRAM Block End Address	Block End Address
CNNx16_2	0	6,912B	16KB	0x5098 0000	0x5098 2FFF	0x5098 3FFF
	1	6,912B	16KB	0x5098 4000	0x5098 6FFF	0x5098 7FFF
	2	6,912B	16KB	0x5098 8000	0x5098 AFFF	0x5098 CFFF
	3	6,912B	16KB	0x5098 C000	0x5098 EFFF	0x5098 FFFF
	4	6,912B	16KB	0x5099 0000	0x5099 2FFF	0x5099 3FFF
	5	6,912B	16KB	0x5099 4000	0x5099 6FFF	0x5099 7FFF
	6	6,912B	16KB	0x5099 8000	0x5099 AFFF	0x5099 BFFF
	7	6,912B	16KB	0x5099 C000	0x5099 EFFF	0x5099 FFFF
	8	6,912B	16KB	0x509A 0000	0x509A 2FFF	0x509A 3FFF
	9	6,912B	16KB	0x509A 4000	0x509A 6FFF	0x509A 7FFF
	10	6,912B	16KB	0x509A 8000	0x509A AFFF	0x509A BFFF
	11	6,912B	16KB	0x509A C000	0x509A 3FFF	0x509A FFFF
	12	6,912B	16KB	0x509B 0000	0x509B 2FFF	0x509B 3FFF
	13	6,912B	16KB	0x509B 4000	0x509B 6FFF	0x509B 7FFF
	14	6,912B	16KB	0x509B 8000	0x509B AFFF	0x509B BFFF
	15	6,912B	16KB	0x509B C000	0x509B EFFF	0x509B FFFF

Table 27-8: CNNx16 Processor Array3 MRAM Mapping Details (APB Accessible)

CNN Quadrant#	MRAM Block	Size	Allocated Size	Memory Start Address	Valid MRAM Block End Address	Block End Address
CNNx16_3	0	6,912B	16KB	0x50D8 0000	0x50D8 2FFF	0x50D8 3FFF
	1	6,912B	16KB	0x50D8 4000	0x50D8 6FFF	0x50D8 7FFF
	2	6,912B	16KB	0x50D8 8000	0x50D8 AFFF	0x50D8 CFFF
	3	6,912B	16KB	0x50D8 C000	0x50D8 EFFF	0x50D8 FFFF
	4	6,912B	16KB	0x50D9 0000	0x50D9 2FFF	0x50D9 3FFF

CNN Quadrant#	MRAM Block	Size	Allocated Size	Memory Start Address	Valid MRAM Block End Address	Block End Address
	5	6,912B	16KB	0x50D9 4000	0x50D9 6FFF	0x50D9 7FFF
	6	6,912B	16KB	0x50D9 8000	0x50D9 AFFF	0x50D9 BFFF
	7	6,912B	16KB	0x50D9 C000	0x50D9 EFFF	0x50D9 FFFF
	8	6,912B	16KB	0x50DA 0000	0x50DA 2FFF	0x50DA 3FFF
	9	6,912B	16KB	0x50DA 4000	0x50DA 6FFF	0x50DA 7FFF
	10	6,912B	16KB	0x50DA 8000	0x50DA AFFF	0x50DA BFFF
	11	6,912B	16KB	0x50DA C000	0x50DA EFFF	0x50DA FFFF
	12	6,912B	16KB	0x50DB 0000	0x50DB 2FFF	0x50DB 3FFF
	13	6,912B	16KB	0x50DB 4000	0x50DB 6FFF	0x50DB 7FFF
	14	6,912B	16KB	0x50DB 8000	0x50DB AFFF	0x50DB BFFF
	15	6,912B	16KB	0x50DB C000	0x50DB EFFF	0x50DB FFFF

27.4 CNN Global Registers (CNN)

See [Table 3-3](#) for the base address of this peripheral/module. There is one instance of the CNN in the MAX78000. The global CNN registers are shown in [Table 27-9](#). See [Table 1-1](#) for an explanation of the read and write access of each field. Unless specified otherwise, all fields are reset on a system reset, soft reset, POR, and the peripheral-specific resets.

Table 27-9: Global CNN Register Summary

Offset	Register	Description
[0x0000]	CNN_FIFO_CTRL	CNN FIFO Control Register
[0x0004]	CNN_FIFO_STAT	CNN FIFO Status Register
[0x0008]	CNN_FIFO_WRO	CNN FIFO 0 Write Register
[0x000C]	CNN_FIFO_WR1	CNN FIFO 1 Write Register
[0x0010]	CNN_FIFO_WR2	CNN Fast FIFO 2 Write Register
[0x0014]	CNN_FIFO_WR3	CNN FIFO 3 Write Register
[0x1000]	CNN_AOD_CTRL	CNN AoD Control Register

27.4.1 Global CNN Register Details

Table 27-10: CNN FIFO Control Register

CNN FIFO Control			CNN_FIFO_CTRL		[0x0000]
Bits	Field	Access	Reset	Description	
31:28	almost_empty_int_en	R/W	0	FIFO Almost Empty Interrupt Enable Each bit of this field maps to one of the CNNx16_n quadrants. Bit 3 maps to CNNx16_n_q3, bit 2 maps to CNNx16_n_q2, bit 1 maps to CNNx16_n_q1, and bit 0 maps to CNNx16_n_q0. Set this field to 1 to enable an interrupt event based on the CNN_FIFO_CTRL.empty_thresh flag. Enable the interrupt for all quadrants by setting this field to 0b1111.	
27:24	almost_full_int_en	R/W	0	FIFO Almost Full Interrupt Enable Each bit of this field maps to one of the CNNx16_n quadrants. Bit 3 maps to CNNx16_n_q3, bit 2 maps to CNNx16_n_q2, bit 1 maps to CNNx16_n_q1, and bit 0 maps to CNNx16_n_q0. Set this field to 1 to enable an interrupt event based on the FIFO almost full threshold flag, CNN_FIFO_CTRL.full_thresh . Enable the interrupt for all quadrants by setting this field to 0b1111.	

CNN FIFO Control			CNN_FIFO_CTRL		[0x0000]
Bits	Field	Access	Reset	Description	
23:20	empty_int_en	R/W	0	FIFO Empty Interrupt Enable Each bit of this field maps to one of the CNNx16_n quadrants. Bit 3 maps to CNNx16_n_q3, bit 2 maps to CNNx16_n_q2, bit 1 maps to CNNx16_n_q1, and bit 0 maps to CNNx16_n_q0. Set this field to 1 to enable an interrupt event based on the FIFO empty flag. Enable the interrupt for all quadrants by setting this field to 0b1111	
19:16	full_int_en	R/W	0	FIFO Full Interrupt Enable Each bit of this field maps to one of the CNNx16_n quadrants. Bit 3 maps to CNNx16_n_q3, bit 2 maps to CNNx16_n_q2, bit 1 maps to CNNx16_n_q1, and bit 0 maps to CNNx16_n_q0. Set this field to 1 to enable an interrupt event based on the FIFO full flag. Enable the interrupt for all quadrants by setting this field to 0b1111.	
15:12	fifo_en	R/W	0	Per FIFO Enable Set this field to 1 to enable the corresponding FIFO for the specified CNNx16_n quadrant. Each bit of this field maps to one of the CNNx16_n quadrants. Bit 3 maps to CNNx16_n_q3, Bit 2 maps to CNNx16_n_q2, Bit1 maps to CNNx16_n_q1 and Bit 0 maps to CNNx16_n_q0. <i>Note: Unused FIFOs must be disabled.</i>	
11	fifo_cpl	R/W	0	FIFO Coupling Setting this bit to 1 forces the FIFOs to operate in lockstep. Data available status is dependent on all FIFOs having identical write pointer values. 0: FIFOs are not coupled 1: FIFOs are coupled and operate in lockstep	
10	-	RO	0	Reserved	
9:7	empty_thresh	R/W	0	FIFO Almost Empty Threshold If the difference between the write and read pointer (read_pointer - write_pointer) falls below this number of bytes, the almost empty flag is set. 0 to 7: Sets the FIFO Almost Empty Threshold to the value written.	
6:5	-	RO	0	Reserved	
4:2	full_thresh	R/W	0	FIFO Almost Full Threshold This flag is set if the difference between the write and read pointer (read_pointer - write_pointer) exceeds this number of bytes, the almost full flag is set. 0 to 7: Sets the FIFO Almost Full Threshold to the value written.	
1:0	rdy_sel	R/W	b11	APB Wait State Selection This field determines the number of wait states added during an APB access. 0b00: 0 wait states 0b01: 1 wait state 0b10: 2 wait states 0b11: 3 wait states <i>Note: Write operations load the data one clock before the end of the cycle.</i>	

Table 27-11: CNN FIFO Status Register

CNN FIFO Status			CNN_FIFO_STAT		[0x0004]
Bits	Field	Access	Reset	Description	
31:22	-	RO	0	Reserved	

CNN FIFO Status			CNN_FIFO_STAT		[0x0004]
Bits	Field	Access	Reset	Description	
21	rp_ptr_eq	R	0	FIFO Read Pointers Equal This bit is set when all active FIFO read pointers are equal	
20	wptr_eq	R	0	FIFO Write Pointers Equal This bit is set when all active FIFO write pointers are equal	
19	fifos_almost_empty	R	0	Aggregate FIFO Almost Empty Status This field is a logical OR of individual FIFO almost empty statuses.	
18	fifos_almost_full	R	0	Aggregate FIFO Almost Full Status This field is a logical AND of individual FIFO almost full statuses.	
17	fifos_empty	R	0	Aggregate FIFO Empty Status This field is a logical OR of individual FIFO empty statuses.	
16	fifos_full	R	0	Aggregate FIFO Full Status This field is a logical AND of individual FIFO full statuses.	
15:12	almost_empty	R	0	Per FIFO Almost Empty Status If a bit in this field reads 1, the corresponding FIFO is almost empty. Each bit corresponds to a FIFO with almost_empty[0] mapped to CNNx16_n FIFO0. This status is persistent until the condition changes or until software disables the FIFO.	
11:8	almost_full	R	0	Per FIFO Almost Full Status If a bit in this field reads 1, the corresponding FIFO is almost full. Each bit corresponds to a FIFO with almost_full[0] mapped to CNNx16_n FIFO0. This status is persistent until the condition changes or until software disables the FIFO.	
7:4	empty	R	0	Per FIFO Empty Status If a bit in this field reads 1, the corresponding FIFO is empty. Each bit corresponds to a FIFO with empty[0] mapped to CNNx16_n FIFO0. This status is persistent until the condition changes or until software disables the FIFO.	
3:0	full	R	0	Per FIFO Full Status If a bit in this field reads 1, the corresponding FIFO is full. Each bit corresponds to a FIFO with full[0] mapped to CNNx16_n FIFO0. The status is persistent until the condition changes or until software disables the FIFO.	

Table 27-12: CNN FIFO 0 Write Register

CNN FIFO 0 Write			CNN_FIFO_WR0		[0x0008]
Bits	Field	Access	Reset	Description	
31:0	data	R/W	0	FIFO 0 Data CNNx16_n FIFO 0 data register.	

Table 27-13: CNN FIFO 1 Write Register

CNN FIFO 1 Write			CNN_FIFO_WR1		[0x000C]
Bits	Field	Access	Reset	Description	
31:0	data	R/W	0	FIFO 1 Data CNNx16_n FIFO 2 data register.	

Table 27-14: CNN FIFO 2 Write Register

CNN FIFO 2 Write			CNN_FIFO_WR2		[0x0010]
Bits	Field	Access	Reset	Description	
31:0	data	R/W	0	FIFO 2 Data CNNx16_n FIFO 2 data register.	

Table 27-15: CNN FIFO 3 Write Register

CNN FIFO 3 Write			CNN_FIFO_WR3		[0x0014]
Bits	Field	Access	Reset	Description	
31:0	data	R/W	0	FIFO 3 Data CNNx16_n FIFO 3 data register.	

Table 27-16: CNN Always On Domain Control Register

CNN Always On Domain Control			CNN_AOD_CTRL		[0x1000]
Bits	Field	Access	Reset	Description	
31:20	-	RO	0	Reserved	
19:16	rm	R/W	0	MRAM Read Margin MSB Each bit of this field maps to one of the four CNNx16_n quadrants. Bit 0 maps to CNNx16_n_q0, Bit 1 maps to CNNx16_n_q1, bit 2 maps to CNNx16_n_q2, and bit 3 maps to CNNx16_n_q3.	
15:12	pd	R/W	0	MRAM Power Down Enable Each bit of this field maps to a CNNx16_n quadrant. Setting the quadrant's bit position to 1 shuts down power to the MRAM in the CNNx16_n quadrant. Write 0b1111 to this field to power down all CNNx16_n quadrants' MRAM. When a CNNx16_n quadrant's MRAM is put into power down mode, the contents of the MRAM are <i>not</i> maintained. 0: In any bit position, the corresponding CNNx16_n's MRAM is not in a power down state. 1: In any bit position, the corresponding CNNx16_n's MRAM is in a power down state. <i>Note: This field is independent of the Low Power Mode settings. See Operating Modes for more information on the system's low-power modes and their effect on the CNN and CNNx16_n quadrant memories.</i>	
11:8	dsleep	R/W	0	MRAM Deep Sleep Enable Each bit of this field maps to a CNNx16_n quadrant. Setting the quadrant's bit position to 1 puts the specified CNNx16_n's MRAM in a deep sleep state. Write 0b1111 to this field to put all four of the CNNx16_n quadrant's MRAM in deep sleep mode. When a CNNx16_n quadrant's MRAM is in deep sleep, the contents of the MRAM are retained, but the memory cannot be accessed. 0: In any bit position, the corresponding CNNx16_n's MRAM is not in deep sleep. 1: In any bit position, the corresponding CNNx16_n's MRAM is in deep sleep. <i>Note: This field is independent of the Low Power Mode settings. See Operating Modes for more information on the system's low-power modes and their effect on the CNN and CNNx16_n quadrant memories.</i>	
7:2	-	RO	0	Reserved	

CNN Always On Domain Control			CNN_AOD_CTRL		[0x1000]
Bits	Field	Access	Reset	Description	
1:0	rdy_sel	R/W	b11	APB Wait State Selection This field determines the number of wait states added during an APB access. 0: 0 wait states 1: 1 wait state 2: 2 wait states 3: 3 wait states <i>Note: Write operations load the data one clock before the end of the cycle.</i>	

27.5 CNNx16 Processor Array (CNNx16_n) Registers

The CNN includes four ×16 Processor Arrays, referred to as CNNx16_n. Each processor array includes a dedicated FIFO interface allowing configuration, status, and write access to each of the four CNNx16_n's individual memory spaces. The table below shows the Base Address for each of the four CNNx16_n processor arrays.

Table 27-17: CNNx16_n Instances and Base Offset Address

Base Offset Address	Processor Array	Description
[0x0010 0000]	CNNx16_0	CNN ×16 Processor Array 0
[0x0050 0000]	CNNx16_1	CNN ×16 Processor Array 1
[0x0090 0000]	CNNx16_2	CNN ×16 Processor Array 2
[0x00D0 0000]	CNNx16_3	CNN ×16 Processor Array 3

27.5.1 CNNx16_n Instances and Base Offset Addresses

Table 27-17 shows the base address for each of the four CNNx16 processor arrays. Each CNNx16 processor array has its own independent set of registers shown in Table 27-18. The base address for a specific CNNx16_n processor array is calculated by adding the CNNx16_n base offset address to the global CNN base address shown in Table 3-3. Register names for a specific instance are defined by replacing "n" with the instance number. As an example, a register PERIPHERALn_CTRL resolves to PERIPHERAL0_CTRL and PERIPHERAL1_CTRL for instances 0 and 1, respectively.

See Table 1-1 for an explanation of the read and write access of each field. Unless specified otherwise, all fields are reset on a system reset, soft reset, POR, and the peripheral-specific resets.

Table 27-18: CNNx16_n Processor Array Registers

Offset	Register	Description
[0x0000]	CNNx16_n_CTRL	CNNx16_n Control Register
[0x0004]	CNNx16_n_SRAM	CNNx16_n SRAM Control Register
[0x0008]	CNNx16_n_LCNT_MAX	CNNx16_n Layer Count Maximum Register
[0x000C]	CNNx16_n_TEST	CNNx16_n SRAM Test Register
[0x0990]	CNNx16_n_IFRM	CNNx16_n Input FIFO Frame Size Register
[0x1000]	CNNx16_n_MLAT	CNNx16_n Mlator Data Register
[0x0010]-[0x008C]	CNNx16_n_Ly_RCNT	CNNx16_n_Ly Row Count Register
[0x0090]-[0x010C]	CNNx16_n_Ly_CCNT	CNNx16_n_Ly Column Count Register
[0x0110]-[0x018C]	CNNx16_n_Ly_ONED	CNNx16_n_Ly One Dimensional Control Register
[0x0190]-[0x020C]	CNNx16_n_Ly_PRCNT	CNNx16_n_Ly Pool Row Count Register
[0x0210]-[0x028C]	CNNx16_n_Ly_PCCNT	CNNx16_n_Ly Pool Column Count Register
[0x0290]-[0x030C]	CNNx16_n_Ly_STRIDE	CNNx16_n_Ly Stride Count Register

Offset	Register	Description
[0x0310]-[0x038C]	CNNx16_n_Ly_WPTR_BASE	CNNx16_n_Ly Write Pointer Base Address Register
[0x0390]-[0x040C]	CNNx16_n_Ly_WPTR_TOFF	CNNx16_n_Ly Write Pointer Timeslot Offset Register
[0x0410]-[0x048C]	CNNx16_n_Ly_WPTR_MOFF	CNNx16_n_Ly Write Pointer Mask Offset Register
[0x0490]-[0x050C]	CNNx16_n_Ly_WPTR_CHOFF	CNNx16_n_Ly Write Pointer Multi-Pass Channel Offset Register
[0x0510]-[0x058C]	CNNx16_n_Ly_RPTR_BASE	CNNx16_n_Ly Read Pointer Base Address Register
[0x0590]-[0x060C]	CNNx16_n_Ly_LCTRL0	CNNx16_n_Ly Layer Control 0 Register
[0x0610]-[0x068C]	CNNx16_n_Ly_MCNT	CNNx16_n_Ly Mask Count Register
[0x0690]-[0x070C]	CNNx16_n_Ly_TPTR	CNNx16_n_Ly TRAM Pointer Register
[0x0710]-[0x078C]	CNNx16_n_Ly_EN	CNNx16_n_Ly Enable Register
[0x0790]-[0x080F]	CNNx16_n_Ly_POST	CNNx16_n_Ly Post Processing Register
[0x0A10]-[0x0A8C]	CNNx16_n_Ly_LCTRL1	CNNx16_n_Ly Layer Control 1 Register
[0x0810]-[0x082C]	CNNx16_n_Sz_STRM0	CNNx16_n_Sz Stream Control 0 Register
[0x0890]-[0x08AC]	CNNx16_n_Sz_STRM1	CNNx16_n_Sz Stream Control 1 Register
[0x0910]-[0x092C]	CNNx16_n_Sz_FBUF	CNNx16_n_Sz Stream Frame Buffer Size Register

27.5.2 CNN Per x16 Processor Register Details

Table 27-19: CNNx16_n Control Register

CNNx16_n Control				CNNx16_n_CTRL	[0x0000]
Bits	Field	Access	Reset	Description	
31	qupac	R/W	0	QuPac Mode The quad processors pack bit enables parallel processing of the same data by each of the 16 processors in the CNNx16_n. 0: Normal processing mode 1: x16 parallel processing mode <i>Note: FIFO mode must be enabled, CNNx16_n_CTRL.ffifo_en set to 1, prior to enabling QuPac mode.</i>	
30	timeshft	R/W	0	Pooling Stage Time Shift When set, one wait state is added to the pooling stage to allow design time closure at a higher clock frequency.	
29:24	fclk_dly	R/W	0	FIFO Clock Delay This field selects the clock delay of the FIFO clock relative to the primary CNN clock. A setting of 0 adds in the minimum delay, and a value of 0x3F adds the maximum delay.	
23	fifogrp	R/W	0	FIFO Group Output Enables sending all "little data" channels to the first 4 processors. When this bit is not set, each byte of FIFO data is directed to the first little data channel of each CNNx16_n processor.	
22	ffifo_en	R/W	0	Fast FIFO Enable This field enables the tightly coupled data path between the MAX78000's CNN_TX fast FIFO and the CNN data SRAMs. The CNN_TX_FIFO is 32 bits wide, with 8 bits being dedicated to each of 4 channels. Channel routing is controlled by the state of the CNNx16_n_CTRL.fifogrp control bit.	

CNNx16_n Control				CNNx16_n_CTRL	[0x0000]
Bits	Field	Access	Reset	Description	
21	simple1b	R/W	0	Simple 1-Bit Weights Enable simple logic for 1-bit weights. Setting this bit disables the wide accumulators used to calculate the convolution products and activates simple one-bit logic functions.	
20	mexpress	R/W	0	Mask Memory Packed Memory Enable Enable loading of the mask memories using packed data. With this bit set, a change in the state of the two least significant bits of the MRAM address triggers a reload of the address counter.	
19	lilbuf	R/W	0	Stream Mode Circular Buffer Enable Setting this bit restricts the associated read buffer's bounds to a circular buffer starting at the CNNx16_n_Ly_RPTR_BASE address and terminating at the CNNx16_n_Sz_FBUF address. When set, the circular buffer is used on all streaming layers.	
18:17	mlatch_sel	R/W	0	SRAM Packed Channel Select Selects which of the four channels in the SRAM read data is packed. 0: Selects channel 0, SRAM data bits 7:0 1: Selects channel 1 SRAM data bits 15:8 2: Selects channel 2 SRAM data bits 23:16 3: Select channel 3 SRAM data bits 31:24	
16	mlat_ld	R/W	0	Mlator Load Data Writing the bit from a 0 to a 1 forces the CNNx16_n_Ly_WPTR_BASE address to be loaded into the Mlator address counter and selects the counter as the SRAM address source. SRAM reads are only possible when CNNx16_n_CTRL.cnn_en is reset to 0.	
15	fifo_en	R/W	0	CNNx16_n_FIFO Enable When set, data for the first (input) layer is taken from the CNN_FIFO_WRN FIFO register. One 4 byte-wide FIFO is dedicated for each of the four processor arrays. The FIFOs are accessed through the APB memory map, and each can be used in a single byte-wide channel mode, a single 4 byte-wide channel mode, or 4 single byte-wide channel mode. The mode is determined by the data configuration written to the FIFO through the APB, the CNNx16_n_CTRL.bigdata/CNNx16_n_Ly_LCTRL0.parallel configuration, and the channel enables.	
14	stream_en	R/W	0	Streaming Mode Enable When set, the streaming mode is enabled for the CNNx16_n processor array. Streaming behavior is defined by the CNNx16_n Processor Stream Registers . See Streaming Mode Configuration for additional information. <i>Note: Unexpected behavior is likely when all four CNNx16_n processor arrays are not configured identically for streaming/non-streaming operation. Each CNN_x16_n processor array should be configured identically for streaming or non-streaming operation. See Streaming Mode Configuration for further details.</i>	
13	poolrnd	R/W	0	Average Pooling Enable When this bit is set, and average pooling is enabled, pooled values are rounded up for remainders greater or equal to 0.5 and down for remainders less than 0.5. 0: Average Pooling Disabled 1: Average Pooling Enabled	

CNNx16_n Control				CNNx16_n_CTRL	[0x0000]
Bits	Field	Access	Reset	Description	
12	cnn_irq	R/W	0	CNN Interrupt Enable This read/write bit indicates when the CNN interrupt request is active. It can be written to zero to reset the interrupt request. This interrupt signals the completion of CNN processing and should be masked in the interrupt control logic if not required. It can also be written to 1 to force an interrupt. 0: CNN interrupt not active. 1: CNN interrupt active, write 0 to clear the CNN interrupt.	
11:9	ext_sync	R/W	0	CNNx16_n External Sync Select Each of these bits enable the external sync input from one of the CNN_x16_n processors. These bits allow the CNNx16_n processors to optionally operate in synchronization with one of the other CNNx16_n processors. In the general case, when all 64 processors are operating on a single convolution, CNNx16_0 processor 0 is selected by all four of the CNNx16_n's as the master byte setting <i>ext_sync</i> = 0b001. Combinations of processors can be configured as long as the groups are made up of sequential processors, without gaps.	
8	oneshot	R/W	0	One Shot Layer Mode With this bit set, only one layer is processed when the CNNx16_n_CTRL.cnn_en bit is set. To advance to the next layer, the CNNx16_n_CTRL.cnn_en bit must be reset to 0 and then set to 1. The low to high transition causes the CNN state machine to advance through the next layer. Memories can be interrogated between layers when CNNx16_n_CTRL.cnn_en is 0. 0: One-shot layer mode disabled 1: One-shot layer mode enabled	
7	apbclk_en	R/W	0	APB Clock Always On Setting this bit forces the APB clock always on. When this bit is set to 0, clocks are only generated to the APB registers during write operations. 0: APB clocks are only on when APB register writes occur 1: APB clocks to the CNN APB registers is always on	
6	bigdata	R/W	0	Big Data Enable This bit globally selects the input data format that uses four data bytes per read for the groups of 4 processors. In other words, the four bytes allocated for a group of four processors is multiplexed to the first processor of the group.	
5	pool_en	R/W	0	Pooling Enable This bit globally enables pooling for all layers. 0: Global pooling disabled 1: Global pooling enabled for all layers. <i>Note: If this bit is set and the CNNx16_n_CTRL.calcmax bit is not set, the per-layer CNNx16_n_Ly_LCTRL0.maxpl_en bits are in effect.</i>	
4	calcmax	R/W	0	Max Pooling Enable This bit globally enables max pooling for all layers when the CNNx16_n_CTRL.pool_en bit is set. <i>Note that this bit will be in effect, per layer, when the global CNNx16_n_CTRL.pool_en bit is 0, and the per-layer CNNx16_n_Ly_LCTRL0.pool_en bits are set.</i>	

CNNx16_n Control				CNNx16_n_CTRL	[0x0000]
Bits	Field	Access	Reset	Description	
3	clk_en	R/W	0	Data Processing Clock Enable Setting this bit enables the clocks to the . This field <i>does not</i> affect the clocks to the APB registers. See the CNNx16_n_CTRL.apbclken bit for the description of the APB clock behavior. 0: Clocks disabled to the Data Processing registers 1: Clocks enabled to the Data Processing registers	
2:1	rdy_sel	R/W	b11	APB Wait State Selection This field determines the number of wait states added during an APB access. 0b00: 0 wait states 0b01: 1 wait state 0b10: 2 wait states 0b11: 3 wait states <i>Note: Write operations load the data one clock before the end of the cycle.</i>	
0	cnn_en	R/W	0	CNN Enable Setting this bit to 1 initiates CNN processing. Processing is triggered by this field to 1. Software must set this field to 0 and back to 1 to perform subsequent CNN processing operations. 0: CNN processing stopped 1: Start CNN processing <i>Note: This field must be written to 0 before writing it to 1 for subsequent CNN processing to start.</i>	

Table 27-20: CNNx16_n SRAM Control Register

CNNx16_n SRAM Control				CNNx16_n_SRAM	[0x0004]
Bits	Field	Access	Reset	Description	
22	lsbram	R/W	0	Bias Memory Light Sleep Setting this bit forces the bias memory into light sleep when the bias memory is not selected by the CNN system or the APB.	
21	lstram	R/W	0	TRAM Light Sleep Setting this bit forces the TRAMs into light sleep when the TRAM is not selected by the CNN system or the APB.	
20	lsmram	R/W	0	MRAM Light Sleep Setting this bit forces the MRAMs into light sleep when the MRAM is not selected by the CNN system or the APB.	
19	lsdram	R/W	0	Data RAM Light Sleep Setting this bit forces the Data RAMs into light sleep when the SRAM is not selected by the CNN system or the APB.	
18:17	-	RO	0	Reserved	
16	pd	R/W	0	CNNx16_n Memory Power Down Enable Set this field to 1 to put the CNNx16_n's SRAM, TRAM, MRAM, and bias memory into a power-down mode. All memory contents are lost in power downstate. 0: CNNx16 memories are not powered down 1: Power down the CNNx16_n's memories	

CNNx16_n SRAM Control			CNNx16_n_SRAM		[0x0004]
Bits	Field	Access	Reset	Description	
15	ds	R/W	0	CNNx16_n Memory Deep Sleep Enable Set this field to 1 to put the CNNx16_n's SRAM, TRAM, MRAM, and bias memory into a deep sleep state. In deep sleep, the memories contents are retained, but peripheral logic is powered down, and the memory cannot be accessed. All memory outputs are set to output 0. 0: Memories are not in deep sleep state. 1: Put the CNNx16_n's memories into a deep sleep state. <i>Note: This field has no effect If the CNNx16_n_SRAM.pd field is set to 1 (memories powered down).</i>	
14	-	RO	0	Reserved	
13:11	wpulse	R/W	0	Write Pulse Width This field determines the bit line pulse width applied to the memory during writes. 0b000: Use the minimum bit line pulse width 0b111: Use the maximum bit line pulse width <i>Note: Values of wpulse between 0 and 7 incrementally set the bit line pulse width between the minimum and maximum values.</i>	
10	wneg_en			Write Negative Voltage Enable This bit enables the CNNx16_n_SRAM.wneg_vol . If this field is 0, the system controls the negative voltage applied to the bit lines. 0: Write negative voltage time is controlled by the system. 1: Write negative voltage time is controlled by the setting in the CNNx16_n_SRAM.wneg_vol field.	
9:8	wneg_vol	R/W	0	Write Negative Voltage This field sets the SRAM negative voltage level applied to the bit lines. This field is only used when the CNNx16_n_SRAM.wneg_en field is set to 1. 0: $V_{DD} - 80\text{mV}$ 1: $V_{DD} - 120\text{mV}$ 2: $V_{DD} - 180\text{mV}$ 3: $V_{DD} - 220\text{mV}$	
7:6	ra	R/W	0	Read Assist Voltage This field controls the Read Assist value for the SRAM bit lines. 0: V_{DD} 1: $V_{DD} - 20\text{mV}$ 2: $V_{DD} - 40\text{mV}$ 3: $V_{DD} - 60\text{mV}$ These bits determine the WL underdrive (Read Assist) value. ra[1:0] = 00 limits the WL voltage to VDD, ra[1:0] = 01 limits the WL to VDD-20mV, ra[1:0] = 10 limits WL to VDD-40mV, and ra[1:0] = 11 limits the WL voltage to VDD-60mV.	
5:2	rmargin	R/W	b0011	SRAM Read Margin When CNNx16_n_SRAM.rm_en is set, this field determines the length of the memory access time. 0b0000: Slowest SRAM access time 0b0001: 0b0010: 0b0011: Fastest SRAM access time (Reset Default) 0b0100-0b1111: Reserved <i>Note: The value of this field has no effect unless the CNNx16_n_CTRL.rm_en field is set to 1.</i>	

CNNx16_n SRAM Control			CNNx16_n_SRAM		[0x0004]
Bits	Field	Access	Reset	Description	
1	rmargin_en	R/W	b1	SRAM Read Margin Enable Set this field to 1 to use the SRAM Access Time setting in the CNNx16_n_CTRL.ram_acc_time field. 0: SRAM access time is set by the hardware 1: SRAM access time is controlled using the CNNx16_n_CTRL.ram_acc_time field.	
0	extacc	R/W	0	SRAM Extended Access Time Enable Set this bit to 1 to enable SRAM access time maximum. Enabling longer SRAM access time increases the power consumption of the SRAM. 0: SRAM Power Optimized, reduced performance 1: SRAM Extended Access, higher power <i>Note, this setting can be used to extend access time but force the memories to consume additional power when active. This bit applies to all SRAMs in the CNNx16_n processor.</i>	

Table 27-21: CNNx16_n Layer Count Maximum Register

CNNx16_n Layer Count Maximum			CNNx16_n_LCNT_MAX		[0x0008]
Bits	Field	Access	Reset	Description	
31:5	-	RO	0	Reserved	
4:0	lcnt	R/W	0	Layer Count Maximum Set this field to the maximum layer number for processing by the CNNx16_n. When the CNNx16_n is enabled, processing starts at layer 0 and completes processing at the layer number set by this field. 0-31: Set to the last layer for processing by the CNNx16_n <i>Note: The CNNx16_n must be inactive(CNNx16_n_CTRL.cnn_en=0) when setting this field.</i>	

Table 27-22: CNNx16_n SRAM Test Register

CNNx16_n SRAM Test			CNNx16_n_TEST		[0x000C]
Bits	Field	Access	Reset	Description	
28	zerodone	R	0	BIST Zeroization Complete This field is set to 1 by hardware when any of the zero run bits are set to 1 by software, and the hardware completes the zeroization. This bit is read only. Clear this bit by setting each of the zero run bits to 0. 1: BIST zeroization complete	
27	bistdone	R	0	BIST Run Complete This field is set to 1 by hardware when any of the BIST run bits are set to 1 by software, and the hardware completes the BIST operation. This bit is read only. Clear this bit by setting each of the BIST run bits to 0. 1: BIST operation complete	
26	bistfail	R	0	BIST Run Failure Detected This field is set to 1 by hardware if a BIST run operation was run and a BIST failure occurred. This bit is read only. Clear this bit by setting each of the BIST run bits to 0. 0: If the CNNx16_n_TEST.bistdone bit reads 1, this bit indicates no BIST failures were detected. 1: BIST failure detected	

CNNx16_n SRAM Test			CNNx16_n_TEST		[0x000C]
Bits	Field	Access	Reset	Description	
25	ballzdone	R/W	0	BRAM Zeroization Complete This field indicates an SRAM zeroization is completed. This field is reset by hardware when software writes the CNNx16_n_TEST.bzerorun field to 0. 1: BRAM zeroization complete	
24	tallzdone	R/W	0	TRAM Zeroization Complete This field indicates a TRAM zeroization is completed. This field is reset by hardware when software writes the CNNx16_n_TEST.tzerorun field to 0. 1: TRAM zeroization complete	
23	mallzdone	R/W	0	MRAM Zeroization Complete This field indicates a MRAM zeroization is completed. This field is reset by hardware when software writes the CNNx16_n_TEST.mzerorun field to 0. 1: MRAM zeroization complete	
22	sallzdone	R/W	0	SRAM Zeroization Complete This field indicates an SRAM zeroization is completed. This field is reset by hardware when software writes the CNNx16_n_TEST.szerorun field to 0. 1: SRAM zeroization complete	
21	ballbdone	R/W	0	BRAM BIST Complete This field indicates a BRAM BIST run is completed. This field is reset by hardware when software writes the CNNx16_n_TEST.bbistrun field to 0. 1: BRAM BIST complete	
20	tallbdone	R/W	0	TRAM BIST Complete This field indicates a TRAM BIST run is completed. This field is reset by hardware when software writes the CNNx16_n_TEST.tbistrun field to 0. 1: TRAM BIST complete	
19	mallbdone	R/W	0	MRAM BIST Complete This field indicates a MRAM BIST run is completed. This field is reset by hardware when software writes the CNNx16_n_TEST.mbistrun field to 0. 1: MRAM BIST complete	
18	sallbdone	R	0	SRAM BIST Complete This field indicates an SRAM BIST run is completed. This field is reset by hardware when software writes the CNNx16_n_TEST.sbistrun field to 0. 1: SRAM BIST complete	
17	ballbfail	R	0	BRAM BIST Result When a BRAM BIST operation was started by setting CNNx16_n_TEST.bbistrun bit to 1, and the operation is completed by hardware (CNNx16_n_TEST.ballbdone is set to 1 by hardware), this field indicates the result of the BRAM BIST operation. Reset this field by writing a 0 to the CNNx16_n_TEST.bbistrun field. 0: BRAM BIST Passed 1: BRAM BIST Failed, indicating an error occurred in one of the BRAMs. <i>Note: This field is only valid after a BRAM BIST operation has started and completed (CNNx16_n_TEST.ballbdone = 1).</i>	

CNNx16_n SRAM Test			CNNx16_n_TEST		[0x000C]
Bits	Field	Access	Reset	Description	
16	tallbfail	R/W	0	SRAM BIST Result When a TRAM BIST operation was started by setting <code>CNNx16_n_TEST.tbistrun</code> bit to 1, and the operation is completed by hardware (<code>CNNx16_n_TEST.tallbdone</code> is set to 1 by hardware), this field indicates the result of the TRAM BIST operation. Reset this field by writing a 0 to the <code>CNNx16_n_TEST.tbistrun</code> field. 0: TRAM BIST Passed 1: TRAM BIST Failed, indicating an error occurred in one of the four SRAMs. <i>Note: This field is only valid after a TRAM BIST operation has started and completes (<code>CNNx16_n_TEST.tallbdone</code> = 1).</i>	
15	mallbfail	RO	0	MRAM BIST Result When a MRAM BIST operation was started by setting <code>CNNx16_n_TEST.mbistrun</code> to one, and the operation is completed by hardware (<code>CNNx16_n_TEST.mallbdone</code> is set by hardware to 1), this field indicates the result of the SRAM BIST operation. Reset this field by writing a 0 to the <code>CNNx16_n_TEST.mbistrun</code> field. 0: MRAM BIST Passed 1: MRAM BIST Failed, indicating an error occurred in one of the 16 MRAMs. <i>Note: This field is only valid after a MRAM BIST operation has started and completed (<code>CNNx16_n_TEST.mallbdone</code> = 1).</i>	
14	sallbfail	RO	0	SRAM BIST Result When an SRAM BIST operation was started by setting <code>CNNx16_n_TEST.sbistrun</code> to one, and the operation is completed by hardware (<code>CNNx16_n_TEST.sallbdone</code> is set by hardware to 1), this field indicates the result of the SRAM BIST operation. Reset this field by writing a 0 to the <code>CNNx16_n_TEST.sbistrun</code> field. 0: SRAM BIST Passed 1: SRAM BIST Failed, indicating an error occurred in one of the four SRAMs. <i>Note: This field is only valid after an SRAM BIST operation has started and completed (<code>CNNx16_n_TEST.sallbdone</code> = 1).</i>	
13:8	bistssel	R/W	0	BIST Controller Status Selection The bits select an individual BIST controller status to be reported in the associated 32 bits of the memory read data bus. <code>CNNx16_n_TEST.bistssel[5]</code> selects the SRAM or bias memory BIST group controller statuses, with: <code>CNNx16_n_TEST.bistssel[2:0]</code> selecting the individual SRAM/bias memory instance with <code>CNNx16_n_TEST.bistssel[2:0] = 100</code> selecting the bias memory Instance. Control bit <code>CNNx16_n_TEST.bistssel[4]</code> selects the MRAM BIST controller statuses, with <code>CNNx16_n_TEST.bistssel[3:0]</code> selecting the individual RAM instance. Control bit <code>CNNx16_n_TEST.bistssel[3]</code> selects the TRAM BIST controller statuses, with <code>CNNx16_n_TEST.bistssel[3:0]</code> selecting the individual RAM instance.	
7	bramz	R/W	0	BIAS Memory Zeroize Setting this bit to 1 will force the BIST to initialize all Bias memory locations to 0. Completion status can be found in the: • <code>CNNx16_n_TEST.sallzdone</code> and • <code>CNNx16_n_TEST.zerodone</code> This bit must be written to 0 to reset the operation prior to writing it to 1. Writing 1 consecutively does not run the memory zeroization again.	

CNNx16_n SRAM Test			CNNx16_n_TEST		[0x000C]
Bits	Field	Access	Reset	Description	
6	bbistrun	R/W	0	Run Bias Memory BIST Setting this bit to 1 will run the BIST for all bias memory instances in the CNNx16_n processor. The BIST will run to completion, and the status is reported in: • CNNx16_n_TEST.ballbdone • CNNx16_n_TEST.ballbfail • CNNx16_n_TEST.bistdone • CNNx16_n_TEST.bistfail If more detailed status is required from the BIST execution, the CNNx16_n_TEST.bistse1 field can be used to extract status from an individual BIST controller. This bit must be written to 0 to reset the BIST operation prior to writing it to 1. Writing 1 consecutively does not run the BIST again.	
5	tramz	R/W	0	TRAM Zeroize Setting this bit to 1 will force the BIST to initialize all TRAM memory locations to 0. Completion status can be found in the CNNx16_n_TEST.tallzdone and CNNx16_n_TEST.zerodone status bit. This bit is edge-triggered and must be toggled from zero to one to run the BIST.	
4	tbistrun	R/W	0	TRAM BIST Run Setting this bit to 1 will run the BIST for all TRAM instances in the CNNx16_n processor. The BIST will run to completion and the status reported in: • CNNx16_n_TEST.tallbdone • CNNx16_n_TEST.tallbfail • CNNx16_n_TEST.bistdone • CNNx16_n_TEST.bistfail If more detailed status is required from the BIST execution, the CNNx16_n_TEST.bistse1 field can be used to extract status from an individual BIST controller. This bit must be written to 0 to reset the BIST operation prior to writing it to 1. Writing 1 consecutively does not run the BIST again.	
3	mramz	R/W	0	MRAM Zeroize Setting this bit to 1 will force the BIST to initialize all MRAM memory locations to 0. Completion status can be found in the: • CNNx16_n_TEST.mallzdone and • CNNx16_n_TEST.zerodon This bit must be written to 0 to reset the operation prior to writing it to 1. Writing 1 consecutively does not run the memory zeroization again.	

CNNx16_n SRAM Test			CNNx16_n_TEST		[0x000C]
Bits	Field	Access	Reset	Description	
2	mbistrun	R/W	0	MRAM BIST Run Setting this bit to 1 will run the BIST for all MRAM instances in the CNNx16_n processor. The BIST will run to completion, and the status is reported in: • CNNx16_n_TEST.mallbdone • CNNx16_n_TEST.mallbfail • CNNx16_n_TEST.bistdone • CNNx16_n_TEST.bistfail If more detailed status is required from the BIST execution, the CNNx16_n_TEST.bistse1 field can be used to extract status from an individual BIST controller. This bit must be written to 0 to reset the BIST operation prior to writing it to 1. Writing 1 consecutively does not run the BIST again.	
1	sramz	R/W	0	SRAM Zeroize Setting this bit to 1 will force the BIST to initialize all SRAM memory locations to 0. Completion status can be found in the • CNNx16_n_TEST.sallzdone and • CNNx16_n_TEST.zerodone This bit must be written to 0 to reset the operation prior to writing it to 1. Writing 1 consecutively does not run the memory zeroization again.	
0	sbistrun	R/W	0	SRAM BIST Run Setting this bit to 1 will run the BIST for all SRAM instances in the CNNx16_n processor. The BIST will run to completion, and the status is reported in: • CNNx16_n_TEST.sallbdone • CNNx16_n_TEST.sallbfail • CNNx16_n_TEST.bistdone • CNNx16_n_TEST.bistfail If more detailed status is required from the BIST execution, the CNNx16_n_TEST.bistse1 field can be used to extract status from an individual BIST controller. This bit must be written to 0 to reset the BIST operation prior to writing it to 1. Writing 1 consecutively does not run the BIST again.	

27.5.2.1 CNNx16_n Per Layer Registers (CNNx16_n_Ly)

Each of the four CNNx16 Processor Arrays supports up to 32 layers. Each layer is controlled using the CNNx16_n's Layer registers. Each layer includes one instance of each layer register. Each layer register is 32 bits wide (4 bytes). Register names for a given layer are defined by replacing "y" with the layer number ranging from 0 to 31 (32 layers maximum). The offset address of each layer's specific register is determined by the layer's base offset address and adding 4 times the layer number in hex. For example, for the CNNx16_n_Ly Row Count Register, the base offset address is [0x0010]. For layer 21, register CNNx16_n_L21_RCNT, the address offset is [0x0010] + (4 × 0x15) = [0x0064].

Table 27-23: CNNx16_n_Ly Row Count Register

CNNx16_n_Ly Row Count			CNNx16_n_Ly_RCNT		[0x0010]-[0x008C]
Bits	Field	Access	Reset	Description	
31:18	-	RO	0	Reserved	

CNNx16_n_Ly Row Count				CNNx16_n_Ly_RCNT	[0x0010]-[0x008C]
Bits	Field	Access	Reset	Description	
17:16	rcnt_pad	R/W	0	Pad Rows This field sets the number of pad rows included at the beginning and end of the frame. <i>Note: The CNNx16_n_Ly_RCNT.rcnt_max is inclusive of two times this value, as the same number of pad rows are included at the beginning and end of the frame.</i>	
15:10	-	RO	0	Reserved	
9:0	rcnt_max	R/W	0	Layer Row Count Maximum This field sets the maximum row count to be processed. Processing begins with row 0 and completes when the processing of the row determined by this field is complete. Set this field to include two times the CNNx16_n_Ly_RCNT.rcnt_pad value. $rcnt_max = (2 \times rcnt_pad) + number\ of\ rows - 1$ <i>Note: The value programmed into this field is the effective image row value, including pad, but excluding rows not processed due to stride restrictions.</i>	

Table 27-24: CNNx16_n_Ly Column Count Register

CNNx16_n_Ly Column Count				CNNx16_n_Ly_CCNT	[0x0090]-[0x010C]
Bits	Field	Access	Reset	Description	
31:18	-	RO	0	Reserved	
17:16	ccnt_pad	R/W	0	Layer Column Pad Count This field determines the number of pad columns included at the beginning and end of the frame. Note that the CNNx16_n_Ly_CCNT.ccnt_max is inclusive of two times this value, as the same number of pad columns are included at the beginning and end of the row.	
15:10	-	RO	0	Reserved	
9:0	ccnt_max	R/W	0	Layer Column Count Maximum When the CNN is enabled, these bits determine the maximum per layer column count to be processed. Processing begins with column 0 and completes when the processing of the column determined by CNNx16_n_Ly_CCNT.ccnt_max is complete. The value programmed into this register, through the CNN APB interface, is the effective image column value including pad, but excluding columns not processed due to stride restrictions.	

Table 27-25: CNNx16_n_Ly One Dimensional Control Register

CNNx16_n_Ly One Dimensional Control				CNNx16_n_Ly_ONED	[0x0110]-[0x018C]
Bits	Field	Access	Reset	Description	
31:22	-	RO	0	Reserved	
21:18	ewise_cnt	R/W	0	Element-Wise Channel Count Determines the number of element-wise channels to be processed. 0: 1 channel 1: 2 channels 2: 3 channels ... 14: 15 channels 15: 16 channels	

CNNx16_n_Ly One Dimensional Control				CNNx16_n_Ly_ONED	[0x0110]-[0x018C]
Bits	Field	Access	Reset	Description	
17	2d_conv	R/W	0	2D Convolution of Element-Wise Result Set this field to 1 to enable 2D convolution of the element-wise result. Standard 2D processing applies. 0: 2D convolution disabled 1: Enable 2D convolution	
16	prepool	R/W	0	Pre-Pooling of Input Data Set this field to 1 to enable pre-pooling of the input data prior to the element-wise operation selected with CNNx16_n_Ly_ONED.ewise_func . 0: Input data is not pre-pooled prior to the element-wise function. 1: Pre-pooling of input data prior to element-wise operation enabled	
15:14	ewise_func	R/W	0	Element-Wise Function Select Selects the element-wise function performed on the input data if . 0b00: Subtract 0b01: Add 0b10: Bitwise OR 0b11: Bitwise XOR	
13	ewise_en	R/W	0	Element-Wise Enable Set this field to 1 to enable the element-wise operations. Prior to enabling element-wise operations, both the CNNx16_n_Ly_ONED.tscnt_max field and the CNNx16_n_Ly_POST.ts_en fields must be set. 0: Element-wise operation disabled 1: Enable element-wise operation	
12	oned_en	R/W	0	One Dimensional Processing Enable Enables 1D input data processing. If the CNNx16_n_Ly_RCNT.rcnt_max field is non-zero, the row count is used to index the input image; otherwise, the column count, CNNx16_n_Ly_CCNT.ccnt_max , value is used.	
11:8	oned_width	R/W	0	One Dimensional Convolution Mask Width One dimensional convolution mask width (0-9 are valid values). One based value with a width > 0 enabling 1D convolution operation.	
7:4	oned_sad	R/W	0	One Dimensional Convolution Start Mask Address One dimensional convolution start mask address (offset within 9-byte mask width) used in conjunction with the mask start address (CNNx16_n_Ly_MCNT.mcnt_sad) to determine that 1D convolution mask starting address.	
3:0	tscnt_max	R/W	0	Maximum Time Slot Count Maximum timeslot count register. When CNNx16_n_Ly_POST.tsen is set, this value determines the number of timeslots required to output data that has been generated in parallel by the CNNx16_n processors. This count is used for pass through, 1x1, elementwise, and mask sharing (CNNx16_n_Ly_LCTRL0.mslave and CNNx16_n_Ly_LCTRL0.sslave) operations.	

Table 27-26: CNNx16_n_Ly Pool Row Count Register

CNNx16_n_Ly Pool Row Count				CNNx16_n_Ly_PRCNT	[0x0190]-[0x020C]
Bits	Field	Access	Reset	Description	
31:4	-	RO	0	Reserved	

CNNx16_n_Ly Pool Row Count				CNNx16_n_Ly_PRCNT	[0x0190]-[0x020C]
Bits	Field	Access	Reset	Description	
3:0	prcnt_max	R/W	0	Pool Row Count Max When the CNN is enabled with one of the CNNx16_n_CTRL.pool_en (global) or CNNx16_n_Ly_LCTRL0.pool_en (per layer) bits set, this field determines the per layer pool row count to be processed. Processing begins with pool row 0 and completes when the processing of the pooling row determined by <i>prcnt_max</i> is complete, or the effective pool row count exceeds the image row count specified in CNNx16_n_Ly_RCNT.rcnt_max . These count values are added to the row count to determine the effective address of the pooled data.	

Table 27-27: CNNx16_n_Ly Pool Column Count Register

CNNx16_n_Ly Pool Column Count				CNNx16_n_Ly_PCCNT	[0x0210]-[0x028C]
Bits	Field	Access	Reset	Description	
31:4	-	RO	0	Reserved	
3:0	pccnt_max	R/W	0	Pool Column Count Max When the CNN is enabled with one of the CNNx16_n_CTRL.pool_en (global) or CNNx16_n_Ly_LCTRL0.pool_en (per layer) bits set, these bits determine the per layer pool column count to be processed. Processing begins with pool column 0 and completes when the processing of the pooling column determined by <i>pccnt_max</i> is complete, or the effective pool column count exceeds the image column count specified in CNNx16_n_Ly_CCNT.ccnt_max . These count values are added to the column count to determine the effective address of the pooled data.	

Table 27-28: CNNx16_n_Ly Stride Count Register

CNNx16_n_Ly Stride Count				CNNx16_n_Ly_STRIDE	[0x0290]-[0x030C]
Bits	Field	Access	Reset	Description	
31:2	-	RO	0	Reserved	
1:0	stride	R/W	0	Stride This field determines the stride length across and down the image. Processing begins with row 0 and column 0. A stride of one is applied until the top row or left column of the receptive field lands in the unpadded image. At that point, the stride value programmed through the APB interface is applied to the column and/or row of the field in the unpadded image. The stride is applied as long as the field remains within the bounds of the padded image.	

Table 27-29: CNNx16_n_Ly Write Pointer Base Address Register

CNNx16_n_Ly Write Pointer Base Address				CNNx16_n_Ly_WPTR_BASE	[0x0310]-[0x038C]
Bits	Field	Access	Reset	Description	
31:17	-	RO	0	Reserved	
16:0	wptr_base	R/W	0	Write Pointer Base This per layer register sets the CNN convolution result SRAM write pointer base address. The base address can be set to any location in any data SRAM in the CNN. Bit 16 allows the write pointer to not point to any SRAM.	

Table 27-30: CNNx16_n_Ly Write Pointer Timeslot Offset Register

CNNx16_n_Ly Write Pointer Timeslot Offset				CNNx16_n_Ly_WPTR_TOFF	[0x0390]-[0x040C]
Bits	Field	Access	Reset	Description	
31:17	-	RO	0	Reserved	
16:0	wptr_tooff	R/W	0	Write Pointer Timeslot Offset When the CNN is enabled with the CNNx16_n_Ly_POST.ts_en bit set and a non-zero time slot count value loaded into CNNx16_n_Ly_ONED.tscnt_max field, this per layer register sets the CNN convolution result SRAM write pointer timeslot offset value. During a convolution result, SRAM data write, this timeslot offset value is multiplied by the timeslot counter and added to the SRAM write address pointer. This offset can be used to determine SRAM write addresses based on the timeslot. The offset can be set to any location in any data SRAM in the CNN.	

Table 27-31: CNNx16_n_Ly Write Pointer Mask Offset Register

CNNx16_n_Ly Write Pointer Mask Offset				CNNx16_n_Ly_WPTR_MOFF	[0x0410]-[0x048C]
Bits	Field	Access	Reset	Description	
31:17	-	RO	0	Reserved	
16:0	wptr_moff	R/W	0	Write Pointer Mask Offset When the CNN is enabled with the CNNx16_n_Ly_EN.mask_en bit set and a mask count value loaded into the CNNx16_n_Ly_MCNT.mcnt_max register that is greater than the CNNx16_n_Ly_MCNT.mcnt_sad value, this per layer register sets the CNN convolution result SRAM write pointer mask count offset value. During a convolution result, SRAM data write, this mask count offset value is multiplied by the mask counter and added to the SRAM write address pointer. This offset can be used to determine SRAM write addresses based on a mask count. The offset can be set to any location in any data SRAM in the CNN.	

Table 27-32: CNNx16_n_Ly Write Pointer Multi-Pass Channel Offset Register

CNNx16_n_Ly Write Pointer Multi-Pass Channel Offset				CNNx16_n_Ly_WPTR_CHOFF	[0x0490]-[0x050C]
Bits	Field	Access	Reset	Description	
31:17	-	RO	0	Reserved	
16:0	wptr_choff	R/W	0	Write Pointer Multi-Pass Offset When the CNN is enabled with the CNNx16_n_Ly_EN.mask_en bit set and a programmed maximum mask counter value ($CNNx16_n_Ly_MCNT.mcnt_max - CNNx16_n_Ly_MCNT.mcnt_sad$) that is greater than the maximum number of available processors programmed into the CNNx16_n_Ly_LCTRL1.xpch_max register (output channel multi-pass is enabled), the rounded mask counter value is divided by the CNNx16_n_Ly_LCTRL1.xpch_max value and multiplied by the CNNx16_n_Ly_WPTR_CHOFF.wptr_choff to create a multi-pass channel offset value. During a convolution result SRAM data write, this multi-pass channel offset value is added to the SRAM write address pointer. This offset can be used to determine SRAM write addresses based on a multi-pass count value. The offset can be set to any location in any data SRAM in the CNN.	

Table 27-33: CNNx16_n_Ly Read Pointer Base Address Register

CNNx16_n_Ly Read Pointer Base Address				CNNx16_n_Ly_RPTR_BASE	[0x0510]-[0x058C]
Bits	Field	Access	Reset	Description	
31:17	-	RO	0	Reserved	

CNNx16_n_Ly Read Pointer Base Address			CNNx16_n_Ly_RPTR_BASE		[0x0510]-[0x058C]
Bits	Field	Access	Reset	Description	
16:0	rptr_base	R/W	0	Read Pointer Base Address When the CNN is enabled, this per layer register sets the CNN convolution result SRAM read pointer base address. The base address can be set to any location in the SRAM dedicated to a specific CNN input channel processor. <i>Note: This field is limited to the 8,192 bytes of SRAM in the MAX78000. Do not write values greater than the memory available.</i>	

Table 27-34: CNNx16_n_Ly Layer Control 0 Register

CNNx16_n_Ly Layer Control 0			CNNx16_n_Ly_LCTRL0		[0x0590]-[0x060C]
Bits	Field	Access	Reset	Description	
31:17	-	RO	0	Reserved	
16	bigdwrt	R/W	0	Big Data Write Enables writing out the current output channel in 32-bit accumulator form. This bit allows the full resolution of the output layer to be written to assist with software-defined softmax operation.	
15:12	cnsi_en	R/W	0	CNN 26-Bit Non-Scaled Non-quantized Sum of Products Feed When set, this field enables the associated CNNx16_n 26-bit non-scaled, non-quantized sum of products data into the output accumulator, allowing sums of 16, 32, 48, or 64 products. Each bit is associated with one of the remaining 3 CNNx16_n processors. In processor 0, bit 1 is associated with CNNx16_n processor 1, bit 2 with CNNx16_n processor 2, and bit 3 with CNNx16_n processor 3. In processor 1, bit 1 is associated with CNNx16_n processor 2, bit 2 with CNNx16_n processor 3, and bit 3 with CNNx16_n processor 0, and so on. When these bits are set to zero, the internal state of the 26-bit data bus is forced to zero. 0b0000: 26-bit data bus set to 0 0b0001-0b1111: See description	
11	sramsrc	R/W	0	SRAM CNNx16_n SRAM Global Write Source Data When set, SRAM data is sourced from the global data busses. The device supports 4 global data busses, one from each of the CNNx16_n processors. When all four CNNx16_n processors are used together to form a single sum-of-products value, all four CNNx16_n outputs an identical address, and based on the natural priority of the decode, processor 0 sources the SRAM write data.	
10	cpad_only	R/W	0	Input Frame Column Pad When set, padding is applied only to the input frame columns. In this case, row padding is ignored. This bit is intended to be used for parallel processing. <i>Note: When this field is set, row padding is ignored.</i>	
9	act_en	R/W	0	ReLU Activation Enable When set, ReLU activation is enabled for each output channel. Activation is applied to the scaled and quantized data. 0: ReLU not enabled 1: ReLU activation enabled for each output channel and is applied to the scaled and quantized data.	

CNNx16_n_Ly Layer Control 0			CNNx16_n_Ly_LCTRL0		[0x0590]-[0x060C]
Bits	Field	Access	Reset	Description	
8	maxpl_en	R/W	0	Max Pooling Enable When set to 1, Max Pooling is selected as the pooling mode. In Max Pooling mode, the maximum value in the pool is selected for the field and written into the TRAM. When this field is set to 0, average pooling mode is selected. In Average Pooling Mode, the average of all pooled values is calculated and selected for use in the field. 0: Average pooling mode selected. 1: Max pooling mode selected <i>Note: This field is only used if the CNNx16_n_Ly_LCTRL0.pool_en field is set to 1.</i>	
7	pool_en	R/W	0	Enable Pooling Setting this bit enables pooling for the associated layer. Pool dimensions are determined by the pool row and column count maximums (CNNx16_n_Ly_PRCNT.prcnt_max and CNNx16_n_Ly_PCCNT.pccnt_max). 0: Disabled 1: Enabled <i>Note: The type of pooling performed is determined by the CNNx16_n_Ly_LCTRL0.maxpl_en field as either average pooling or max pooling.</i>	
6	parallel	R/W	0	Parallel Mode Enable Setting this bit to one enables a single input channel's data to use 4 bytes of memory instead of one. In parallel mode, data is read in byte order from byte 0 to byte 3 and then by memory depth. The purpose of the mode is to allow additional memory to be used for the input layer data to support larger images. When set, the receptive field for the data will be generated in the first processor in each group of 4 processors, provided the processor is enabled.	
5	master	R/W	0	CNNx16_n Processor Group Master Select Enables a CNNx16_n group of processors to independently calculate a sum-of-products result for all adjacent ascending CNNx16_n processor groups not configured for master operation.	
4	mslave	R/W	0	CNNx16_n Processor 0 Mask Leader When this bit is set, all 16 processors within the CNNx16_n share the receptive field with processor 0. This allows the generation of additional output channels using the mask values distributed across the 16 processors. Each processor applies a 3x3 mask of the selected width to be applied to the processor 0 field. <i>Note: Use of timeslots is required to write the parallel generated output channels to memory.</i>	
3:0	sslave	R/W	0	CNNx16_n Processor Group Slave Mode Within a CNNx16_n, this field enables each of the 4 groups of four processors to share the receptive field with the first processor of the x4 group (channels 0,4,8,12). When set, the receptive field of the first processor in the associated x4 group to be passed to the remaining 3 processors in the x4 group. Masks associated with the slaved group of three processors can be applied to the field of the first processor and additional output channels generated. Use of the timeslot counter is required to write the additional output channels to memory.	

Table 27-35: CNNx16_n_Ly Layer Mask Count Register

CNNx16_n_Ly Mask Count				CNNx16_n_Ly_MCNT	[0x0610]-[0x068C]
Bits	Field	Access	Reset	Description	
31:16	mcnt_sad	R/W	0	Layer Mask RAM start Address Mask RAM address counter increments sequentially from this word and bit address during layer processing. Counter restarts each stride and increments once per output channel: • <i>mcnt_sad</i>[15:4]→SRAM word (72bits) address • <i>mcnt_sad</i>[2:0]→bit address	
15:0	mcnt_max	R/W	0	Mask RAM Layer Maximum Address Mask RAM address counter increments sequentially by word and bit address during layer processing to this address. Counter restarts each stride and increments once per output channel: • <i>mcnt_max</i>[15:4]→SRAM word (72-bits) address • <i>mcnt_max</i>[2:0]→bit address.	

Table 27-36: CNNx16_n_Ly TRAM Pointer Register

CNNx16_n_Ly TRAM Pointer				CNNx16_n_Ly_TPTR	[0x0690]-[0x070C]
Bits	Field	Access	Reset	Description	
31:27	-	RO	0	Reserved	
26:16	tptr_sad	R/W	0	TRAM Start Address This field determines the per layer TRAM pointer start address for initial processing and rollover events.	
15:11	-	RO	0	Reserved	
10:0	tptr_max	R/W	0	TRAM Max Address This field determines the rollover point of the TRAM address pointer. This field, <i>tptr_max</i> , is used together with the TRAM address pointer start address value (CNNx16_n_Ly_TPTR.tptr_sad) to reflect the usable input image row size, including pad.	

Table 27-37: CNNx16_n_Ly Enable Register

CNNx16_n_Ly Enable				CNNx16_n_Ly_EN	[0x0710]-[0x078C]
Bits	Field	Access	Reset	Description	
31:16	mask_en	R/W	0	CNNx16_n Processor Mask Enable Each bit in this per layer field enables the state of one processor's kernel logic in the CNNx16_n. Bit 0 controls processor 0's mask application (the master processor), and bit 15 controls processor 15's mask application.	
15:0	pro_en	R/W	0	CNNx16_n Processor Enable Each bit in this per layer field controls the enable state of one processor in the CNNx16_n. Bit 0 controls processor 0's (the master processor) enable and bit 15 controls processor 15's enable.	

Table 27-38: CNNx16_n_Ly Post Processing Register

CNNx16_n_Ly Post Processing				CNNx16_n_Ly_POST	[0x0790]-[0x080F]
Bits	Field	Access	Reset	Description	
31: 29	-	RO	0	Reserved	

CNNx16_n_Ly Post Processing			CNNx16_n_Ly_POST		[0x0790]-[0x080F]
Bits	Field	Access	Reset	Description	
28	deconv_en	R/W	0	Deconvolution Enable Virtually expands the input image size by adding a 0 byte after each actual input data byte is shifted into the TRAM, and a row of 0s following each row of column expanded input data is shifted into the TRAM. The receptive field remains at 3×3 scanned across the expanded data. 0: Deconvolution disabled 1: Deconvolution enabled	
27	flatten_en	R/W	0	Flatten Enable Enables flattening all of the input channel data supporting a series of 1×1 convolutions emulating a fully connected network. Setting this bit forces the use of an extended multi-pass count allowing for up to 256 neurons. This bit is used in conjunction with the CNNx16_n_Ly_LCTRL1.inpchexp , CNNx16_n_Ly_POST.xpmp_cnt , and CNNx16_n_Ly_POST.onexone_en fields to enable Multi-Layer Processing.	
26	out_abs	R/W	0	Absolute Value Output Enable Convert the scaled, quantized convolution output to an absolute value. This bit, along with the activation enable bit, CNNx16_n_Ly_LCTRL0.act_en determine the final processing operation prior to writing the out channel to memory. 0: Output is not converted to an absolute value 1: Output is converted to an absolute value <i>Note: The CNNx16_n_Ly_POST.out_abs has priority over the activation enable bit, CNNx16_n_Ly_LCTRL0.act_en.</i>	
25	onexone_en	R/W	0	Pass-Thru/1×1 Convolution Mode Enable This bit forces all sixteen processors in the CNNx16_n to either directly pass through the result of the pooling logic or compute a 1 data byte by 1 byte (1,2,4,8 bit) weight product. Control of a pass through or 1×1 convolution is made for each of the 16 processors using the CNNx16_n_Ly_EN.mask_en control field. 0: Pass-Thru Enabled 1: 1×1 Enabled	
24	ts_en	R/W	0	Timeslot Mode Enable This bit is used to enable the timeslot counter. When enabled, the number of timeslots programmed into the timeslot counter, CNNx16_n_Ly_ONED.tscnt_max , are added to each output channel slot. The timeslot counter allows pass through, 1×1, and elementwise values calculated in parallel to be written sequentially to memory. 0: Timeslot Mode Disabled 1: Timeslot Mode Enabled	
23:22	mask_size	R/W	0	Mask Size Selection This field determines the mask size multiplied with each 8-bit data value in the convolution. 0b00: 8-bits 0b01: 1-bit 0b10: 2-bits 0b11: 4-bits	
21:18	xpmp_cnt	R/W	0	Expanded Multi-Pass (MP) Count Adds 4 additional bits to the MP counter for flattening (MLP) operations. This field is only used when the CNNx16_n_Ly_POST.flatten_en bit is set to 1. This field is appended to the CNNx16_n_Ly_LCTRL1.inpchexp bits and makes up the 4 most significant bits of the count.	

CNNx16_n_Ly Post Processing				CNNx16_n_Ly_POST	[0x0790]-[0x080F]
Bits	Field	Access	Reset	Description	
17	scale_shft	R/W	0	Scale Shift Control This bit selects the shift direction of the pre-activation sum-of-products result of the convolution. 0: Left Shift 1: Right Shift	
16:13	scale_reg	R/W	0	Scale Shift Number This field sets the number of bits to shift the pre-activation sum-of-products result of the convolution. Valid values are 0 to 16-bits, and the direction of the shift is controlled by <i>CNNx16_n_Ly_POST.scale_shft</i> .	
12	bptr_en	R/W	0	Bias Enable This field enables the addition of a scaled bias, stored in each bias location, to the result of the convolution. Bias values are automatically scaled by a shift left of seven bits in hardware.	
11:0	bptr_sad	R/W	0	Bias Pointer Start Address Bias register file address byte counter increments once each mask count increment. <i>Note: The x16 Bias values can be enabled and used independently across the four output processors.</i>	

Table 27-39: CNNx16_n_Ly Layer Control 1 Register

CNNx16_n_Ly Layer Control 1				CNNx16_n_Ly_LCTRL1	[0x0A10]-[0x0A8C]
Bits	Field	Access	Reset	Description	
31:17	-	RO	0	Reserved	
16:8	xpch_max	R/W	0	Expansion Mode Maximum Processors This field selects the maximum channel processor number used in channel expansion mode. This allows for fewer than 64 processors to be used in a multi-pass or channel expansion configuration. <i>Note: Processor management was shown to be important for mask management when input channel processing draws on a relatively small number of channels for a potentially large number of masks.</i> 0-64: Selects the number of processors to use for channel expansion mode <i>Note: This field contains 9 bits; the upper 3 bits are reserved. Only values up to 64 are supported.</i>	
7:4	wptr_inc	R/W	0	Write Pointer Increment This field determines the increment for the write pointer counter after all output channels within a given stride are written to memory. Non-zero values in this field are used for multi-pass operations. <i>Note: The Write Pointer is always incremented by 1 or by 4 in parallel mode. The value in this field is added to the internal write pointer increment.</i>	

CNNx16_n_Ly Layer Control 1			CNNx16_n_Ly_LCTRL1		[0x0A10]-[0x0A8C]
Bits	Field	Access	Reset	Description	
3:0	inpchexp	R/W	0	Multi-Pass Input Channel Expansion Count This field determines the number of sequential memory locations associated with a single convolution. For example, if a value of 15 (0xf) is programmed into this field, 16 channels are assumed to be sequentially stored in memory (channel then byte order) for each of the 64 processors. This allows for up to 1024 input channels to be processed in a single convolution. Similarly, if this field is set to 1, 2 channels are assumed to be sequentially stored in memory (channel then byte order) as channels for the convolution, totaling 128 channels if all 64 processors are used. A value of zero disables the multi-pass feature allowing for a single channel for each processor.	

Table 27-40: CNNx16_n Mlator Data Register

CNNx16_n Mlator Data			CNNx16_n_MLAT		[0x1000]
Bits	Field	Access	Reset	Description	
31:0	mлатdat	RO	0	Packed Channel Data This register accumulates four bytes of output channel data to be read by the host MCU. Channel data is usually stored in the memory depth of a single byte of the SRAM data word, and four channels make up the memory data word. The Mlator automatically fetches four bytes in the memory depth to generate a packed 4-byte word for efficient reading by the MCU. The target channel is selected using the CNNx16_n_CTRL.mlat_sel bits, and the target SRAM address is determined by the CNNx16_n_Ly_WPTR_BASE register. Setting the CNNx16_n_CTRL.mlat_ld bit to 1 loads the CNNx16_n_Ly_WPTR_BASE.wptr_base value into the address counter and initiates the read of the first 4 bytes of channel data. When the current 4 bytes are accumulated in the CNNx16_n_MLAT.mlatdat is read through the APB interface, the next 4 bytes are read in sequence. Reading continues until halted by the MCU. Four clock cycles are required after the completion of the last read or setting of the CNNx16_n_CTRL.mlat_ld bit to 1 to accumulate the 4 bytes of data. It is up to the MCU software to ensure there is adequate time between reads to accumulate the 4 bytes of data.	

27.5.2.2 CNNx16_n Processor Stream Registers

Each of the four CNNx16 Processor arrays supports up to 8 simultaneous streams for the first 8 layers. Each stream is controlled using the CNNx16_n's Stream registers. Each stream includes one instance of each Stream register. Each Stream register is 32 bits wide (4 bytes). Register names for a given stream are defined by replacing "z" with the stream number ranging from 0 to 7 (8 streams maximum). The offset address of each stream's specific register is determined by the stream's base offset address and adding 4 times the stream number in hex. For example, for the CNNx16_n_Sz Stream Control 0 Register, the base offset address is [0x0810]. For Stream 5, register CNNx16_n_S5_STRM0, the address offset is [0x0810] + (4 × 0x05) = [0x0824].

Table 27-41: CNNx16_n_Sz Stream Control 0 Register

CNNx16_n_Sz Stream Control 0			CNNx16_n_Sz_STRM0		[0x0810]-[0x082C]
Bits	Field	Access	Reset	Description	
31:14	-	RO	0	Reserved	

CNNx16_n_Sz Stream Control 0			CNNx16_n_Sz_STRM0		[0x0810]-[0x082C]
Bits	Field	Access	Reset	Description	
13:0	strm_isval	R/W	0	Per-Stream Start Count The per-stream start count is based on the previous layer's <i>tptr_inc</i> count. When streaming is enabled (<i>CNNx16_n_CTRL.stream_en</i> = 1), each time a byte in the prior or input layer is written into the TRAM, the internal stream start counter is incremented. When the counter reaches this field's value, <i>CNNx16_n_Sz_STRM0.strm_isval</i> , processing of the current layer is enabled. This mechanism allows adequate receptive field data to be accumulated before convolution processing in a stream layer begins. Once the counter value reaches this field's value, <i>CNNx16_n_Sz_STRM0.strm_isval</i> , counting is halted, and the terminal count value remains in the counter.	

Table 27-42: CNNx16_n_Sz Stream Control 1 Register

CNNx16_n_Sz Stream Control 1			CNNx16_n_Sz_STRM1		[0x0890]-[0x08AC]
Bits	Field	Access	Reset	Description	
31:28	-	RO	0	Reserved	
27:16	strm_dsval2	R/W	0	Per Stream Multi-Row Delta Count This field is based on the previous layer's <i>CNNx16_n_Ly_TPTR</i> count. When streaming is enabled (<i>CNNx16_n_CTRL.stream_en</i> = 1), this field determines the number of bytes written into the TRAM of the prior layer between active processing of the current layer. This APB accessible R/W register is used to set the processing cadence across an image row boundary. When the internal delta count counter reaches the value stored in this field, <i>CNNx16_n_Sz_STRM1.strm_dsval2</i> , processing of the current layer is enabled for one stride (input pooling plus output channel generation), and the delta counter is reset to zero.	
15:9	-	RO	0	Reserved	
8:4	strm_dsval1	R/W	0	Per Stream In-Row Delta Count This field is based on the previous layer's <i>tptr_inc</i> count. When streaming is enabled (<i>CNNx16_n_CTRL.stream_en</i> = 1), this field determines the number of bytes written into the TRAM of the prior layer between active processing of the current layer. This APB accessible R/W register is used to set the processing cadence of each column across an image row. When the internal delta count counter reaches the value stored in this field, <i>CNNx16_n_Sz_STRM1.strm_dsval1</i> , processing of the current layer is enabled for one stride (input pooling plus output channel generation), and the delta counter is reset to zero.	
3:0	strm_invol	R/W	0	Per Stream Input Volume Offset This field is based on the stream count. When streaming is enabled (<i>CNNx16_n_CTRL.stream_en</i> = 1), this field determines the input volume offset applied to each stream. The value programmed into this field is multiplied by the stream count to calculate the stream count input volume offset. The CNN supports up to 16 independent input volumes. The input volumes are split between multi-pass and streaming. The streaming input volume offset allows the streaming input volume selection to "skip over" the input volumes used for multi-pass processing.	

Table 27-43: CNNx16_n_Sz Stream Frame Buffer Size Register

CNNx16_n_Sz Stream Frame Buffer Size			CNNx16_n_Sz_FBUF		[0x0910]-[0x092C]
Bits	Field	Access	Reset	Description	
31:17	-	RO	0	Reserved	
16:0	fbuf_max	R/W	0	Per Stream Circular Buffer Max Count When streaming is enabled (<i>CNNx16_n_CTRL.stream_en</i> = 1), the per-layer SRAM read pointer base register value (<i>CNNx16_n_Ly_RPTR_BASE.rptr_base</i>) is subtracted from the internally generated read pointer counter and compared to the value stored in this field. When the adjusted read pointer is equal to this field, the rollover point is detected, and the <i>CNN_FIFO_STAT.rptr_base</i> value is loaded into the read pointer counter.	

Table 27-44: CNNx16_n Input FIFO Frame Size Register

CNNx16_n Input FIFO Frame Size			CNNx16_n_IFRM		[0x0990]
Bits	Field	Access	Reset	Description	
31:17	-	RO	0	Reserved	
16:0	ifrm_reg	R/W	0	Streaming Input Frame Size Byte Count When streaming is enabled (<i>CNNx16_n_CTRL.stream_en</i> = 1), this field determines the number of bytes read from the data FIFOs. An internal counter counts the number of input frame bytes read from the input FIFOs and compares the count to the value in this register. Once the value in this field is reached, the terminal count value is retained, incrementing of this counter is inhibited, and processing of the input data is terminated.	