[Document Name] Specification

[Title of the Invention] Neural Network Processor

[Technical Field]

　This invention relates to a neural network processor.

[Background Technology]

　A neural network processor is a processor that performs inferential computations on neural network models. Neural network processors include, for example, neuromorphic processors, which are computing chips that mimic neurons. The neuromorphic processor also has a configuration that mimics neurons and has been proposed, for example, as a spiking neural network processor. In a spiking neural network processor, spikes propagating between neurons are mapped to a single bit of data.

　Neural network processors disclosed herein are not limited to neuromorphic processors or spiking neural network processors. However, in the following embodiment, a neuromorphic processor (hereinafter referred to as NM processor) and a spiking neural network (hereinafter referred to as SNN) processor, etc., which are examples of a neural network processor, are disclosed.

　A neural network model has a plurality of first neurons (nodes) that receive input data and a plurality of second neurons connected to the first neurons via their respective synapses (edges). Such a configuration is called a convolutional layer or a fully connected layer. In the neural network model, the input data of a plurality of first neurons and the synaptic weights are product summed, and the second neuron calculates the activation output that the activation function of the second neuron calculates based on the result of the product sum operation. The synapse weights are one of the parameters that are fine-tuned in the deep learning of neural networks.

　In the SNN processor, on the other hand, a single bit of data that mimics a spike is propagated between neurons as input and output data. Therefore, operations by neurons in SNN are lightweight and low-power consumption is possible, and SNN is expected to be an edge computer for artificial intelligence models.

[Prior art references]

　[Patent documents]

　　　[Patent document 1] U.S. Patent Publication No. 2022/0129769

　　　[Patent document 2] U.S. Patent No. 10,621,489

　　　[Patent document 3] U.S. Patent No. 9,924,480

　　　[Patent document 4] U.S. Patent No. 10,824,937

　　　[Patent document 5] Patent Publication No. 20-43541

　　　[Patent document 6] WO 2020/008642 A1

　[Non-patent document]

　　　[Non-patent document 1] An In-Situ Dynamic Quantization With 3D Stacking Synaptic Memory for Power-Aware Neuromorphic Architecture, IEEE Access, Volume 11, 2023 August 3, 2023, NGO-DOANH NGUYEN, XUAN-TU TRAN, ABDERAZEK BEN ABDALLAH, and KHANH N. DANG

[Summary of Invention]

　　[Problem to be solved]

　When neural network models or SNN models are mapped to a processor chip, the memory that stores parameters such as synaptic weights occupy a large area in the processor chip. As a result, the distance between the plurality of computing cores mapped to neurons increases.

　Furthermore, the large power consumption when accessing parameters such as weights stored in the memory, and the large power consumption of the memory for these parameters is a problem that must be solved to realize an edge computer. Another issue is that defects in the memory cells of the memory, especially in the memory cells that store the upper bits of parameters such as weights, may cause a decrease in the accuracy of operations by the neural network processor.

　Therefore, the purpose of the first aspect of the embodiment is to provide a neural network processor that solves the above problem.

　　[To solve the problem]

　The first aspect of the embodiments is a neural network processor comprises

at least one computing chip layer having a plurality of neural computing cores and a network-on-chip connecting the plurality of neural computing cores; and

a plurality of memory chip layers each having a plurality of memory blocks each storing a plurality of parameters,

wherein, each of said plurality of neural computing cores has a network interface connected to said network-on-chip and a neuron array having a plurality of physical neurons performing operations on a neuronal model, and

said parameters are quantized to a fixed-point format having a first number of bits,

wherein, the quantized parameters comprise at least a first partial parameter having an upper bit and a number of bits smaller than the first number of bits, and a second partial parameter having a lower bit below the upper bit and a number of bits smaller than the first number of bits,

the stacked plurality of memory chip layers comprises a first memory chip layer storing the first partial parameter and a second memory chip layer storing the second partial parameter.

　　[Effects of the Invention]

　According to the first aspect, the above issues can be solved.

[Brief description of the drawing].

　Figure 1: Figure 1 shows an example of the configuration of a neuromorphic processor (NM processor), which is an example of a Neural network processor (hereinafter referred to as NN processor)

　Figure 2: Side view of the NM processor in the first embodiment.

　Figure 3: Exploded perspective view of an example of the configuration of each chip layer of the NM processor in the first embodiment.

　Figure 4: Exploded perspective view of an example of the configuration of each chip layer of the NM processor in the second embodiment.

　Figure 5: This figure shows the quantization process of the weights and the decomposition process that divides the quantized weights into partial weights.

　Figure 6] This chart shows various scenarios of the dynamic quantization process.

　Figure 7: This figure shows an example of storing the plurality of partial weights in the plurality of memory chip layers 207\_1 to 207\_4.

　Fig. 8: This figure shows the overall flowchart of the power supply control method.

　Figure 9: Flowchart of a power supply control for the memory chips layer in the low power mode 605\_1 in the first power supply control method.

　Figure 10: Flowchart of a power supply control for the memory chip layer in the high-performance mode 604\_1.

　Figure 11: Flowchart of a power supply control for memory chip layer in the low power mode 605\_2 in the second power supply control method.

　Figure 12: Flowchart of a supply control for memory chip layer power in high-performance mode 604\_2 in the second power supply control method.

　Figure 13: The figure shows the change in power supply and power consumption of the memory chip layer in the power supply control method.

　Figure 14: An example of an arithmetic unit in the computing chip layer.

　Figure 15: This figure shows an example of the operation of the multiplexer of the arithmetic unit in Figure 14.

　Figure 16: This figure shows the first reallocation method of partial weights in a defective memory block.

　Figure 17: This figure shows the second reallocation method of partial weights in a defective memory block.

　[Fig. 18] The figure shows four patterns of the second reallocation method.

　Figure 19: An example of an arithmetic unit in the computing chip layer in the third embodiment.

　Figure 20: This diagram illustrates the operation of the circuit in the input stage of the adder in Figure 19.

[Embodiments according to the invention]

　Neural network processors perform convolution operations of input data and parameters like weights and etc. of a neural network. The parameters of a neural network include various data such as bias in addition to weights. In the embodiment, the configuration of the memory chip layer that stores the parameters and the processing of the parameters are described with weights, one of the parameters, as an example.

　Figure 1 shows an example of structure of a neuromorphic processor (NM processor), which is an example of a neural network processor (hereinafter referred to as NN processor). The NM processor 101 shown in Figure 1 is a semiconductor chip, having a plurality of neural computing cores 102 arranged in a matrix and a two-dimensional mesh network-on-chip 103 that enables communication between the cores. The NM processor 101 is connected via inter-chip connections 108 to a host processor chip 109.

　The neural computing core 102 has a network interface (NI) 104, a weight memory (WM) 105, and a neuron array (NA) 106. The network interface 104 is connected to a router 107 and performs the interface process of communications between the neural computing core 102 and the network-on-chip 103. The router 107 corresponding to each core 102 is provided on the network-on-chip 103 and routes the communication propagating through the network-on-chip 103 to the direction of the destination core. Further, the router 107 routes communications destined for a core in the direction of that core.

　As mentioned above, the network interface 104 of the core receives the input routed by the router 107 from the network-on-chip 103. This input is, for example, an arithmetic request issued by the host processor 109 that decodes instruction. The network-on-chip 103 decodes the received computation request and sends the decoded request to the neuron array 106 and weight memory 105. Based on the respective requests, the weight memory 105 reads the weights and sends them to the neuron array, and the neuron array 106 performs the operation included in the request on the input data in the request and the read weight data. The results of the operations are encoded and returned to the router 107, which forwards them to the next destination neural computing core 102.

　The neuron array 106 has a plurality (N) of physical neurons 110, indexed by identification numbers SN1 to SNN. The plurality of physical neurons 110 correspond to the plurality of neurons in the neural network. The weight memory 105 has a plurality of memory blocks (WMBs) 111 and the weight data stored in each memory block 111 correspond to edges between neurons in the neural network. These mappings are performed by the host processor 109, which issues an arithmetic request based on the mapping and sends it to the neural computing core 102, the destination of the arithmetic request.

Each physical neuron 110 has an arithmetic unit and performs the requested operation. The physical neuron 110 receives a request containing input data from the network interface 104 and weight data read based on the operation request from the weight memory 105. The physical neuron 110 then performs the requested operation using the input data and weight data.

　If the requested operation is a convolution operation, the physical neuron 110 receives the input data transmitted from the input neuron and the weight data corresponding to the edges of the input neuron. The physical neuron 110 then performs a sum-of-products operation on the input data and the weight data, inputs the result of the sum-of-products operation into an activation function of the neuron model, and output result of the activation function. The operations of the physical neurons are executed synchronously with the clock cycles in the NM processor, for example, and the results of the operations are sent to the next physical neuron that executes next operation.

That is, if the input data sent from M input neurons are input1 - inputM , the weight of each edge is weight1 - weightM , and the activation function is factivation, then the output of the physical neuron performing the operation is as follows.  
　output = factivation (Σ (i=1-M) inputi \* weight )i  
 where M is the number of input data and weights. The activation function is the ReLU function, etc.

　[First embodiment]  
　Figure 2 is a side view of the NM processor in the first embodiment. Figure 3 is an exploded perspective view of the NM processor showing an example of the configuration of each chip layer of the NM processor. As shown in Figures 2 and 3, the NM processor 213 is composed of a plurality of chip layers 201 and 207. The plurality of chip layers includes a computing chip layer 201 with a plurality of neural computing cores 205 and a network-on-chip 204. Further, the plurality of chip layers includes a plurality of memory chip layers 207 stacked on the computing chip layer 201.

　The plurality of chip layers 201, 207 are mounted on a package substrate 215. The plurality of chip layers 201, 207 are enabled to communicate via interlayer vias 209. An interlayer communication section 208 in the plurality of chip layers 201 and 207 communicates between the plurality of chips via the interlayer vias 209. The plurality of memory chip layers 207 have high-speed memory, such as SRAM and ReRAM, for example.

　The network-on-chip 204 in the computing chip layer 201 is connected to the host processor 211 via an inter-chip connection 214. The network-on-chip 204 and router 202 enable communication between the plurality of neural computing cores 205. The router 202 is provided for each core 205.

　The neural computing core 205 in the computing chip layer 201 has a network interface (NI) 203, a neuron array (NA) 212, and an interlayer communication section 208. However, core 205 does not have a weight memory.

　On the other hand, the plurality of memory chip layers 207 each has a weight memory 206 that stores weights and other parameters. Therefore, the neural computing core 205 in the computing chip layer 201 does not have a weight memory. Thus, the area of the core is smaller than the example in Figure 1. This shortens the distance between the plurality of routers 202 located on the network-on-chip 204. This means that the communication time between cores 205 will be shorter. Also, more cores 205 can be placed on the computing chip layer 201.

　The NM processor stores the plurality of weights and other parameters in the weight memory 206 in the memory chip layer 207. The floating-point weights are quantized in a fixed-point format with a first number of bits, and the quantized weights are stored in the weight memory 206. Accordingly, the arithmetic unit in the physical neuron is a fixed-point arithmetic unit that input the fixed-point weights to execute a calculation.

　The plurality of memory chip layers 207\_1 to 207\_N store the partial weights, each of which is generated by dividing the quantized weight into N quantized weights. That is, the quantized weight of the first number of bits is decomposed into the first through the Nth partial weights of the second number of bits less than the first number. In the example in Figure 2, N = 4. The first partial weight with the most significant bit MSB is stored in the weight memory in the first memory chip layer 207\_1, and the Nth partial weight with the least significant bit LSB is stored in the weight memory in the Nth memory chip layer 207\_N. The first through N partial weights are divided in such a way that the bits of the partial weights are decremented in order, and are stored in the first through N memory chip layers, respectively. The divided partial weights do not necessarily need to have the same number of bits (second number of bits), but may have different number of bits.

　When the plurality of partial weights are stored separately in the first through N memory chip layers as described above, the first memory chip layer stores the partial weight including MSB, the second through N-1 memory chip layers store partial weights descending in order of digit, and the Nth memory chip layer stores the partial weight including LSB. The specifics of the four memory chip layers are shown later in Figures 5 through 7.

　According to the first embodiment, first, since partial weights having upper bits to lower bits are stored separately in N memory chip layers 207, the power of the memory chip layer that stores partial weights having lower bits can be turned off to save power. In this case, the partial weights having the lower bits stored in the memory chip layer whose power is turned off are no longer available, but since they are lower bits, the NM processor's accuracy degradation can be suppressed.

　Second, the partial weights having the upper bits in the memory block where defects occur can be overwritten in the non-defective memory block that stores the partial weights of the lower bits to improve the fault tolerance of the weight memory. In this case, the weights of the lower bits are unavailable, but since they are lower bits, the NM processor's accuracy degradation can be suppressed.

　[Second embodiment]

　Figure 4 is an exploded perspective view of an example configuration of each chip layer of the NM processor in the second embodiment. The NM processor 213 in Figure 4 has the same configuration as the NM processor shown in Figures 2 and 3. That is, the NM processor 213 is stacked with a computing chip layer 201 and first through N memory chip layers 207\_1 through 207\_N. The computing chip layer 201 has a plurality of neural computing cores 205 and a network-on-chip 204 used for communication between them. The weights and other parameters are stored within the N memory chip layers 207\_1 to 207\_N.

　Unlike Figure 3, Figure 4 provides a power rail 1505, a power control circuit 1502, and a voltage regulator 1503 in the plurality of memory chip layers 207\_1 to 207\_N. In addition, a power rail 1505 and a power input/output port 1500 are provided in the computing chip layer 201. The power supply rails 1505 of these chip layers 207 and 201 are connected via power supply vias 1501. The power input/output port 1500 is connected to an external power supply and power is supplied to the NM processor 213. The power supplied from the external power supply is supplied to the electronic circuits of each chip layer 201 and 207 via the power supply rails and power supply vias. The configuration of the power supply rails, power supply vias, power control circuitry, and voltage regulators in Figure 4 differs from that in Figure 3.

　Based on the power control signals (not shown) from the host processor 211, the power supply control circuit 1502 in each memory chip layer 207\_1 to 207\_N controls the power supply in the memory chip layer on and off. When the power supply control circuit 1502 controls power on, the power is supplied to the voltage regulator 1503, and the voltage regulator 1503 controls the power supply voltage of the power rail 1505 to rise or fall based on the voltage control signal from the host processor 211. The power supply voltage controlled by the voltage regulator is supplied to each circuit module in the memory chip layer via the power rail 1505.

　Figure 5 shows the quantization process of the weights and the decomposition process that divides the quantized weights into partial weights. An example of a neural network model is shown at the top of Figure 5. In this example, the neural network model 301 has an input neuron model 302\_IN, an output neuron model 302\_OUT, and a weight model 303 for the edges between both neuron models. The weight models 303 are usually weights Wik and Wjk in floating point format. The weights suffix i, j are the input side neuron model i and the output side neuron model j.

　The quantization process 304 quantizes the floating point weights 303 into a fixed number of bits, e.g., 8-bit fixed point weights Wik [0:7], Wjk [0:7]. The quantization weights 305 are quantized in 8-bit fixed-point format. The 0 side of [0:7] corresponds to the MSB and the 7 side of [0:7] to the LSB.

　The decomposition process 306 decomposes the 8-bit quantization weights Wik [0:7] and Wjk [0:7] into, for example, four partial weights 307. For example, the partial weights are 2 bits. The specifics are shown below.  
 The first partial weight 307\_1 is Wik [0:1], Wjk [0:1], which is a 2-bit partial weight including the MSB and the second digit.  
 The second partial weights 307\_2 are Wik [2:3] and Wjk [2:3], which are 2-bit partial weights including the third and fourth digits.  
 The third partial weights 307\_3 are Wik [4:5] and Wjk [4:7], which are 2-bit partial weights including the 5th and 6th digits.  
 The fourth partial weights 307\_4 are Wik [6:7] and Wjk [6:7], which are 2-bit partial weights including the seventh digit and LSB.

　Figure 6 is a chart showing various scenarios of the dynamic quantization process. In the figure, column 401 shows an example of a floating-point weight model 303, column 402 shows a binary number of the weight model 303, column 403 shows an 8-bit quantized weight 305, column 404 shows a decimal number of the quantized weight 305, and column 405 shows a partial weight 307 that is broken down into 2-bit pieces.

　Furthermore, column 406 shows the partial weights 307 when the lowest partial weight bit is inverted. Inverted bits are underlined. Column 407 shows the value of the inverted bits in decimal. This shows that the change in the value of the weights is limited when the bit of the lowest partial weight is flipped.

　Figure 7 shows an example of a plurality of partial weights stored in a plurality of memory chip layers 207\_1 to 207\_4. In the figure, a computing chip layer 201 and four memory chip layers 207\_1 to 207\_4 are stacked. However, the two upper memory chip layers 207\_3 and 207\_4 are shown side by side on the right. The four partial weights Wik [0:1] to Wik [6:7] are stored separately in the memory blocks MB\_1 to MB\_4 of the weight memories 206\_1 to 206\_4 of each memory chip layer 207\_1 to 207\_4 respectively.

　Preferably, the partial weights Wik [0:1], Wik [2:3], Wik [4:5], and Wik [6:7] between neurons i,k are stored in memory blocks MB\_1 to MB\_4 at the same on-chip address in weight memories 206\_1, 206\_2, 206\_3, 206\_4 respectively. The partial weights Wjk [0:1], Wjk [2:3], Wjk [4:5], and Wjk [6:7] between neurons j and k are similarly stored in memory blocks at the same on-chip address.

This mapping of the memory blocks storing the four partial weights to the same address in the memory chip layer simplifies the reading process of the four partial weights. In other words, the host processor 211 can read the four partial weights Wik [0:1], Wik [2:3], Wik [4:5], and Wik [6:7] from the four memory chip layers 207 respectively by specifying the address in the chip of the weight Wik [0:7]. The physical neuron operator can then merge the bits of the four partial weights read out to obtain the 8-bit weights Wik [0:7]. The merged 8-bit weights are then input to the fixed-point operator. However, a plurality of partial weights of a given weight do not have to be stored in a memory block at the same on-chip address. In that case, the arithmetic instruction includes the on-chip address for each of the four partial weights, and the network interface in the core accesses the memory block that stores the partial weights at the respective on-chip address.

　Furthermore, the first partial weights W[0:1] are preferably stored in the weight memory 206\_1 in the first memory chip layer 207\_1 that is closest to the computing chip layer 201. The reason for this is that the first partial weight W[0:1], which includes the most significant bit MSB of the quantization weights W[0:7], has the greatest impact on the accuracy of the inference of the neural network processor. The computing chip layer 201 can read the first partial weights of the MSBs from the nearest first memory chip layer 207\_1 at high speed.

　［Power control method]

　In the second embodiment, power supply control is used to reduce the power consumption of the plurality of memory chip layers when power consumption is higher than the power supply of the NM processor. The two power supply control methods are described below.

In the power control method, the host processor 211 sends power control signals to the power control circuit 1502 and the voltage regulator 1503 in the memory chip layer to execute power control of the memory chip layer. In the NM processor shown in Figure 4, the power control signal issued by the host processor 211 is sent to the power control circuit 1502 in the memory chip layer to be controlled via the inter-chip connection 214, the computing chip layer 201, and the interlayer via 209.

　Figure 8 shows the overall flowchart of the power control method. The host processor 211, which performs power control 600, controls the memory chip layer in high-performance mode (604) when the power supply of the NM processor, Psupply, is greater than the power consumed by the NM processor, Pconsumed (YES in 601). On the other hand, the host processor 211 controls the memory chip layer in low power mode (605) when the power supply of the NM processor Psupply is less than the NM processor's power consumption Pconsumed (YES in 602). Then, the host processor 211 maintains the current control mode (603) when the power supply of the NM processor Psupply is equal to the NM processor’s power consumption Pconsumed (NO in 601 and 602). The above power control is executed every predetermined time T.

　［First power control method (power control by dynamic quantization)］

　Figure 9 shows a flowchart of the power control of the memory chip layers in low power mode 605\_1 in the first power control method. The power on and off control of the four memory chip layers corresponding to the time axis is shown in the lower right corner of the flowchart.

　The host processor 211 starts power control from the index k of the topmost memory chip layer whose power supply is in the active state (on state) (701). If all memory chip layers are in the power on state, the host processor starts power control from the fourth memory chip layer that is the topmost layer. If the power supply to the NM processor, Psupply, is less than or equal to the power consumption of the NM processor, Pconsumed (YES in 702), the host processor enters the low power mode 605. In the low power mode, if the index k has not reached the lowest index "1" of the memory chip layer (NO in 703), the power of the memory chip layer k is controlled to be off (704). Then, the index k is reduced by one (705), and the same low power mode processing (702, 703, 704) is performed as described above. When the index k reaches the lowest index "1" (YES in 703), the low-power mode ends because the power supply of the lowest memory chip layer that stores the MSBs of the partial weights cannot be turned off (706).

　As shown in the lower right corner of the flowchart, when the power control enters the low power mode while all memory chip layers are powered on (k=4), the NM processor is controlled to power off from the top memory chip layer (k=4) to the bottom layer. When the NM processor is powered off from the top memory chip layer (k = 4) to the second memory chip layer (k = 2), one above the first memory chip layer, the NM processor continues to infer using the partial weights of the MSBs stored in the first memory chip layer that is powered on. In this case, the inference accuracy is reduced, but the reduction in the inference accuracy is limited because the calculation is performed with the first partial weights that include MSBs.

　Figure 10 shows the flowchart of power control for the memory chip layer in high-performance mode 604\_1. The host processor 211 starts at index k of the topmost memory chip layer where the power supply is active (on) (801), and if the NM processor power supply Psupply is greater than or exceeds the NM processor power consumption Pconsumed (YES in 802), the power control enters the high-performance mode 604\_1.

　In high-performance mode, while index k has not reached the top N of the memory chip layer (NO in 803), index k is increased by one (804) and memory chip layer k is controlled to be powered on (805). Since the number of memory chip layers powered on has increased, it is checked whether the power supply of the NM processor, Psupply, is less than the power consumed by the NM processor, Pconsumed (806). If this comparison 806 is NO, the same high performance mode process (802-805) as described above is performed. If this comparison 806 is YES, the power of the memory chip layer of index k that was controlled to be turned on immediately before is turned back off (807), and the high-performance mode is terminated. If index k reaches the top layer N (YES in 803), the high-performance mode is also terminated.

　As shown in the lower right corner of the flowchart, if the power control enters the high-performance mode when the memory chip layers from the topmost layer to the second layer are power off, while the NM processor power supply Psupply is greater than the NM processor power consumption Pconsumed, the memory chip layers from the second layer to the topmost layer are controlled to be turned on in order from the second memory chip layer to the top memory chip layer.

　Entering the low power mode 605\_1 of Figure 9 above, the partial weights of the lower bits become unavailable in sequence, and the number of bits of quantized weights being available decreases. On the other hand, when entering the high-performance mode 604\_1 of Figure 10 above, the partial weights of the lower bits become available in sequence, and the number of bits of quantized weights being available increases. Thus, the number of bits of quantized weights is dynamically changed and controlled.

　In Figure 9, the index k of the memory chip layer at the start of control is not necessarily the fourth memory chip layer at the top layer (k = 4); if the top layer in the power-on state is the third memory chip layer, the index k at the start is 3. Similarly, it may start at k = 2. Conversely, in Figure 10, the index k of the memory chip layer at the start is not necessarily the lowest first memory chip layer (k=1); if the top layer in the power-on state is the second memory chip layer, the index k at the start is 2. Similarly, it may start at k = 3.

　[Second power control method (power control by dynamic quantization and voltage scaling)]

　In the second power control method, in low power mode, the host processor 211 reduces the supply voltage of a memory chip layer before turning off the active power supply of the memory chip layer. The host processor then turns off the power of the memory chip layer, if the power supply is still less than or equal to the consumed power when the power supply voltage reaches the minimum voltage Vmin.

Conversely, in high-performance mode, the host processor increases the supply voltage of one lower memory chip layer before turning on the inactive power supply of the memory chip layer. The host processor then turns on the power supply of one upper memory chip layer, if the power supply is still greater than or exceeds the consumed power when the power supply voltage reaches the maximum voltage Vmax. The operation of the second power supply control method is the same as that of the first power supply control method except for the increase or decrease (scaling) of the power supply voltage.

　Figure 11 shows a flowchart of the power supply control of the memory chip layer in low power mode 605\_2 in the second power supply control method. In this flowchart, the host processor 211 decreases its power supply voltage Vk by a certain amount (709) if the power supply voltage Vk of the index k to be controlled has not reached the minimum operable voltage Vmin (NO in 708). If the supply voltage Vk has reached the minimum operable voltage Vmin (NO in 708), the host processor controls the memory chip layer of index k to power off (704, 705), when k is not equal to 1 (NO at 708). The host processor 211 then performs the same control as the low power mode 605\_1 in the first power control method of Figure 9, except for this process 708 and 709.

　Figure 12 shows a flowchart of the power supply control of the memory chip layer in high-performance mode 604\_2 in the second power supply control method. In this flowchart, the host processor 211 increases the power supply voltage Vk of the index k to be controlled by a certain amount (809) if the power supply voltage V has not reached the maximum voltage Vmax (NO in 808). If the supply voltage Vk has reached the maximum voltage Vmax (NO in 808), the host processor controls the memory chip layer of index k+1 on one upper layer to power on (804-807), when k is not equal to N (No in 803). The host processor 211 then performs the same control as the high-performance mode 604\_1 in the first power control method in Figure 10, except for this process 808 and 809.

　Figure 13 shows the power supply and power consumption variations of the memory chip layer power control methods. 605\_1 and 604\_1 in the upper row of Fig. 13 are examples of power changes for dynamic quantization only of the first power supply control method. The lower 605\_2 and 604\_2 in Figure 13 are examples of power changes for dynamic quantization and supply voltage scaling of the second power supply control method.

In the upper 605\_1 and 604\_1 of Figure 13, power on/off control of the memory chip layer is performed, resulting in a large staircase-like change in power consumption. In 605\_2 and 604\_2 in the lower part of Figure 13, in addition to power on/off control of the memory chip layer, power supply voltage in the memory chip layer is controlled to rise and fall, resulting in a small staircase shape of power consumption change. In the high-performance mode 604\_1 and 604\_2, power consumption is controlled to increase once and then decrease according to steps 704 and 805 in the flowchart.

　Figure 14 shows an example of an arithmetic unit in the computing chip layer. The adder ADDER, an arithmetic unit, takes the weight input in1 and the input in2 of the input node, performs an addition operation, and outputs the result of the operation out. The weight input in1 of the adder is an 8-bit fixed-point number. In this mode, the memory chip layer’s power supply is controlled and the NM processor is controlled to operate in high-performance mode or power-saving mode. In the high-performance power mode, for example, all partial weights are used as weights, while in the power-saving mode, for example, only the MSB partial weights are used as weights.

　When the number of partial weights read from the partial weight memory varies due to power supply control, the adder inputs the valid partial weights read from the memory and the 2-bit signal 00 instead of the invalid partial weights through multiplexers MUX11 to MUX14. In the figure, pw1 to pw4 are partial weights read from the memory chip layer. on1 to on4 are control signals corresponding to the power on/off of the memory chip layer.

　The partial weights pw1 to pw4 read from the memory chip layer include valid bits read from the power-on memory chip layer and invalid bits read from the power-off memory chip layer. Based on the control signals on1 to on4 corresponding to power on/off, the multiplexer replaces the invalid partial weight bits pw1 to pw4 with the 2-bit signal 00 and outputs the replaced 2-bit signal (when the control signal on1 to on4 is or are 0) or outputs the valid partial weight bits pw1 to pw4 as is (when the control signalson1 to on4 is ore are 1).

　Figure 15 shows an example of the operation of the multiplexer of the arithmetic unit in Figure 14. The assumption is that the four partial weights pw1-pw4 are all 2 bits 01. Each of the four partial weights pw1 to pw4 read out is high impedance z for invalid bits.  
　In the example in Figure 15, the memory chip layers are turned off in order from the layer on the lower bit side as time TIME elapses. At the leftmost clock timing CK1, all control signals on1 to on4 are "1", so all multiplexers output the valid partial weights pw1 to pw4 read from the memory chip layer as is. When the clock timing moves to CK2 to CK4, the partial weights of the lower bits become invalid bits z in sequence and are replaced by the 2-bit signal 00. Of the 8 bits of the weight input in1 input to the arithmetic unit, the underlined bits 00 are replaced.

　As described above, in the second embodiment, since the plurality of partial weights are stored in the plurality of memory chip layers separately, in the power-saving mode, the host processor controls power off starting from the memory chip layer of the division weight of the lowest bit in order. In the high-performance power mode, the host processor controls power-on starting from the memory chip layer with the second highest bit division weight. This allows the host processor to control the power supply in the power-saving mode and the high-performance power mode while preventing the NM processor's inference accuracy from degrading.

[Third embodiment]

　In the third embodiment, when a certain memory block in the memory chip layer becomes defective and the partial weights pw stored in it become defective, the partial weights pw in the defective memory block are reallocated (reassigned) to an undefective memory block that stores partial weights for the lower bits. This allows the NM processor to continue to use the partial weights of the upper bits, which have a large impact on the NM processor's calculation accuracy. The method of reallocating partial weights in defective memory blocks is described below.

　[First Realocate Method].

　Figure 16 illustrates the first reallocation method of partial weights in a defective memory block. Figure 16 shows a computing chip layer 201 and four memory chip layers 207\_1 to 207\_4. Suppose that memory block MB\_1, which stores partial weights Wik [0:1] including the most significant bits, is defective and error bit Faulty pw is detected in the partial weight Wik [0:1] read from it.

The host processor 211 lists the partial weights with error bits. The host processor checks whether the partial weights with error bits are in the memory block of the memory chip layer that stores the LSB partial weights. If the check result is false, the host processor 211 overwrites the partial weight Wik [0:1] stored by the defective memory block with the memory block MB\_4 that stores the partial weight Wik [6:7] containing its least significant bit to reallocates the partial weight Wik [0:1]. This stops the use of partial weights Wik [6:7] including the least significant bits, but allows the use of partial weights Wik [0:1] including the most significant bits to continue.

　If the check result is true, the host processor 211 stops using the partial weights containing the least significant bits of the error bits without reallocating them.

　As a result of the reallocation of partial weights Wik [0:1], including MSBs, from memory chip layer 207\_1 to memory chip layer 207\_4, the host processor 211 converts the address of partial weights Wik [0:1] in the calculation request issued by the NM processor to the address of the reallocation destination. Based on this address translation, the network interface NI of the NM processor reads the partial weight Wik [0:1] from the memory chip layer 207\_4 based on the address of the reallocation destination in the computation request.

　[Second reallocation method].

　Figure 17 shows a second reallocation method of partial weights in a defective memory block. Figure 17 has one computing chip layer 201 and four memory chip layers 207\_1 to 207\_4 as in Figure 16, where partial weights Wik [0:1] to Wik [6:7] and Wjk [0:1] to Wjk [6:7] are stored as well. And the same is true for the defective memory block MB\_1, which stores the partial weights Wik [0:1], including the most significant bits.

The assumption is that each of the four partial weights Wik [0:1]-Wik [6:7] and Wjk [0:1]-Wjk [6:7] are stored in memory blocks at the respective same address in the four memory chip layers.

　In the second reallocation method, the host processor 211 lists the partial weights with error bits. The host processor checks whether the partial weights with error bits are in the memory block of the memory chip layer that stores the LSB partial weights.

If the check result is false, the host processor 211 reallocates the partial weights Wik [0:1], Wik [2:3], and Wik [4:5] in memory chip layers 207\_1, 207\_2, and 207\_3 to the memory blocks in memory chip layers 207\_2, 207\_3, and 207\_4 one layer above. This is shown in Reallocation ReA in the figure. In other words, the bundle of partial weights from the partial weights Wik [0:1] where the defect occurred to the partial weights Wik [4:5] of one bit higher than the partial weights Wik [6:7] of the LSB are reallocated to the memory block in the memory chip layers one layer above, respectively. When reallocated in this manner, the host processor 211 does not change the address of the partial weights in the calculation request, but changes the control signals to the multiplexers in the input stage of the arithmetic unit. As described below, the reallocated bundle of partial weights can be input to the arithmetic unit as the output of the original memory chip layer simply by changing the control signals. Thus, the load on the host processor 211 can be reduced.

　If the check result is true, the host processor 211 stops using the partial weights containing the least significant bit of the error bit without reallocating the partial weights.

　Figure 18 shows the four patterns of the second reallocation method. The five columns in each table show the memory chip layer “Layer”, the undefective or defective “T/F”, the partial weights stored in each memory chip layer before reallocation “original”, the partial weights stored in the memory chip layer after reallocation ”realct”, and the partial weights W[0:1]-W[ 6:7] at the wight input of the arithmetic unit “in1”. The multiplexer in the input stage of the arithmetic unit, shown in Figure 19, maps the reallocated partial weights read from the memory chip layer to the bits of the input in1 of the arithmetic unit.

　The first reallocation pattern 1611 is the example shown in Figure 17. In other words, this is an example where the memory block in the memory chip layer storing the partial weight pw1 including the most significant bit is defective F, and the bundle of partial weights w[0:1] in the defective memory block to w[2:3] and w[4:5] are reallocated to memory chip layers pw2 to pw4 on one upper layer, respectively. The host processor 211 reads the reallocated partial weights w[0:1], w[2:3], and w[4:5] from the memory chip layers one layer above, respectively, without changing the addresses of the weights in the calculation request. The read partial weights are then shifted to the original bit positions of the partial weights before reallocation by the multiplexer in the input stage of the arithmetic unit, as described below, and are input to the arithmetic unit.

　The second reallocation pattern 1612 is an example where the memory block storing the partial weight pw2 of the second memory chip layer from the most significant bit is defective F, and the bundle of partial weights w[2:3] of the defective memory block to w[4:5] are transferred to the memory chip layers pw3 to pw4, one layer above. In this case, too, the host processor 211 reads the reallocated partial weights w[2:3] and w[4:5] from the memory chip layer one layer above, respectively, without changing the addresses of the weights in the calculation request. The read partial weights are then shifted to the original bit positions of the partial weights before reallocation by the multiplexer in the input stage of the arithmetic unit, as described below, and are input to the arithmetic unit.

　The third reallocation pattern 1613 is an example where the memory block storing the partial weights pw3 in the third memory chip layer from the most significant bit becomes defective F and the partial weights w[4:5] of the defective memory block are reallocated to the memory chip layer pw4 one layer above. In this case, too, the host processor 211 does not change the address of the weights in the calculation request and reads out the reallocated partial weights w[4:5] from the memory chip layer one layer above, respectively. The read partial weights w[4:5] are then shifted to the original bit positions of the partial weights before reallocation by the multiplexer in the input stage of the arithmetic unit, as described below, and are input to the arithmetic unit.

　Finally, the fourth reallocation pattern 1614 is an example where the memory block storing the partial weight pw4 of the least significant bit memory chip layer is defective F and is not reallocated. The partial weights w[6:7] of the defective memory block are converted to the fixed 2-bit 00 by the multiplexer in the input stage and input to the arithmetic unit.

　Figure 19 shows an example of an arithmetic unit in the computing chip layer in the third embodiment. As in Figure 14, the adder ADDER, which is an arithmetic unit, takes the weight input in1 and the input node input in2, performs an addition operation, and outputs the result of the operation out. The weight input in1 is 8-bit fixed-point format data.

　In this implementation, partial weights in defective memory blocks in the memory chip layer are reallocated to undefective memory blocks in other memory chips. Therefore, partial weights read from the four memory chip layers are read from the memory chip layer to which they are reallocated.

　In Figure 19, pw1 to pw4 are partial weights read from memory chip layers 207\_1 to 207\_4. The control signals on1 to on4 correspond to valid or invalid of the partial weights in the memory chip layer. The control signals s1 to s3 correspond to whether the partial weights pw2 to pw4 read from the memory chip layer are shifted or not.

In the adder in Figure 19, the partial weight pw1 to pw4 read from the memory chip layer to which the partial weight is or are reallocated are shifted down to the original bit positions of the weights by multiplexers MUX1 to MUX3. The reallocated partial weight bits are then input to the adder as the original bits of the 8-bit weights. Another multiplexers, MUX11-MUX14, convert the invalid partial weights to fixed 2-bit 00. The invalid partial weight bit is converted to 00 and is input to the adder.

　Figure 20 illustrates the operation of the circuitry in the input stage of the adder in Figure 19. The partial weights pw1 to pw4 read from the four memory chip layers, control signals on1 to on4, and control signals s1 to s3 at clock timings CK1 to CK4 shown in Clock Clock.

　Clock CK1 shows the signals when all memory chip layers are no defects. The partial weights pw1 to pw4 read from the memory chip layer are "01", "10", "01", and "10" as an example. Since there are no defects, the control signals on1 to on4 and s1 to s3 sent from the host processor 211 are all 0.

　Clock CK2 signals the case of a defect in the memory block of the most significant bit partial weights in the lowest memory chip layer. In this case, the host processor 211 reallocates a bunch of partial weights w[0:1], w[2:3], and w[4:5] to memory chip layers pw2 to pw4 on one upper layer, respectively. In other words, it corresponds to the first reallocation pattern 1611 described above.

　In this case, the host processor 211 sets the control signals s1-S3 to "111". Based on this control signal, the three multiplexers MUX1 to MUX3 shift down the partial weights pw2 to pw4 read from the memory chip layers to the original bit positions of the original weights. Furthermore, the host processor sets the control signal on1 to on4 to "0001" and causes multiplexer MUX14 to output fixed bit 00. The remaining multiplexers MUX11 to MUX13 output the shifted-down partial weights. As a result, the input in1 of the adder becomes "01100100".

　Next, clock CK3 shows the signal when all memory chip layers are no defect. It is the same as clock CK1. Then, at clock CK4, the signal is shown for the case of a defect in the memory block of the third memory chip layer. In this case, the host processor 211 reallocates the partial weight w[4:5] = 01 to the memory chip layer pw4, one layer above. This example corresponds to the third reallocation pattern 1613 described above.

　The host processor sets the control signals s1-S3 to "001". Based on this control signal, multiplexer MUX3 shifts down the partial weight pw4 read from the memory chip layer to the original bit position of the original weight. Furthermore, the host processor sets the control signals on1 to on4 to "0001" and causes multiplexer MUX14 to output fixed bit 00. The remaining multiplexers MUX11 to MUX13 output the read partial weights w[0:1], w[2:3] and the shifted-down partial weights w[4:5]. As a result, the input in1 of the adder becomes "01100100", the same as for clock CK2.

　In the above two examples, the input in1 of the adder will be "01100110" if there is no defect and "01100100" if there is a defect. In the case of a defect, the partial weight of the least significant bit is "00". Therefore, the degradation of the calculation accuracy of the processor is suppressed.

　As described above, according to the third embodiment, a plurality of divided partial weights are respectively stored in a plurality of memory chip layers, and when a defect occurs in a memory block of a memory chip layer that stores a partial weight of a certain weight, the partial weight with the defect is stored in a memory chip layer that stores a partial weight of a lower bit than it. This allows the partial weights of the defective upper bits to continue to be used, thereby suppressing the degradation of the NM processor chip's arithmetic precision.

[Conclusion]

　In the above embodiments, the weights, which are one of the parameters of the neural network, are quantized, the quantized weights are decomposed into a plurality of partial weights, and the plurality of partial weights are stored separately in the plurality of memory chip layers. However, parameters other than weights, e.g., biases of convolutional operations of the neural network, etc., may be quantized and decomposed into a plurality of partial parameters, and the plurality of partial parameters may be stored separately in the plurality of memory chip layers.

[Description of Signs].

201: Computing chip layer

202: Router

203: Network interface

204: Network-on-a-chip

205: Neural Computing Core, Core

206: Weight memory

207: Memory chip layer

208: Interlayer communication section

209: Interlayer via

211: Host processor

213: NM processor, Neuromorphic processor

214: Chip-to-chip connection

215: Package substrate

302: Neuron model

303: Weight model

305: Quantization weights

307: Partial weights, partial quantization weights

MB: Memory block

MUX: Multiplexer

pw1 to pw4: partial weights, memory chip layer output

on1-4: Control signals, power on/off signals, valid/invalid signals

s1-s3: Control signals, shift signals

ADDER: Adder, an example of an arithmetic unit

in1, in2: Adder inputs

[Name of document] Patent claims

[Claim 1]

　A neural network processor comprising:

at least one computing chip layer having a plurality of neural computing cores and a network-on-chip connecting the plurality of neural computing cores,  
　a plurality of memory chip layers each having a plurality of memory blocks each storing each of a plurality of parameters,  
　each of said plurality of neural computing cores has a network interface connected to said network-on-chip and a neuron array having a plurality of physical neurons performing operations on a neuron model,

　the plurality of parameters are quantized to a fixed-point format having a first number of bits,  
　the quantized parameters are decomposed into at least a first partial parameter having upper bits and a bit number less than the first bit number and a second partial parameter having lower bits below the upper bits and a bit number less than the first bit number,  
　the plurality of memory chip layers are,  
　　a first memory chip layer storing said first partial parameter; and  
 a second memory chip layer storing said second partial parameter. (First embodiment)

[Claim 2]

　The neural network as claimed in claim 1, wherein

the plurality of memory chip layers are stacked on the computing chip layer, and interlayer vias for communication between the computing chip layer and the plurality of memory chip layers are provided,

furthermore, in the stacked plurality of memory chip layers, the second memory chip layer is stacked on the first memory chip layer. (First embodiment, stacking order of the stacked plurality of memory chip layers)

[Claim 3]

　The neural network processor as claimed in claim 2, wherein

the neural network processor is controlled to one of two states:

a first state in which the first and second memory chip layers are powered on and

a second state in which the first memory chip layer is powered on and the second memory chip layer is powered off. (Second embodiment)

[Claim 4]

　The neural network processor as claimed in claim 3, wherein

　when the supply power of said neural network processor is greater than the power consumption of said neural network processor, the neural network processor is controlled to said first state,  
　when said supply power is less than said power consumption of said neural network processor, the neural network processor is controlled to the second state. (Figure 9, second embodiment, power off control of the first power control)

[Claim 5]

　The neural network processor as claimed in claim 4, wherein

when the power supply is greater than the power consumption while controlled to the second state, the neural network processor is controlled to the first state. (Figure 10, second embodiment, power on control of the first power control)

[Claime 6]

　The neural network processor as claimed in claim 3, wherein

　when the supply power of the neural network processor is greater than the power consumption of the neural network processor, the neural network processor is controlled to the first state,  
　During said first state, when the supply power is less than the power consumption, the supply voltage of the second memory chip layer is controlled to be reduced,  
　when the supplied power is less than the consumed power while the supply voltage of the second memory chip layer is reduced to the minimum supply voltage that is operable, the neural network processor is controlled to the second state. (Figure 11, second embodiment of power control, second power supply voltage drop and power off control)

[Claim 7]

　The neural network processor as claimed in claim 6, wherein

when the power supplied is greater than the power consumed during the second state, the second memory chip layer is controlled to turn on and increase the power supply voltage from the minimum supply voltage. (Figure 12, second embodiment, second power control power on and power supply voltage increase)

[Claim 8]

　The neural network processor as claimed in claim 3, wherein

　the quantized parameter is decomposed into N partial parameters, from the first partial parameter having the most significant bit and having a bit number less than the first bit number, to the Nth partial parameter having the least significant bit and having a bit number less than the first bit number, in descending order of bit digits of the partial parameters, said N is an integer greater than or equal to 2,  
　the stacked plurality of memory chip layers are,  
　stacked as N memory chips layer from the first memory chip layer storing the first partial parameter to the Nth memory chip layer storing the Nth partial parameter, in descending order of bit digits of the partial parameters  
　when the power supply of the first to N memory chip layers is on and the power supply becomes lower than the power consumption, the power supply of each memory chip layer is controlled to turn off in turn from the Nth to the second memory chip layer. (Second embodiment, Figures 9 and 10)

[Claim 9]

　The neural network processor as claimed in claim 2, wherein

　when a memory block in the first memory chip layer becomes defective, the first partial parameter is stored in the second memory chip layer and the first partial parameter is read from the second memory chip layer to input to the physical neuron. Neural network processor. (Third embodiment)

[Claim 10]

　The neural network processor as claimed in claim 9, wherein

　the first partial parameter stored in the first memory chip layer that is in a defective state is excluded from the read data. (Third embodiment, Figure 16)

[Claim 11]

　The neural network processor as claimed in claim 9, wherein

　the quantized parameter is decomposed into N partial parameters, from the first partial parameter having the most significant bit and having a bit number less than the first bit number, to the Nth partial parameter having the least significant bit and having a bit number less than the first bit number, in descending order of bit digits of the partial parameters, said N is an integer greater than or equal to 2,  
　the stacked plurality of memory chip layers are,  
　stacked as N memory chip layers stacked from the first memory chip layer storing the first partial parameter to the Nth memory chip layer storing the Nth partial parameter, wherein the bits of the partial parameters are decremented from the bottom layer to the top layer in sequence,

the first through N partial parameters that are decomposed from said quantized parameters are stored in the memory blocks of the same address in the first through N memory chip layers, and  
　when the memory block storing the Mth (M is an integer between 1 and N-1) partial parameter among the first through Nth partial parameters becomes defective, the Mth through N-1th partial parameters are reallocated to the memory blocks at the same address in the M+1th through Nth memory chip layers, and said Mth through N-1th partial parameters are read from said reallocated memory block. (Third embodiment, Figures 17 and 18)

[Claim 12]

　The neural network processor as claimed in claim 11, wherein

　said Mth partial parameters stored in the Mth memory chip layer that is in a defective state are excluded from the read data. (Third embodiment, Figures 17 and 18)

[Claim 13]

　The neural network processor as claimed in claim 11, wherein

　reallocation information including M is received,

the partial parameters read from the memory block of the same address in the M+1th to Nth memory chip layer are input as the Mth to N-1th partial parameters based on the reallocation information. (Third embodiment, Figures 19 and 20)

[Claim 14]

　The neural network processor as claimed in claim 1, wherein the parameter is either a neural network weight or a bias.

[Document Name] Abstract

[Abstract]

[Problems].

Solve problems related to memory for storing neural network parameters including distance between computing cores, memory power consumption, and memory fault tolerance, etc.

[Means of Solution]

A neural network processor including at least one computing chip layer having a plurality of neural computing cores and a network-on-chip connecting the plurality of neural computing cores; and a plurality of memory chip layers each having a plurality of memory blocks each storing a plurality of parameters, wherein, each of said plurality of neural computing cores has a network interface connected to said network-on-chip and a neuron array having a plurality of physical neurons performing operations on a neuronal model, and said parameters are quantized to a fixed-point format having a first number of bits, wherein, the quantized parameters comprise at least a first partial parameter having an upper bit and a number of bits smaller than the first number of bits, and a second partial parameter having a lower bit below the upper bit and a number of bits smaller than the first number of bits, the stacked plurality of memory chip layers comprises a first memory chip layer storing the first partial parameter and a second memory chip layer storing the second partial parameter

[Selected figure] Figure 3.