



(51) International Patent Classification:

G06N 3/04 (2006.01) G11C 16/00 (2006.01)
G06N 3/08 (2006.01)

(21) International Application Number:

PCT/US2021/013738

(22) International Filing Date:

15 January 2021 (15.01.2021)

(25) Filing Language:

English

(26) Publication Language:

English

(30) Priority Data:

63/068,349 20 August 2020 (20.08.2020) US
17/148,200 13 January 2021 (13.01.2021) US

(71) Applicant: MICROCHIP TECHNOLOGY INC.
[US/US]; 2355 W. Chandler Blvd., Chandler, Arizona
85224 (US).

(72) Inventors: ZUOLO, Lorenzo; Via Marasso, 1170, 45020
Lusia (IT). MICHELONI, Rino; Via Como, 8, 22078 Tu-
rate (IT).

(74) Agent: GLASS, Kenneth; Glass & Associates, 236 N. San-
ta Cruz Ave., Suite 243, Los Gatos, California 95030 (US).

(81) Designated States (unless otherwise indicated, for every
kind of national protection available): AE, AG, AL, AM,
AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ,

CA, CH, CL, CN, CO, CR, CU, CZ, DE, DJ, DK, DM, DO,
DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN,
HR, HU, ID, IL, IN, IR, IS, IT, JO, JP, KE, KG, KH, KN,
KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD,
ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO,
NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW,
SA, SC, SD, SE, SG, SK, SL, ST, SV, SY, TH, TJ, TM, TN,
TR, TT, TZ, UA, UG, US, UZ, VC, VN, WS, ZA, ZM, ZW.

(84) Designated States (unless otherwise indicated, for every
kind of regional protection available): ARIPO (BW, GH,
GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ,
UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ,
TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK,
EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV,
MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM,
TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW,
KM, ML, MR, NE, SN, TD, TG).

Published:

— with international search report (Art. 21(3))

(54) Title: PARTITIONABLE NEURAL NETWORK FOR SOLID STATE DRIVES

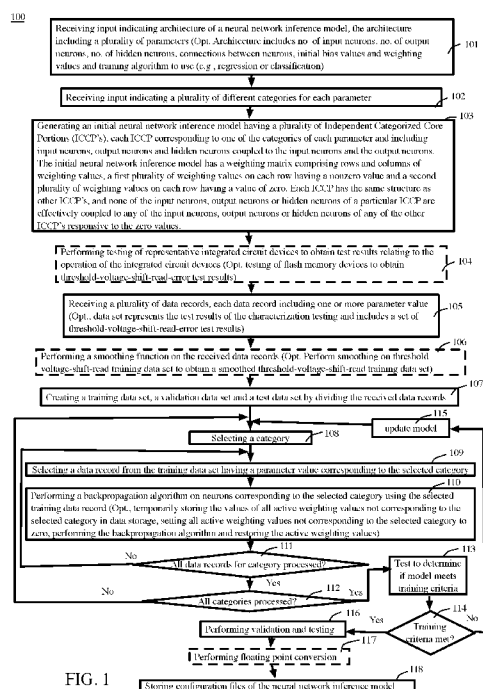


FIG. 1

(57) Abstract: A method includes storing configuration files of a Multi-Core Neural Network Inference (MCNNI) model having Independent Categorized-Core-Portions (ICCP's). Each ICCP corresponds to one of a plurality of categories for each parameter. A first plurality of weighting values on each row of the weighting matrix of the MCNNI model have a nonzero value and a second plurality of weighting values on each row having a value of zero. The configuration files are loaded into a neural network engine. The operation of the integrated circuit device is monitored to identify a usage value corresponding to each of the parameters. A single neural network operation is performed using the usage values as input to generate, at the output neurons of each ICCP, output values indicating an estimation of one or more variable. The output values of the ICCP that corresponds to the input usage values are identified and are sent as output.

Partitionable Neural Network for Solid State Drives

CROSS-REFERENCE TO RELATED APPLICATIONS

[001] The present application claims priority to United States Provisional Patent Application Serial No. 63/068,349 filed on August 20, 2020 and United States Non-Provisional Patent Application Serial No. 17/148,200 filed on January 13, 2021, the contents of each of which are incorporated by reference herein in their entirety.

BACKGROUND

[002] Neural networks are effective for identifying solutions to questions relating to complex systems. However, when the complex system includes a large number of factors that affect the outcome of the inquiry and/or the complexity of the training data set requires significant numbers of hidden neurons to achieve accurate results, the size of the network becomes so large that it is impractical to implement it in a stand-alone system such as a dedicated integrated circuit device for performing a specific function. In particular, the significant number of hidden neurons of the neural network dictates the use of a neural engine with so many gates that implementation on a single integrated circuit device becomes impractical because of the required semiconductor surface area, power requirements, cooling requirements, cost, inference latency, configuration file space, etc.

[003] One method for solving this problem is to categorize each factor and to generate a different inference model for each category. Though this type of solution allows for processing of neural network operations on a single integrated circuit device, this solution requires a significant number of neural network configuration files, and requires that the configuration files be loaded into the neural network engine prior to each neural network operation. The required loading of many configuration files consumes power and adds to inference latency.

[004] What is needed is a way to partition neural networks so that the same physical neural

network can be used to generate multiple predictions of one or more output variable in parallel. Also, there is a need for a method and apparatus for identifying the best threshold voltage shift offset value to use in performing reads of a Solid State Drive (SSD).

[005] The methods and apparatus of the present invention provide for partitioning neural networks such that the same physical neural network can be used to generate multiple predictions of one or more output variable in parallel. In addition, the methods and apparatus of the present invention allows for identifying the best threshold voltage shift offset value to use in performing reads of an SSD.

SUMMARY OF THE INVENTION

[006] A method for generating a neural network inference model includes: receiving input indicating architecture of a neural network inference model, the architecture including a plurality of parameters; and receiving input indicating a plurality of different categories for each parameter. An initial neural network inference model is generated that includes a plurality of Independent Categorized-Core-Portions (ICCP's), each ICCP corresponding to one of the categories of each parameter and including input neurons, output neurons and hidden neurons coupled to the input neurons and the output neurons. The initial neural network inference model has a weighting matrix comprising rows and columns of weighting values, a first plurality of weighting values on each row having a nonzero value and a second plurality of weighting values on each row having a value of zero. Each ICCP has the same structure as other ICCP's, and none of the input neurons, output neurons or hidden neurons of a particular ICCP are effectively coupled to any of the input neurons, output neurons or hidden neurons of any of the other ICCP's responsive to the zero values. A plurality of data records are received, each data record including one or more parameter value. A training data set, a validation data set and a test data set are created by dividing the received data records. The initial neural network inference model is trained by: selecting a category; selecting a data record from the training data set having a parameter value corresponding to the selected category; performing a backpropagation algorithm on neurons corresponding to the selected category using the selected training data record;

repeating the selecting a data record and the performing a backpropagation algorithm until all data records in the training data set corresponding to the selected category have been processed; continuing the selecting a category, selecting a data record, performing a backpropagation algorithm and the repeating until all categories have been processed. Configuration files of the neural network inference model are then stored.

[007] In one example the output neurons of each ICCP are configured to generate an output value that indicates a number of errors of a Threshold-Voltage-Shift Read-Error (TVS-RE) curve at a particular Threshold Voltage Shift Offset (TVSO) value. In another example, the output neurons of each ICCP are configured to generate an output value indicating a coefficient of a Threshold-Voltage-Shift Read-Error (TVS-RE) curve.

[008] A method for performing an inference operation includes storing configuration files of a neural network inference model on an integrated circuit device, the neural network inference model including a plurality of ICCP's, each ICCP corresponding to one of a plurality of categories for each parameter of the neural network inference model and including input neurons, output neurons and hidden neurons coupled to the input neurons and the output neurons, the neural network inference model having a weighting matrix comprising rows and columns of weighting values, a first plurality of weighting values on each row having a nonzero value and a second plurality of weighting values on each row having a value of zero. Each ICCP has the same number of input neurons, the same number of hidden neurons, the same number of output neurons, and the same connections between input neurons, hidden neurons and output neurons as the other ICCP's, and none of the input neurons, output neurons or hidden neurons of a particular ICCP effectively coupled to any of the input neurons, output neurons or hidden neurons of any of other ICCP responsive to zero values of the weighting matrix. The configuration files of the neural network inference model are loaded into a neural network engine of the integrated circuit device to form a neural network inference core. The operation of the integrated circuit device is monitored to identify a usage value corresponding to each of the parameters. A single neural network operation is performed on the neural network inference core using the usage values as input to each of the ICCP's to generate, at the output neurons of each ICCP output values

indicating an estimation of one or more variable. The output values of the ICCP that correspond to the input usage values are identified. The identified output values are then sent as output.

[009] An integrated circuit device is disclosed that includes operating circuitry for performing one or more operations and generating values relating to the one or more operations and a data storage configured for storing a Multi-Core Neural Network Inference (MCNNI) model. The MCNNI model including a plurality of Independent Categorized-Core-Portions (ICCP's), each ICCP corresponding to one of a plurality of categories for each parameter in the MCNNI model and including input neurons, output neurons and hidden neurons coupled to the input neurons and the output neurons, each ICCP having the same number of input neurons, the same number of hidden neurons, the same number of output neurons, and the same connections between input neurons, hidden neurons and output neurons as the other ICCP's, and none of the input neurons, output neurons or hidden neurons of a particular ICCP coupled to any of the input neurons, output neurons or hidden neurons of any of the other ICCPs. The neural network inference model having a weighting matrix comprising rows and columns of weighting values, a first plurality of weighting values on each row having a nonzero value and a second plurality of weighting values on each row having a value of zero. The integrated circuit device is configured to identify usage values corresponding to the operations of the integrated circuit device. A neural network engine is coupled to the data storage, wherein the integrated circuit device is configured to load the MCNNI model into the neural network engine to form a MCNNI core and the neural network engine is configured for performing a single neural network operation of the MCNNI core using the usage values as input to the single neural network operation to generate, for each ICCP an output value at each output neuron that corresponds to the categories represented by the particular ICCP core. The integrated circuit device includes a selection circuit configured for identifying the MCNNI core corresponding to the usage values input into the single neural network operation. The selection circuit is further configured to send the output values generated by the output neurons of the selected MCNNI core to one or more other circuit of the integrated circuit device.

[0010] The method and apparatus of the present invention models multiple factors that affect

UBER and generates, on the flash controller, output that indicates a minimum value of a TVS-RE curve that accurately reflects the physical characteristics of the location that is to be read, both the structural characteristics of the location that is to be read (e.g., the wordline and the block that is to be read), the current age/physical degradation of the location that is to be read (e.g., the number of P/E cycles) and the current transitory characteristics of the location that is to be read (e.g., closed-block read disturb and closed-block retention time). Because the generated TVS-RE curve accurately represents the factors that affect UBER, the generated TVSO value corresponding to the minimum value of the TVS-RE curve will be the appropriate TVSO value for performing the read. Using the best TVSO value for performing the read results in a reduced number of read errors at the flash controller. The reduced number of read errors of the method and apparatus of the present invention effectively maintains UBER within acceptable levels during the lifetime of the SSD.

BRIEF DESCRIPTION OF THE DRAWINGS

[0011] The accompanying drawings are included to provide a further understanding of the invention, and are incorporated in and constitute a part of this specification. The drawings illustrate embodiments of the invention and, together with the description, serve to explain the principles of the invention.

[0012] FIG. 1 is block diagram illustrating a method for generating a neural network inference model in accordance with an embodiment of the present invention.

[0013] FIGS. 2-3 are diagrams illustrating an exemplary neural network inference model and how a Multi-Core Neural Network Inference (MCNNI) model can be generated in accordance with an embodiment of the present invention.

[0014] FIG. 4 is a diagram illustrating an MCNNI model in accordance with an embodiment of the present invention.

[0015] FIG. 5 is a diagram illustrating a weighting matrix of an MCNNI model in accordance with an embodiment of the present invention.

[0016] FIG. 6 is block diagram illustrating a method for performing a neural network operation on an integrated circuit device in accordance with an embodiment of the present invention.

[0017] FIG. 7 is a block diagram illustrating an SSD in accordance with an embodiment of the present invention.

[0018] FIG. 8 is a diagram illustrating a flash controller and an exemplary NAND flash memory device and illustrates the communication between the flash controller and the flash memory device in accordance with an embodiment of the present invention.

[0019] FIG. 9 is a diagram illustrating a testing and model generation system in accordance with an embodiment of the present invention.

[0020] FIG. 10 is a diagram illustrating wordline categories for an MCNNI model in accordance with an embodiment of the present invention.

[0021] FIG. 11 is a diagram illustrating categories for an MCNNI model in accordance with an embodiment of the present invention.

[0022] FIG. 12 is a graph illustrating a number of errors on the vertical axis and threshold voltage shift offset on the horizontal axis that illustrates errors in reading an exemplary flash memory device in accordance with an embodiment of the present invention.

[0023] FIG. 13 is a graph illustrating exemplary threshold-voltage-shift read-error curves generated from an exemplary threshold-voltage shift-read training data set in accordance with an embodiment of the present invention.

[0024] FIG. 14 is a graph illustrating exemplary smoothed threshold-voltage-shift read-error curves generated from an exemplary smoothed threshold-voltage shift-read training data set in accordance with an embodiment of the present invention.

[0025] FIGS. 15-19 are diagrams illustrating a multi-category neural network inference models in accordance with embodiments of the present invention.

DETAILED DESCRIPTION

[0026] Fig. 1 illustrates a method (100) for generating a neural network inference model that includes receiving input (101) indicating architecture of a neural network inference model, the architecture including a plurality of parameters. In one example, the architecture includes one or more file that indicates parameters for the neural network model. The architecture may also include hyperparameters for the neural network model and may include the number of input neurons, the number of hidden neurons, the number of output neurons, the connections between neurons, initial bias values and weighting values and the training algorithm to use (e.g., a particular regression algorithm or a classification algorithm).

[0027] Input indicating a plurality of different categories for each parameter is received (102). In one example, an electronic file is received as input indicating categories for each parameter. Alternatively, categories for each parameter are input via a graphical user interface of a computer that is being used to perform one or more steps of method 100.

[0028] An initial neural network inference model is generated (103) having a plurality of ICCP's, each ICCP corresponding to one of the categories of each parameter and including input neurons, output neurons and hidden neurons coupled to the input neurons and the output neurons. The initial neural network inference model has a weighting matrix comprising rows and columns of weighting values, a first plurality of weighting values on each row having a nonzero value and a second plurality of weighting values on each row having a value of zero. Each ICCP has the same structure as other ICCP's, and none of the input neurons, output neurons or hidden neurons of a particular ICCP are effectively coupled to any of the input neurons, output neurons or hidden neurons of any of the other ICCP's responsive to the zero values. Physically a path may potentially exist between the input neurons, output neurons or hidden neurons of a particular ICCP and output neurons or hidden neurons of other ICCP's, however due to the zero weighting value they are not effectively coupled.

[0029] The mathematical operation of method 100 is illustrated in FIGs. 2-3 that show input parameter values received at particular input neurons as x_0-x_s , where parameters a, b, c and d are

weights to generate output values y_0 - y_7 representing the value of hidden neurons (not shown) plus a bias value B_0 - B_7 . This is illustrated by input parameter values x_0 - x_5 multiplied by weighting values corresponding to parameters a, b, c and d of a weighting index that includes weighting values on each. The dotted line is a virtual line separating the top portion from the bottom portion, and is implemented by setting weighting values to zero in the appropriated row/columns of the weighting index. Setting the weighting values to zero thus separates the interconnected neural network into separate ICCPs.

[0030] In one example, the outputs of a Multi-Core Neural Network Inference (MCNNI) model are generated in accordance with the following exemplary equations, where the variable “y” represents the output value of each ICCP, the variables “x” represent inputs to each ICCP that may be referred to as “input parameter values”, the variables a, b, c and d represent parameters (that may also be referred to as “weighting values”) and the variables B_0 - B_7 represent bias values.

$$[0031] \quad y_0 = (x_0 * a) + (x_1 * a) + (x_2 * a) + (x_3 * 0) + (x_4 * 0) + (x_5 * 0) + B_0$$

$$[0032] \quad y_1 = (x_0 * b) + (x_1 * b) + (x_2 * b) + (x_3 * 0) + (x_4 * 0) + (x_5 * 0) + B_1$$

$$[0033] \quad y_2 = (x_0 * c) + (x_1 * c) + (x_2 * c) + (x_3 * 0) + (x_4 * 0) + (x_5 * 0) + B_2$$

$$[0034] \quad y_3 = (x_0 * d) + (x_1 * d) + (x_2 * d) + (x_3 * 0) + (x_4 * 0) + (x_5 * 0) + B_3$$

$$[0035] \quad y_4 = (x_0 * 0) + (x_1 * 0) + (x_2 * 0) + (x_3 * a) + (x_4 * a) + (x_5 * a) + B_4$$

$$[0036] \quad y_5 = (x_0 * 0) + (x_1 * 0) + (x_2 * 0) + (x_3 * b) + (x_4 * b) + (x_5 * b) + B_5$$

$$[0037] \quad y_6 = (x_0 * 0) + (x_1 * 0) + (x_2 * 0) + (x_3 * c) + (x_4 * c) + (x_5 * c) + B_6$$

$$[0038] \quad y_7 = (x_0 * 0) + (x_1 * 0) + (x_2 * 0) + (x_3 * d) + (x_4 * d) + (x_5 * d) + B_7$$

[0039] Regression neural network models typically refer to the variable representing input into each input neuron (represented by the variable “ x_n ” in the above equation) as a “parameter” and such terminology is used in the present application, with the input values received at a particular input neuron referred to as “input parameter values.” However, the methods and apparatus of the present invention may also use other types of neural networks. In some of these other types of

neural networks, the input into each input neuron is referred to as a “feature.” The term “parameter” as used in the present application shall not be construed as being limited to receiving input relating to a regression neural network, but shall include “features” when such term is applicable.

[0040] In one example step 103 includes generating an ICCP for each combination of the received categories. In the following examples, independent core portions are indicated with subscripts in which the first subscript indicates the categories of the first parameter, the second subscript indicates the category of the second parameter, and so forth. For example, in an embodiment in which there are three categories for a first parameter and two categories for a second parameter that is illustrated in FIG. 4, six ICCP’s will be generated to form MCNNI model 40. The six ICCP’s include a first ICCP corresponding the first category of the first parameter and the first category of the second parameter ($ICCP_{11}$), a second ICCP corresponding to the first category of the first parameter and the second category of the second parameter ($ICCP_{12}$), a third ICCP corresponding to the second category of the first parameter and the first category of the second parameter ($ICCP_{21}$), a fourth ICCP corresponding to the second category of the first parameter and the second category of the second parameter ($ICCP_{22}$), a fifth ICCP corresponding to the third category of the first parameter and the first category of the second parameter ($ICCP_{31}$) and a sixth ICCP corresponding to the third category of the first parameter and the second category of the second parameter ($ICCP_{32}$). MCNNI model 40 includes a plurality of input neurons 41, a plurality of output neurons 45 and layers 42-44 of hidden neurons. It can be seen that each ICCP in MCNNI model 40 has the same structure as the other ICCP’s, and none of the input neurons, output neurons or hidden neurons of a particular ICCP are effectively coupled to any of the input neurons, output neurons or hidden neurons of any of the other ICCP’s. The term “not effectively coupled” or “none of... are effectively coupled” as used in the present application shall mean either 1) not physically coupled or 2) physically coupled, but physically coupled in such a way that the “0” values in the weighting matrix make the physical coupling inoperable.”

[0041] Input neurons 41 of $ICCP_{11}$ receive input parameter values x_0 , x_1 , x_2 , x_3 and x_4 . Input

neurons 41 of ICCP₁₂ receive input parameter values x_5, x_6, x_7, x_8 and x_9 . Input neurons 41 of ICCP₂₁ receive input parameter values $x_{10}, x_{11}, x_{12}, x_{13}$ and x_{14} . Input neurons 41 of ICCP₂₂ receive input parameter values $x_{15}, x_{16}, x_{17}, x_{18}$ and x_{19} . Input neurons 41 of ICCP₃₁ receive input parameter values $x_{20}, x_{21}, x_{22}, x_{23}$ and x_{24} . Input neurons 41 of ICCP₃₂ receive input parameter values $x_{25}, x_{26}, x_{27}, x_{28}$ and x_{29} .

[0042] FIG. 5 is a diagram illustrating a weighting matrix of a MCNNI model in accordance with an embodiment of the present invention. Weighting values having a nonzero value (which may be referred to hereinafter as active weighting values) form the various inference cores in the MCNNI model. More particularly, the weighting values (a, b, c, d) in rows 1-4 and columns 1-4 form the inference core for a first ICCP₁₁, the weighting values (a, b, c, d) in rows 5-8 and columns 5-8 form the inference core for a second ICCP₁₂ and so forth. It can be seen that all weighting values on each row that are not active weighting values have a value of zero. More particularly, the first four columns and the first n rows (where n is four) include active values for first ICCP₁₁, the next four columns and n rows include active values for second ICCP₁₂, the next four columns and n rows include active values for third ICCP₂₁, the next four columns and n rows include active values for fourth ICCP₂₂, the next four columns and n rows include active values for fifth ICCP₃₁ and the last four columns and n rows include active values for sixth ICCP₃₂.

[0043] Optionally testing of representative integrated circuit devices is performed (104) to obtain test results relating to the operation of the integrated circuit devices as will be discussed in further detail below.

[0044] A plurality of data records are received (105), each data record including one or more parameter value. The data records are labeled examples, each of which includes a plurality of features (a.k.a attributes), some or all of which may be parameters of the model (e.g., the variables " x_i ", in the model that is received at a particular input neuron) and a corresponding label (e.g., a " y " value), i.e. the output of the respective ICCP.

[0045] Optionally a smoothing function may be performed on the data records (106) as will be discussed in detail below.

[0046] A training data set, a validation data set and a test data are created (107) set by dividing the received data records. In one example the received data records are divided randomly into the respective data sets.

[0047] The initial neural network inference model generated in 103 is trained by selecting a category (108), selecting a data record (109) from the training data set having a parameter value corresponding to the selected category and performing a backpropagation algorithm (110) on neurons corresponding to the selected category using the selected training data record. In the above example in which there are three categories for a first parameter and two categories for a second parameter, if the selected category is the first category the neurons of ICCP's belonging to the first category of the first parameter will be trained. More particularly, the neurons of ICCP₁₁ and ICCP₁₂ will be trained using the selected data record. The selecting a data record (109) and the performing a backpropagation algorithm (110) are repeated until all data records in the training data set corresponding to the selected category have been processed (111).

[0048] Steps 108-111 are repeated until all categories have been processed (112). Thus, in the above example, the process of steps 108-111 is repeated a total of five times, once for each category. In this example, a first iteration of steps 108-111 will be performed to train the neurons of the ICCP₁₁ and ICCP₁₂ (those that are responsive to input parameter values x_0-x_9 , i.e. those neurons for which the weighting value for these parameter values have not been set to zero) using data records that correspond to the first category of the first parameter. A second iteration of steps 108-111 will be performed to train the neurons of ICCP₂₁ and ICCP₂₂ (those that are responsive to input parameter values $x_{10}-x_{19}$, i.e. those neurons for which the weighting value for these parameter values have not been set to zero) using data records that correspond to the second category of the first parameter. A third iteration of steps 108-111 will be performed to train the neurons of ICCP₃₁ and ICCP₃₂ (those that are responsive to input parameter values input neurons $x_{20}-x_{29}$, i.e. those neurons for which the weighting value for these parameter values have not been set to zero) using data records corresponding to the third category of the first parameter. A fourth iteration of steps 108-111 is performed to train the neurons of ICCP₁₁ and ICCP₂₁ and ICCP₃₁ using data records corresponding to the first category of the second parameter. The fifth

iteration of steps 108-111 trains the neurons of ICCP₁₂, ICCP₂₂, and ICCP₃₂ using data records corresponding to the second category of the second parameter.

[0049] In one example step 110 includes temporarily storing all weighting values in the first plurality of weighting values not corresponding to the ICCP that is being trained in data storage (so that they will not be altered by the backpropagation); performing the backpropagation algorithm; and then restoring each of the weighting values in the other ICCP's back to its respective temporarily stored value after the backpropagation algorithm has been performed. By storing and replacing weighting values during training only the ICCP to be trained is modified during the training iteration.

[0050] The algorithm used for training may be a regression algorithm so as to identify coefficients or values relating to a curve that represents the training data. Alternatively, a classification algorithm or linear regression algorithm may be used.

[0051] The resulting trained neural network inference model is tested (113) to determine whether the trained neural network inference model meets training criteria, and if the training criteria are not met (114-115) the model is updated (115) and the process of steps 108-114 is repeated. The criteria used in steps 113 may be a cost function that determines "the goodness" of the training process. In one example, inference operations are performed using data records in the threshold-voltage-shift-read training data set and a Min-Squared-Error (MSE) function is used to identify the MSE achieved by training using the training data set.

[0052] When the training criteria are met in step 114 validation and testing of the trained neural network inference model are performed (116).

[0053] Optionally a floating point conversion is performed (117). In one example step 117 includes converting the configuration files of the MCNNI model from floating point to fixed point, e.g. 40 bits fixed point to speed up the inference time of the resulting final MCNNI model.

[0054] Configuration files of the trained, validated and tested neural network inference model are stored (118).

[0055] FIG. 6 illustrates a method for performing an inference operation that includes generating an MCNNI model (201). The MCNNI model may be formed in accordance with method 100 of FIG. 1. The configuration files of the MCNNI model are stored (202) on an integrated circuit device. The MCNNI model generated (201) and stored (202) includes a plurality of ICCP's, each ICCP corresponding to one of a plurality of categories for each parameter of the neural network inference model and including input neurons, output neurons and hidden neurons coupled to the input neurons and the output neurons, the neural network inference model having a weighting matrix comprising rows and columns of weighting values, a first plurality of weighting values on each row having a nonzero value and a second plurality of weighting values on each row having a value of zero. Each ICCP has the same number of input neurons, the same number of hidden neurons, the same number of output neurons, and the same connections between input neurons, hidden neurons and output neurons as the other ICCP's, and none of the input neurons, output neurons or hidden neurons of a particular ICCP effectively coupled to any of the input neurons, output neurons or hidden neurons of any of other ICCP responsive to zero values of the weighting matrix. Physically a path may potentially exist between the input neurons, output neurons or hidden neurons of a particular ICCP and output neurons or hidden neurons of other ICCP's, however due to the zero weighting values they are not effectively coupled.

[0056] The configuration files of the neural network inference model are loaded (203) into a neural network engine of the integrated circuit device to form a neural network inference core. The operation of the integrated circuit device is monitored (204) to identify a usage value corresponding to each of the parameters. A single neural network operation is performed (206) on the neural network inference core using the usage values as input to each of the ICCP's to generate, at the output neurons of each ICCP output values indicating an estimation of one or more variable.

[0057] Optionally the neural network operation is performed in response to receiving a read instruction (205) at the integrated circuit device as will be further discussed below.

[0058] In the neural network operation, a corresponding input neuron of each core receives the

same input as a corresponding input neuron of each other core. In the example shown in FIG. 4, input parameter values $X_0, X_5, X_{10}, X_{15}, X_{20}$ and X_{25} each represent the same first usage value as input, input parameter values $X_1, X_6, X_{11}, X_{16}, X_{21}$ and X_{26} each represent the same second usage value as input, input parameter values $X_2, X_7, X_{12}, X_{17}, X_{22}$ and X_{27} each represent the same third usage value as input and input parameter values $X_3, X_8, X_{13}, X_{18}, X_{23}$ and X_{28} each represent the same fourth usage value as input, and input parameter values $X_4, X_9, X_{14}, X_{19}, X_{24}$ and X_{29} each represent the same fifth usage value as input.

[0059] In the neural network operation, a corresponding output neuron of each core generates an output representing the same output variable as a corresponding output neuron of each other core. In the example shown in FIG. 4, output neurons 45 having output values $Y_0, Y_4, Y_8, Y_{12}, Y_{16}$ and Y_{20} represent a generated first output variable, output neurons 45 having output values $Y_1, Y_5, Y_9, Y_{13}, Y_{17}$ and Y_{21} represent a generated second output variable, output neurons 45 having output values $Y_2, Y_6, Y_{10}, Y_{14}, Y_{18}$ and Y_{22} represent a generated third output variable and output neurons 45 having output values $Y_3, Y_7, Y_{11}, Y_{15}, Y_{19}$ and Y_{23} represent a generated fourth output variable. In one example the first output variable is a prediction of a first coefficient representing a curve, the second output variable is a prediction of a second coefficient of the curve, and so forth.

[0060] The output values of the ICCPs are identified (207) that correspond to the input usage values and the identified output values are sent (208) as output. Since each combination of categories is represented in a single ICCP, the ICCP corresponding to the input usage values may be determined by identifying which ICCP corresponds to the categories of each of the usage values input into the single neural network operation. In one example a lookup table is stored that includes usage values for each category and corresponding index values identifying one of ICCP's. Step 207 includes performing a lookup operation using the usage values input into the single neural network operation to identify the corresponding ICCP.

[0061] In the example shown in FIG. 4, if the first usage value is within the first category and the second usage value is within the first category, the output of core ICCP₁₁ is identified in step 207 and output in step 208 (output values Y_0 - Y_3), if the first usage value is within the first category

and the second usage value is within the second category, the output of core ICCP₁₂ is identified in step 207 and output in step 208 (output values Y₄-Y₇).

[0062] Optionally a minimum of a TVSO curve is identified (TVSO-min value) is identified (209) using the identified output values and a read of a flash memory device will be performed (210) using the identified TVSO-min value as will be further discussed below.

[0063] FIG. 7 shows an exemplary SSD 10 that includes a flash controller 3 coupled to a plurality of flash memory devices 9 for storing data. In the present embodiment, the flash memory devices 9 are NAND devices and SSD 10 includes one or more circuit boards onto which a host connector receptacle 14, flash controller 3 and flash memory devices 9 are attached. SSD 10 may also include one or more memory device 13 such as a Dynamic Random Access Memory (DRAM), that may be a separate integrated circuit device attached to a circuit board on which flash controller 3 is mounted, and electrically coupled directly to flash controller 3.

[0064] Flash controller 3 is configured to receive read and write instructions from a host computer through host connector receptacle 14, and to perform program operations, erase operations and read operations on memory cells of flash memory devices 9 to complete the instructions from the host computer. For example, upon receiving a write instruction from the host computer via host connector receptacle 14, flash controller 3 is operable to store data in SSD 10 by performing program operations (and when required, erase operations) to program codewords on to one or more flash memory device 9.

[0065] Flash controller 3 includes a write circuit 8, a read circuit 6, a decoder 7, a status circuit 5, a minimum finder 17 and a data storage 4. Flash controller 3 also includes a neural network engine 11 coupled to the data storage 4 and a selection circuit 16. Flash controller 3 may optionally include one or more selection table 15 and a TVSO-min table 15a coupled to selection circuit 16, optionally included within data storage 4. Read circuit 6 is coupled to decoder 7, status circuit 5, selection circuit 16, data storage 4 and neural network engine 11. Status circuit 5 is coupled to write circuit 8, neural processing module 11, selection circuit 16 and data storage 4. Selection circuit 16 may be a dedicated circuit or may include a combination of software and

hardware that is operable to communicate with one or more of data storage 4, status circuit 5, neural network engine 11 and read circuit 6. Minimum finder 17 is coupled to selection circuit 16 and to read circuit 6 may be a dedicated circuit or may include a combination of software and hardware. In one embodiment minimum finder 17 is a software program operable on a processor of controller 3 for receiving input and identifying a value corresponding to a minimum of a curve represented by the input. Configuration files for one or more MCNNI (e.g. MCNNI-TVSO) inference model 12 is included within data storage 4, which MCCNI model may be a TVSO model for reading flash memory devices 9. In one example, neural network engine 11 includes specialized hardware module (e.g., a specialized configurable accelerator) specifically configured to perform a neural network operation.

[0066] Each flash memory device 9 may be a packaged semiconductor die or “chip” that is coupled to flash controller 3 by conductive pathways that couple instructions, data and other information between each flash memory device 9 and flash controller 3. In the embodiment shown in FIG. 8 each flash memory device 9 (e.g., a NAND device) includes registers 31, a microcontroller 32 and a memory array 33, and is coupled to flash controller 3 by a chip enable signal line (CE#), a command latch enable signal line (CLE), a read enable signal line (RE#), an address latch enable signal line (ALE), a write enable signal line (WE#), a read/busy signal line (R/B) and input and output signal lines (DQ). Upon receiving a write instruction from a host computer, write circuit 8 is operable to encode received data into a codeword that is sent to registers 31 along with a corresponding program instruction. Microcontroller 32 is operable to perform the requested write instruction and retrieve the codeword from register 31 and store the codeword in memory array 33 by programming cells of memory array 33 (e.g., as a logical page).

[0067] In one example, each flash memory device 9 includes NAND memory cells that are organized into blocks and pages, with each block composed of NAND strings that share the same group of wordlines. Each logical page is composed of cells belonging to the same wordline. The number of logical pages within each logical block is typically a multiple of 16 (e.g. 64, 128). In the present embodiment, a logical page is the smallest addressable unit for reading from, and

writing to, the NAND memory cells of each flash memory device 9 and a logical block is the smallest erasable unit. However, it is appreciated that in embodiments of the present invention, programming less than an entire logical page may be possible, depending on the structure of the NAND array. Though flash memory devices 9 are illustrated as being NAND devices, it is appreciated that flash memory devices 9 may be any type of memory storage device that uses a threshold voltage for reading memory cells of the flash memory device 9.

[0068] In response to receiving a read instruction from the host computer, read circuit 6 is configured to perform a read of a respective flash memory device 9 using a threshold-voltage-shift read operation. More particularly, read circuit 6 is operable to send a threshold-voltage-shift read instruction 34 to flash memory device 9. Microcontroller 32 reads the respective memory array 33 in response to the threshold-voltage-shift read instruction and outputs the read results at registers 31. The read results are sent to decoder 7 that is operable to decode the read results to obtain the stored codeword.

[0069] Flash memory devices 9 may be SLC, MLC, TLC or QLC NAND devices. In the present embodiment flash memory devices 9 are capable of performing a wide range of threshold-voltage-shift reads, including reads specified by whole number offset values such as $-n \dots -2, -1, 0, +1, +2, \dots n$ without limitation.

[0070] An erased block of a flash memory device 9 may be referred to as a “free block.” When data is programmed into a block that has been erased the block is referred to as an “open block” until all pages of the block have been programmed. Once all pages of the block have been programmed the block is referred to as a “closed block” until it is again erased.

Flash Controller Example – Generating the MCNNI Model

[0071] Following is an example in which method 100 is used to generate an MCNNI model that can be used for selecting TVSO values for performing reads of flash memory devices 9, that may be referred to hereinafter as a “MCNNI-TVSO inference model”. The method is illustrated with reference to system 70 of FIG. 9. However, it is appreciated that the MCNNI-TVSO inference model could also be generated by flash controller 3. System 70 includes an input 71, an output

72, a processor 74, a data storage 79 and a minimum finder program 83 that are coupled together via a network 80. Minimum finder program 83 is a software program operable on processor 74, and may thus represent electronically readable instructions, that is operable to identify a minimum value of a function. Test and model generation system 70 includes a bench test program 82, coupled to network 80, which may represent electronically readable instructions, that is operable on processor 74 for testing representative flash memory devices 73 in such a way as to identify the number of errors that occur as the representative flash memory devices 73 age. Representative flash memory devices 73 may be inserted into one or more test fixture that couples to test and model generation system 70, and may particularly couple to network 90. Representative flash memory devices 73 are devices that are similar to, or identical to, flash memory devices 9, and may be the same type of device (e.g. the same type of NAND device), made by the same manufacturer as flash memory devices 9. Machine learning module 81, coupled to network 80, may represent electronically readable instructions, that is operable on processor 74 for generating the neural network, and may include a computer program operable on processor 74.

[0072] The architecture of the neural network inference model is received as input (101) and the plurality of different categories are received as input (102) through input 71 and temporarily stored in data storage 79. In the present embodiment, the plurality of categories for each parameter span the full range of values expected to be received as input to the inference operation relating to the particular parameter. Machine learning module 81, coupled to network 80, is operable to generate the initial MCNNI-TVSO inference model in step 103, and the architecture input into machine learning module 81 controls the generation and testing of machine learning module 81 during the process of steps 103-118.

[0073] In one embodiment, parameters include a parameter indicating a wordline of a flash memory device (a wordline value), a parameter indicating a block of a flash memory device (a block value), a parameter indicating a retention time of a flash memory device (a retention time value), a parameter indicating a number of read disturbs of a flash memory device (a read-disturb value) and a parameter indicating the number of program and erase cycles of a flash memory

device (a program/erase value). In this example, each categorized-core-portion corresponds to a category of wordline values, a category of block values, a category of retention time values, a category of read-disturb values or a plurality of periods in the life of a flash memory device (e.g., indicated by program/erase value).

[0074] Wordlines are grouped by their physical location on the chip and the boundaries between different groups are determined by running a test on exemplary semiconductor devices in a test lab to identify a custom defined metric that relates to errors of TVSO values. In an embodiment illustrated in FIG. 10 it can be seen that there are discontinuities in the test results. In the present example categories are selected such that category boundaries correspond to the discontinuities. A first group of wordlines 51 including wordlines 1-30, a second group of wordlines 52 that includes wordlines 31-80, a third group of wordlines 53 that includes wordlines 81-195, a fourth group of wordlines 54 that include wordlines 196-220, a fifth group of wordlines 55 that includes wordlines 221-250 and a sixth group of wordlines 56 that includes wordlines 250-n wherein n is the total number of wordlines. Accordingly, wordlines are divided into six different wordline categories.

[0075] In one example, block categories group blocks by their physical location on the chip (e.g., n adjoining blocks are a first category, then the next n adjoining blocks are a second category, and so on), with blocks divided into five different block categories. In one exemplary embodiment in which SSD 10 includes 128 flash memory devices 9, each of which includes 2048 blocks, blocks may be assigned block numbers 0 through 262,143 with the block value input into the single neural network operation being the block number of a page that is to be read (e.g., when a read is to be performed as shown in step 205).

[0076] Retention time is grouped into different periods based on closed-block retention time, with some or all periods having the same duration. In one example retention time is grouped into 50 periods, with each period having a duration of 40 hours (e.g., a first group has a closed-block retention time less than or equal to 40 hours, a second group has a closed-block retention time of greater than 40 hours and less than or equal to 80 hours, a third group has a closed-block

retention time of greater than 80 hours and less than or equal to 120 hours, and so on).

[0077] Read disturb is categorized based the number of closed-block reads of the block that is to be read. In one example read disturb is divided into four categories, with a first category including from 0 to 100,000 closed-block reads, a second category including from 100,001 to 200,000 closed-block reads, a third category including from 200,001 to 300,000 closed-block reads and a fourth category that is more than 300,001 closed-block reads.

[0078] In one example the lifetime of the flash memory device is grouped into periods based on the number of program and erase cycles and a category is generated for each of the periods. In one example, the periods in the life of the flash memory device are divided into seven different categories, a first category for the first 1,000 P/E cycles, a second category that is 1,001 P/E cycles to 2,000 P/E cycles, a third category that is 2,001 P/E cycles to 3,000 P/E cycles, a fourth category that is 3,001 P/E cycles to 4,000 P/E cycles, a fifth category that is 4,001 P/E cycles to 5,000 P/E cycles, a sixth category that is 5,001 P/E cycles to 6,000 P/E cycles, a seventh category that is 6,001 P/E cycles to 7,000 P/E cycles (which is the end of life of the SSD).

[0079] In the present example, step 103 generates an initial neural network inference model that is a regression neural network model, that includes ICCP's that generate output identifying a curve represented by the parameters. In one embodiment, the definition of ICCP's in the model is done in an automated manner by machine learning model 81 to define the ICCP's in the initial MCNNI-TVSO model.

[0080] Figure 11 illustrates the ICCP's of an exemplary MCNNI-TVSO model and how the ICCP's are generated. In this example, there are six wordline categories shown as W1-W6 and there are five block categories, shown as B1-B5, fifty retention time categories shown as RT1-RT50, four read disturb categories shown as RD1-RD4 and seven program/erase categories shown as PE1-PE7. It can be seen that there are a large number of combinations of these categories, each of which is implemented as an ICCP of the respective MCNNI- TVSO model. The last column represents each of the ICCP's in the MCNNI- TVSO model. More particularly the MCNNI-TVSO model includes a first core portion, shown as W1B1RT1RD1PE1 (the first

category of each parameter), a second core portion W1B1RT1RD1PE2 which is the first category of each of the first four parameters and the second program/erase category, a third core portion W1B1RT1RD1PE3 which is the first category of each of the first four parameters and the third program/erase category, and so forth so that each possible combination of categories comprises an ICCP. It is appreciated that only the first and last category of each parameter is expanded in this illustration and that all categories will be expanded in the same manner to form ICCP's.

[0081] In optional step 104 flash characterization testing is performed on representative flash memory devices 73 to obtain one or more sets of threshold-voltage-shift read test results. Numerous representative flash memory devices 73 are tested under varying conditions to obtain one or more sets of threshold-voltage-shift read test results 85 that may be stored in data storage 79. The set(s) of test results represent different characterized points or periods in the lifetime of a characterized flash memory device and different operating conditions of the characterized flash memory device as a result of different testing conditions applied during the testing of flash memory devices 73. Figure 12 illustrates a graph representing exemplary flash characterization testing for an exemplary memory cell that illustrates different TVSO values on the x-axis and the number of errors on the y axis. The result of this process, i.e. the collection of the number of errors at different VT-shift values (dots 91), is a curve, where a portion of the curve is shaped like a "valley." The MCNNI- TVSO inference model 78 of the present invention generates output that predicts a curve that includes this valley for each of the conditions simulated during the testing of step 104.

[0082] In one example step 104 of FIG. 1 includes performing tests on representative flash memory devices 73 that identify a number of errors at each of a plurality of different threshold voltage shift offsets, in which the following variables are used in the testing: a REFERENCE VALUE of the respective representative flash memory device 73 that is being tested, a BLOCK VALUE identifying the block of the page that was read in the test, a WORDLINE VALUE indicating a wordline of the page that was read in the test, a READ DISTURB VALUE indicating the number of closed-block reads of the block containing the page that was read in the

test, a RETENTION TIME VALUE indicating the closed-block retention time of the block containing the page that was read in the test and a PROGRAM/ERASE VALUE of the block containing the page that was read in the test.

[0083] In step 105 data records are received that indicate the results of the tests on representative flash memory devices 73. In one example, threshold-voltage-shift read test results 85 are received (e.g. they are received at input 71 or machine learning module 81 retrieves them from data storage 79). FIG. 13 shows an example of TVS-RE curves generated from an exemplary threshold-voltage shift-read training data set. More particularly, the number of errors is shown on the y-axis and TVSO is shown on the x-axis. As can be seen in Fig. 13, because of the nature of the NAND flash read circuitry there are fluctuations (noise) in the number of errors of each valley. This noise negatively impacts the learning process of the MCNNI-TVSO inference model which may impact the actual position of the minimum of the valley and lead to calculation of incorrect TVSO values. To avoid this problem a smoothing algorithm such as a moving average or multi-polynomial interpolation is optionally applied to the training data set as is shown by step 106 prior to training to generate a smoothed threshold-voltage-shift-read training data set (where the shape of the valleys represented by the target data set are smoothed). Figure 14 shows an example of TVS-RE curves generated from an exemplary smoothed threshold-voltage shift-read training data set. Accordingly, because the smoothing function (106) is performed prior to training, the MCNNI-TVSO inference model is not trained to predict the TVS-RE curve (e.g., rebuild the “valley”) corresponding to the exact number of errors measured during the device’s characterization, but rather is trained to predict the TVS-RE curve (e.g., rebuild the “valley”) corresponding to the smoothed threshold-voltage-shift-read training data set.

[0084] In one example, the training data set, the validation data set and the test data set include flash memory device characterization test results that indicate the number of errors in reads performed at different threshold-voltage-shift-offset values. In one example, machine learning module 81 is operable to separate (e.g., by a random process) the threshold-voltage-shift read test results 85 into a training database 75 that includes a threshold-voltage-shift-read training data set, a validation database 76 that includes a threshold-voltage-shift-read validation data set and a test

database 77 that includes a threshold-voltage-shift-read test data set, each of which are coupled to network 80.

[0085] In the present example, the first wordline category is selected in step 108 of FIG. 1 and a data record is selected in step 109 that includes a wordline value within the limits of the first wordline category (e.g., a data record having a wordline of 1-30) and a backpropagation algorithm is performed (110) on data input into input neurons 41 to train the ICCP as shown by block 61 of FIG. 10. The data record may include a number of errors for a read operation of an exemplary flash memory device, a WORDLINE VALUE identifying the wordline that was read, a BLOCK VALUE indicating the block number of the block containing the page that was read, a READ DISTURB VALUE indicating the number of closed-block reads of the block containing the page that was read, a RETENTION TIME VALUE indicating the closed-block retention time for the block containing the page that was read and a PROGRAM/ERASE value indicating a current number of program and erase cycles of the block containing the page that was read. The backpropagation algorithm may be a regression algorithm that generates a curve representing the errors at different TVSO values.

[0086] Once all data records having a wordline value within the limits of the first wordline category have been processed, the process moves to the second wordline category and the backpropagation algorithm is performed on data records that include a wordline value within the limits of the second wordline category (e.g., a data record having a wordline of 31-80) as shown by block 62. Once all data records having a wordline value within the limits of the second wordline category have been processed, the process moves to the third wordline category and the backpropagation algorithm is performed on data records that include a wordline value within the limits of the third wordline category as shown by block 63. The backpropagation algorithm is performed on data records that include a wordline value within the limits of the fourth wordline category as shown by block 64, the backpropagation algorithm is performed on data records that include a wordline value within the limits of the fifth wordline category as shown by block 65 and then the backpropagation algorithm is performed on data records that include a wordline value within the limits of the sixth wordline category as shown by block 66. Once training has

been performed using all wordline categories, data records that include block values within the limits of each category of block values are used to perform, training. The process continues in the same manner, until all block categories have been trained. Training is then performed in the same manner for each retention time category, each read disturb category and each P/E category.

[0087] In the testing of step 113 inference operations are performed using data records in the threshold-voltage-shift-read training data set and an error function, such as a Min-Squared-Error (MSE) function, without limitation, is used to identify the error achieved by training using the threshold-voltage-shift-read training data set.

[0088] When the trained MCNNI-TVSO inference model meets predetermined training criterial (114-115) the MCNNI-TVSO inference model is validated using the threshold-voltage-shift-read validation data set, stored in validation database 76, by performing inference operations using data records in the threshold-voltage-shift-read validation data set and computing the error function on the results. When the error computed by the error function is below a predetermined threshold the MCNNI-TVSO model is tested by performing inference operation using data records in the threshold-voltage-shift-read test data set, stored in test database 77, and the output of the MCNNI-TVSO model generated using the test data set is compared to optimal threshold voltage shift values, that may be referred to as “golden VT-opt” values calculated without using the MCNNI-TVSO model (e.g., are calculated by generating error curves using the test bench program 82 and using minimum finder program 83 to identify golden VT-opt values for the various testing samples). The method passes to step 117 when the predicted TVSO-min values calculated using MCNNI-TVSO model are within a range of the corresponding golden VT-opt values.

[0089] Optionally a floating point conversion is performed (117) by converting the configuration files of the MCNNI-TVSO model from floating point to fixed point, e.g. 40 bits fixed point to speed up the inference time of the resulting final MCNNI-TVSO model. In one example, step 117 produces configuration files of MCNNI-TVSO model 78 shown in FIG. 9. The conversion from floating point to fixed point enables a neural network engine 11 with a lower gate count and

reduces latency.

[0090] Configuration files of the trained, validated and tested neural network inference model are stored (118). In one example a trained, validated and tested neural network inference model 78 is stored in data storage 79 in the form of configuration files. Generation of the MCNNI model may be performed prior to sale and use of a SSD 10 and prior to delivery of a flash controller 3 to a customer for use in fabrication of a SSD 10 or it may be performed on SSD 10. If it is performed prior to sale and use of a SSD 10 and prior to delivery of a flash controller 3 to a customer for use in fabrication of a SSD 10 it may be performed using a testing and model generation system as will be discussed in more detail below.

Flash Controller Example – Method 200 and SSD 10

[0091] Following is an example of method 200 that illustrates the operation of SSD 10. In this example, a MCNNI- TVSO model is generated (201) and the configuration files for the MCNNI-TVSO model are stored (202) in data storage on flash controller 3 as configuration files 12 or on a memory device 13 that is coupled to the flash controller 3, preferably prior to delivery of flash controller 3 to a customer. Configuration files of the MCNNI-TVSO model are loaded (203) into neural network engine 11 to form a MCNNI-TVSO core.

[0092] In step 204, status circuit 5 is operable to monitor the operation of each flash memory device 9 to identify one or more usage values of flash memory devices 9. The determined usage values may be stored in registers of flash controller 3, in data storage 4 of flash controller 3 or in memory device 13. In this example, the usage values are values determined during usage of a flash memory device 9 that can affect threshold voltage distribution such as, for example, a value indicating the current age/physical degradation of the location that is to be read (e.g., the number of P/E cycles or Bit Error Rate (BER)) or indicating the current transitory characteristics of the location that is to be read (e.g., closed-block read disturb and closed-block retention time).

[0093] The usage values may include a read disturb value. In one example, each time that a block is closed, status circuit 5 is operable to count the number of reads of the block while the block is closed and the number of reads of the block while the block is closed is stored as a read

disturb value. When a block is erased the read disturb value of the erased block is reset to zero.

[0094] The usage values may include a closed-block retention time. In one example, each time that a block is closed, status circuit 5 is operable to start a timer to determine the amount of time that has elapsed since the block was closed. The elapsed time as determined by the timer at any point in time is defined as a closed-block retention time value. When a block is erased the timer for that block is stopped and the closed-block retention time value of the erased block is reset to zero.

[0095] The usage values may include a number indicating the number of program and erase operations of each block. In one example, status circuit 5 is operable for counting the number of program and erase cycles of each block of the flash memory device during the entire lifetime of the flash memory device and storing the count in data storage 4 or memory device 13, which may be a dynamic read access memory (DRAM).

[0096] The usage values may also include a BER of a block or a group of blocks. In one example status circuit 5 is operable to periodically determine a BER of each block and the BER for each block is stored as a BER value. In one example, each time that a BER value is calculated it is stored in data storage 4 or memory device 13 so as to overwrite a previously stored BER value for the particular block.

[0097] A single neural network operation is performed (206) by neural network engine 11. Optionally, the single neural network operation is performed in response to receiving a read instruction (205). In one example, upon receiving a read instruction indicating a read address corresponding to memory cells of the flash memory device, status circuit 5 is configured to identify one or more usage values corresponding to the memory cells that are to be read.

[0098] In step 206 the single neural network operation is performed on the MCNNI-TVSO core using the usage value as input to each of the ICCP's to generate, at the output neuron of each ICCP an output value indicating an estimation of the variable.

[0099] The output values of the categorized-core-portions corresponding to the input usage

values are identified (207) and are sent as output (208). In one example selection circuit 16 is operable for performing steps 207-208. In one embodiment selection circuit 16 comprises a processor that is operable for performing instructions so as to perform a lookup in selection table 15 using the input usage values to identify the core portion corresponding to the input usage values, and identify the corresponding output values that are coupled as output either to data storage 4, to minimum finder 17, or to read circuit 6. In this example selection table 15 includes input usage values and index values identifying corresponding core portions, and selection circuit 16 looks up the index value indicating the core portion to use. The index values is then used to obtain the corresponding output values, by receiving all output values (all ICCP's from the operation of step 206) from neural network engine 11 and outputting only the output values corresponding to the identified core portion. Alternatively, upon completion of the neural network operation of step 206 the output of all ICCP's is stored in data storage 4 and selection circuit 16 reads the output values of the ICCP corresponding to the identified core portion.

[00100] In one example the mathematical equation solved by each categorized-core-portions to generate the output value are equivalent and the values of the first plurality of weighting values are not the same for each categorized core portion, such that the output value indicating an estimation of the variable corresponds to the first plurality of weighting values for each categorized core portion. In one example the input neurons of each independent categorized core portion are identical to the input neurons of each of the other independent categorized-core-portions and are configured to receive the same usage values as the input neurons of each of the other independent categorized-core-portions.

[00101] When the output indicates coefficients of a TVSO-RE curve or the amount of error TVSO values, minimum finder program 83 is operable (e.g., by running the program on a processor of flash controller 3) in step 209 to find the minimum point of the curve and identify the TVSO-min corresponding to the identified minimum point in the curve.

[00102] In step 210 a flash memory device is read using a threshold-voltage-shift read operation performed using the identified TVSO-min. The single neural network operation of step 206 is

performed each time that a read instruction is received from a host computer. More particularly, in response to receiving a read instruction at host connector receptacle 14, steps 206-209 are performed so as to generate a TVSO-min value and read circuit 6 uses the TVSO-min value identified in step 209 to perform the read of step 210.

[00103] In another example steps 206-209 are performed by flash controller 3 in the background, without any trigger from the host. In this example steps 206-209 are performed by flash controller 3 upon an internal controller event such as an event by a timer or a counter (e.g., a background read) to monitor the reliability state of the flash memory devices 9 and the TVSO-min values identified in step 209 are stored in a table. In one example selection circuit 16 is operable to store the values in table 15a that includes usage characteristics and corresponding TVSO-min values, that may be referred to as a “TVSO-min table”. In this example, when the read instructed in step 205 is to be performed the usage values are identified (step 204) and the identified usage values are used by selection circuit 16 to look up the corresponding TVSO-min value to use in the read. Alternatively, the look-up process can be performed by read circuit 6.

[00104] In one example, each ICCP corresponds to a single category of wordline values, a single category of block values, a single category of retention time values, a single category of read-disturb values and a single value indicating a period in the life of a flash memory device, and a first input neuron of each ICCP is configured to receive a wordline value, a second input neuron of each ICCP is configured to receive a block value, a third input neuron of each ICCP is configured to receive a retention time value, a fourth input neuron of each ICCP is configured to receive a read-disturb value and a fifth input neuron of each ICCP is configured to receive a program/erase value.

[00105] In one example, the output generated at each output neuron of an ICCP can be represented by a mathematical equation that includes a variable corresponding to the value provided to each input neuron of the ICCP, weighting values and bias values, and wherein the output neurons of each particular ICCP solves the same equations as the output neurons of the other ICCPs such that the only difference between the output of different ICCPs is a result of

different weighting values and bias values of the particular ICCP.

[00106] Figure 15 shows a diagram of an exemplary RNN inference model 40a that includes a plurality of input neurons 41a, a plurality of output neurons 45a and layers 42-44 of hidden neurons. The input parameter values for the RNN inference model 40a includes a reference value (REFERENCE VALUE). In one embodiment the reference value is an identifying value that identifies one or more characteristic of the flash memory device that is to be read, such as for example, a manufacturer, a part number, and/or a manufacturing lot number, without limitation. By including a reference value identifying the type of flash memory device, an RNN inference model can be used that can identify a TVS-RE curve for different types of flash memory devices. In another embodiment the reference value is a read reference number that is a number specified by the manufacturer of the flash memory device that indicates which read reference voltage is to be used to read a page. For example, in one type of Flash memory device read reference numbers 1 and 5 are associated with the lower page, references 2, 4 and 6 are associated with the middle page and 3 and 7 are associated with the upper page. In one example the read reference number is one or more value identifying a read reference voltage that is used to perform a read (e.g., the read reference number(s) corresponding to to a particular read operation).

[00107] The input parameter values further include one or more value identifying a page that is to be read, which in this example includes a BLOCK VALUE identifying the block of the page that is to be read in step 205 and a WORDLINE VALUE indicating a wordline of the page that is to be read. In one exemplary embodiment in which SSD 10 includes 128 flash memory devices 9, each of which includes 2048 blocks, blocks may be assigned block numbers 0 through 262,143 with the BLOCK VALUE being the block number of the page that is to be read in step 205.

[00108] In this example, step 204 includes counting a number of closed-block reads, and an input parameter value for the neural network operation includes a value indicating the number of closed-block reads of the block containing the page that is to be read (i.e. READ DISTURB

VALUE). Step 204 also includes determining a closed-block retention time, the input for the neural network operation including a value indicating the closed-block retention time of the block containing the page that is to be read (i.e. RETENTION TIME VALUE). In addition, step 204 includes counting a number of program and erase cycles of each block of the flash memory device, and the current point in the lifetime of the flash memory device is determined to be a current number of program and erase cycles of the block containing the page that is to be read (i.e. PROGRAM/ERASE VALUE).

[00109] Output neurons 45a generate output in the form of exponent values that define a TVS-RE curve. The exemplary output of RNN inference model 40a is shown to include six exponents, shown as FIRST EXPONENT, SECOND EXPONENT, THIRD EXPONENT, FOURTH EXPONENT, FIFTH EXPONENT and SIXTH EXPONENT. It is appreciated that the RNN inference model 40a may have any number of exponents, and that the number of exponents may be altered to achieve a balance between processing time and accuracy. In this embodiment, in step 209 the exponents are coupled to minimum finder 82, that is operable to receive as input the exponents and identify the TVSO value corresponding to the minimum of the curve (TVSO_{min}).

[00110] In the example shown in FIG. 16, output neurons 45b generate values that indicate the number of errors at locations corresponding to TVSO values. In this example, output neurons 45b include an output neuron generating an output for each TVSO value (for n TVSO values), such as, for example an output neuron that predicts the error when a first TVSO value is used (PRED-ERR TVSO1), an output neuron that predicts the error when a second TVSO value is used (PRED-ERR TVSO2), an output neuron that predicts the error when a third TVSO value is used (PRED-ERR TVSO3), an output neuron that predicts the error when a fourth TVSO value is used (PRED-ERR TVSO4), an output neuron that predicts the error when a fifth TVSO value is used (PRED-ERR TVSO5) and so on, to an output neuron that predicts the error when an nth TVSO value is used (PRED-ERR TVSO_n). In one specific example, TVSO values from -32...0...+32 are used and PRED-ERR TVSO1 is the predicted error when a TVSO of -32 is used, PRED-ERR TVSO2 is the predicted error when a TVSO of -31 is used, PRED-ERR

TVSO3 is the predicted error when a TVSO of -30 is used, PRED-ERR TVSO4 is the predicted error when a TVSO of -29 is used, PRED-ERR TVSO5 is the predicted error when a TVSO of -28 is used, and PRED-ERR TVSO_n is the predicted error when a TVSO of +32 is used.

[00111] When the output is values that indicate the magnitude of the number of errors at locations corresponding to TVSO values as is shown in FIG. 16, in step 209 minimum finder program 83 need only identify which output generates the lowest error to determine the TVSO value corresponding to the minimum value of the TVSE curve since it is known which TVSO value corresponds to each output neuron. However, when the output neurons generate output in the form of exponents, minimum finder program 83 includes an algorithm for identifying where the minimum value is (e.g., where on the x-axis) and that identifies the TVSO value that corresponds to the identified minimum point on the curve (e.g., the closest whole number TVSO value to the identified minimum point on the curve).

[00112] FIG. 17 shows an exemplary MCNNI-TVSO inference Model 40c that is the same as the MCNNI-TVSO inference model 40a shown in FIG. 15 except that each ICCP includes input neurons 41b having one less input neuron than does model 40a, and it does not include a reference value as an input. FIG. 18 shows an exemplary MCNNI-TVSO inference model 40d that is the same as the MCNNI-TVSO inference Model 40b shown in FIG. 16 except that each ICCP includes input neurons 41b having one less input neuron than does model 40a, and it does not include a reference value as an input.

[00113] FIG. 19 illustrates a MCNNI-TVSO model that generates output at an output neuron 45c that indicates a Threshold Voltage Shift Offset (TVSO) value corresponding to a minimum number of errors for a Threshold-Voltage-Shift Read-Error (TVS-RE) curve (TVSO-min). Since this embodiment indicates the TVSO-min, step 209 need not be performed and minimum finder 17 is not required. Instead, the TVSO-min corresponding to the TVSO-min that corresponds to the input usage values identified in step 207 is output in step 208 to the read circuit 6 or stored in data storage 4 (e.g., in table 15a).

[00114] The method and apparatus of the present invention models multiple factors that affect

UBER and generates, on the flash controller, output identifying a TVSO_{min} of a TVS-RE curve that accurately reflects the physical characteristics of the location that is to be read, both the structural characteristics of the location that is to be read (e.g., the wordline and the block that is to be read), the current age/physical degradation of the location that is to be read (e.g., the number of P/E cycles) and the current transitory characteristics of the location that is to be read (e.g., closed-block read disturb and closed-block retention time). Because the generated TVS-RE curve accurately represents the factors that affect UBER, the TVSO_{min} will be the appropriate TVSO value for performing the read (corresponding to the TVSO value that provides a minimum number of errors during device characterization). Using the appropriate TVSO value for performing the read results in a reduced number of read errors at the flash controller. The reduced number of read errors of the method and apparatus of the present invention effectively maintains UBER within acceptable levels during the lifetime of the SSD.

[00115] Further, for purposes of discussing and understanding the embodiments of the invention, it is to be understood that various terms are used by those knowledgeable in the art to describe techniques and approaches. Furthermore, in the description, for purposes of explanation, numerous specific details are set forth in order to provide a thorough understanding of the present invention. It will be evident, however, to one of ordinary skill in the art that the present invention may be practiced without these specific details. In some instances, well-known structures and devices are shown in block diagram form, rather than in detail, in order to avoid obscuring the present invention. These embodiments are described in sufficient detail to enable those of ordinary skill in the art to practice the invention, and it is to be understood that other embodiments may be utilized and that logical, mechanical, electrical, and other changes may be made without departing from the scope of the present invention.

What is claimed is:

1. A method for generating a neural network inference model, the method comprising:

receiving input indicating architecture of a neural network inference model, the architecture including a plurality of parameters;

receiving input indicating a plurality of different categories for each parameter;

generating an initial neural network inference model having a plurality of Independent Categorized-Core-Portions (ICCP's), each ICCP corresponding to one of the categories of each parameter and including input neurons, output neurons and hidden neurons coupled to the input neurons and the output neurons, the initial neural network inference model having a weighting matrix comprising rows and columns of weighting values, a first plurality of weighting values on each row having a nonzero value and a second plurality of weighting values on each row having a value of zero, each ICCP having the same structure as other ICCP's, and none of the input neurons, output neurons or hidden neurons of a particular ICCP effectively coupled to any of the input neurons, output neurons or hidden neurons of any of the other ICCP's responsive to the zero values;

receiving a plurality of data records, each data record including one or more parameter value;

creating a training data set, a validation data set and a test data set by dividing the received data records;

training the initial neural network inference model by:

selecting a category;

selecting a data record from the training data set having a parameter value corresponding to the selected category;

performing a backpropagation algorithm on neurons corresponding to the selected category using the selected training data record;

repeating the selecting a data record and the performing a backpropagation algorithm until all data records in the training data set corresponding to the selected category have been processed;

continuing the selecting a category, selecting a data record, performing a backpropagation algorithm and the repeating until all categories have been processed; and

storing configuration files of the neural network inference model.

2. The method of claim 1 further comprising performing validation using the validation data set and performing testing using the test data set, the storing comprising storing the trained, validated and tested neural network inference model.

3. The method of claim 2 further comprising, after the performing validation and testing, performing floating point conversion to generate the configuration files.

4. The method of claim 2 further wherein the storing further comprises storing the configuration files on an integrated circuit device, where the parameters relate to the operation of the integrated circuit device and each output neuron generates an estimate of a variable that relates to the operation of the integrated circuit device.

5. The method of claim 2 further comprising:

wherein each ICCP corresponds to a single category of wordline values, a single category of block values, a single category of retention time values, a single category of read-disturb values and a single value indicating a period in the life of a flash memory device, and

wherein a first input neuron of each ICCP is configured to receive a wordline value, a second input neuron of each ICCP is configured to receive a block value, a third input neuron of each ICCP is configured to receive a retention time value, a fourth input neuron of each ICCP is configured to receive a read-disturb value and a fifth input neuron of each ICCP is configured to receive a program/erase value.

6. The method of claim 2 wherein each of the training data set, the validation data set and the test data set comprise flash memory device characterization test results that indicate the number of

errors in reads performed at different threshold-voltage-shift-offset values.

7. The method of claim 1 wherein the output neurons of each ICCP are configured to generate an output value indicating a coefficient of a threshold-voltage-shift read-error curve.

8. The method of claim 1 wherein the output neurons of each ICCP are configured to generate an output value that indicates a number of errors of a Threshold-Voltage-Shift Read-Error (TVS-RE) curve at a particular Threshold Voltage Shift Offset (TVSO) value.

9. The method of claim 1 wherein the performing a backpropagation algorithm further comprises:

temporarily storing all weighting values in the first plurality of weighting values not corresponding to the selected category in data storage;

setting all weighting values in the first plurality of weighting values not corresponding to the selected category to a value of zero;

performing the backpropagation algorithm; and

restoring each of the weighting values in the first plurality of weighting values not corresponding to the selected category back to its respective temporarily stored value.

10. The method of claim 1 wherein the output generated at each output neuron of an ICCP can be represented by a mathematical equation that includes a variable corresponding to each input neuron of the ICCP, weighting values and bias values, and wherein the output neurons of each particular ICCP solves the same equations as the output neurons of the other ICCPs such that the only difference between the output of different core portions is a result of different weighting values and bias values of the particular core portion.

11. A method for performing an inference operation comprising:

storing configuration files of a neural network inference model on an integrated circuit device, the neural network inference model including a plurality of Independent Categorized-Core-Portions (ICCP's), each ICCP corresponding to one of a plurality of categories for each

parameter of the neural network inference model and including input neurons, output neurons and hidden neurons coupled to the input neurons and the output neurons, the neural network inference model having a weighting matrix comprising rows and columns of weighting values, a first plurality of weighting values on each row having a nonzero value and a second plurality of weighting values on each row having a value of zero, each ICCP having the same number of input neurons, the same number of hidden neurons, the same number of output neurons, and the same connections between input neurons, hidden neurons and output neurons as the other ICCP's, and none of the input neurons, output neurons or hidden neurons of a particular ICCP effectively coupled to any of the input neurons, output neurons or hidden neurons of any of other ICCP responsive to zero values of the weighting matrix;

loading the configuration files of the neural network inference model into a neural network engine of the integrated circuit device to form a neural network inference core;

monitoring the operation of the integrated circuit device to identify a usage value corresponding to each of the parameters;

performing a single neural network operation on the neural network inference core using the usage values as input to each of the ICCP's to generate, at the output neurons of each ICCP, output values indicating an estimation of one or more variable;

identifying the output values of the ICCP that corresponds to the input usage values;

sending as output the identified output values.

12. The method of claim 11 further comprising generating the neural network inference model by:

receiving input indicating architecture of a neural network inference model, the architecture including a plurality of parameters;

receiving input indicating a plurality of different categories for each parameter;

generating an initial neural network inference model having the plurality of ICCP's;

receiving a data set containing data records, each data record including one or more parameter value;

creating a training data set, a validation data set and a test data set by dividing the received data records;

training the initial neural network inference model by:

selecting a category;

selecting a data record from the training data set having a parameter value corresponding to the selected category;

performing a backpropagation algorithm on neurons corresponding to the selected category using the selected training data record;

repeating the selecting a data record and the performing a backpropagation algorithm until all data records in the training data set corresponding to the selected category have been processed; and

continuing the selecting a category, selecting a data record, performing a backpropagation algorithm and the repeating until all categories have been processed.

13. The method of claim 11 further comprising:

wherein each ICCP corresponds to a single category of wordline values, a single category of block values, a single category of retention time values, a single category of read-disturb values and a single category indicating a period in the life of a flash memory device,

wherein a first input neuron of each ICCP is configured to receive a wordline value, a second input neuron of each ICCP is configured to receive a block value, a third input neuron of each ICCP is configured to receive a retention time value, a fourth input neuron of each ICCP is configured to receive a read-disturb value and a fifth input neuron of each ICCP is configured to receive a program/erase value.

14. The method of claim 13 wherein the output neurons of each ICCP are configured to generate

an output value indicating a coefficient of a threshold-voltage-shift read-error curve.

15. The method of claim 13 wherein each of the plurality of ICCPs includes one or more output neurons configured to generate an output value that indicates a number of errors of a Threshold-Voltage-Shift Read-Error (TVS-RE) curve at a particular Threshold Voltage Shift Offset (TVSO) value.

16. The method of claim 11 wherein the output generated at each output neuron of an ICCP can be represented by a mathematical equation that includes a variable corresponding to each input neuron of the ICCP, weighting values and bias values, and wherein the output neurons of each particular ICCP solves the same equations as the output neurons of the other ICCPs such that the only difference between the output of different ICCPs is a result of different weighting values and bias values of the particular ICCP.

17. An integrated circuit device comprising:

operating circuitry for performing one or more operations and generating values relating to the one or more operations;

a data storage configured for storing a Multi-Core Neural Network Inference (MCNNI) model, the MCNNI model including a plurality of Independent Categorized-Core-Portions (ICCP's), each ICCP corresponding to one of a plurality of categories for each parameter in the MCNNI model and including input neurons, output neurons and hidden neurons coupled to the input neurons and the output neurons, each ICCP having the same number of input neurons, the same number of hidden neurons, the same number of output neurons, and the same connections between input neurons, hidden neurons and output neurons as the other ICCP's, the neural network inference model having a weighting matrix comprising rows and columns of weighting values, a first plurality of weighting values on each row having a nonzero value and a second plurality of weighting values on each row having a value of zero, none of the input neurons, output neurons or hidden neurons of a particular ICCP effectively coupled to any of the input neurons, output neurons or hidden neurons of any of the other ICCPs responsive to the zero values of the weighting matrix; and

a neural network engine coupled to the data storage,

wherein the integrated circuit device is configured to identify usage values corresponding to the operations of the integrated circuit device;

wherein the integrated circuit device is configured to load the MCNNI model into the neural network engine to form a MCNNI core;

wherein the neural network engine is configured to perform a single neural network operation of the MCNNI core using the usage values as input to the single neural network operation to generate, for each ICCP an output value at each output neuron that corresponds to the categories represented by the particular ICCP core; and

wherein the integrated circuit device includes a selection circuit configured for identifying the MCNNI core corresponding to the usage values input into the single neural network operation, the selection circuit further configured to send the output values generated by the output neurons of the selected MCNNI core to one or more other circuit of the integrated circuit device.

18. The integrated circuit device of claim 17 wherein the integrated circuit device comprises a flash controller,

wherein each ICCP corresponds to a single category of wordline values, a single category of block values, a single category of retention time values, a single category of read-disturb values and a single category indicating a period in the life of a flash memory device, and

wherein a first input neuron of each ICCP is configured to receive a wordline value, a second input neuron of each ICCP is configured to receive a block value, a third input neuron of each ICCP is configured to receive a retention time value, a fourth input neuron of each ICCP is configured to receive a read-disturb value and a fifth input neuron of each ICCP is configured to receive a program/erase value.

19. The integrated circuit device of claim 17 wherein the integrated circuit device comprises a flash controller and wherein each of the plurality of ICCPs includes one or more output neurons

configured to generate an output value that indicates a number of errors of a Threshold-Voltage-Shift Read-Error (TVS-RE) curve at a particular Threshold Voltage Shift Offset (TVSO) value.

20. The integrated circuit device of claim 17 wherein the integrated circuit device comprises a flash controller and wherein the output generated at each output neuron of an ICCP can be represented by a mathematical equation that includes a variable corresponding to each input neuron of the ICCP, weighting values and bias values, and wherein the output neurons of each particular ICCP solves the same equations as the output neurons of the other ICCPs such that the only difference between the output of different ICCPs is a result of different weighting values and bias values of the particular ICCP.

100

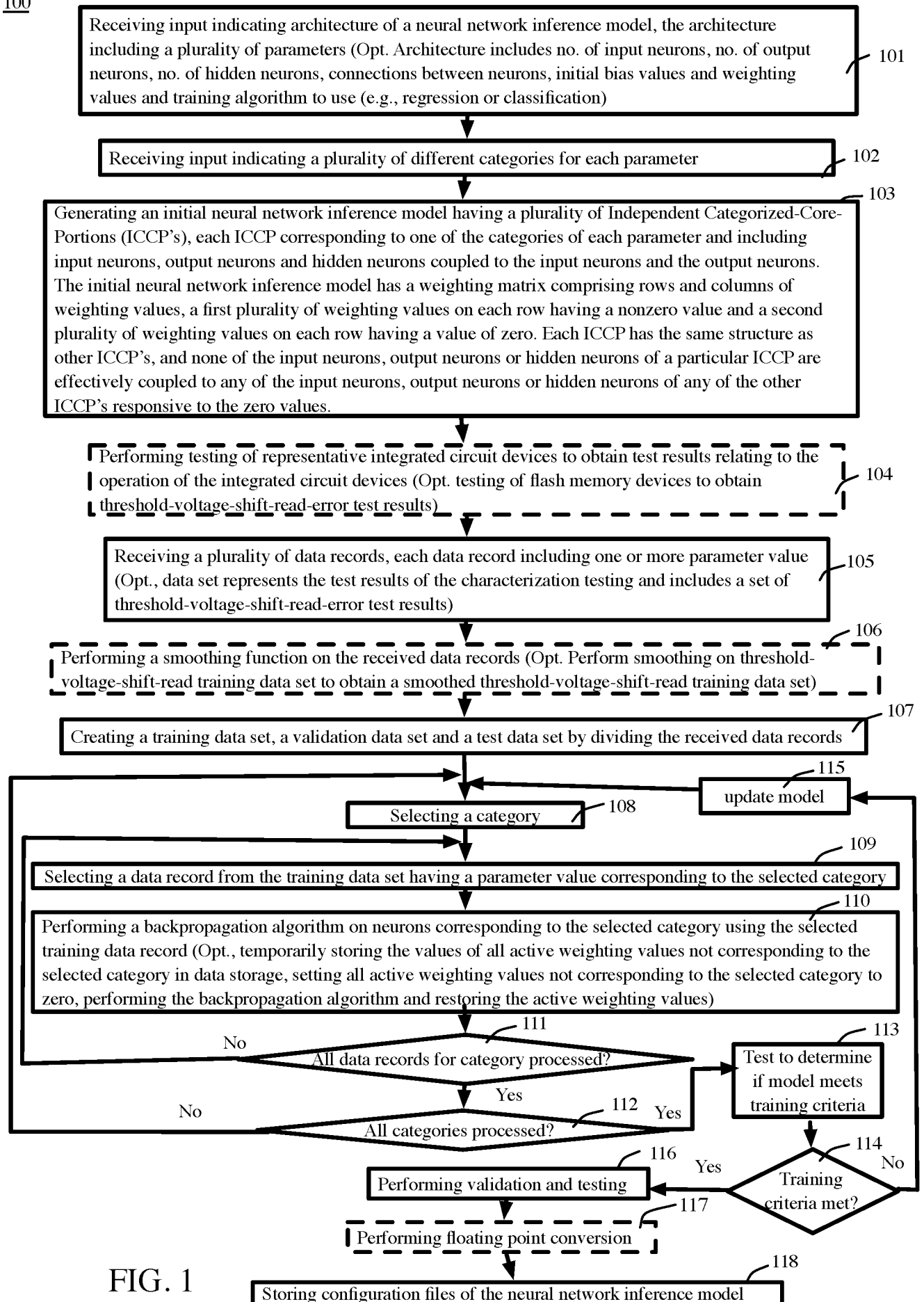


FIG. 1

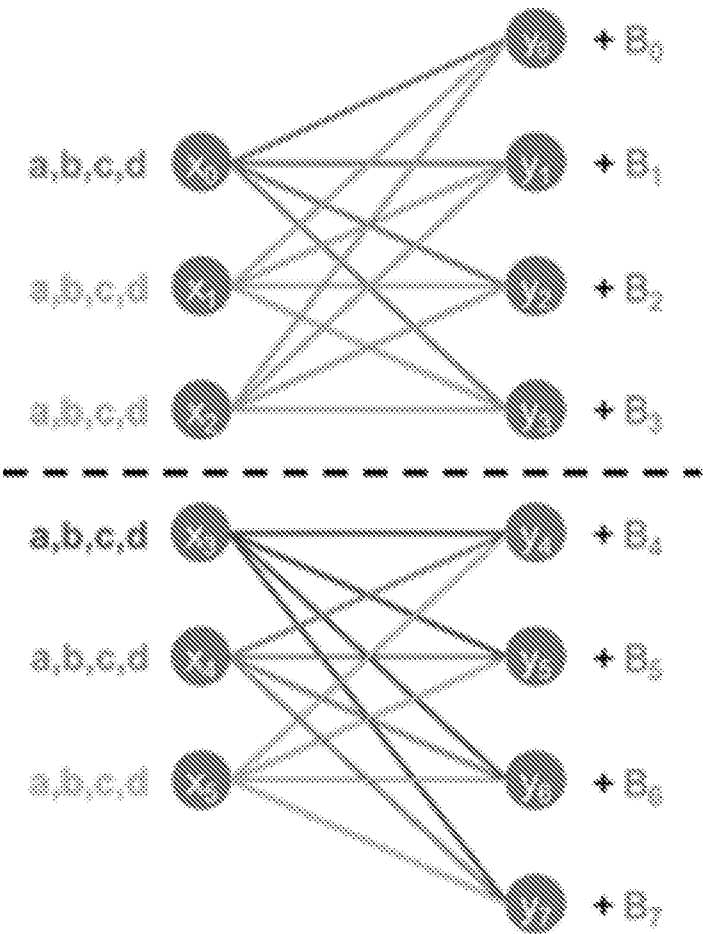


FIG. 2

$$\begin{bmatrix} \mathbf{x}_0 & \mathbf{x}_1 & \mathbf{x}_2 & \mathbf{x}_3 & \mathbf{x}_4 & \mathbf{x}_5 \end{bmatrix} \mathbf{X} \begin{bmatrix} a & b & c & d & 0 & 0 & 0 & 0 \\ a & b & c & d & 0 & 0 & 0 & 0 \\ a & b & c & d & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & a & b & c & d \\ 0 & 0 & 0 & 0 & a & b & c & d \\ 0 & 0 & 0 & 0 & a & b & c & d \end{bmatrix}$$

FIG. 3

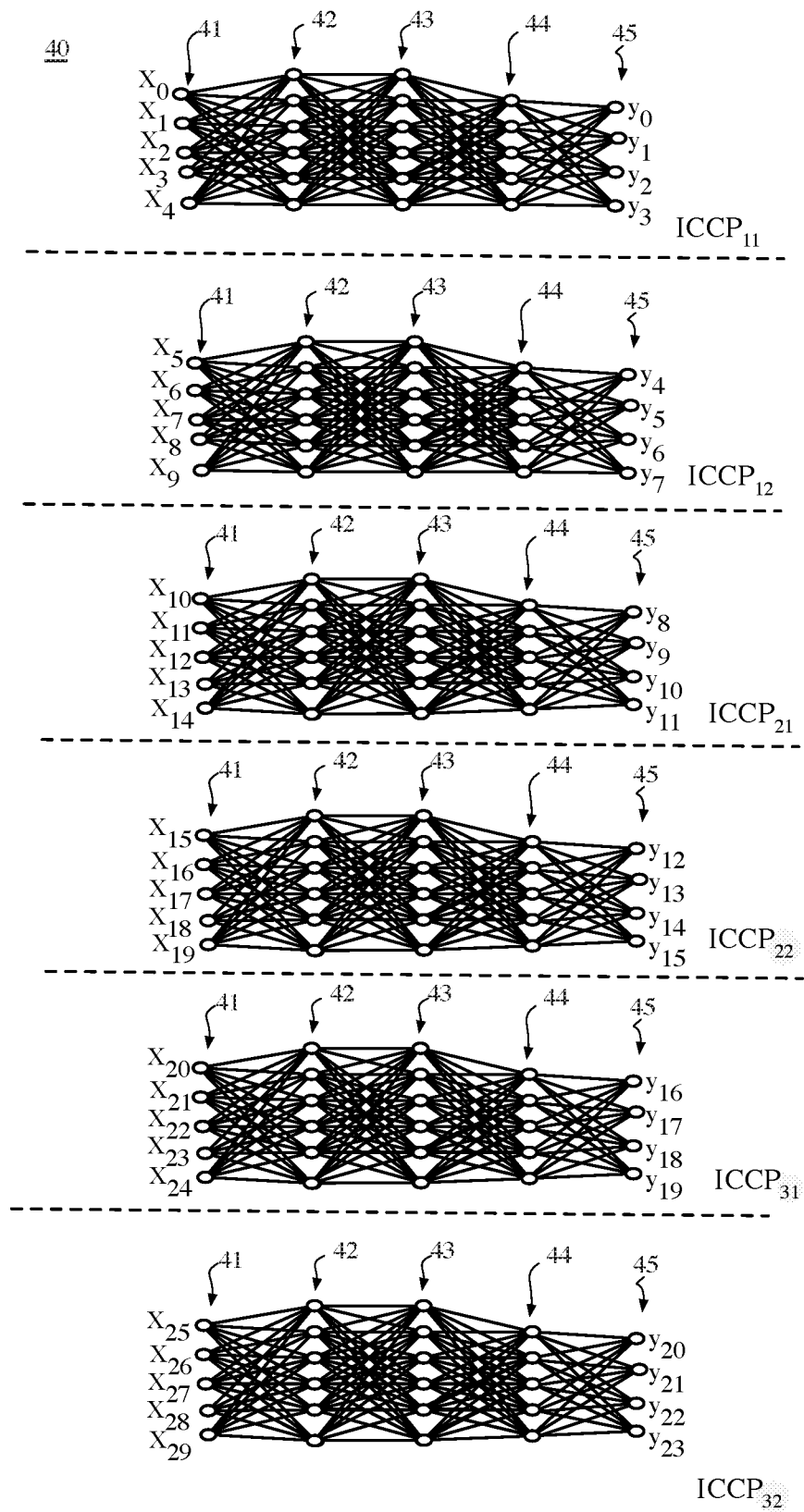


FIG. 4

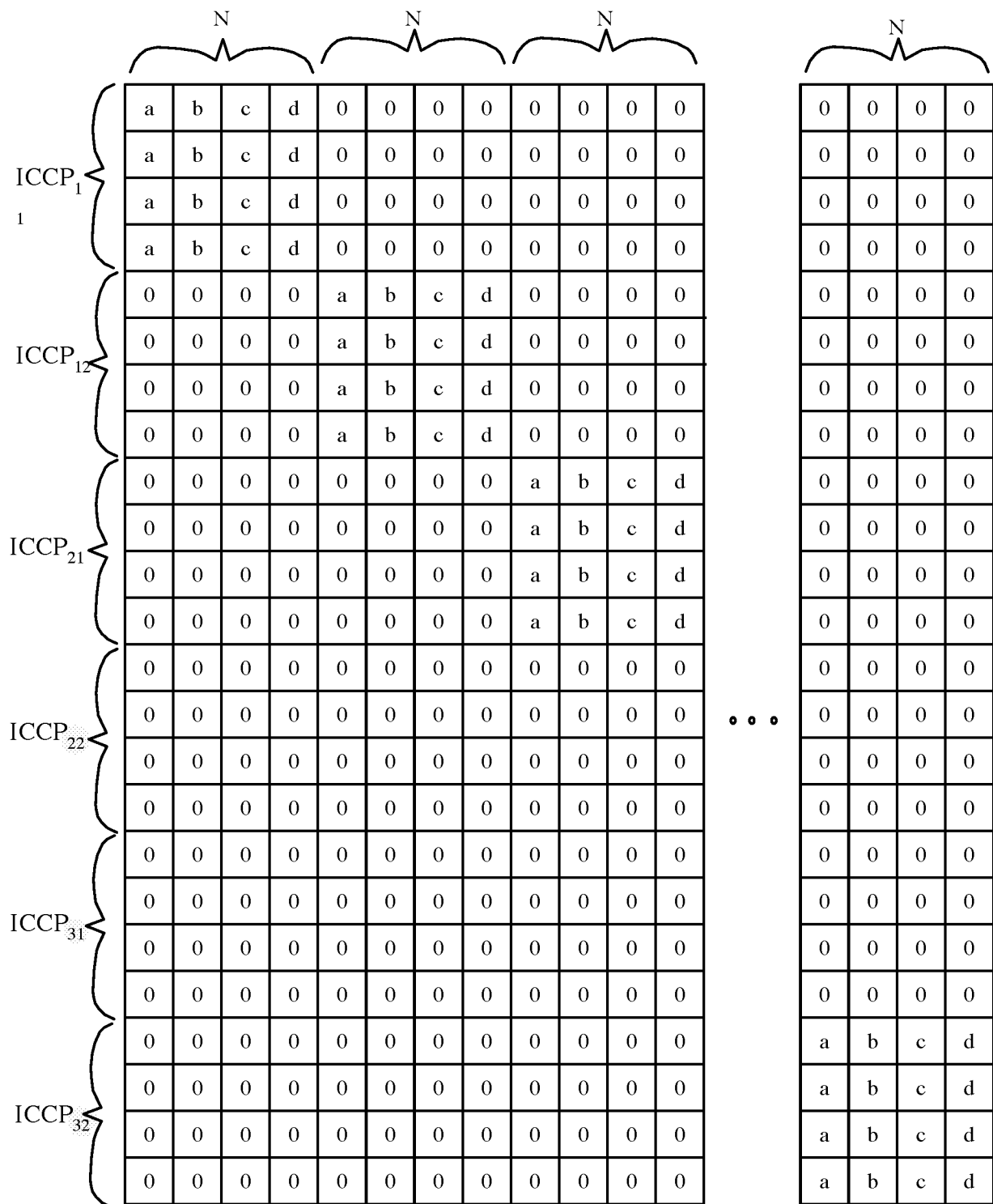


FIG. 5

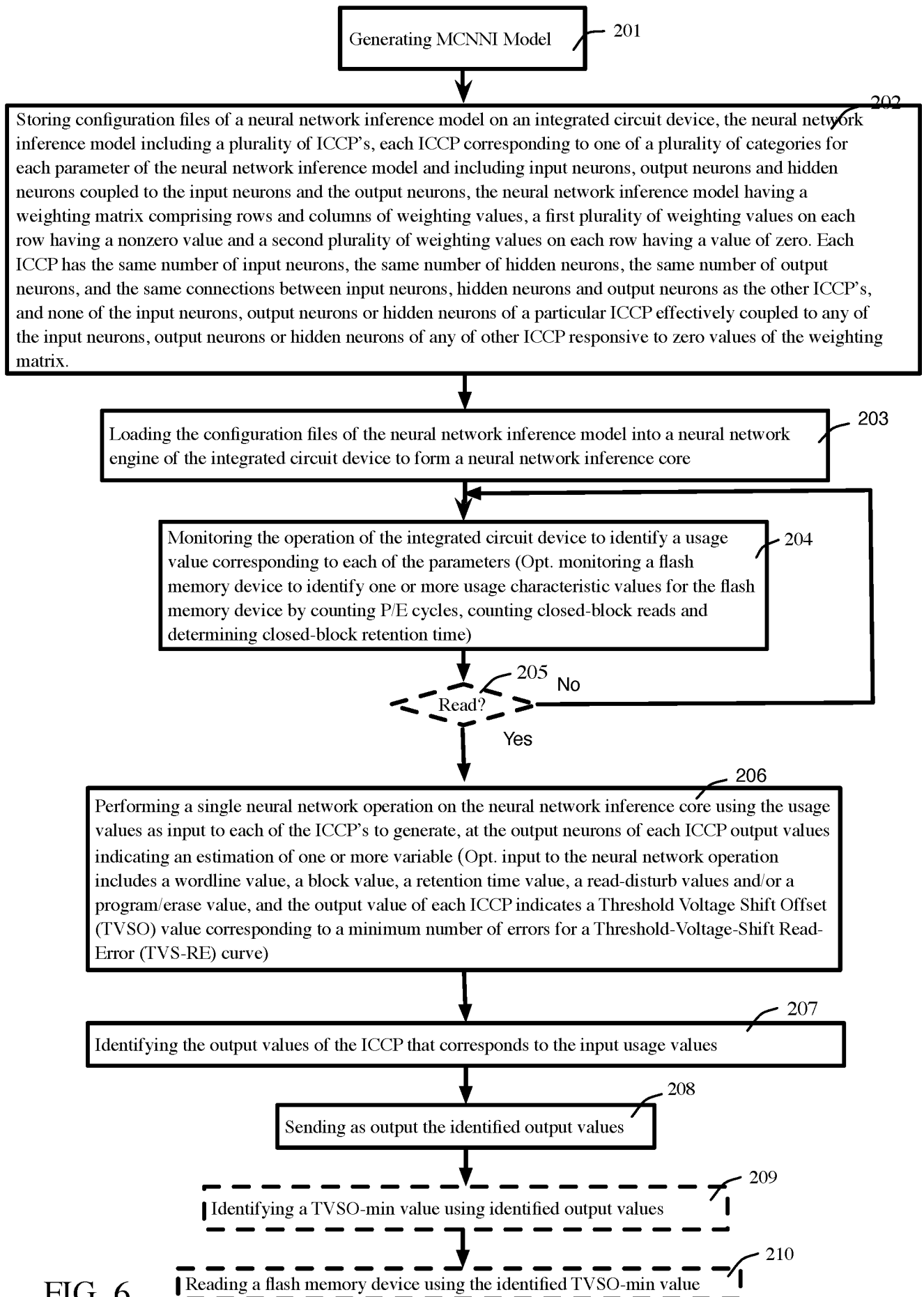


FIG. 6

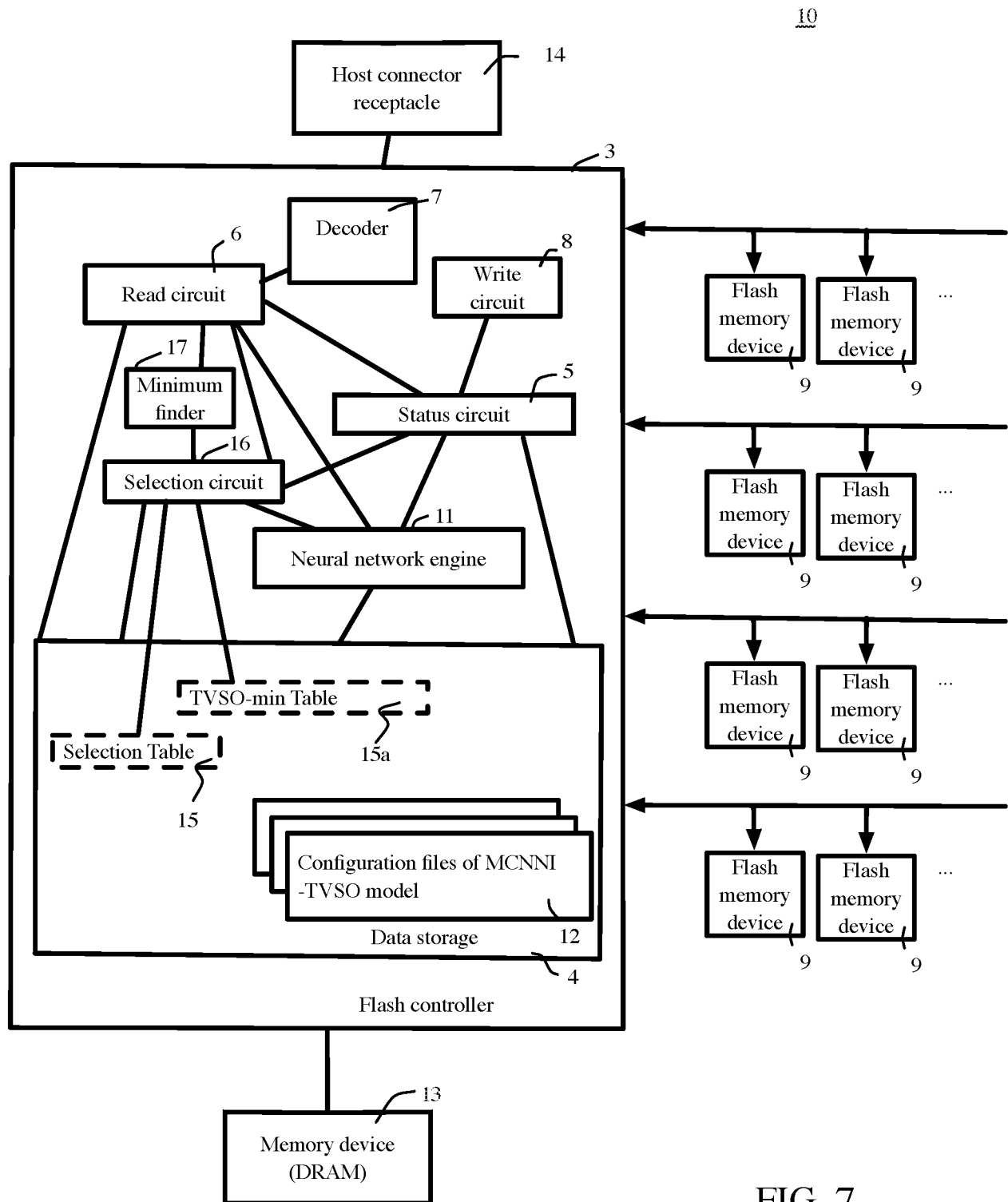


FIG. 7

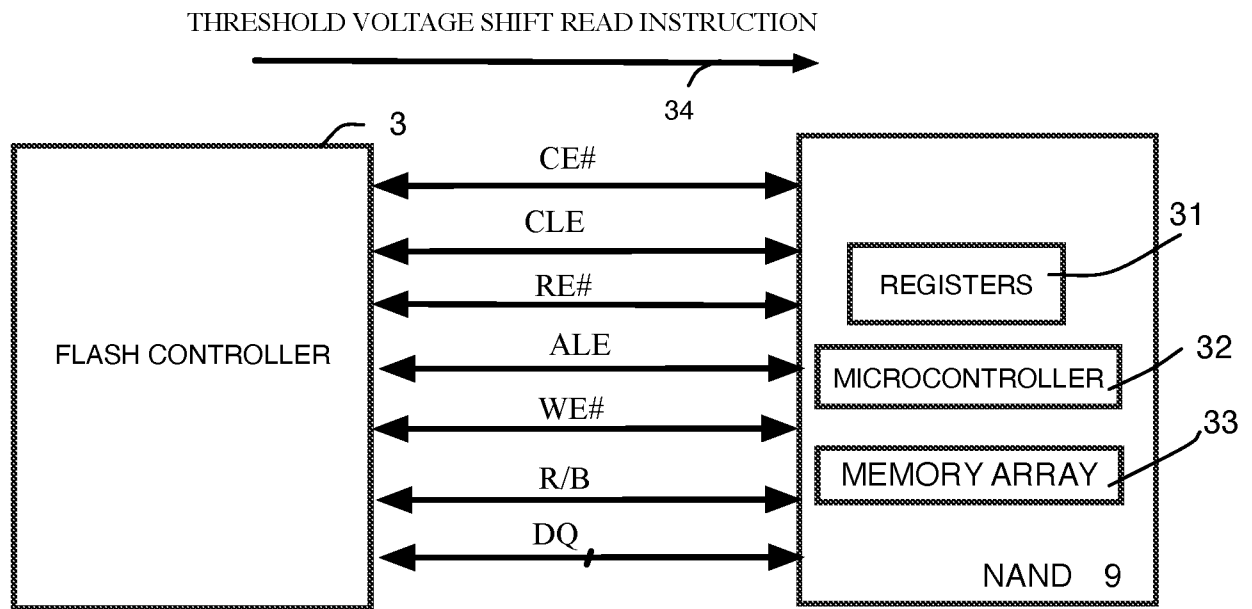


FIG. 8

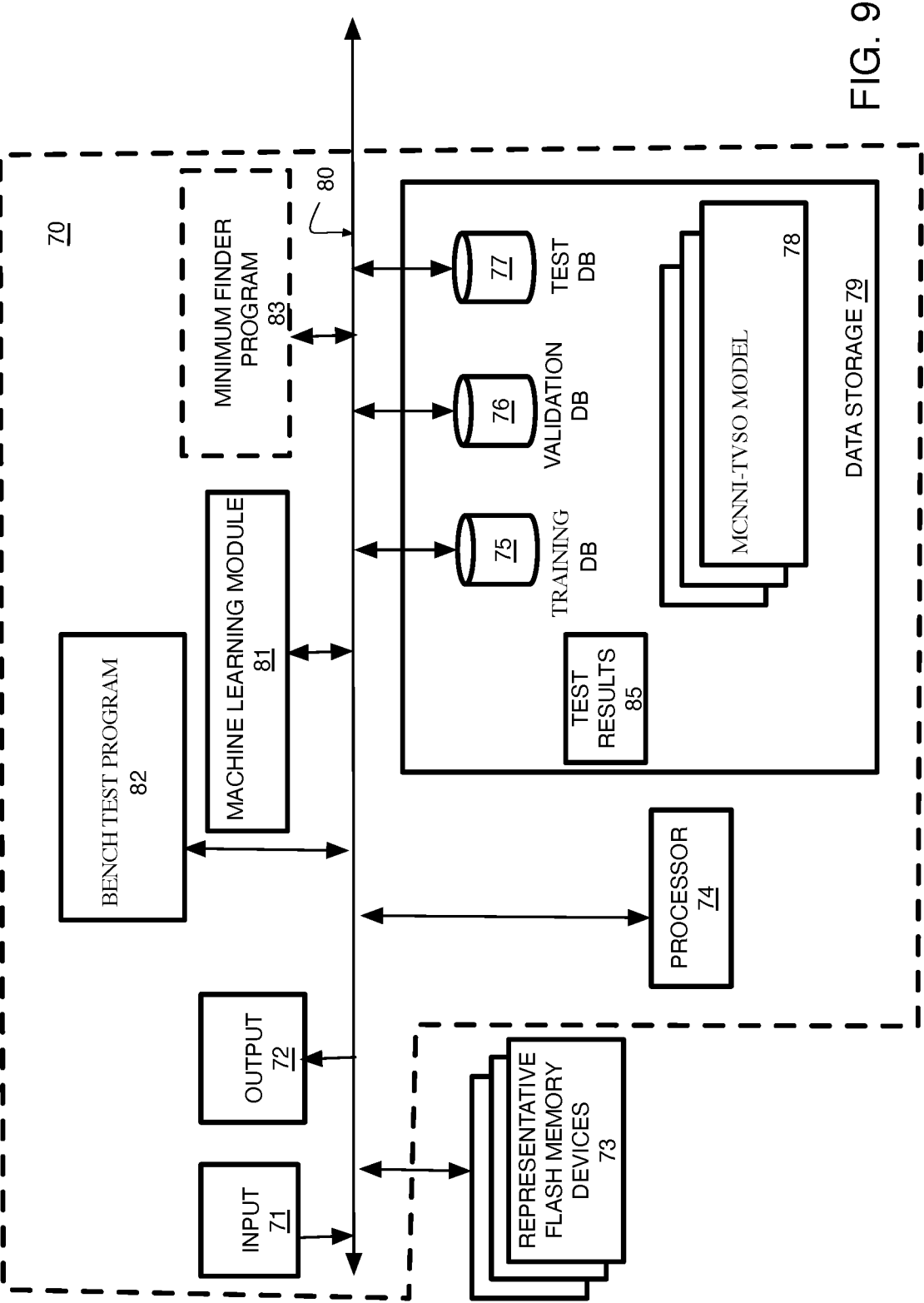
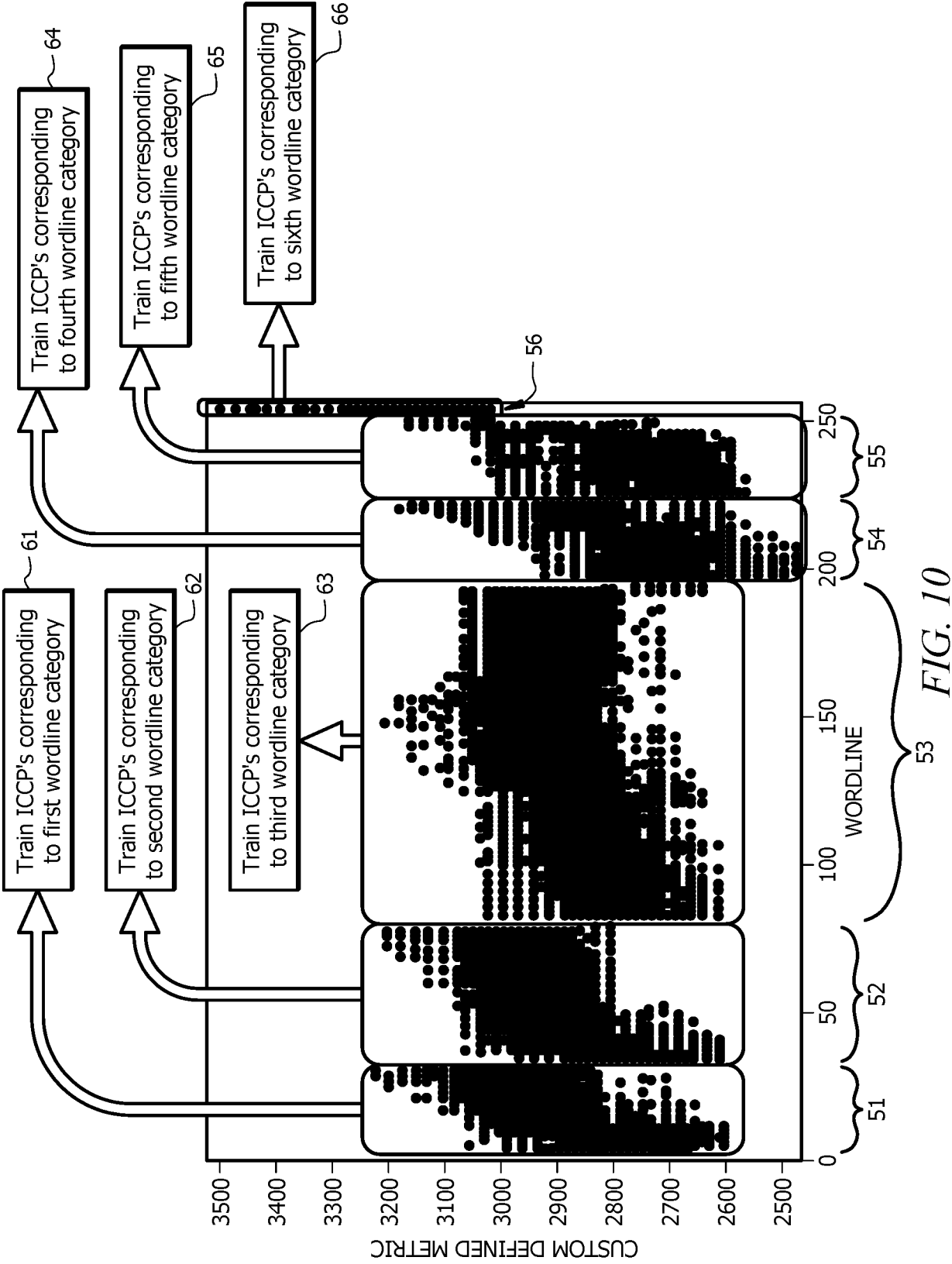


FIG. 9



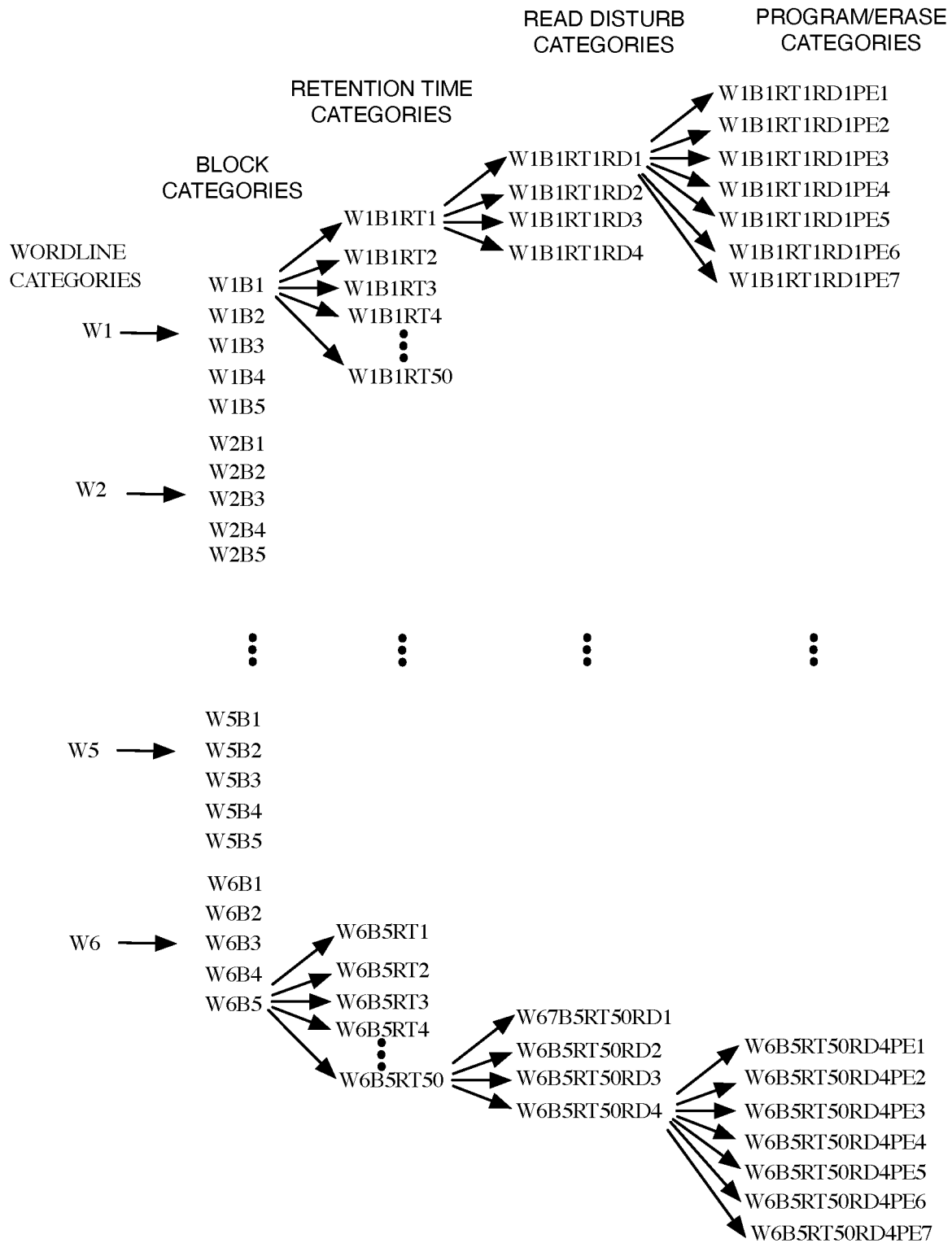


FIG. 11

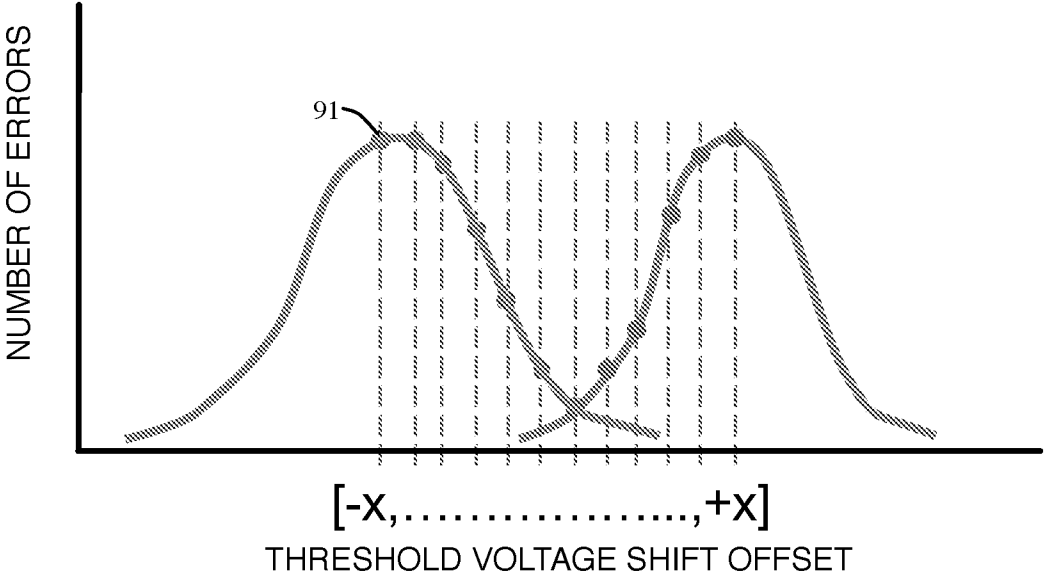


FIG. 12

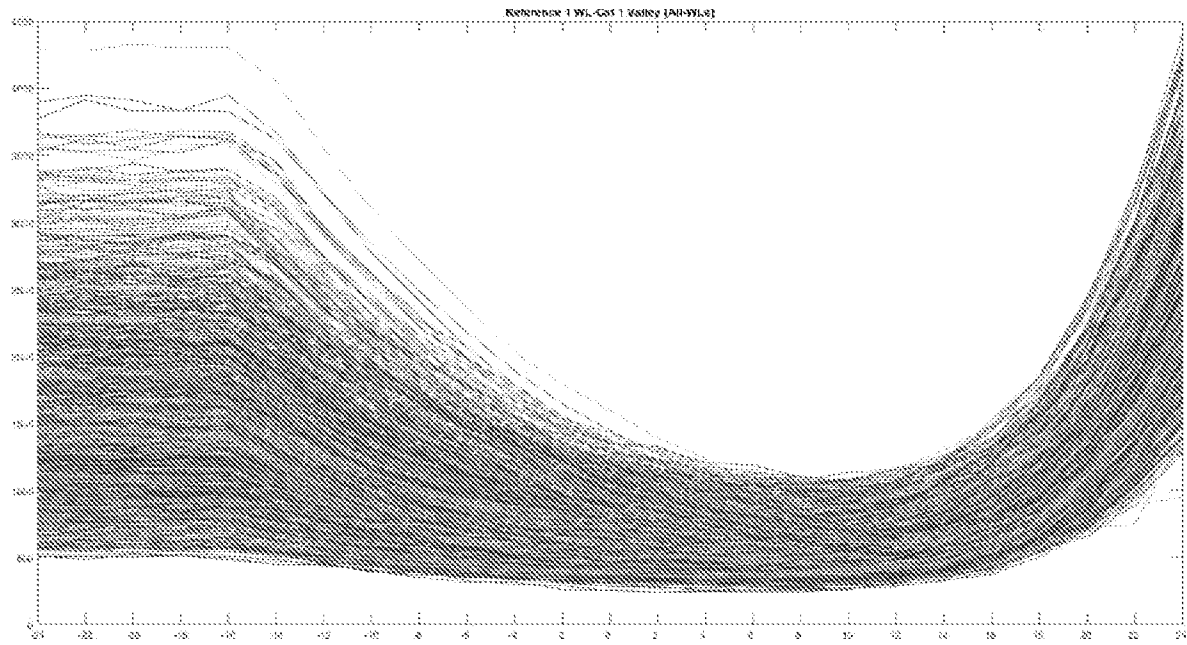


FIG. 13

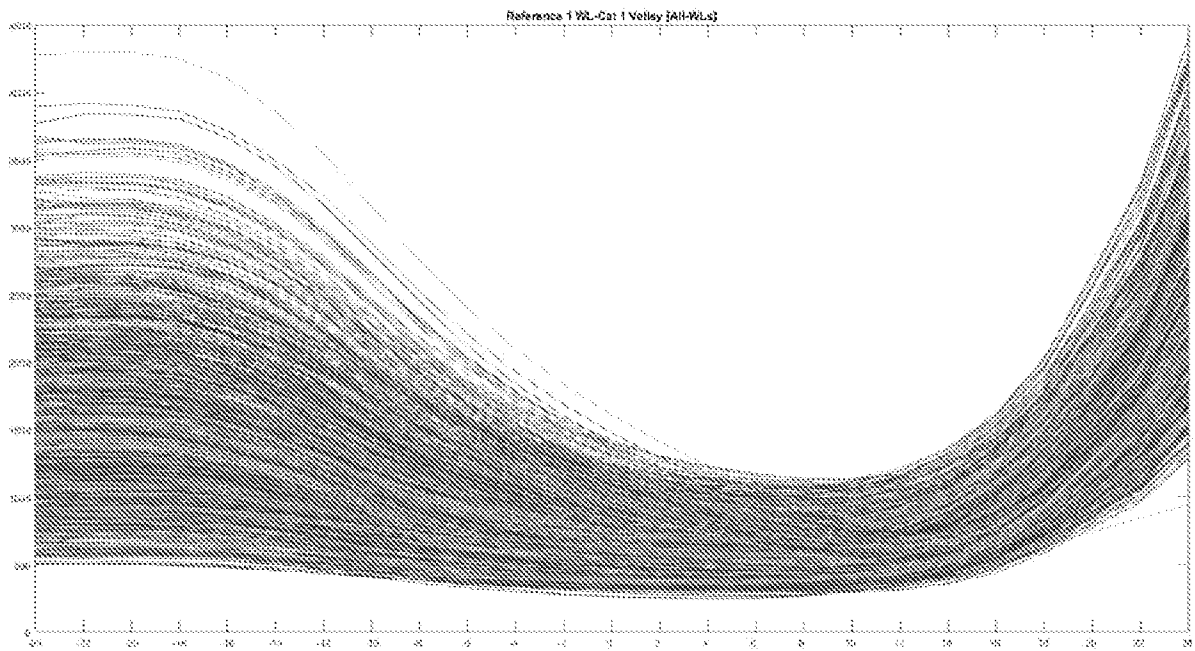


FIG. 14

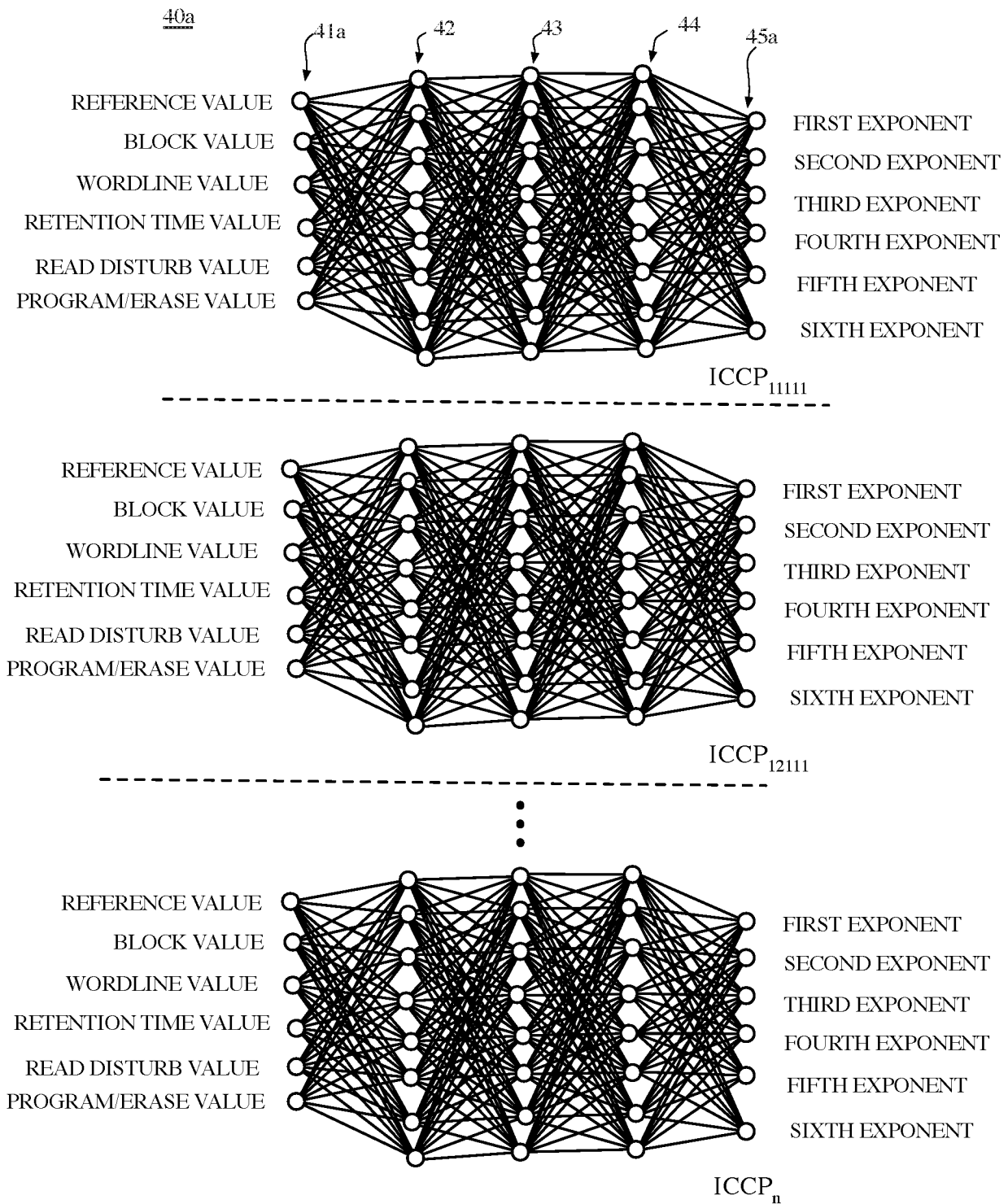


FIG. 15

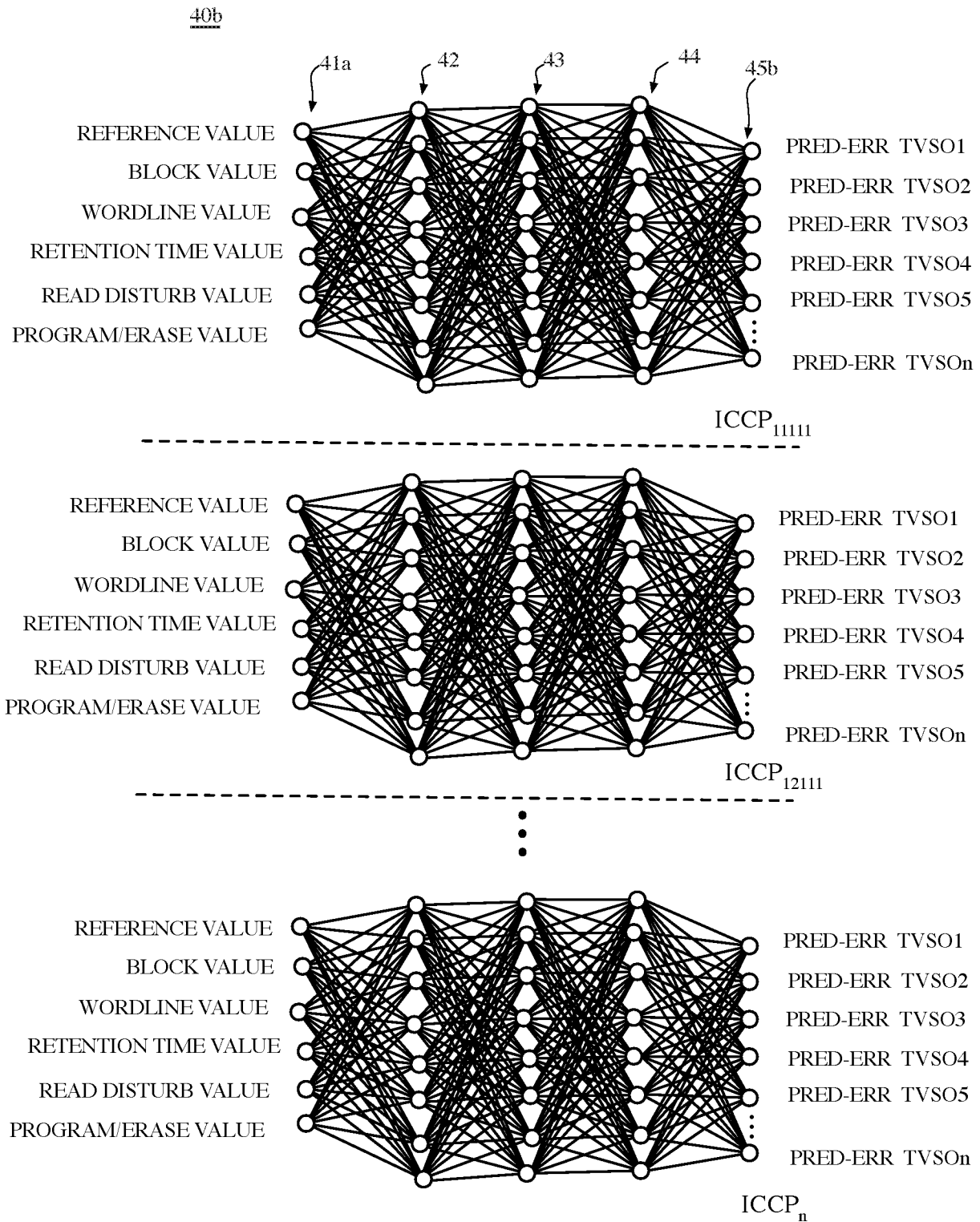


FIG. 16

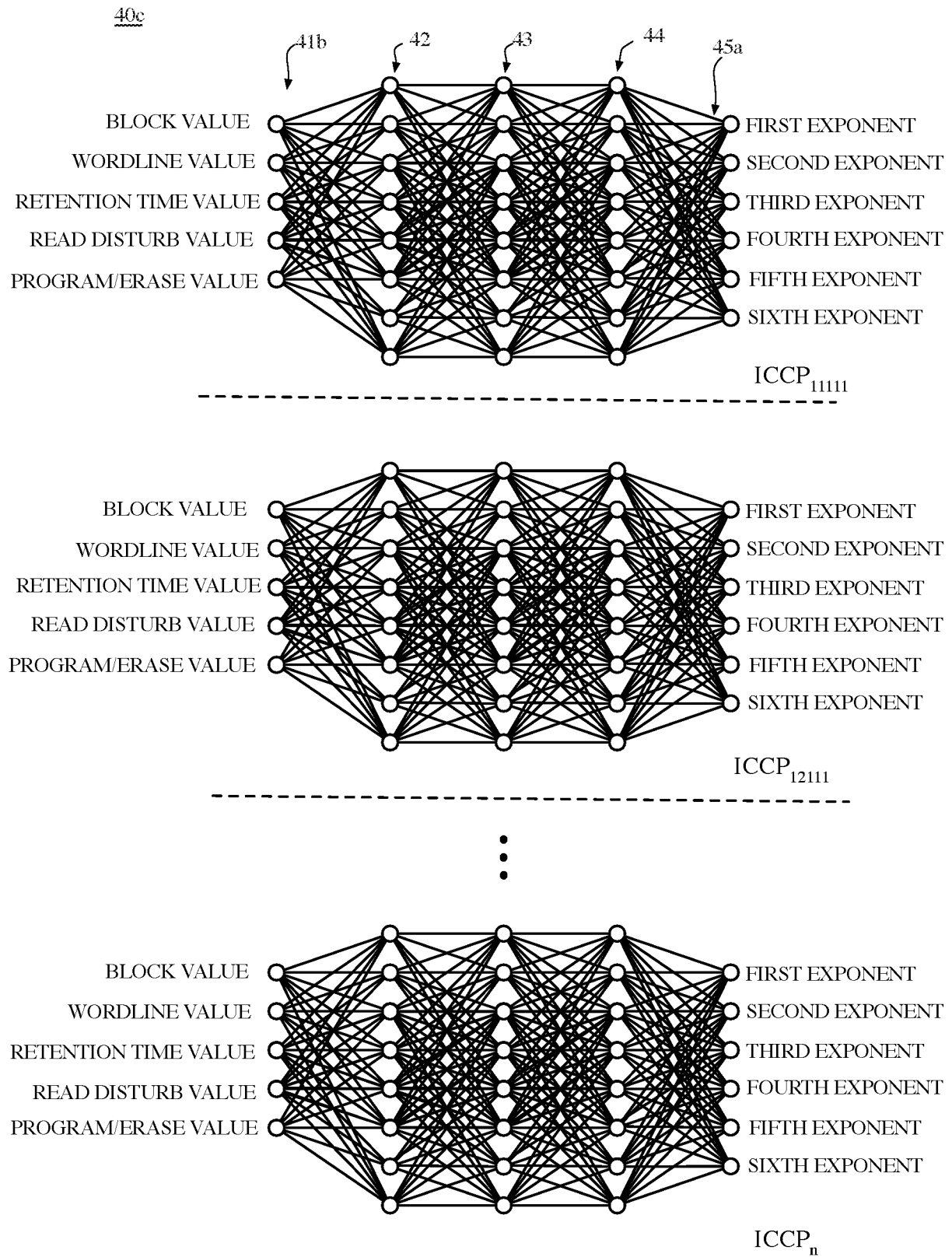


FIG. 17

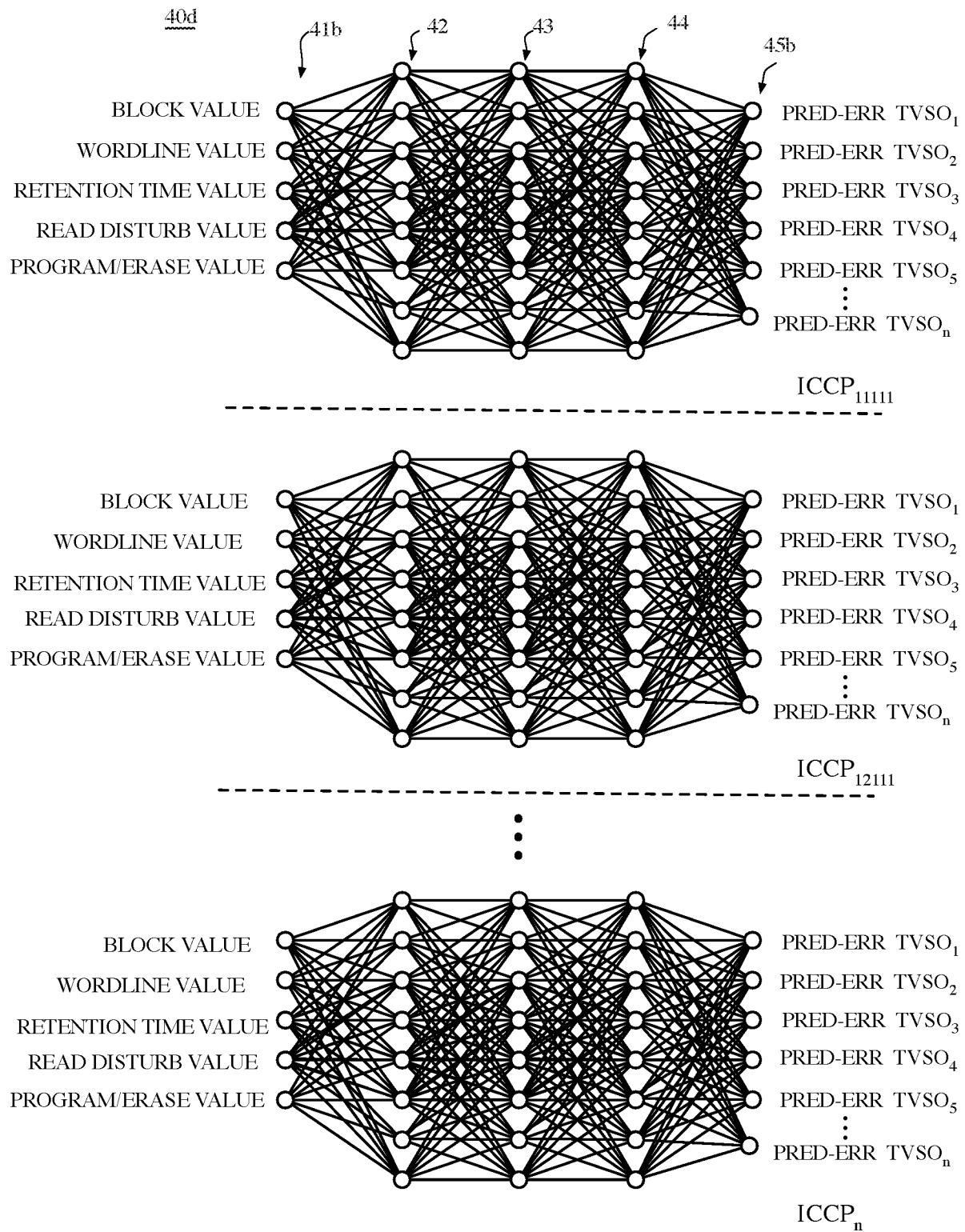


FIG. 18

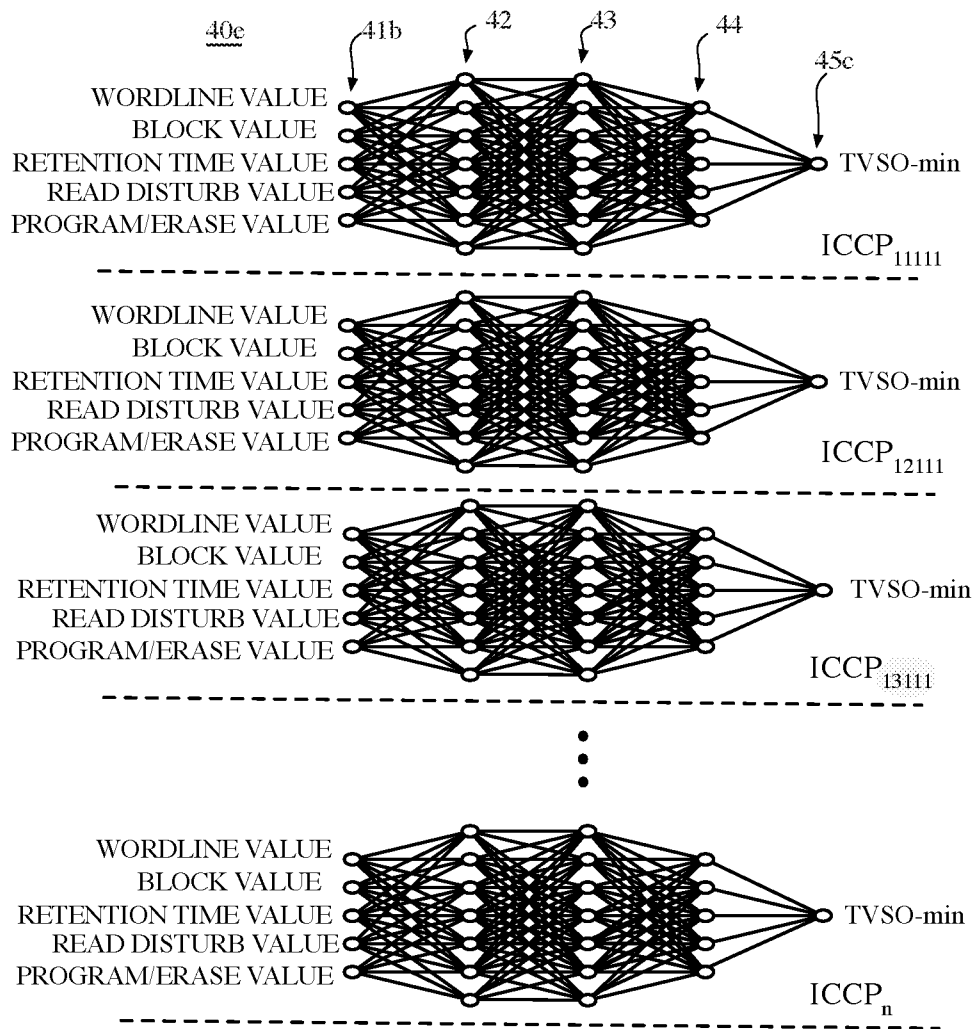


FIG. 19

INTERNATIONAL SEARCH REPORT

International application No
PCT/US2021/013738

A. CLASSIFICATION OF SUBJECT MATTER
INV. G06N3/04 G06N3/08 G11C16/00
ADD.

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
G06N G11C G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

EPO-Internal, WPI Data, INSPEC

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	NOAM SHAZEER ET AL: "Outrageously Large Neural Networks: The Sparsely-Gated Mixture-of-Experts Layer", ARXIV.ORG, CORNELL UNIVERSITY LIBRARY, 201 OLIN LIBRARY CORNELL UNIVERSITY ITHACA, NY 14853, 23 January 2017 (2017-01-23), XP080750950, sections 1, 2 and 5 -----	1-20
X	US 2020/125955 A1 (KLINGER TAMIR [US] ET AL) 23 April 2020 (2020-04-23) claims 1-7; figures 1-3 ----- -/-	1-20



Further documents are listed in the continuation of Box C.



See patent family annex.

* Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search

21 May 2021

Date of mailing of the international search report

07/06/2021

Name and mailing address of the ISA/

European Patent Office, P.B. 5818 Patentlaan 2
NL - 2280 HV Rijswijk
Tel. (+31-70) 340-2040,
Fax: (+31-70) 340-3016

Authorized officer

Douarche, Nicolas

INTERNATIONAL SEARCH REPORT

International application No
PCT/US2021/013738

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT		
Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	YU CAI ET AL: "Errors in Flash-Memory-Based Solid-State Drives: Analysis, Mitigation, and Recovery", ARXIV.ORG, CORNELL UNIVERSITY LIBRARY, 201 OLIN LIBRARY CORNELL UNIVERSITY ITHACA, NY 14853, 28 November 2017 (2017-11-28), XP081304598, the whole document -----	1-20
A	US 2020/066361 A1 (IOANNOU NIKOLAS [CH] ET AL) 27 February 2020 (2020-02-27) the whole document -----	1-20

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No

PCT/US2021/013738

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2020125955	A1	23-04-2020	NONE

US 2020066361	A1	27-02-2020	NONE
