(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
02 February 2023 (02.02.2023)

WIPO | PCT

(10) International Publication Number
WO 2023/007115 A1

(51) International Patent Classification:
G06F 7/544 (2006.01)

(21) International Application Number:
PCT/GB2022/051592

(22) International Filing Date:
22 June 2022 (22.06.2022)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
17/388,285 29 July 2021 (29.07.2021) US

(71) Applicant: **CIRRUS LOGIC INTERNATIONAL SEMICONDUCTOR LIMITED** [GB/GB]; 7B Nightingale Way, Quartermile, Edinburgh EH3 9EG (GB).

(72) Inventors: **LESSO, John Paul**; 7B Nightingale Way, Quartermile, Edinburgh EH3 9EG (GB). **MELANSON, John L.**; 800 West 6th Street, Austin, Texas 78701 (US).

(74) Agent: **HASELTINE LAKE KEMPNER LLP**; Redcliff Quay, 120 Redcliff Street, Bristol Bristol BS1 6HU (GB).

(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, IQ, IR, IS, IT, JM, 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.

(54) Title: CIRCUITRY FOR PERFORMING A MULTIPLY-ACCUMULATE OPERATION

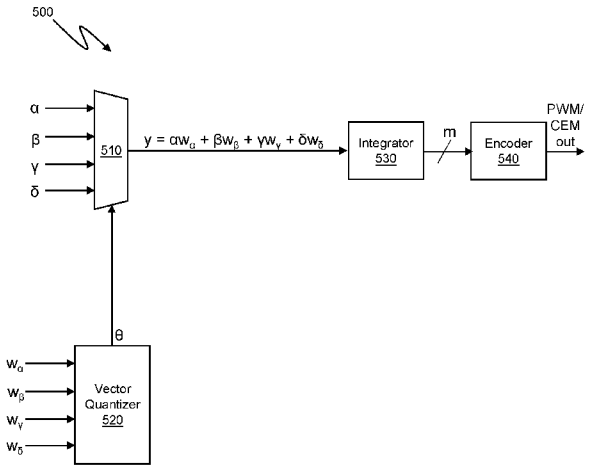


Figure 5

(57) Abstract: The present disclosure relates to circuitry for performing a multiply-accumulate (MAC) operation. The circuitry comprises a first multiplexer having a plurality of inputs for receiving a plurality of unary-coded input signals representing operands of the MAC operation and an output for outputting a multiplexer output signal representing a result of the MAC operation and a first vector quantizer configured to receive a plurality of weighting signals, each representing a proportion of a computation time period for which a respective one of the unary-coded input signals should be selected by the multiplexer and to output a first selector signal to the multiplexer to cause the multiplexer to select each of the input signals in accordance with the plurality of weighting signals.

WO 2023/007115 A1

[Continued on next page]

(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))*

CIRCUITRY FOR PERFORMING A MULTIPLY-ACCUMULATE OPERATION

Field of the Invention

The present disclosure relates to circuitry for performing a multiply-accumulate operation in stochastic computing.

Background

In stochastic computing, computation is performed on bitstreams representing the operands to a computing operation. The bitstreams may be random or pseudo-random, and the value of the operand is represented by the probability of observing a one in that bitstream. For example, for a bitstream that is eight bits long and contains four ones and four zeros distributed randomly or pseudo-randomly throughout the bitstream, the probability of observing a one at a randomly selected bit position in the bitstream is 4/8 = 1/2, and so the value represented by the bitstream is 1/2.

Stochastic computing permits the use of simple logic to perform complex operations. For example, a multiply operation can be performed with a single AND gate performing a bitwise AND operation on input bitstreams representing the operands to the multiply operation. In the example shown in **Figure 1**, to perform the multiplication operation 1/2 x 3/4, the first operand (1/2) represented by the bitstream 01101010 (i.e. an 8 bit bitstream containing four ones) is input to a first input of a logic AND gate, and the second operand (3/4) represented by the bitstream 1011011 (i.e. an 8 bit bitstream containing six ones) is input to a second input of the logic AND gate. The 8 bit bitstream output by the AND gate is 00101001, i.e. has three ones, and thus the probability of observing a one at a randomly selected bit position in the output bitstream is 3/8, which is the correct result of the operation 1/2 x 3/4.

Thus, provided that there is no correlation between the operands, a multiplication operation can be performed simply by performing a bitwise AND operation, with the accuracy of the result being dependent upon the number of bits examined at the output of the AND gate.

Modern computing systems typically make extensive use of matrix multiplication operations, which are typically implemented using multiply-accumulate (MAC)

operations. While multiplication is simple in stochastic computing, a multiply accumulate operation is more complicated, as each product (i.e. the result of each individual multiplication operation) must be accumulated to reach the final result of the MAC operation.

Summary

The present disclosure relates to circuitry for performing a multiply-accumulate (MAC) operation, in which operands of the MAC operation are represented by unary coded bitstreams. Multiplexer circuitry receives the bitstreams representing the operands and selects each bitstream for output by the multiplexer circuitry for a proportion of a total computation time period. Over the computation period the multiplexer thus generates and outputs a time division multiplexed signal based on the bitstreams input to the multiplexer. The proportion of the total computation time period for which each bitstream is selected for output by the multiplexer circuitry is determined by a vector quantizer based on weighting signals received by the vector quantizer.

According to a first aspect, the invention provides circuitry for performing a multiply-accumulate (MAC) operation, the circuitry comprising:

a first multiplexer having a plurality of inputs for receiving a plurality of unary-coded input signals representing operands of the MAC operation and an output for outputting a multiplexer output signal representing a result of the MAC operation; and

a first vector quantizer configured to receive a plurality of weighting signals, each representing a proportion of a computation time period for which a respective one of the unary-coded input signals should be selected by the multiplexer and to output a first selector signal to the multiplexer to cause the multiplexer to select each of the input signals in accordance with the plurality of weighting signals.

The circuitry may further comprise an integrator for integrating the multiplexer output signal over a period of time.

The circuitry may further comprise an encoder for encoding a signal output by the integrator into a unary coded output signal.

The circuitry may further comprise a consecutive edge modulator (CEM) configured to encode the multiplexer output signal into a consecutive edge modulated signal.

The CEM may comprise:

a first counter configured to generate a first pulse width modulated signal based on the number of high bits in a first frame of the multiplexer output signal;

a second counter configured to generate a second pulse width modulated signal based on the number of high bits in a second frame of the multiplexer output signal; and

a selector configured to select the first pulse width modulated signal as a first frame of a CEM output signal and to invert the second pulse width modulated signal in time to generate a second frame of the CEM output signal.

The second frame of the multiplexer output signal may immediately follow the first frame of the multiplexer output signal, and the second frame of the CEM output signal may immediately follow the first frame of the CEM output signal.

The first frame of the multiplexer output signal may be an odd-numbered frame of the multiplexer output signal and the second frame of the multiplexer output signal may be an even-numbered frame of the multiplexer output signal.

The first vector quantizer may comprise:

a plurality of integrators, each configured to receive a respective weighting signal; and

a processing unit configured to, in each cycle of a computation period of the circuitry:

determine a maximum among the outputs of the plurality of integrators;

output a control signal to the first multiplexer to cause the first multiplexer to select an input corresponding to the weighting signal received by the integrator whose output is maximum; and

reset the integrator whose output is maximum.

Resetting the integrator may comprise resetting the output of the integrator to 0 or to a random value.

The first vector quantizer may comprises a tree-based modulator, for example.

The circuitry may further comprise a second multiplexer configured to receive a second selector signal from the first vector quantizer.

The circuitry may further comprise a second multiplexer and a second vector quantizer, the second multiplexer being configured to receive a second selector signal from the second vector quantizer.

Each of the plurality of unary-coded input signals may be encoded according to a common encoding scheme.

At least one of the plurality of unary-encoded input signals may be encoded according to a different encoding scheme than at least one other of the plurality of unary-encoded input signals.

According to a second aspect, the invention provides convolutional Neural Network (CNN) circuitry comprising MAC circuitry according to the first aspect.

According to a third aspect, the invention provides stochastic to consecutive edge modulation (CEM) converter comprising:

a first counter configured to generate a first pulse width modulated signal based on the number of high bits in a first frame of a received stochastic signal;

a second counter configured to generate a second pulse width modulated signal based on the number of high bits in a second frame of the received stochastic signal; and

a selector configured to select the first pulse width modulated signal as a first frame of a CEM output signal and to invert the second pulse width modulated signal in time to generate a second frame of the CEM output signal.

According to a fourth aspect, the invention provides an integrated circuit comprising the circuitry of first, second or third aspect.

According to a fifth aspect, the invention provides a device comprising the circuitry of The first, second or third aspect.

The device may comprise a portable device, a battery powered device, a mobile telephone, a tablet or laptop computer, a smart speaker, an accessory device, a headset device, smart glasses, headphones, earphones or earbuds.

According to a sixth aspect, the invention provides circuitry for performing a multiply-accumulate (MAC) operation, the circuitry comprising multiplexer circuitry configured to output a time division multiplexed output signal comprising portions of a plurality of unary coded multiplexer input signals selected according to a control signal received by the multiplexer circuitry from vector quantizer circuitry.

According to a seventh aspect, the invention provides circuitry for performing a convolution operation between a kernel and a set of data, the circuitry comprising:

vector quantizer circuitry configured to receive one or more input signals representing the kernel;

first multiplexer circuitry configured to receive a first plurality of unary coded input signals representing a first subset of the data; and

second multiplexer circuitry configured to receive a second plurality of unary coded input signals representing a second subset of the data,

wherein the vector quantizer circuitry is configured to:

output a first selector signal to the first multiplexer circuitry to cause the first multiplexer circuitry to select each of the first plurality of unary coded input signals for a respective proportion of a computation period, wherein the proportion of the computation period for which each of the first plurality of unary coded input signals is selected is based on the kernel; and

output a second selector signal to the first multiplexer circuitry to cause the first multiplexer circuitry to select each of the second plurality of unary coded input signals for a respective proportion of the computation period, wherein the proportion of the computation period for which each of the second plurality of unary coded input signals is selected is based on the kernel.

According to an eighth aspect, the invention provides circuitry for performing a mean pooling function for a plurality of unary coded data signals, the circuitry comprising:

multiplexer circuitry configured to receive the plurality of unary coded data signals; and

vector quantizer circuitry configured to output a selection signal to cause the multiplexer circuitry to select each of the plurality of input signals for an equal proportion of a computation period.

Brief Description of the Drawings

Embodiments of the invention will now be described, strictly by way of example only, with reference to the accompanying drawings, of which:

Figure 1 is a schematic diagram illustrating a the use of stochastic computing techniques to perform a multiplication operation using a logical AND operation;

Figure 2 is a schematic diagram illustrating one approach to performing a multiply-accumulate (MAC) operation;

Figure 3 is a schematic diagram illustrating a further approach to performing a MAC operation;

Figure 4 is a schematic diagram illustrating a further approach to performing a MAC operation;

Figure 5 is a schematic diagram illustrating circuitry for performing a MAC operation;

Figure 6 is a schematic diagram illustrating circuitry for implementing a vector quantizer suitable for use in the circuitry of Figure 5;

Figure 7 is a schematic diagram illustrating encoder circuitry for encoding a stochastic signal into consecutive edge modulated ((CEM) signal;

Figure 8 illustrates signals in the encoder circuitry of Figure 7;

Figure 9 is a schematic diagram illustrating alternative circuitry for performing a MAC operation;

Figure 10 is a schematic diagram illustrating further alternative circuitry for performing a MAC operation;

Figure 11 is a schematic diagram illustrating functional blocks of a typical convolutional neural network (CNN);

Figure 12 is a schematic diagram illustrating example operations performed by circuitry implementing a convolutional neural network;

Figure 13 is a schematic diagram illustrating circuitry for implementing convolution operations;

Figure 14 is a schematic diagram illustrating circuitry for implementing a non-linear function;

Figure 15 illustrates a transfer function of the circuitry of Figure 14; and

Figure 16 is a schematic diagram illustrating is a schematic representation of circuitry that may be used to implement a pooling block or layer in circuitry implementing a CNN.

Detailed Description

An approach to performing a MAC operation is illustrated generally at 200 in **Figure 2**, and uses a multiplexer 210 having first and second inputs for receiving first and second unary coded bitstreams α, β and a selector signal θ, which is a unary coded bitstream representing a random number between 0 and 1. The complement of the selector signal θ is defined as $\bar{\theta} = 1 - \theta$. In some examples, the selector signal θ and the first and second unary coded bitstreams α, β may be synchronised to a common clock signal (not shown in Figure 2), in which case the behaviour of the multiplexer is well defined. In examples where there is no common clock signal, the selector signal θ and the first and second unary coded bitstreams α, β may be pleisiochronous, in the sense that they are approximately but not instantaneously frequency-aligned. In such examples, this approximate frequency alignment may be considered an extra source of noise.

In each clock period of the common clock signal, the multiplexer 210 selects and outputs the value (logic 0 or logic 1) of a current bit of the unary coded bitstream α at its first input when a current bit of the selector bitstream θ is at logic 1. Thus, as shown in Table 1 below, when the current bit of both the first unary coded bitstream α and the selector bitstreams θ are at logic 1, the output y of the multiplexer 210 is also logic 1. Otherwise the output y is logic 0. Accordingly the multiplexer 210 performs a bitwise logic AND operation on the first unary coded signal α and the selector signal θ. As explained above, this is equivalent to a multiplication of the first unary coded signal α and the selector signal θ.

α	θ	output y
0	0	0
0	1	0
1	0	0
1	1	1

Table 1

The multiplexer 210 is configured to select and output a value of the current bit of the second unary coded bitstream β at the second input when the current bit of the selector bitstream θ is at logic 0 (and thus the complement $\bar{\theta}$ of the selector bitstream θ is at logic 1). Thus, as shown in Table 2 below, when the current bit of both the second unary coded bitstream β and the complement $\bar{\theta}$ of the selector bitstream θ are at logic 1, the output y of the multiplexer 210 is also logic 1. Otherwise the output y is logic 0. Accordingly the multiplexer 210 performs a bitwise logic AND operation on the second unary coded bitstream β and the complement $\bar{\theta}$ of the selector bitstream θ. As explained above, this is equivalent to a multiplication of the second unary coded bitstream β and the complement $\bar{\theta}$ of the selector bitstream θ.

β	$\bar{\theta}$	output y
0	0	0
0	1	0
1	0	0
1	1	1

Table 2

As the selector bitstream θ and its complement $\bar{\theta}$ cannot (by definition) take the same value at any given time, the multiplexer 210 effectively acts as an AND gate for the first unary coded bitstream α and the selector bitstream θ for a first proportion of a computation period, and acts as an AND gate for the second unary coded bitstream β and the complement $\bar{\theta}$ of the selector bitstream θ for a second proportion of the computation period.

Thus over the whole of a computation period (comprising, for example, a number of cycles of the common clock signal that is equal to the length of the input bitstreams) the output y will be a combination of the result of a first logic AND operation between the first

unary coded bitstream α and the selector bitstream θ and the result of a second logic AND operation between the second unary coded bitstream β and the complement $\bar{\theta}$ of the selector bitstream θ, i.e.:

$$y = \alpha\theta + \beta\bar{\theta}.$$

As will be appreciated, this represents a MAC operation, as the result of the second multiplication operation $\beta\bar{\theta}$ is accumulated with the result of the first multiplication operation $\alpha\theta$.

Under this approach it is possible for the operands represented by the first and second unary bitstreams α, β to be correlated, as the output y will be correct if there is no correlation between either α or β and θ, provided that the durations of the first and second proportions of the computation period are correct.

Figure 3 is a schematic representation of a further approach to performing a MAC operation. As shown generally at 300, in this approach first and second multiplexers 310, 320 are daisy-chained together, with an output of the first multiplexer 310 connected to an input of the second multiplexer.

A first input of the first multiplexer 310 receives a first unary coded bitstream signal α. A second input of the first multiplexer 310 receives a constant logic 0 signal. A first selector signal θ1 in the form of a unary coded bitstream is received at a control input of the first multiplexer 310.

A first input of the second multiplexer 320 receives a second unary coded bitstream signal β. A second input of the second multiplexer 320 is coupled to the output of the first multiplexer 320, and a second selector signal θ2, also in the form of a unary coded bitstream, is received at a control input of the second multiplexer 320. The first and second signals α, β and the first and second selector signals θ1, θ2 are synchronised to a common clock signal.

Thus, the output y of the chained multiplexers 310, 320 over a whole computation period can be expressed as:

$y = \beta\theta2 + \alpha\theta1\bar{\theta}2$, where $\bar{\theta}2$ is the complement of the second selector signal θ2.

Again, the first and second unary coded signals α, β can be correlated without adversely affecting the accuracy of the result of the computation, provided that there is no correlation between either α or β and θ1 or θ2, and that that the durations of the first and second proportions of the computation period are correct.

In order to perform the calculation

$$y = \frac{\alpha w_\alpha + \beta w_\beta}{2}$$

(where w_α and w_β are weightings to be applied to the variables α and β respectively) using the arrangement shown in Figure 3, the correct values θ1 and θ2 can be calculated as follows:

$$\beta\theta 2 + \alpha\theta 1\bar{\theta}2 = \frac{\alpha w_\alpha + \beta w_\beta}{2},$$

so:

$$\theta 2 = \frac{w_\beta}{2} \text{ and } \theta 1 = \frac{w_\alpha}{2\bar{\theta}2}$$

The approach illustrated in Figure 3 can be improved by using a multiple input multiplexer, as shown generally at 400 in **Figure 4**.

In the implementation shown generally at 400 in Figure 4 a multiple input multiplexer 410 having, in this example, four inputs, receives first to fourth unary coded input signals α, β, γ, δ. A selector signal θ is received at a control input of the multiplexer 410. The selector signal θ in this example is a random number with a probability distribution function that causes the multiplexer 410 to select each input signal α, β, γ, δ for the correct proportion of each computation period, such that over a given computation period the output y of the multiplexer 410 is a time division multiplexed signal containing appropriately weighted (in time) versions of the input signals and can be expressed as:

$y = \alpha w_\alpha + \beta w_\beta + \gamma w_\gamma + \delta w_\delta$ (where w_α, w_β, w_γ, and w_δ are weightings to be applied to the variables α, β, γ and δ respectively).

Figure 5 is a schematic representation of circuitry implementing a further approach to performing a multiply-accumulate (MAC) operation on a plurality of operands represented by unary coded bitstreams.

The circuitry, shown generally at 500 in Figure 5, includes multiplexer circuitry 510 having a plurality of inputs for receiving a plurality of unary coded bitstreams α, β, γ, δ representing operands of the MAC operation. The circuitry 500 further includes vector quantizer circuitry 520 having a plurality of inputs for receiving a plurality of weighting signals w_α, w_β, w_γ, w_δ, each weighting signal corresponding to one of the plurality of bitstreams α, β, γ, δ and representing a proportion of a computation time period for which that unary coded bitstream should be selected by the multiplexer circuitry 510 in order to correctly perform a MAC operation using the operands.

The circuitry 500 in this example further includes integrator circuitry 530 coupled to the output of the multiplexer circuitry 510 and configured to integrate the signal output by the multiplexer circuitry 510 over a time period p to generate an m-bit integrator output signal, and encoder circuitry 540 configured to encode the integrator output signal into a unary coded output signal. In some examples the encoder circuitry 540 comprises pulse width modulator circuitry, such that the unary coded output signal is a pulse width modulated (PWM) signal. In other examples, the integrator circuitry 530 encoder circuitry 540 may be configured to generate a consecutive edge modulated (CEM) signal as the unary coded output signal, as will be explained in more detail below.

In operation of the circuitry 500, the vector quantizer circuitry 520 receives the plurality of weighting signals w_α, w_β, w_γ, w_δ and outputs a selector signal θ to cause the multiplexer circuitry 510 to select each input signal α, β, γ, δ for the correct proportion of each computation period, such that over a given computation period the output y of the multiplexer circuitry 510 is a time division multiplexed signal of the form

$$y = \alpha w_\alpha + \beta w_\beta + \gamma w_\gamma + \delta w_\delta.$$

The integrator circuitry 530 is configured to integrate the output signal y over a time period p and to output an integrated signal to the encoder 540, which encodes the integrated signal into a unary output signal. The integrator circuitry 530 may be generally conventional, and the encoder 540 may comprise, for example, a pulse width modulator.

Alternatively, the integrator circuitry 530 and the encoder 540 may be provided by consecutive edge modulator circuitry, as described below with reference to Figure 7.

Figure 6 is a schematic representation of example vector quantizer circuitry suitable for use in the circuitry 500 of Figure 5.

The vector quantizer circuitry, shown generally at 600 in Figure 6, comprises a plurality (in this example four) of integrators 610 – 640, each having an input for receiving a respective weighting signal w_α, w_β, w_γ, w_δ. An output of each of the integrators 610 – 640 is coupled to an input of a processing unit 650 which is configured to calculate, in each cycle of the computation period, the maximum among the signals received from the integrators 610 – 640, and to output a selector signal θ based on the determined maximum, to cause the multiplexer circuitry (e.g. multiplexer circuitry 510 in Figure 5) to select the corresponding input signal α, β, γ, δ for that cycle of the computation period.

In each cycle of the computation period the output of each integrator 610 – 640 increases by the respective weighting signal w_α, w_β, w_γ, w_δ, and the processing unit 650 determines the maximum among the signals output by the integrators 610 – 640 (i.e. which of the integrator output signals has the greatest value or magnitude). The processing unit 650 outputs a selector signal θ to cause the multiplexer to select the input signal α, β, γ, δ that corresponds to the weighing signal w_α, w_β, w_γ, w_δ whose integrator output is maximum, and resets the output of that integrator to 0.

In the example illustrated in Table 3 below, the weighting signals w_α, w_β, w_γ, w_δ have values 0.1, 0.2, 0.3 and 0.4 respectively, and the weighting signals w_α, w_β, w_γ, w_δ are provided to the first to fourth integrators 610 – 640 respectively. In a first computation cycle (cycle 1 in Table 3 below), the processing unit 650 determines that the output of the fourth integrator 640 has the maximum value (0.4) and thus outputs a selector signal θ of a suitable value (e.g. 4) to cause the multiplexer (e.g. multiplexer 510) to select the input signal δ that corresponds to the weighting signal w_δ that is provided to the fourth integrator 640. The processing unit then resets the output of the fourth integrator 640 to zero for the next computation cycle.

In cycle 2, the processing unit 650 determines that the output of the third integrator 630 has the maximum value (0.6) and thus outputs a selector signal θ of a suitable value (e.g. 3) to cause the multiplexer (e.g. multiplexer 510) to select the input signal β that

corresponds to the weighting signal w_γ that is provided to the third integrator 630. The processing unit then resets the output of the third integrator 630 to zero for the next computation cycle.

In cycle 3, the processing unit 650 determines that the output of the second integrator 620 has the maximum value (0.6) and thus outputs a selector signal θ of a suitable value (e.g. 2) to cause the multiplexer (e.g. multiplexer 510) to select the input signal β that corresponds to the weighting signal w_β that is provided to the second integrator 620. The processing unit then resets the output of the second integrator 620 to zero for the next computation cycle.

In cycle 4, the processing unit 650 determines that the output of the fourth integrator 640 has the maximum value (0.8) and thus outputs a selector signal θ of a suitable value (e.g. 4) to cause the multiplexer (e.g. multiplexer 510) to again select the input signal δ that corresponds to the weighting signal w_δ that is provided to the fourth integrator 640. The processing unit then resets the output of the fourth integrator 640 to zero for the next computation cycle.

	Integrator 1 o/p	**Integrator 2 o/p**	**Integrator 3 o/p**	**Integrator 4 o/p**
Cycle no.				
1	0.1	0.2	0.3	*0.4 (max)*
2	0.2	0.4	*0.6 (max)*	0
3	0.3	*0.6 (max)*	0	0.4
4	0.4	0	0.3	*0.8 (max)*
5	0.5	0.2	*0.6 (max)*	0
6	*0.6 (max)*	0.4	0	0.4
7	0	0.6	0.3	*0.8 (max)*
8	0.1	*0.8 (max)*	0.6	0
9	0.2	0	*0.9 (max)*	0.4
10	0.3	0.2	0	*0.8 (max)*

Table 3

The processing unit 650 continues to output a selector signal θ of a suitable value to cause the multiplexer to select the input signal that corresponds to the weighting signal that is provided to the integrator that outputs the maximum value in a computation cycle

until the total number of computation cycles that make up the computation period have elapsed.

In this way the multiplexer 510 can be made to select each input signal for the correct proportion of the computation time to produce the correct result of the MAC operation.

In the example shown in Table 3 below, the weights assigned to the input signals α, β, γ, δ are 0.1, 0.2, 0.3 and 0.4 respectively, meaning that input signal α should be selected for 0.1 of a computation period, input signal β should be selected for 0.2 of a computation period, input signal γ should be selected for 0.3 of a computation period, and input signal δ should be selected for 0.4 of a computation period. As can be seen from Table 3 above, the output of the first integrator 610 has the maximum value once in a computation period of ten cycles, such that the input signal α will be selected for 0.1 of the duration of the computation period. Similarly, the output of the second integrator 620 has the maximum value twice in a computation period, such that the input signal β will be selected for 0.2 of the duration of the computation period, the output of the third integrator 630 has the maximum value three times in a computation period, such that the input signal γ will be selected for 0.3 of the duration of the computation period, and the output of the fourth integrator 640 has the maximum value four times in a computation period, such that the input signal δ will be selected for 0.4 of the duration of the computation period.

It may be desirable to add a degree of dither to interrupt or break up periodic behaviours in the output of the vector quantizer circuitry 600. To effect dither, whenever one of the integrators 610 – 640 is reset a small random number can be added to its output, so that the integrator is initialised to a small random number rather than zero.

It will be appreciated that the vector quantizer circuitry 600 of Figure 6 is just one example of vector quantizer circuitry that is suitable for use as the vector quantizer circuitry 520 in the circuitry 500 of Figure 5, and that other vector quantizer architectures, e.g. a tree-based modulator architecture, could equally be used as the vector quantizer circuitry 520.

Figure 7 is a schematic diagram illustrating example modulator circuitry suitable for integrating and encoding the signal output by the multiplexer 510 of Figure 500 into a consecutive edge modulated (CEM) output signal.

The modulator circuitry, shown generally at 700 in Figure 7 (which may be referred to as a stochastic to CEM converter), comprises a demultiplexer 710, first and second counters 720, 730 and a selector 740. The first and second counters may be, for example, Mobius counters, ring counters, shift registers or any other form of counter suitable for counting the number of logic ones in an input bitstream signal.

The demultiplexer 710 has an input for receiving an input bitstream y, e.g. from the multiplexer 510 of the circuitry 500 of Figure 5. A first output of the demultiplexer is coupled to an input of the first counter 720, and a second output of the demultiplexer 730 is coupled to an input of the second counter 730. An output of the first counter 720 is coupled to a first input of the selector 740, and an output of the second counter 730 is coupled to a second input of the selector 740.

In operation of the modulator circuitry 700, the input bitstream y is received at the input of the demultiplexer 710. The demultiplexer 710 is operative to couple the input of the first counter 720 to the input of the demultiplexer 710 for odd-numbered frames of the input signal y, and to couple the input of the second counter 730 to the input of the demultiplexer 710 for even-numbered frames of the input signal y. (A frame of the input signal y comprises a predefined number of bits of the input signal, e.g. 8 bits).

The first counter 720 is configured to count the number of logic ones in an odd-numbered frame of the input signal y that it receives from the demultiplexer 710, and to generate a bitstream signal comprising a number of logic ones that is equal to the number of logic ones present in the input signal y in the frame period, followed by a number of logic zeros that is equal to the number of logic zeros present in the input signal y in the frame period. In other words, the first counter 720 is operative to generate a pulse width modulated output signal having a positive pulse whose width is dependent upon the number of logic ones present in the input signal y in the frame period.

Similarly, the second counter 730 is configured to count the number of logic ones in an even-numbered frame of the input signal y that it receives from the demultiplexer 710, and to generate a bitstream signal comprising a number of logic ones that is equal to the number of logic ones present in the input signal y in the frame period, followed by a number of logic zeros that is equal to the number of logic zeros present in the input signal y in the frame period. Thus, the second counter 730 is also operative to generate a pulse

width modulated output signal having a positive pulse whose width is dependent upon the number of logic ones present in the input signal y in the frame period.

The selector 740 is configured to select the output of the first counter 720 during even-numbered frames of the input signal y, and to select the output of the second counter 730 during odd-numbered frames of the input signal.

During even-numbered frames of the input signal y, the selector 740 reads the bitstream generated by the first counter 720 and outputs the bitstream as the CEM output signal. During odd-numbered frames of the input signal y, the selector 740 reads the bitstream generated by the second counter 730 and outputs the bitstream, in reverse order, as the CEM output signal.

For example, the first and second counters 720, 730 may each comprise a register of length equal to frame length of input signal (i.e. if frame length is 8 bits then register length is also 8 bits). At the start of a frame period of the input signal the relevant register (i.e. the register of the first counter 720 if the frame period is odd-numbered or the register of the second counter 730 if the frame period is even-numbered) is reset to all 0. Every time a 1 is detected in the input signal y the bits in the register shift one position to the right and the most significant bit (MSB) of the register is set to one. Thus at the end of frame period the register will contain a number of ones equal to the number of ones in the frame of the input signal y, starting at the MSB position, followed by a number of 0s equal to the number of zeros in the input frame.

During even-numbered frame periods of the input signal y, the selector 740 reads the contents of the register of the first counter 720 (which were entered into the register during the immediately preceding frame period of the input signal) in most significant bit (MSB) to least significant bit (LSB) order, such that output signal of the selector 740 during even-numbered frame periods of the input signal is the same as the contents of the register of the first counter. During odd-numbered frame periods of the input signal the selector 740 reads the contents of the register of the second counter 730 (which were entered into the register during the immediately preceding frame period of the input signal) in LSB to MSB order, such that output signal of the selector 740 is inverted in time with respect to the contents of register of the second counter 720.

Table 4 below illustrates the operation of the first and second counters 720, 730 and the selector 740 for an example input signal y comprising four 8 bit frames. It will be noted that the selector output for frame 2 is identical to the bitstream generated by the first counter 720 during frame 1. The selector output for frame 3 is a time-reversed version of the bitstream generated by the second counter 730 during frame 2. The selector output for frame 4 is identical to the bitstream generated by the first counter 720 during frame 3, and the selector output for frame 5 is a time-reversed version of the bitstream generated by the second counter 730 during frame 4.

	Frame 1	**Frame 2**	**Frame 3**	**Frame 4**	**Frame 5**
Input signal y	10011011	01011010	10111011	01001100	
Counter 1	11111000		11111100		
Counter 2		11110000		11100000	
Selector output		11111000	00001111	11111100	00000111

Table 4

Figure 8 shows the example input signal y shown in Table 4 above as trace 810, the signals generated by the first and second counters 720, 730 (traces 820 and 830 respectively) and the CEM output signal generated by the selector 740 (trace 840).

Figure 8 also shows, in trace 850, a PWM signal representing the input signal y. As can be seen, the PWM signal 850 has seven transitions between logic 0 and logic 1 states, whereas the CEM signal output by the selector 740 has only 4 such transitions. As will be appreciated, the reduced number of transitions in the CEM output signal 840, as compared to the PWM signal 850 gives rise to reduced power consumption in the CEM modulator circuitry 700, as compared to conventional PWM modulator circuitry.

Further, the use of a CEM signal as the output signal is beneficial where the circuitry 500 is used for applications such as circuitry implementing a convolutional neural network (CNN), as the CEM signal facilitates implementation of a non-linear activation function, as discussed in more detail below.

Figure 9 is a schematic diagram illustrating alternative circuitry for performing a MAC operation with improved efficiency, as compared to the circuitry 500 of Figure 5, by performing a plurality of MAC operations using the same weighting signals in parallel.

The circuitry, shown generally at 900 in Figure 9, includes first multiplexer circuitry 910 having a plurality of inputs for receiving a first plurality of unary coded input bitstreams α_1, β_1, γ_1, δ_1 representing operands of a first MAC operation. The circuitry 900 further includes first vector quantizer circuitry 920 having a plurality of inputs for receiving a plurality of weighting signals w_α, w_β, w_γ, w_δ, each weighting signal representing a proportion of a computation time period for which a respective one of the first plurality of unary coded input bitstreams α_1, β_1, γ_1, δ_1 should be selected by the first multiplexer circuitry 910 in order to correctly perform the first MAC operation using the operands.

The circuitry 900 further includes second multiplexer circuitry 930 having a plurality of inputs for receiving a second plurality of unary coded input bitstreams α_2, β_2, γ_2, δ_2 representing operands of a second MAC operation. The circuitry 900 further includes second vector quantizer circuitry 940 having a plurality of inputs for receiving the plurality of weighting signals w_α, w_β, w_γ, w_δ, which again represent a proportion of a computation time period for which a respective one of the second plurality of unary coded input bitstreams α_2, β_2, γ_2, δ_2 should be selected by the second multiplexer circuitry 920 in order to correctly perform the second MAC operation using the operands. Thus, the same weightings are applied to the first plurality of unary coded input bitstreams and to the second plurality of unary coded input bitstreams. If the first and second vector quantizer circuitry 920, 940 are dithered differently (e.g. a source of dither for the first vector quantizer circuitry 920 is independent from a source of dither for the second vector quantizer circuitry 940), then the circuitry 900 will accelerate convergence.

The circuitry 900 further includes adder circuitry 950 configured to receive output bitstream signals y1, y2 of the first and second multiplexer circuitry 910, 930 and to combine the received bitstreams into a single output bitstream y representing the sum of the results of the combination first and second MAC operations.

The circuitry 900 may further include integrator circuitry 530 and an encoder 540 of the kind described above with reference to Figure 5, for integrating the output signal y and encoding the integrated signal into a unary coded signal. Alternatively, the circuitry 900 may further comprise CEM encoder circuitry 700 of the kind described above with reference to Figure 7, for encoding the output signal y as a CEM signal.

The first and second vector quantizer circuitry 920, 940 may each comprise vector quantizer circuitry of the kind described above with reference to Figure 6, or alternatively may comprise circuitry implementing some other vector quantizer architecture such as a tree-based modulator.

In operation of the circuitry 900, the first and second vector quantizer circuitry 920, 940 each receive the plurality of weighting signals w_α, w_β, w_γ, w_δ. The first vector quantizer circuitry 920 outputs a first selector signal $\theta 1$ to cause the first multiplexer circuitry 910 to select each input signal α_1, β_1, γ_1, δ_1 for the correct proportion of each computation period, such that over a given computation period the output y1 of the first multiplexer circuitry 910 is a time division multiplexed signal of the form

$$y1 = \alpha_1 w_\alpha + \beta_1 w_\beta + \gamma_1 w_\gamma + \delta_1 w_\delta.$$

Similarly, the second vector quantizer circuitry 940 outputs a second selector signal $\theta 2$ to cause the second multiplexer circuitry 930 to select each input signal α_2, β_2, γ_2, δ_2 for the correct proportion of each computation period, such that over a given computation period the output y2 of the second multiplexer circuitry 930 is a time division multiplexed signal of the form

$$y2 = \alpha_2 w_\alpha + \beta_2 w_\beta + \gamma_2 w_\gamma + \delta_2 w_\delta.$$

The adder circuitry 950 receives the output bitstream signals y1 and y2 and combines them into a combined output bitstream y containing both of the bitstream signals y1 and y2. By varying the dither between the first and second vector quantizer circuitry 920, 940 (e.g. as discussed above in relation to the vector quantizer circuitry 600 of Figure 6), it is possible to ensure that the bitstream signals y1 and y2 are different.

The circuitry 900 of Figure 9 is particularly suited to applications where the same weights are to be applied to different sets of input signals, for example circuitry that implements a convolutional neural network (CNN) or the like.

It will be appreciated that the circuitry 900 illustrated in Figure 9 could be extended to include further multiplexer circuitry, each additional instance of multiplexer circuitry having an associated instance of vector quantizer circuitry. For example, in an implementation having six instances of multiplexer circuitry for receiving six sets of input

signals, there will also be six instances of vector quantizer circuitry for receiving a common set of weighting signals to be applied to each set of input signals.

Figure 10 is a schematic diagram illustrating further alternative circuitry for performing a MAC operation.

The circuitry, shown generally at 1000 in Figure 10, includes first multiplexer circuitry 1010 having a plurality of inputs for receiving a first plurality of unary coded bitstreams α_1, β_1, γ_1, δ_1 representing operands of a first MAC operation and second multiplexer circuitry 1020 having a plurality of inputs for receiving a second plurality of unary coded bitstreams α_2, β_2, γ_2, δ_2 representing operands of a second MAC operation.

The circuitry 1000 further includes vector quantizer circuitry 1030 which is common to both the first and the second multiplexer circuitry 1010, 1020. The vector quantizer circuitry 1030 has a plurality of inputs for receiving a plurality of weighting signals w_α, w_β, w_γ, w_δ, each weighting signal representing a proportion of a computation time period for which a respective unary coded bitstream of each of the first and second plurality of input unary bitstreams should be selected by the first and second multiplexer circuitry 1010, 1020 in order to correctly perform the first and second MAC operations using the operands.

As in the circuitry 900, the circuitry 1000 includes adder circuitry configured to receive output bitstream signals y1, y2 of the first and second multiplexer circuitry 1010, 1020 and to combine the received bitstreams into a single output bitstream y representing the sum of the results of the combination first and second MAC operations.

The circuitry 1000 may further include integrator circuitry 530 and an encoder 540 of the kind described above with reference to Figure 5, for integrating the output signal y and encoding the integrated signal into a unary coded signal. Alternatively, the circuitry 1000 may further comprise CEM encoder circuitry 700 of the kind described above with reference to Figure 7, for encoding the output signal y as a CEM signal.

The vector quantizer circuitry 1030 may comprise vector quantizer circuitry of the kind described above with reference to Figure 6, or alternatively may comprise circuitry implementing some other vector quantizer architecture such as a tree-based modulator.

In operation of the circuitry 1000, the vector quantizer circuitry 1030 receives the plurality of weighting signals w_α, w_β, w_γ, w_δ and outputs a first selector signal $\theta 1$ to cause the first multiplexer circuitry 1010 to select each input signal α_1, β_1, γ_1, δ_1 for the correct proportion of each computation period, such that over a given computation period the output y1 of the first multiplexer circuitry 1010 is a time division multiplexed signal of the form

$$y1 = \alpha_1 w_\alpha + \beta_1 w_\beta + \gamma_1 w_\gamma + \delta_1 w_\delta.$$

The vector quantizer circuitry 1030 also outputs a second selector signal $\theta 2$ to cause the second multiplexer circuitry 1020 to select each input signal α_2, β_2, γ_2, δ_2 for the correct proportion of each computation period, such that over a given computation period the output y2 of the second multiplexer circuitry 1020 is a time division multiplexed signal of the form

$$y2 = \alpha_2 w_\alpha + \beta_2 w_\beta + \gamma_2 w_\gamma + \delta_2 w_\delta.$$

The adder circuitry 1040 receives the output bitstream signals y1 and y2 and combines them into a combined output bitstream y containing both of the bitstream signals y1 and y2.

The circuitry 1000 of Figure 10 is also well suited to applications where the same weights are to be applied to different sets of input signals, for example circuitry that implements a convolutional neural network (CNN) or the like. In comparison with the circuitry 900 of Figure 9, the circuitry 1000 of Figure 10 has reduced power consumption, because instead of having a plurality of multiplexers, each having an associated vector quantizer, a single vector quantizer can be shared between multiple (two, in the example of Figure 10) multiplexers.

As before, it will be appreciated that the circuitry 1000 could be extended to include more instances of multiplexer circuitry, with additional instances of vector quantizer circuitry as necessary to support and control the additional multiplexer circuitry.

An architecture comprising a vector quantizer and a plurality of multiplexers is particularly suited for use in circuitry implementing a convolution layer of a convolutional neural network (CNN).

Figure 11 is a schematic diagram illustrating functional blocks of a typical CNN. As shown, the CNN 1100 includes a convolution block 1110, a non-linear function block 1120 and a pooling block 1130.

The convolution block 1110 is configured to apply a kernel (e.g. a common set of weights) to different subsets of a set of input data. The non-linear function block 1120 is configured to receive output data from the convolution block and to perform a non-linear activation function such as a rectified linear unit (ReLU) activation function on the received data. The pooling block 1130 is configured to perform a pooling function on data output by the non-linear function block 1120.

Figure 12 is a schematic diagram illustrating example operations performed by circuitry implementing a convolutional neural network (CNN).

In order to perform convolution of the kernel with the input data set, the kernel is applied in turn to different subsets of the input data set (which in some examples may be overlapping subsets of the input data set), typically using a matrix multiplication operation. The matrix multiplication operations are represented in Figure 12 by first and second convolution operations 1210, 1220 (though it will be appreciated by those of ordinary skill in the art that a typical convolution block or layer will perform more than two such convolution operations).

As shown, a first convolution operation 1210 is performed on a first subset (e.g. a first vector) D1 of input data and a kernel W1 of weight data. A first non-linear activation function 1220 is performed on the result of the first convolution operation 1210. A second convolution operation 1230 is performed on a second subset (e.g. a second vector) D2 of the input data and the kernel W1, and a second non-linear activation function 1240 is performed on the result of the second convolution operation 1230.

As will be apparent from the foregoing disclosure, because the convolution operations 1210, 1230 are matrix multiplication operations, they can be implemented as multiply-accumulate (MAC) operations using stochastic computing techniques, and thus can be implemented in circuitry using a combination of a plurality of multiplexers and a vector quantizer that is coupled to each of the plurality of multiplexers, as shown in **Figure 13**.

The circuitry, shown generally at 1300 in Figure 13, comprises vector quantizer circuitry 1310 and first and second multiplexer circuitry 1320, 1330.

The vector quantizer circuitry 1310 is configured to receive the kernel W1 as a set of weighting signals w_α, w_β, w_γ, w_δ. The first multiplexer circuitry 1320 is configured to receive the first subset D1 as a first set of unary coded bitstreams α_1, β_1, γ_1, δ_1, and the second multiplexer circuitry 1330 is configured to receive the second subset D2 of input data as a second set of unary coded bitstreams α_2, β_2, γ_2, δ_2 of input data.

The first and second multiplexer circuitry 1320, 1330 are coupled to the vector quantizer circuitry 1310 so as to receive respective first and second selector signals θ1, θ2 from the vector quantizer circuitry 1310.

In operation of the circuitry 1300, the vector quantizer circuitry 1310 outputs the first selector signal θ1 to cause the first multiplexer circuitry 1320 to select each input signal α_1, β_1, γ_1, δ_1 for the correct proportion of each computation period, such that over a given computation period the output y1 of the first multiplexer circuitry 1320 is a time division multiplexed signal of the form

$$y1 = \alpha_1 w_\alpha + \beta_1 w_\beta + \gamma_1 w_\gamma + \delta_1 w_\delta.$$

Similarly, the vector quantizer circuitry 1310 outputs the second selector signal θ2 to cause the second multiplexer circuitry 1330 to select each input signal α_2, β_2, γ_2, δ_2 for the correct proportion of each computation period, such that over a given computation period the output y2 of the second multiplexer circuitry 1330 is a time division multiplexed signal of the form

$$y2 = \alpha_2 w_\alpha + \beta_2 w_\beta + \gamma_2 w_\gamma + \delta_2 w_\delta.$$

As will be appreciated, in a practical implementation of a CNN, the convolution block or layer will be required to perform more than two matrix multiplication operations in order to perform convolution of the whole input data set with the kernel. Each additional matrix multiplication operation can be implemented simply by coupling an additional multiplexer to the vector quantizer circuitry 1310. Thus, if the input data set were divided into four subsets D1 – D4, the circuitry 1300 would include four multiplexers coupled to the vector

quantizer circuitry 1310, each multiplexer receiving a different one of the subsets D1 – D4 of input the input data.

Thus the architecture shown in Figure 13 permits low-power performance of the convolution operations required by the convolution block or layer of a CNN.

As discussed briefly above, the use of a consecutive edge modulated (CEM) signal facilitates implementation of a non-linear activation function.

Figure 14 is a schematic representation of circuitry for implementing a non-linear function (such as the non-linear functions 1120, 1220, 1240 of Figures 11 – 13). The circuitry, shown generally at 1400 in Figure 14, comprises first CEM modulator circuitry 1410, an OR gate 1420, and second CEM modulator circuitry 1430.

The first CEM modulator circuitry 1410 receives a unary coded signal representing the value 0.5, and outputs a corresponding CEM signal to a first input of the OR gate 1420. A second input of the OR gate 1420 receives the signal output by a multiplexer (e.g. the first or second multiplexer 1310, 1320 of Figure 13). The output of the OR gate 1420 is coupled to an input of the second CEM modulator circuitry 1430, which outputs a CEM modulated signal representing the signal received at its input.

The circuitry 1400 has a transfer function of the form shown in **Figure 15**, which is corresponds to the transfer function of a ReLU function.

Figure 16 is a schematic representation of circuitry that may be used to implement a pooling block or layer in circuitry implementing a CNN (e.g. the pooling block 1130 of Figure 11).

The circuitry, shown generally at 1600 in Figure 16, is configured to perform mean pooling, and includes multiplexer circuitry 1610, vector quantizer circuitry 1620 and CEM modulator circuitry 1630.

The multiplexer circuitry 1610 is configured to receive a plurality (in this example four) of CEM encoded input signals $d_1 – d_4$, each output by non-linear function circuitry (e.g. the circuitry 1400 of Figure 14). A control input of the multiplexer circuitry 1610 is coupled

to an output of the vector quantizer circuitry 1620 so as to receive a selector signal θ from the vector quantizer circuitry 1620.

The vector quantizer circuitry 1620 is configured to receive one or more input signals representing a desired weighing to be applied to each of the CEM encoded input signals. In this example, because the circuitry 1600 is configured to perform mean pooling and there are four CEM encoded input signals $d_1 – d_4$, the input signal represents the value ¼. The selector signal θ output by the vector quantizer circuitry 1620 thus causes the multiplexer circuitry 1610 to select each of the CEM encoded input signals $d_1 – d_4$ for one quarter of each computation period, such that over a given computation period the output y of the first multiplexer circuitry 1610 is a time division multiplexed signal of the form

$$y = ¼d_1 + ¼d_2 + ¼d_3 + ¼d_4.$$

More generally, the selector signal θ output by the vector quantizer circuitry 1620 causes the multiplexer circuitry 1610 to select each of the plurality of CEM encoded input signals for an equal proportion of the computation period. Thus, if there are N input signals to the multiplexer circuitry 1610, then the input signal to the vector quantizer circuitry will represent a value 1/N.

An output of the multiplexer circuitry 1610 is coupled to an input of the CEM modulator circuitry 1630. The CEM modulator circuitry 1630 thus outputs a CEM modulated signal representing the signal y output by the multiplexer circuitry 1610.

As will be appreciated, the present disclosure describes efficient architectures for performing MAC operations using unary coded input signals. Such architectures are particularly suited for use in circuitry implementing a convolutional neural network (CNN).

There are a variety of unary codes that can be used for the input unary coded bitstream signals to the multiplexer circuitry in the circuitry 500, 900, 1000 of the present disclosure. Any unary coding scheme may be used to generate the input bitstream signals, and different unary codes may be used for different ones of the plurality of input unary coded bitstream signals. Examples of suitable unary codes include:

Stochastic codes, which are generated by comparing a random sequence to a reference value to generate a sequence with the correct mean value;

Pseudo-random codes. Here instead of comparing the input to a real random number generator, the comparison is performed between the input and pseudo-random number such as generated from a LFSR (Linear Feedback Shift Register);

Low-Discrepancy codes. These are quasi-random codes that aim to improve on LFSR by converging faster, at the cost of additional complexity required to generate them. Commonly used methods include Halton and Sobol sequences;

Pulse Width Modulation (PWM). Here the data is encoded in the duty cycle of the waveform;

Pulse Density Modulation (PDM). Here the information is encoded in the how often the pulses are high. Typically generated via a sigma delta modulator.

Rate coding. This is a neuromorphically inspired method where the pulses are ended by how often they occur. They can be generated via an Integrate and Fire Neuron.

Time encoding modulation. Also known as asynchronous sigma delta modulation. This method encodes the information in the duty cycle, but the duty cycle scales with the amplitude to provide more accuracy.

Consecutive Edge Modulation (CEM). Here the information is encoded in duty cycle, but spread over multiple cycles to minimise the number of edges. Hence it is an optimal method for unary computing since it has a minimum number of edges.

Binary Code modulation (BCM). Here the data is encoded considering each bit of the word to be encoded in turn, producing a number of pulses equal to two the power of its position, with same value as the bit. For example 1 sample of an LSB is sent (which can be high or low), then 2 samples of the next bit, then 4 samples of the next bit, then 8 of the next etc., until the whole value has been encoded. For example, the binary number 1001 would be coded as the sequence 1, then 00, then 0000, then 11111111 i.e. 100000011111111, which has a mean value of 9.

In the examples described above the unary coded bitstreams are described in terms of a unipolar coding scheme in which a value is represented by a stream of bits which can

take the value 0 or 1. It will be appreciated by those of ordinary skill in the art that a bipolar coding scheme in which a value is represented by a stream of bits that can take the value -1 or +1, for example.

The circuitry and/or converter of the present disclosure may be used in a wide variety of applications. For example, the circuitry and/or converter may be used as part of a glance detection system, e.g. a system which monitors the gaze of a user to detect where the user is looking. A glance detection system of this kind may be used, for example, to cause a device such as a mobile telephone to switch between operational modes (e.g. a sleep mode and an active mode) when the user looks at the device or a region or area of the device.

Embodiments may be implemented as an integrated circuit which in some examples could be a codec or audio DSP or similar. Embodiments may be incorporated in an electronic device, which may for example be a portable device and/or a device operable with battery power. The device could be a communication device such as a mobile telephone or smartphone or similar. The device could be a computing device such as a notebook, laptop or tablet computing device. The device could be a wearable device such as a smartwatch. The device could be a device with voice control or activation functionality such as a smart speaker. In some instances the device could be an accessory device such as a headset, headphones, earphones, earbuds or the like to be used with some other product.

The skilled person will recognise that some aspects of the above-described apparatus and methods, for example the discovery and configuration methods may be embodied as processor control code, for example on a non-volatile carrier medium such as a disk, CD- or DVD-ROM, programmed memory such as read only memory (Firmware), or on a data carrier such as an optical or electrical signal carrier. For many applications, embodiments will be implemented on a DSP (Digital Signal Processor), ASIC (Application Specific Integrated Circuit) or FPGA (Field Programmable Gate Array). Thus the code may comprise conventional program code or microcode or, for example code for setting up or controlling an ASIC or FPGA. The code may also comprise code for dynamically configuring re-configurable apparatus such as re-programmable logic gate arrays. Similarly the code may comprise code for a hardware description language such as Verilog™ or VHDL (Very high speed integrated circuit Hardware Description Language). As the skilled person will appreciate, the code may be distributed between

a plurality of coupled components in communication with one another. Where appropriate, the embodiments may also be implemented using code running on a field-(re)programmable analogue array or similar device in order to configure analogue hardware.

It should be noted that the above-mentioned embodiments illustrate rather than limit the invention, and that those skilled in the art will be able to design many alternative embodiments without departing from the scope of the appended claims. The word "comprising" does not exclude the presence of elements or steps other than those listed in a claim, "a" or "an" does not exclude a plurality, and a single feature or other unit may fulfil the functions of several units recited in the claims. Any reference numerals or labels in the claims shall not be construed so as to limit their scope.

CLAIMS

1. Circuitry for performing a multiply-accumulate (MAC) operation, the circuitry comprising:

a first multiplexer having a plurality of inputs for receiving a plurality of unary-coded input signals representing operands of the MAC operation and an output for outputting a multiplexer output signal representing a result of the MAC operation; and

a first vector quantizer configured to receive a plurality of weighting signals, each representing a proportion of a computation time period for which a respective one of the unary-coded input signals should be selected by the multiplexer and to output a first selector signal to the multiplexer to cause the multiplexer to select each of the input signals in accordance with the plurality of weighting signals.

2. Circuitry according to claim 1, wherein the circuitry further comprises an integrator for integrating the multiplexer output signal over a period of time.

3. Circuitry according to claim 2, wherein the circuitry further comprises an encoder for encoding a signal output by the integrator into a unary coded output signal.

4. Circuitry according to any of the preceding claims, further comprising a consecutive edge modulator (CEM) configured to encode the multiplexer output signal into a consecutive edge modulated signal.

5. Circuitry according to claim 4, wherein the CEM comprises:

a first counter configured to generate a first pulse width modulated signal based on the number of high bits in a first frame of the multiplexer output signal;

a second counter configured to generate a second pulse width modulated signal based on the number of high bits in a second frame of the multiplexer output signal;

a selector configured to select the first pulse width modulated signal as a first frame of a CEM output signal and to invert the second pulse width modulated signal in time to generate a second frame of the CEM output signal.

6. Circuitry according to claim 5, wherein the second frame of the multiplexer output signal immediately follows the first frame of the multiplexer output signal, and wherein

the second frame of the CEM output signal immediately follows the first frame of the CEM output signal.

7. Circuitry according to claim 5 or claim 6, wherein the first frame of the multiplexer output signal is an odd-numbered frame of the multiplexer output signal and the second frame of the multiplexer output signal is an even-numbered frame of the multiplexer output signal.

8. Circuitry according to any of the preceding claims, wherein the first vector quantizer comprises:

a plurality of integrators, each configured to receive a respective weighting signal; and

a processing unit configured to, in each cycle of a computation period of the circuitry:

determine a maximum among the outputs of the plurality of integrators;

output a control signal to the first multiplexer to cause the first multiplexer to select an input corresponding to the weighting signal received by the integrator whose output is maximum; and

reset the integrator whose output is maximum.

9. Circuitry according to claim 8, wherein resetting the integrator comprises resetting the output of the integrator to 0 or to a random value.

10. Circuitry according to any of the preceding claims, wherein the first vector quantizer comprises a tree-based modulator.

11. Circuitry according to any of the preceding claims, further comprising a second multiplexer configured to receive a second selector signal from the first vector quantizer.

12. Circuitry according to any of the preceding claims, further comprising a second multiplexer and a second vector quantizer, the second multiplexer being configured to receive a second selector signal from the second vector quantizer.

13. Circuitry according to any of the preceding claims, wherein each of the plurality of unary-coded input signals is encoded according to a common encoding scheme.

14. Circuitry according to any of the preceding claims, wherein at least one of the plurality of unary-encoded input signals is encoded according to a different encoding scheme than at least one other of the plurality of unary-encoded input signals.

15. Convolutional Neural Network (CNN) circuitry comprising MAC circuitry according to any of the preceding claims.

16. A stochastic to consecutive edge modulation (CEM) converter comprising:

a first counter configured to generate a first pulse width modulated signal based on the number of high bits in a first frame of a received stochastic signal;

a second counter configured to generate a second pulse width modulated signal based on the number of high bits in a second frame of the received stochastic signal; and

a selector configured to select the first pulse width modulated signal as a first frame of a CEM output signal and to invert the second pulse width modulated signal in time to generate a second frame of the CEM output signal.

17. An integrated circuit comprising the circuitry of any of claims 1 – 15.

18. A device comprising the circuitry of any of claims 1 – 15.

19. A device according to claim 18, wherein the device comprises a portable device, a battery powered device, a mobile telephone, a tablet or laptop computer, a smart speaker, an accessory device, a headset device, smart glasses, headphones, earphones or earbuds.

20. Circuitry for performing a multiply-accumulate (MAC) operation, the circuitry comprising multiplexer circuitry configured to output a time division multiplexed output signal comprising portions of a plurality of unary coded multiplexer input signals selected according to a control signal received by the multiplexer circuitry from vector quantizer circuitry.

21. Circuitry for performing a convolution operation between a kernel and a set of data, the circuitry comprising:

vector quantizer circuitry configured to receive one or more input signals representing the kernel;

first multiplexer circuitry configured to receive a first plurality of unary coded input signals representing a first subset of the data; and

second multiplexer circuitry configured to receive a second plurality of unary coded input signals representing a second subset of the data,

wherein the vector quantizer circuitry is configured to:

output a first selector signal to the first multiplexer circuitry to cause the first multiplexer circuitry to select each of the first plurality of unary coded input signals for a respective proportion of a computation period, wherein the proportion of the computation period for which each of the first plurality of unary coded input signals is selected is based on the kernel; and

output a second selector signal to the first multiplexer circuitry to cause the first multiplexer circuitry to select each of the second plurality of unary coded input signals for a respective proportion of the computation period, wherein the proportion of the computation period for which each of the second plurality of unary coded input signals is selected is based on the kernel.

22. Circuitry for performing a mean pooling function for a plurality of unary coded data signals, the circuitry comprising:

multiplexer circuitry configured to receive the plurality of unary coded data signals; and

vector quantizer circuitry configured to output a selection signal to cause the multiplexer circuitry to select each of the plurality of input signals for an equal proportion of a computation period.

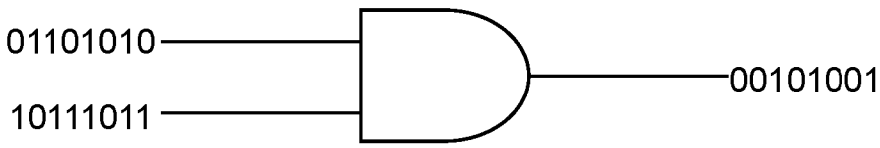


Figure 1

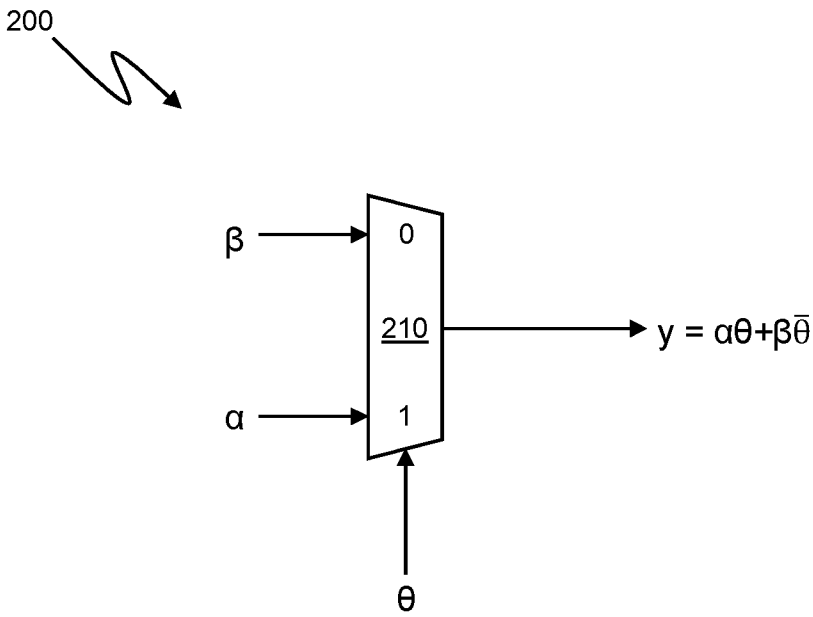


Figure 2

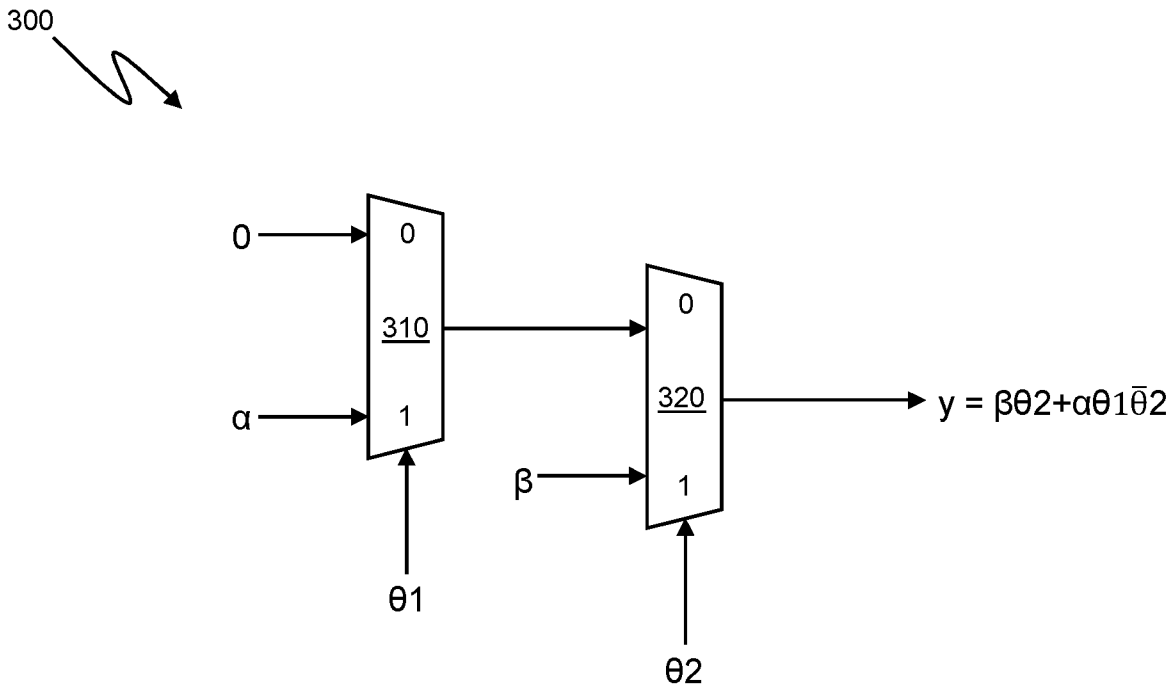


Figure 3

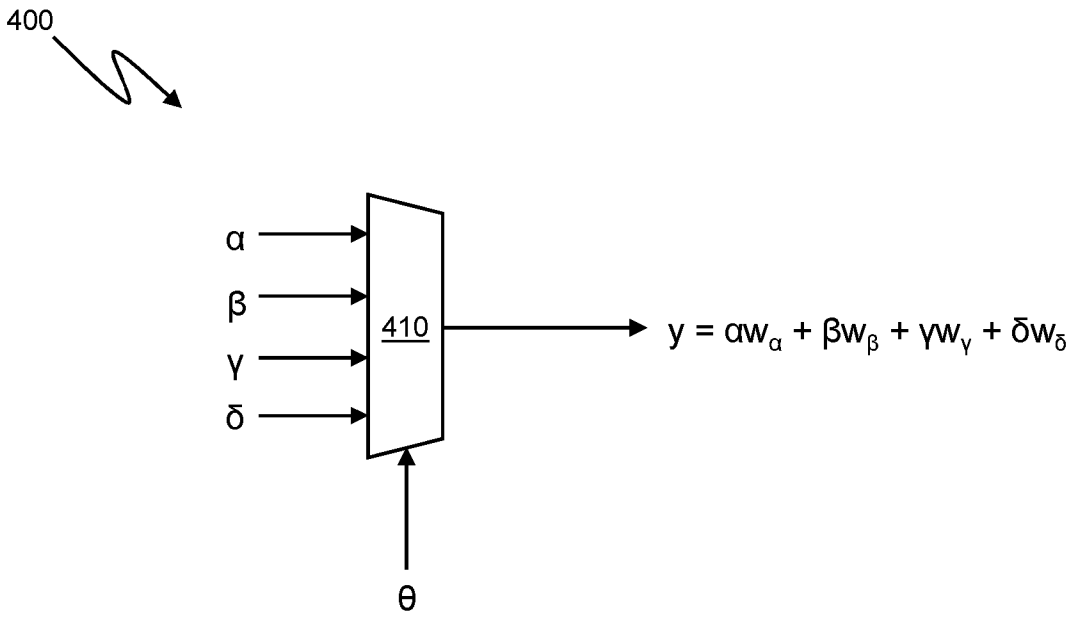


Figure 4

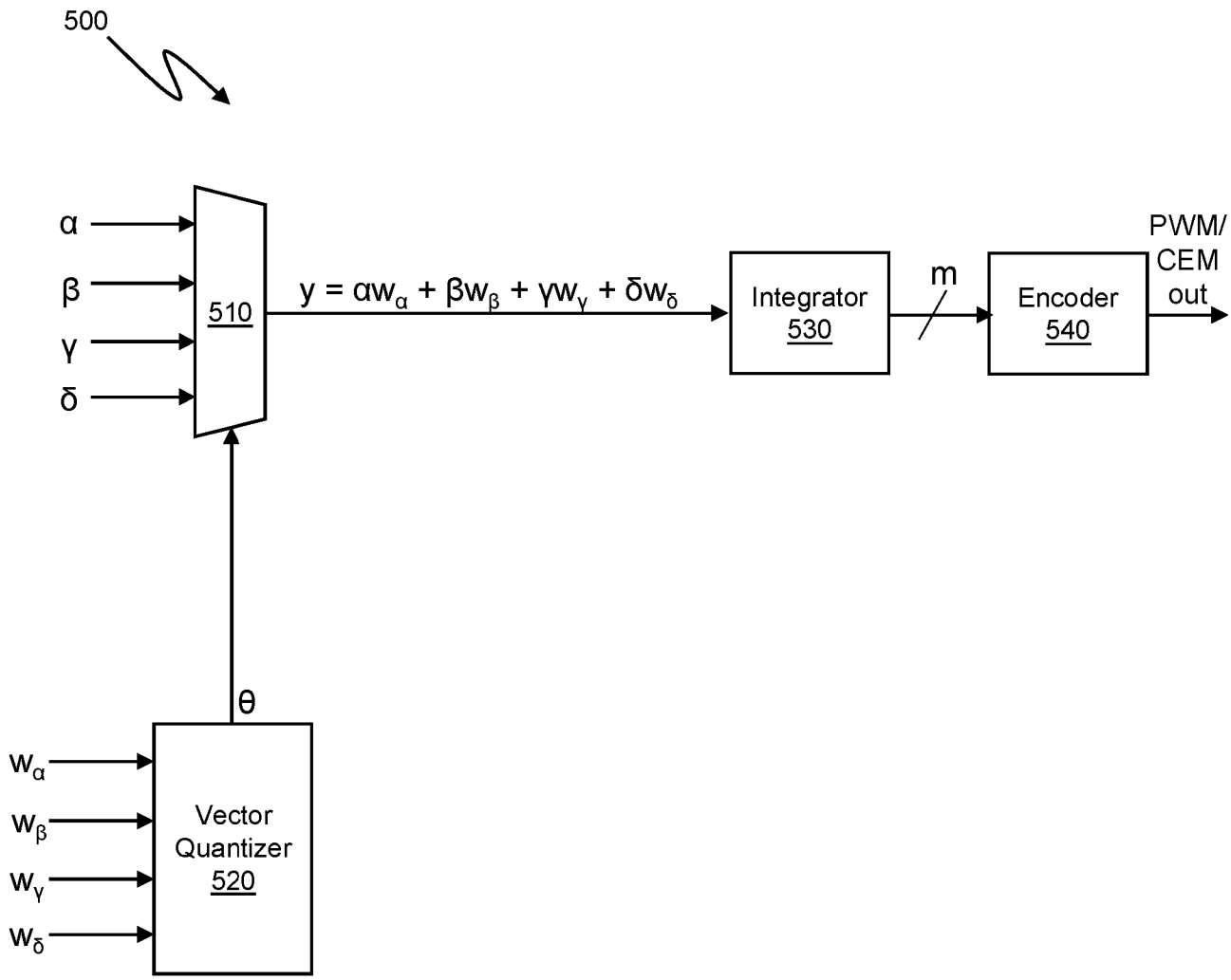

500
α
β
γ
δ
510
y = αw_α + βw_β + γw_γ + δw_δ
Integrator
530
m
Encoder
540
PWM/
CEM
out
θ
w_α
w_β
w_γ
w_δ
Vector
Quantizer
520

Figure 5

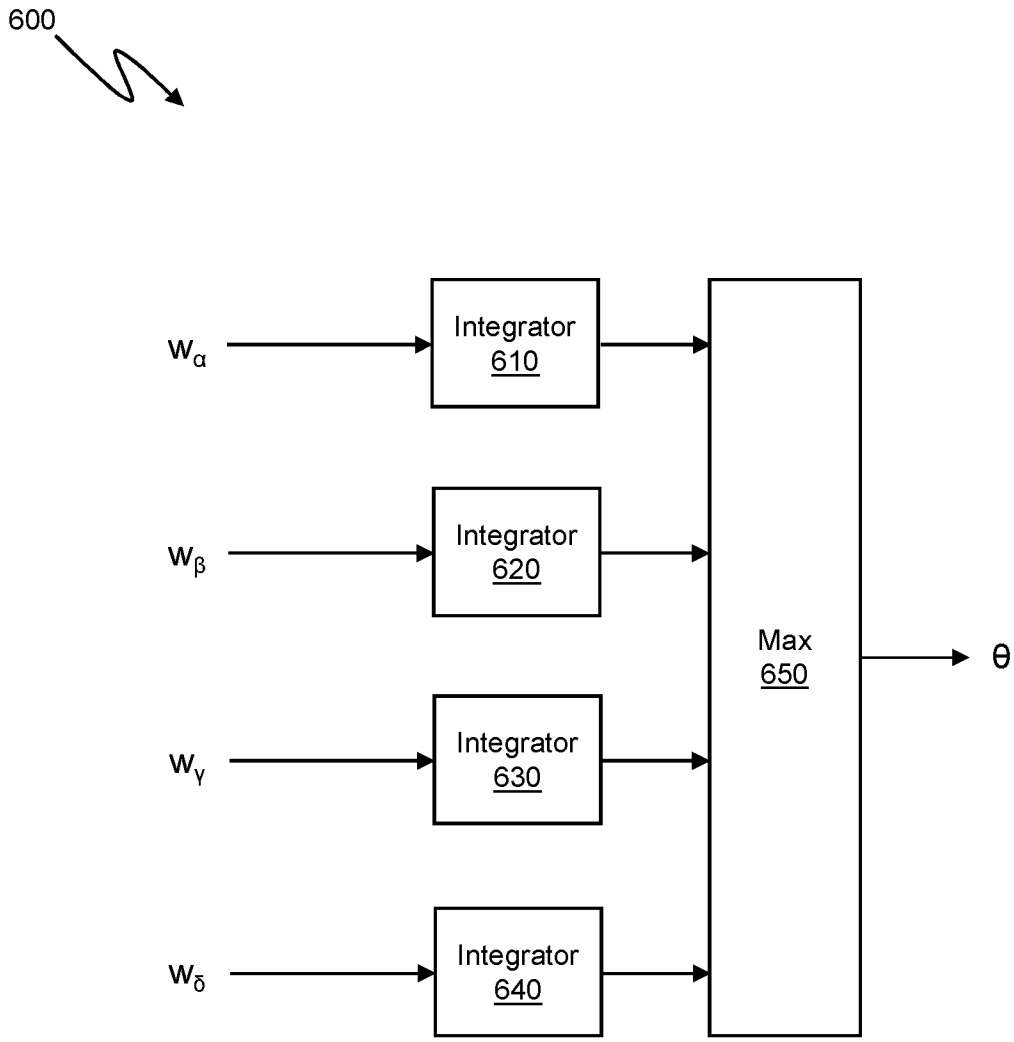

600
w_α
Integrator
610
w_β
Integrator
620
w_γ
Integrator
630
w_δ
Integrator
640
Max
650
θ

Figure 6

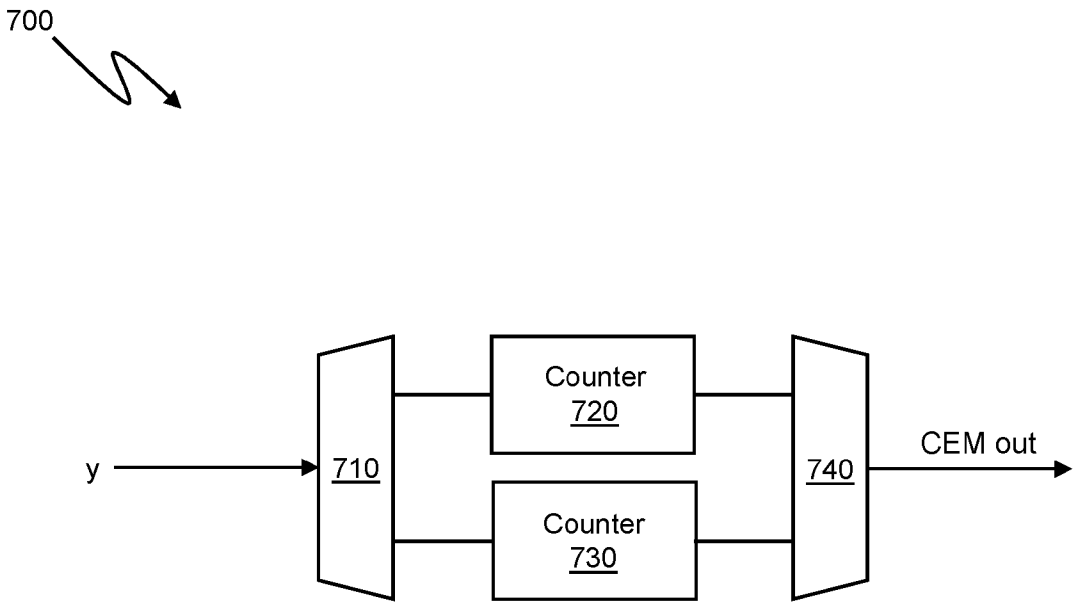


Figure 7

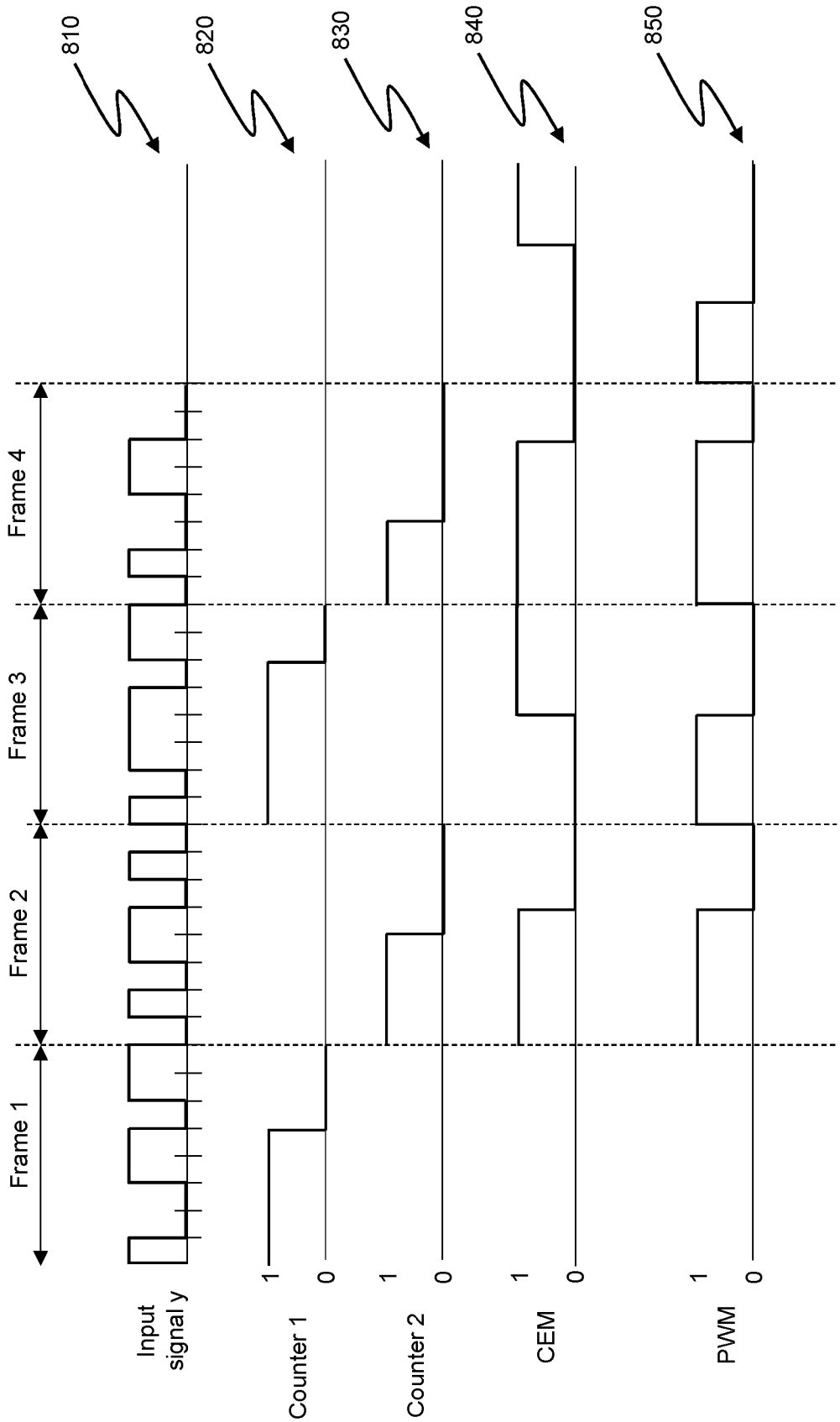

810
820
830
840
850
Frame 4
Frame 3
Frame 2
Frame 1
Input signal y
Counter 1
Counter 2
CEM
PWM
1
0

Figure 8

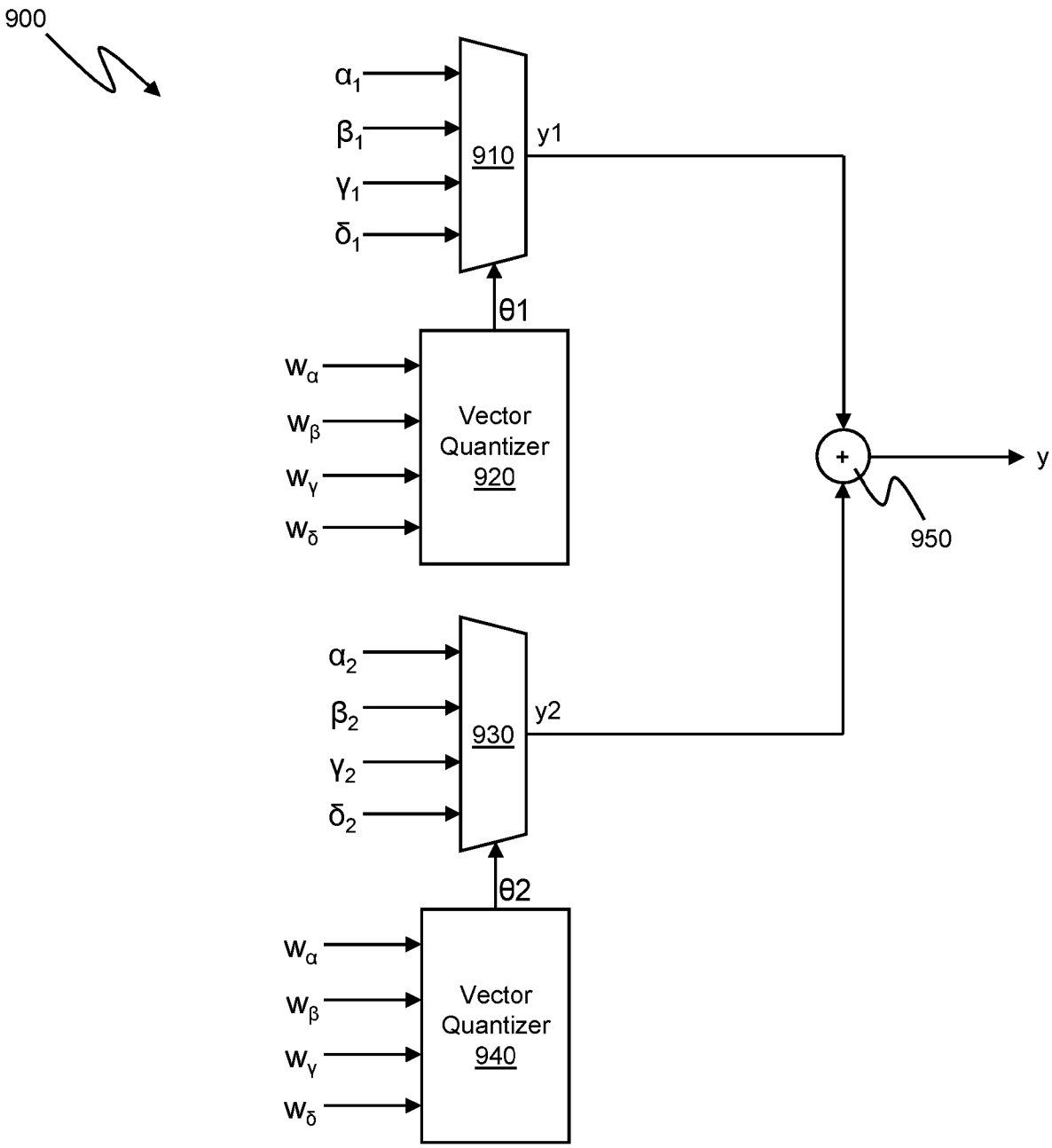

900
α1
β1
γ1
δ1
910
y1
θ1
wα
wβ
wγ
wδ
Vector Quantizer 920
+
y
950
α2
β2
γ2
δ2
930
y2
θ2
wα
wβ
wγ
wδ
Vector Quantizer 940

Figure 9

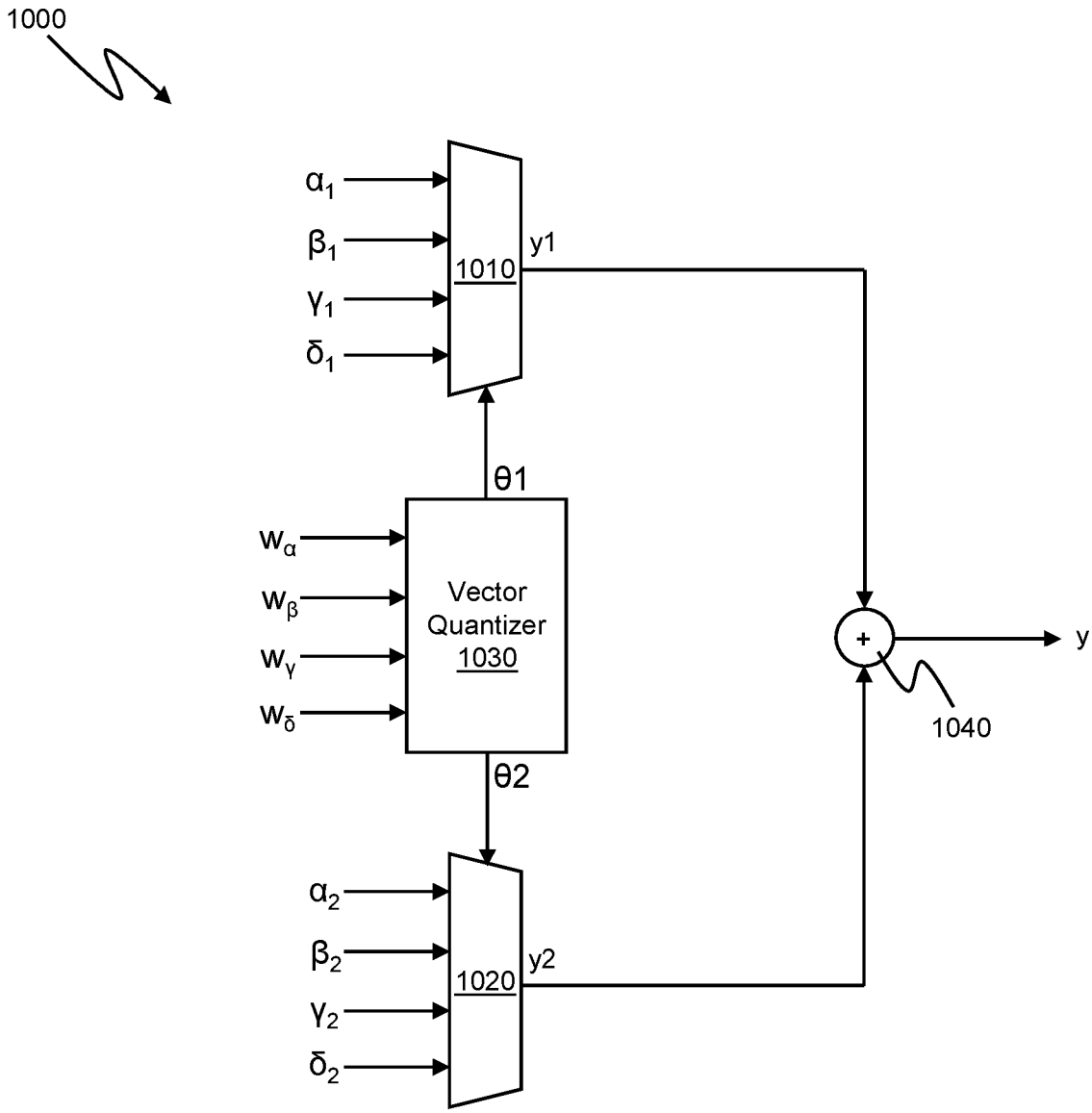

1000
α1
β1
γ1
δ1
1010
y1
θ1
wα
wβ
wγ
wδ
Vector
Quantizer
1030
θ2
α2
β2
γ2
δ2
1020
y2
+
y
1040

Figure 10

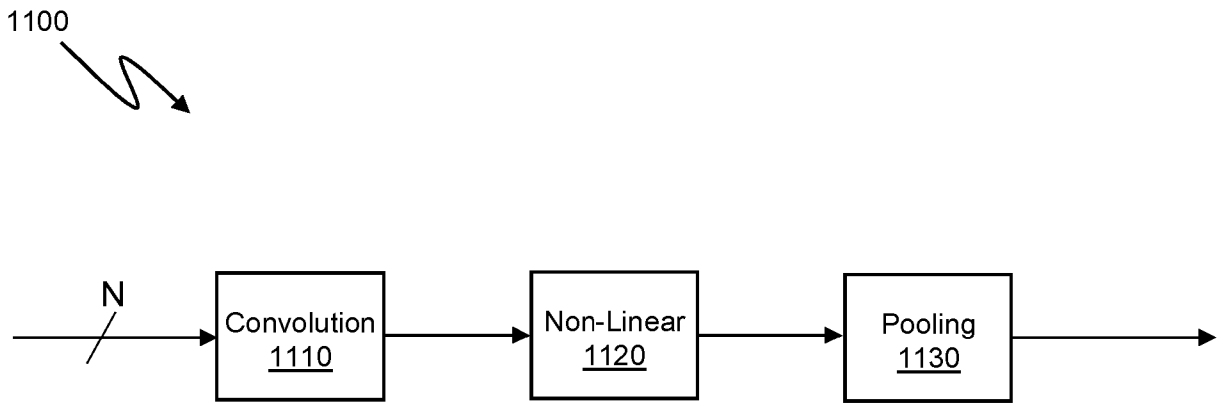


Figure 11

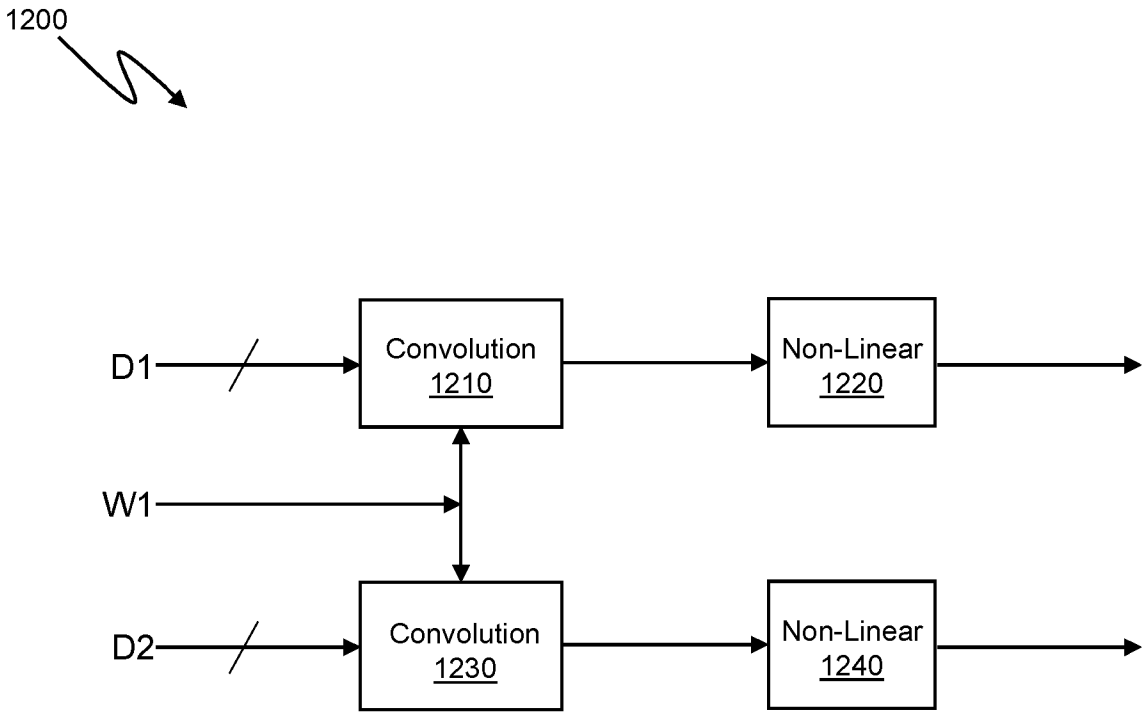

1200
D1
Convolution
1210
Non-Linear
1220
W1
D2
Convolution
1230
Non-Linear
1240

Figure 12

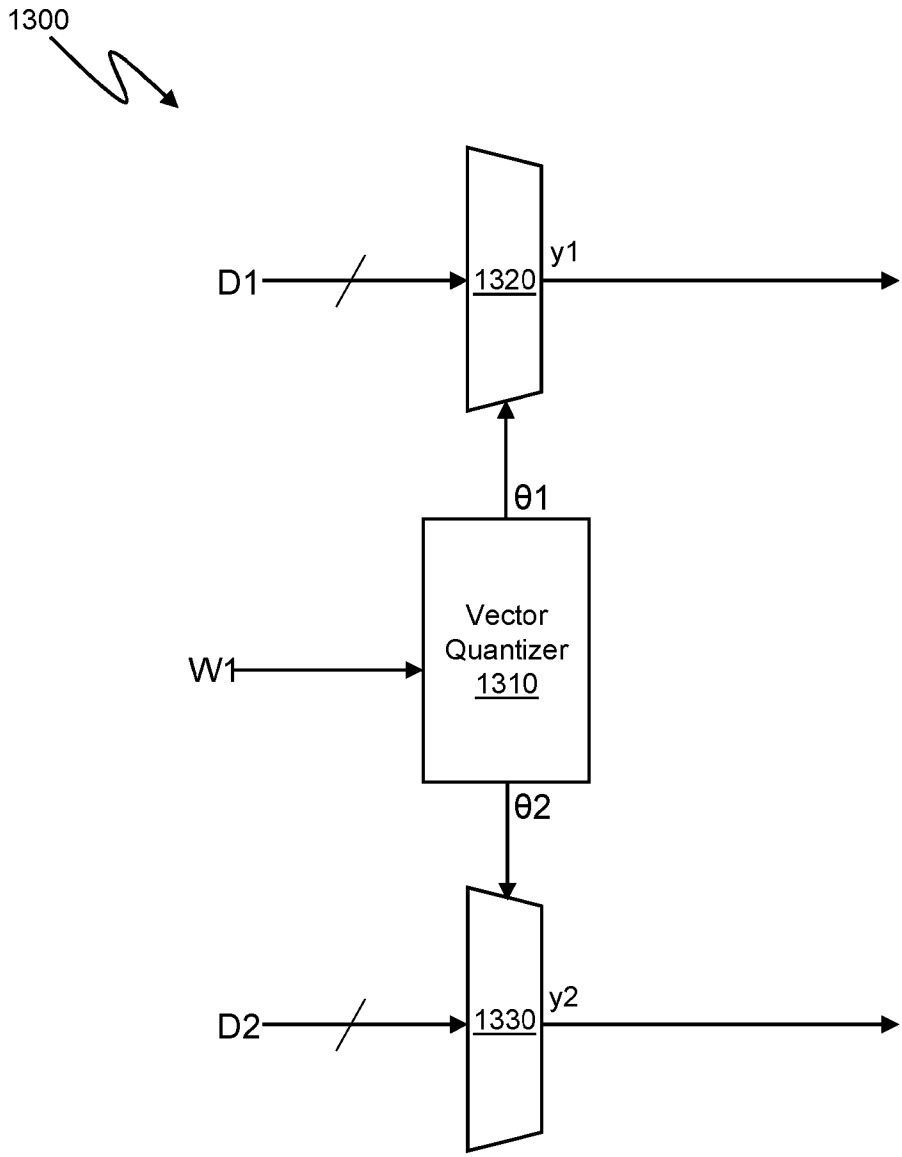


Figure 13

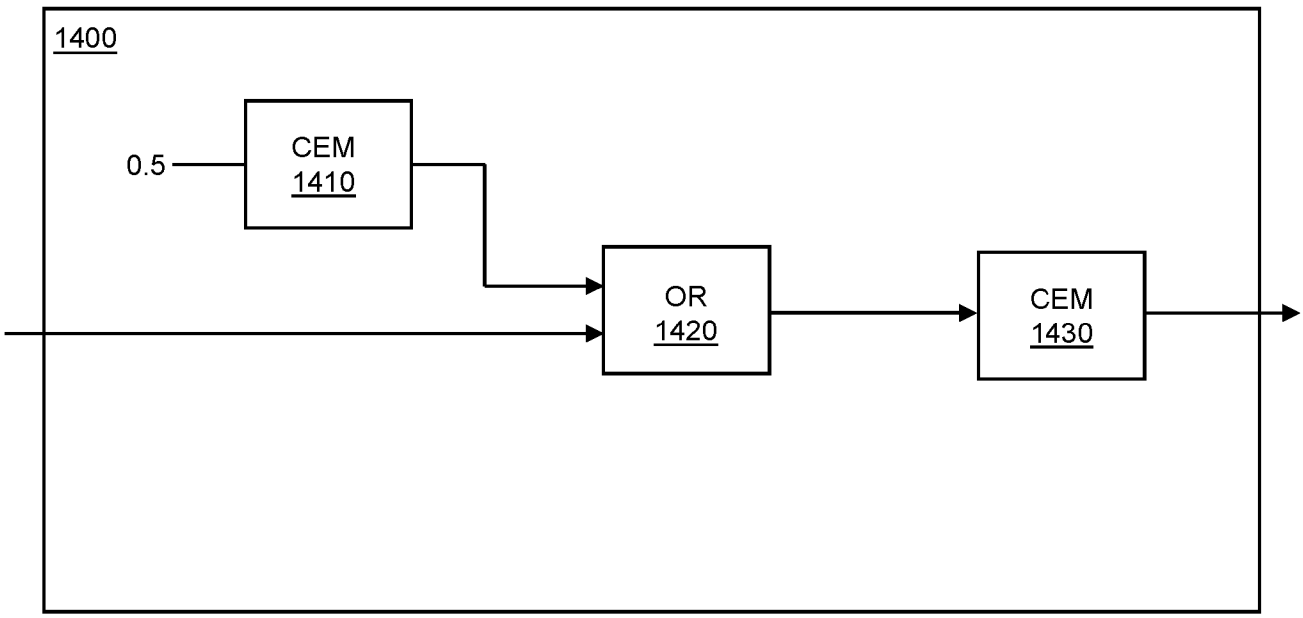


Figure 14

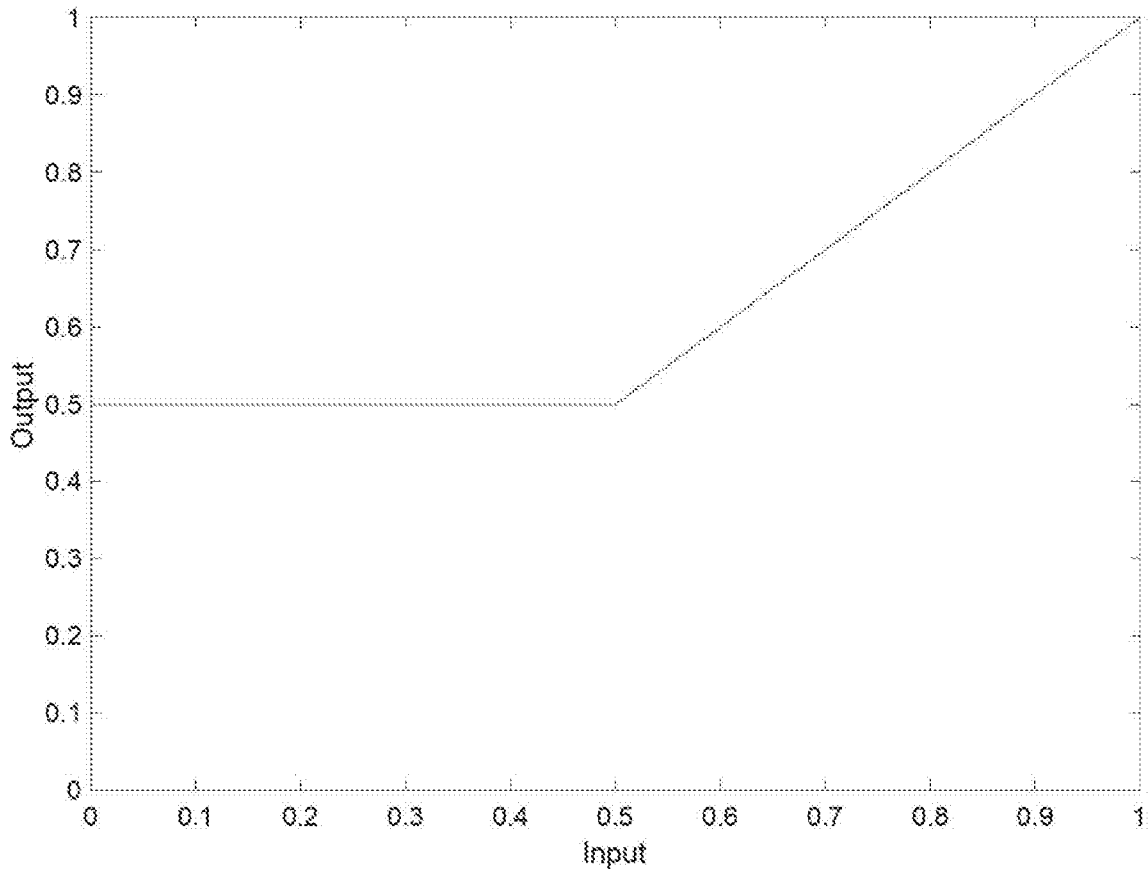


Figure 15

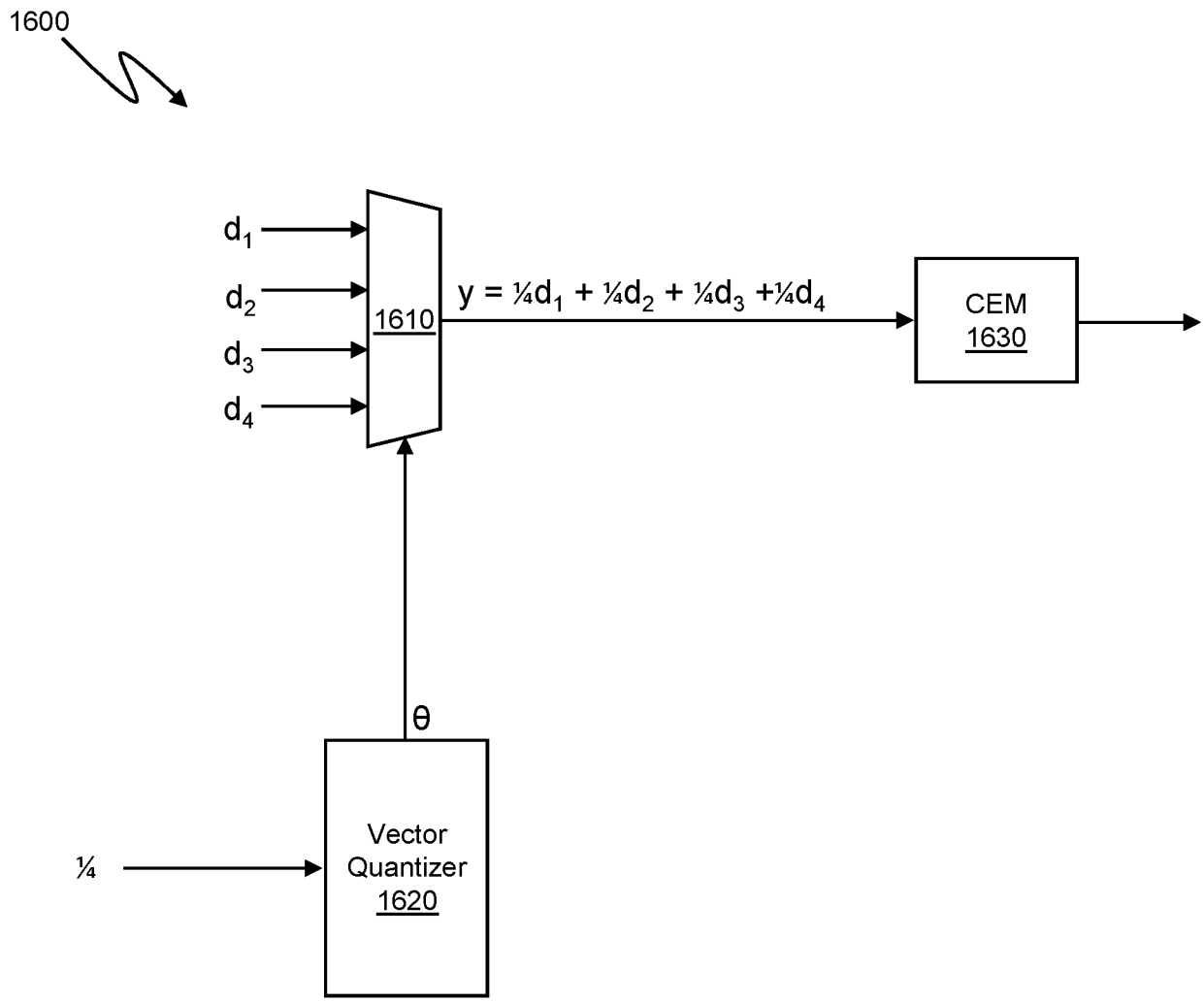

1600
d1
d2
d3
d4
1610
y = ¼d1 + ¼d2 + ¼d3 +¼d4
CEM
1630
θ
¼
Vector
Quantizer
1620

Figure 16

INTERNATIONAL SEARCH REPORT

International application No
PCT/GB2022/051592

A. CLASSIFICATION OF SUBJECT MATTER
INV. G06F7/544
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)
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

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	WELLER DENNIS D ET AL: "Printed Stochastic Computing Neural Networks", 2021 DESIGN, AUTOMATION & TEST IN EUROPE CONFERENCE & EXHIBITION (DATE), EDAA, 1 February 2021 (2021-02-01), pages 914-919, XP033941250, DOI: 10.23919/DATE51398.2021.9474254 Sections I, III; figure 5	1-15, 17-21

-/--

[X] 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	Date of mailing of the international search report
5 September 2022	03/11/2022

Name and mailing address of the ISA/	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016	Tenbieg, Christoph

1

Form PCT/ISA/210 (second sheet) (April 2005)

INTERNATIONAL SEARCH REPORT

International application No.
PCT/GB2022/051592

Box No. II Observations where certain claims were found unsearchable (Continuation of item 2 of first sheet)

This international search report has not been established in respect of certain claims under Article 17(2)(a) for the following reasons:

1. ☐ Claims Nos.:
because they relate to subject matter not required to be searched by this Authority, namely:

2. ☐ Claims Nos.:
because they relate to parts of the international application that do not comply with the prescribed requirements to such an extent that no meaningful international search can be carried out, specifically:

3. ☐ Claims Nos.:
because they are dependent claims and are not drafted in accordance with the second and third sentences of Rule 6.4(a).

Box No. III Observations where unity of invention is lacking (Continuation of item 3 of first sheet)

This International Searching Authority found multiple inventions in this international application, as follows:

see additional sheet

1. ☐ As all required additional search fees were timely paid by the applicant, this international search report covers all searchable claims.

2. ☐ As all searchable claims could be searched without effort justifying an additional fees, this Authority did not invite payment of additional fees.

3. ☐ As only some of the required additional search fees were timely paid by the applicant, this international search report covers only those claims for which fees were paid, specifically claims Nos.:

4. ☒ No required additional search fees were timely paid by the applicant. Consequently, this international search report is restricted to the invention first mentioned in the claims;; it is covered by claims Nos.:
1-15, 17-21

Remark on Protest

☐ The additional search fees were accompanied by the applicant's protest and, where applicable, the payment of a protest fee.

☐ The additional search fees were accompanied by the applicant's protest but the applicable protest fee was not paid within the time limit specified in the invitation.

☐ No protest accompanied the payment of additional search fees.

FURTHER INFORMATION CONTINUED FROM PCT/ISA/ 210

This International Searching Authority found multiple (groups of) inventions in this international application, as follows:

1. claims: 1-15, 17-21

Device for performing a multiply-accumulate operation.

---

2. claim: 16

Device for converting data

---

3. claim: 22

Device for performing a mean pooling function

INTERNATIONAL SEARCH REPORT

International application No
PCT/GB2022/051592

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	SIM HYEONUK ET AL: "Cost-effective stochastic MAC circuits for deep neural networks", NEURAL NETWORKS, ELSEVIER SCIENCE PUBLISHERS, BARKING, GB, vol. 117, 20 May 2019 (2019-05-20), pages 152-162, XP085751685, ISSN: 0893-6080, DOI: 10.1016/J.NEUNET.2019.04.017 [retrieved on 2019-05-20] Section 2 -----	1-15, 17-21
A	ALIREZA KHADEM: "Design Challenges of Neural Network Acceleration Using Stochastic Computing", ARXIV.ORG, CORNELL UNIVERSITY LIBRARY, 201 OLIN LIBRARY CORNELL UNIVERSITY ITHACA, NY 14853, 8 June 2020 (2020-06-08), XP081694077, Section II, IV, V -----	1-15, 17-21

1