US 20060104361A1

(19) United States

(12) Patent Application Publication (10) Pub. No.: US 2006/0104361 A1

Suzuki (43) Pub. Date: May 18, 2006

(54) VARIABLE-LENGTH CODING APPARATUS AND VARIABLE-LENGTH CODING METHOD

(76) Inventor: **Koichi Suzuki**, Tokyo (JP)

Correspondence Address:
COWAN LIEBOWITZ & LATMAN P.C.
JOHN J TORRENTE
1133 AVE OF THE AMERICAS
NEW YORK, NY 10036 (US)

(21) Appl. No.: **11/274,570**

(22) Filed: **Nov. 15, 2005**

(30) **Foreign Application Priority Data**

Nov. 15, 2004 (JP) 2004-331107

Publication Classification

(51) **Int. Cl.**

H04N 11/04	(2006.01)
H04B 1/66	(2006.01)
H04N 11/02	(2006.01)
H04N 7/12	(2006.01)

(52) **U.S. Cl.** **375/240.23**; 375/240.24

(57) **ABSTRACT**

A variable-length coding apparatus which receives image data block by block, and generates coded data from syntax elements includes a storage unit that temporarily stores the image data, a first syntax element calculating unit that calculates a first syntax element from the image data, at the same time the image data is stored in the storage unit, a first coding unit that generates coded data of the first syntax element from a value of the first syntax element, a second syntax element calculating unit that reads out the image data stored in the storage unit, and calculates a value of a second syntax element from the image data, a second coding unit that generates coded data of the second syntax element on the basis of the values of the first and second syntax elements, and a packing unit that connects the coded data of the first and second syntax elements.

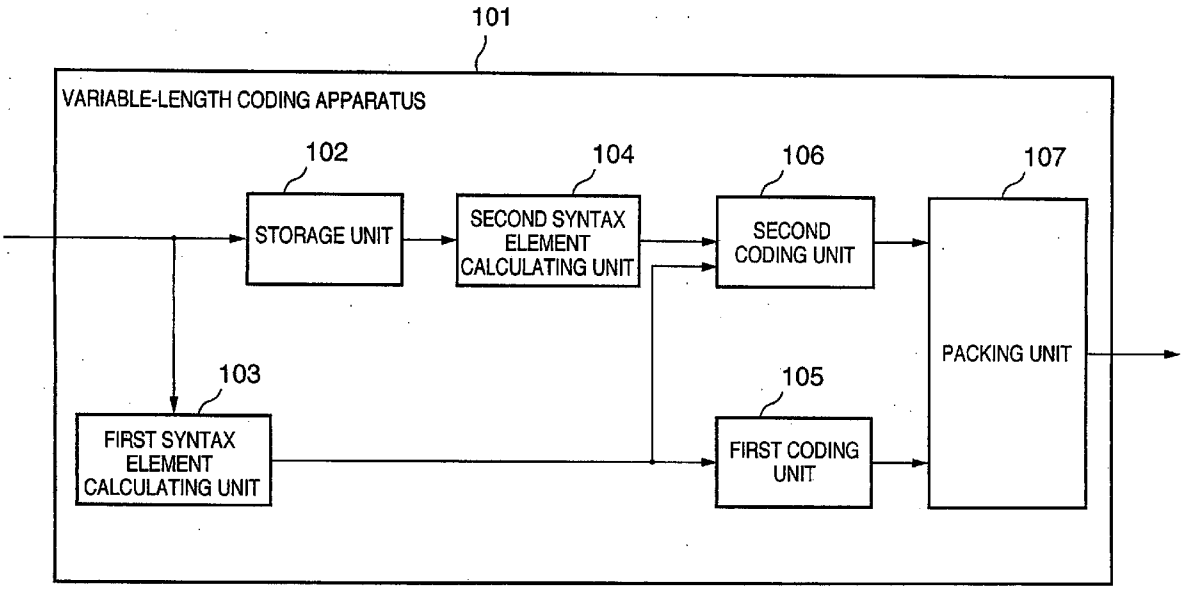
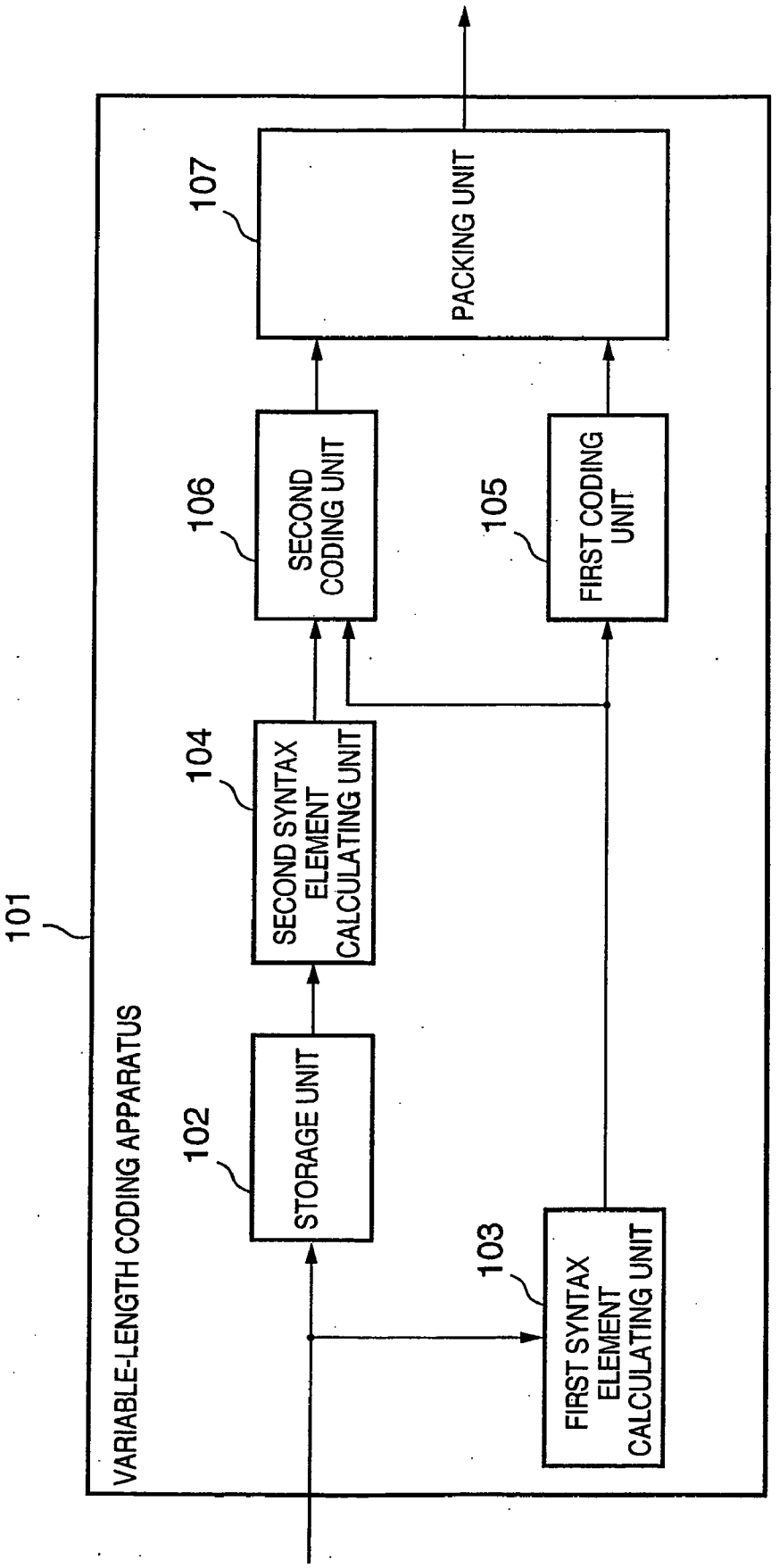


FIG. 1


101
VARIABLE-LENGTH CODING APPARATUS
102
STORAGE UNIT
103
FIRST SYNTAX ELEMENT CALCULATING UNIT
104
SECOND SYNTAX ELEMENT CALCULATING UNIT
105
FIRST CODING UNIT
106
SECOND CODING UNIT
107
PACKING UNIT

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

RASTER SCAN ORDER

FIG. 2A

16	15	11	10
14	12	9	4
13	8	5	3
7	6	2	1

REVERSE SCAN ORDER

FIG. 2B

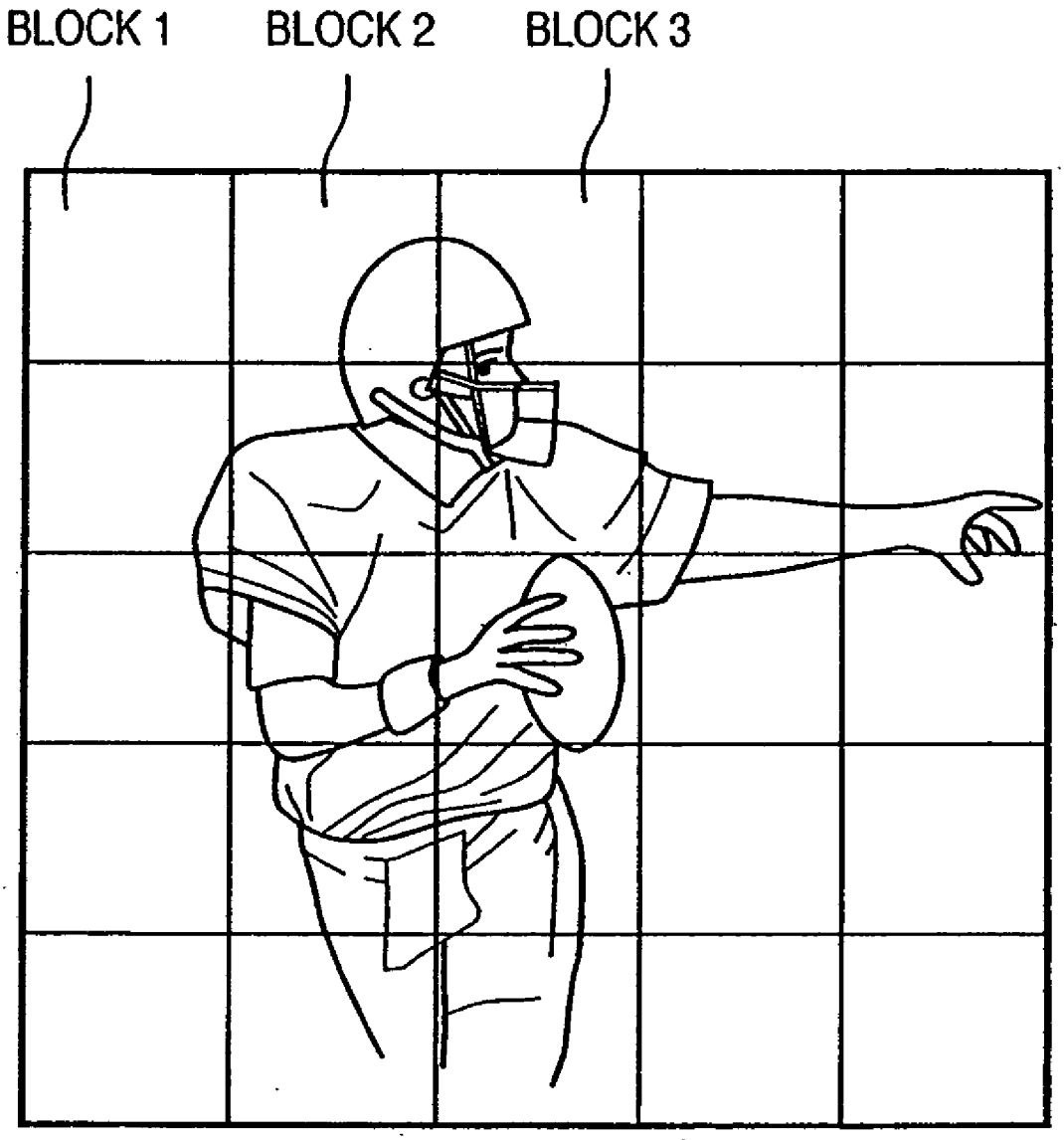


FIG. 3

FIG. 4

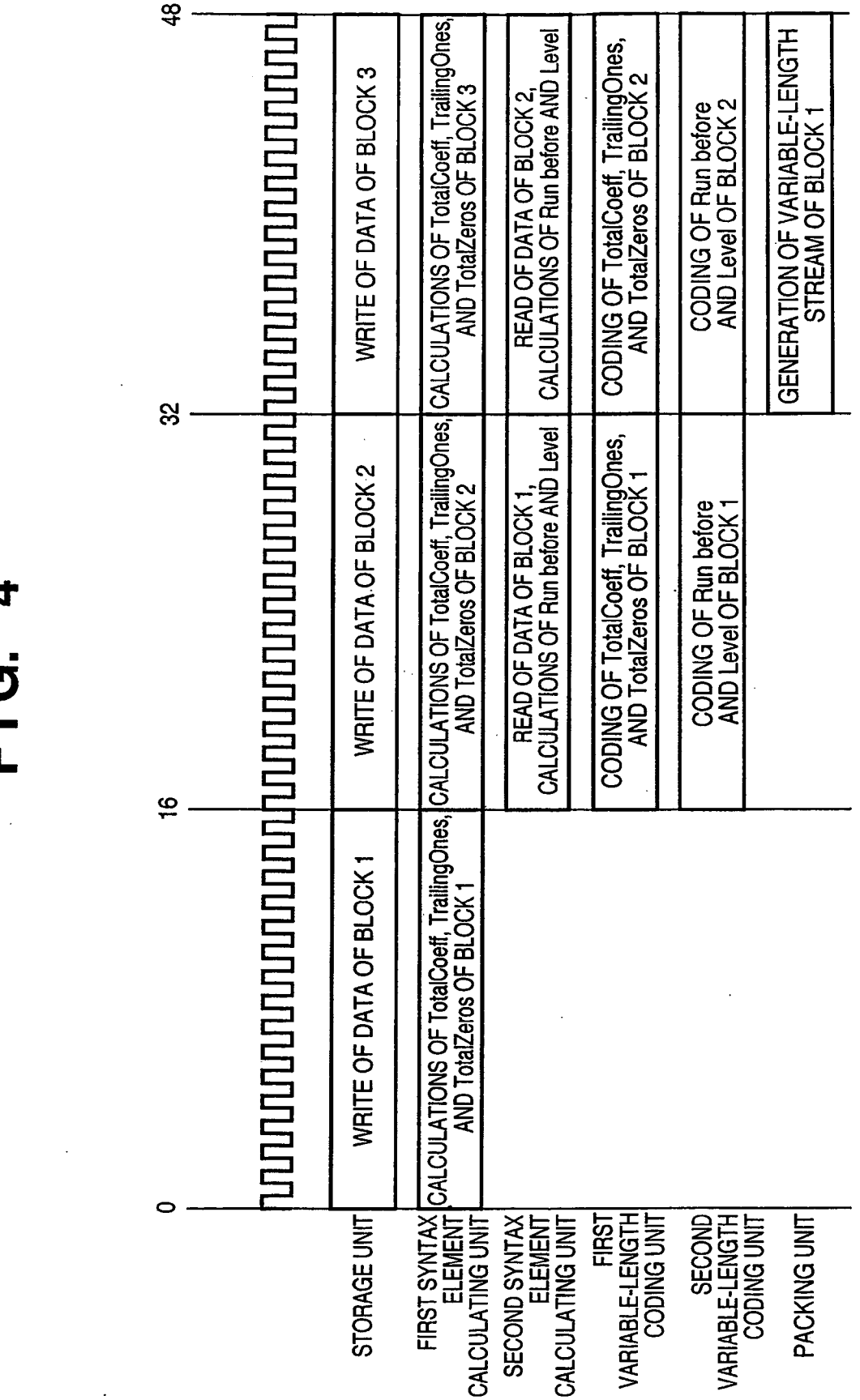

0
16
32
48
STORAGE UNIT
WRITE OF DATA OF BLOCK 1
WRITE OF DATA OF BLOCK 2
WRITE OF DATA OF BLOCK 3
FIRST SYNTAX ELEMENT CALCULATING UNIT
CALCULATIONS OF TotalCoeff, TrailingOnes, AND TotalZeros OF BLOCK 1
CALCULATIONS OF TotalCoeff, TrailingOnes, AND TotalZeros OF BLOCK 2
CALCULATIONS OF TotalCoeff, TrailingOnes, AND TotalZeros OF BLOCK 3
SECOND SYNTAX ELEMENT CALCULATING UNIT
READ OF DATA OF BLOCK 1, CALCULATIONS OF Run before AND Level
READ OF DATA OF BLOCK 2, CALCULATIONS OF Run before AND Level
FIRST VARIABLE-LENGTH CODING UNIT
CODING OF TotalCoeff, TrailingOnes, AND TotalZeros OF BLOCK 1
CODING OF TotalCoeff, TrailingOnes, AND TotalZeros OF BLOCK 2
SECOND VARIABLE-LENGTH CODING UNIT
CODING OF Run before AND Level OF BLOCK 1
CODING OF Run before AND Level OF BLOCK 2
PACKING UNIT
GENERATION OF VARIABLE-LENGTH STREAM OF BLOCK 1

FIG. 5

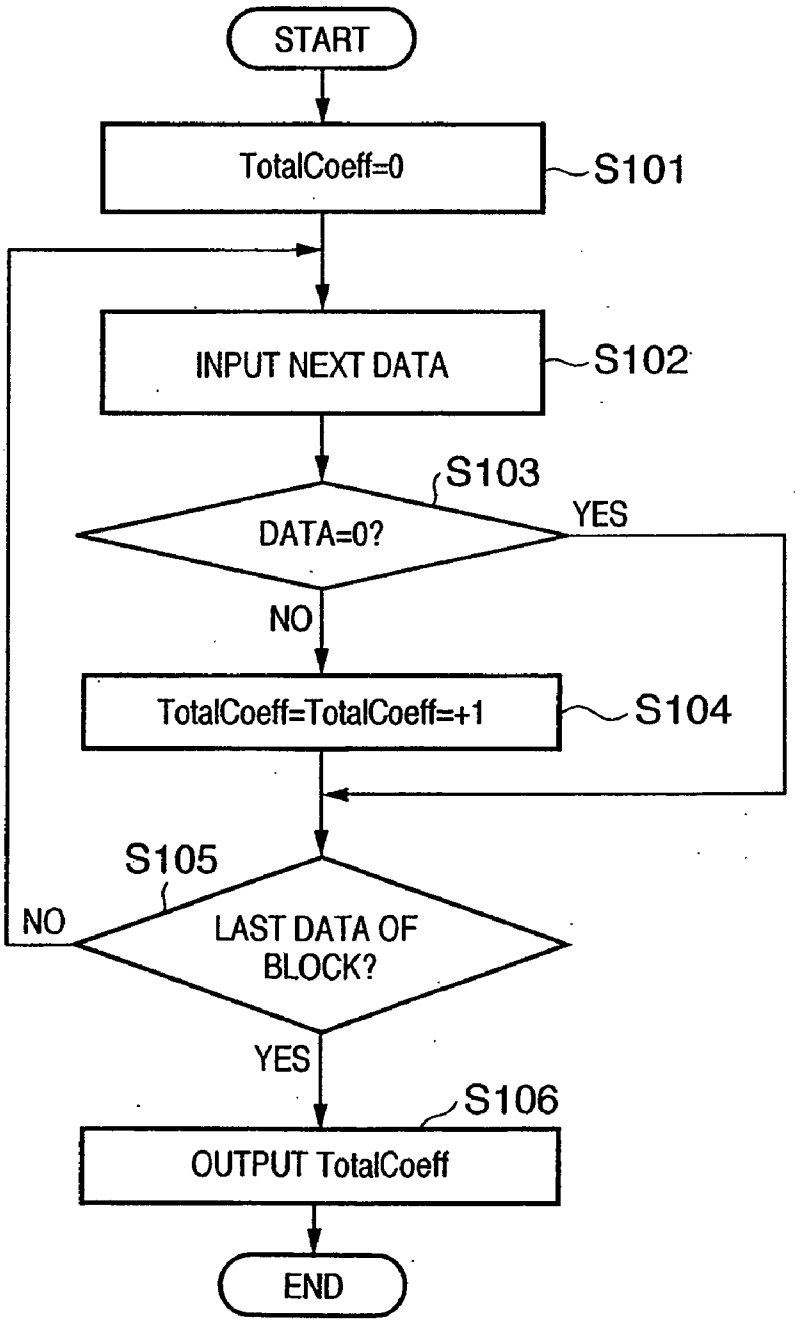
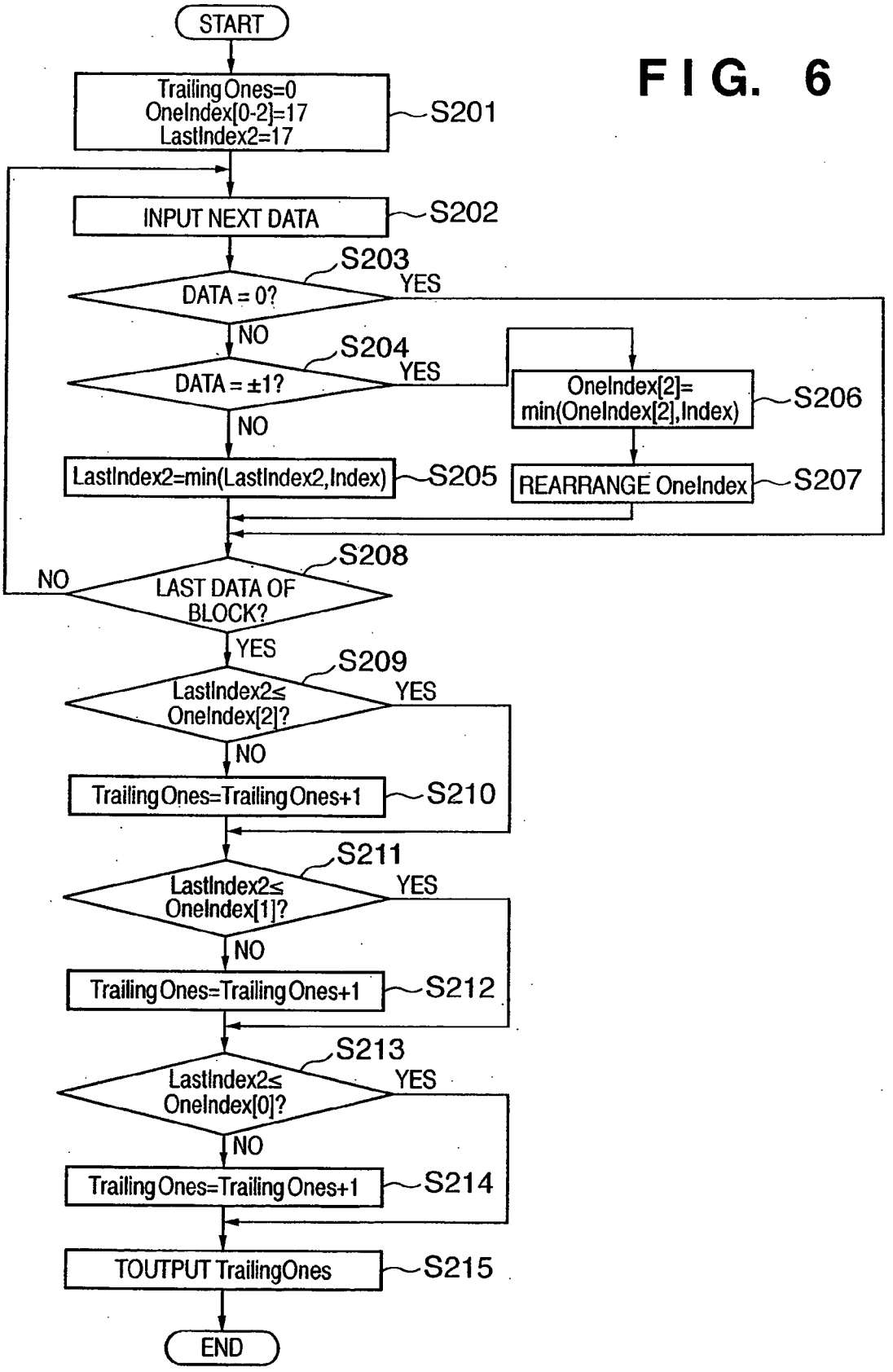

START
TotalCoeff=0
S101
INPUT NEXT DATA
S102
S103
DATA=0?
YES
NO
TotalCoeff=TotalCoeff=+1
S104
S105
NO
LAST DATA OF BLOCK?
YES
S106
OUTPUT TotalCoeff
END

FIG. 6


START
TrailingOnes=0
OneIndex[0-2]=17
LastIndex2=17
S201
INPUT NEXT DATA
S202
S203
DATA = 0?
YES
NO
S204
DATA = ±1?
YES
NO
OneIndex[2]=
min(OneIndex[2],Index)
S206
LastIndex2=min(LastIndex2,Index)
S205
REARRANGE OneIndex
S207
S208
NO
LAST DATA OF
BLOCK?
YES
S209
LastIndex2≤
OneIndex[2]?
YES
NO
TrailingOnes=TrailingOnes+1
S210
S211
LastIndex2≤
OneIndex[1]?
YES
NO
TrailingOnes=TrailingOnes+1
S212
S213
LastIndex2≤
OneIndex[0]?
YES
NO
TrailingOnes=TrailingOnes+1
S214
TOUTPUT TrailingOnes
S215
END

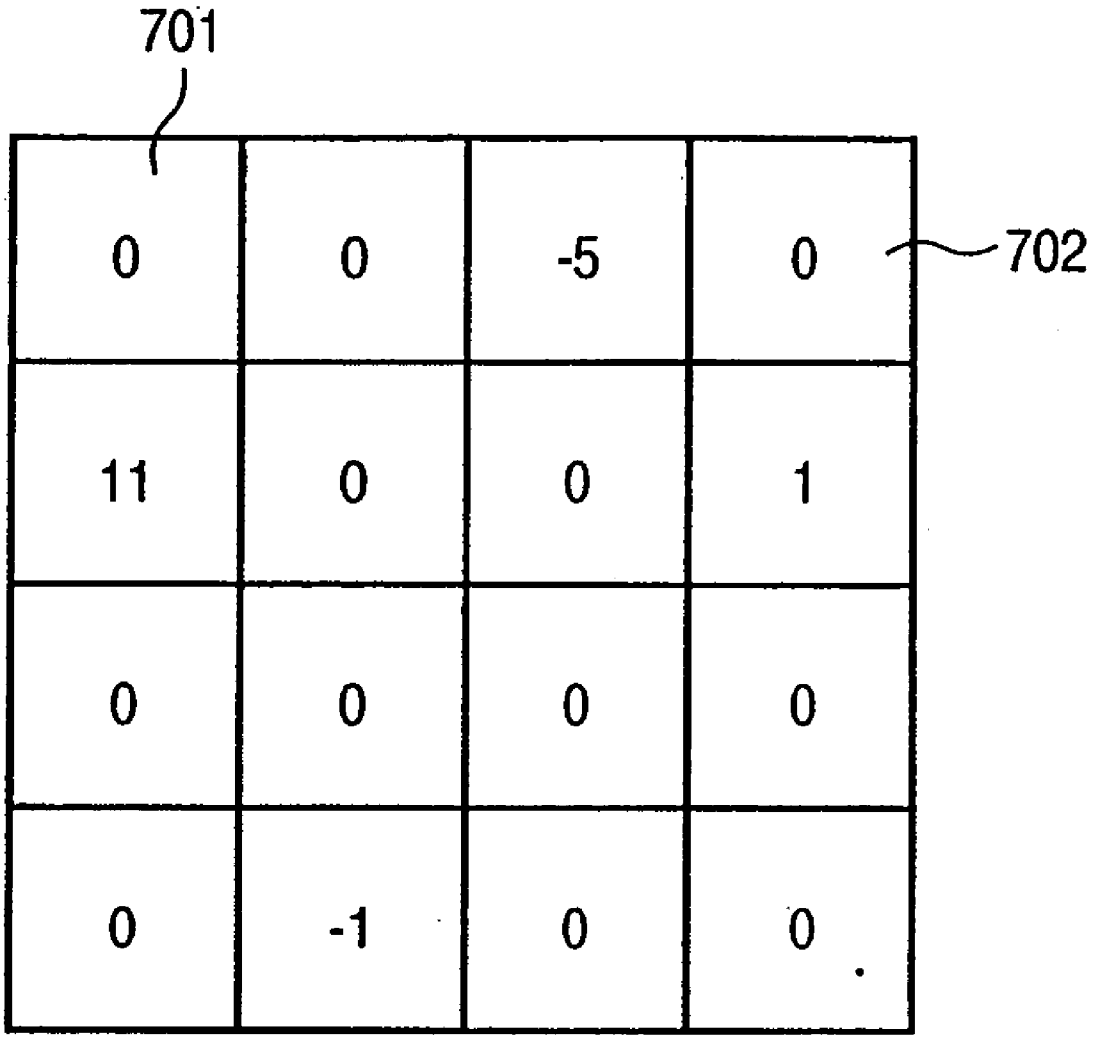


FIG. 7

FIG. 8

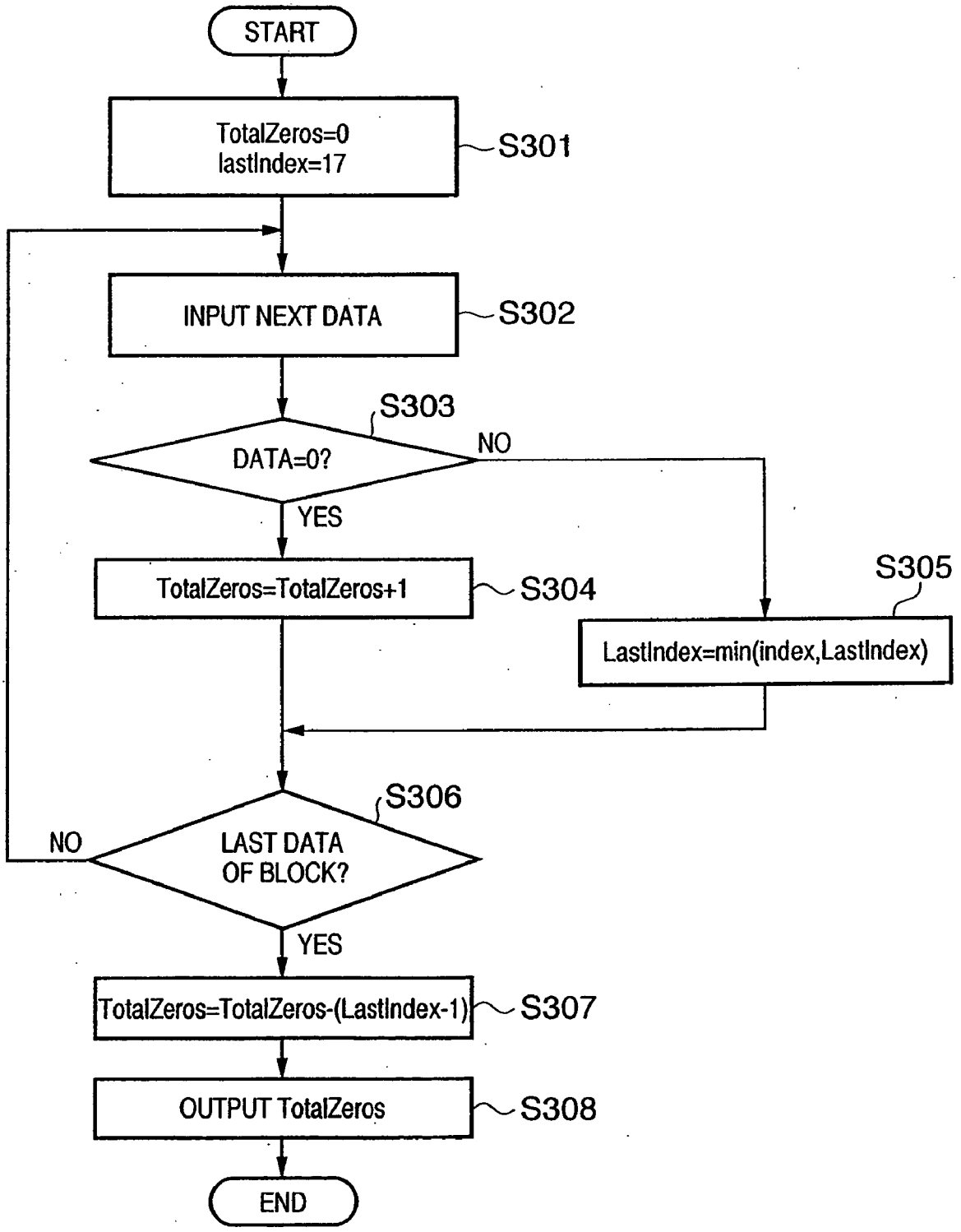

START
TotalZeros=0
lastIndex=17
S301
INPUT NEXT DATA
S302
S303
DATA=0?
NO
YES
TotalZeros=TotalZeros+1
S304
S305
LastIndex=min(index,LastIndex)
S306
NO
LAST DATA
OF BLOCK?
YES
TotalZeros=TotalZeros-(LastIndex-1)
S307
OUTPUT TotalZeros
S308
END

FIG. 9

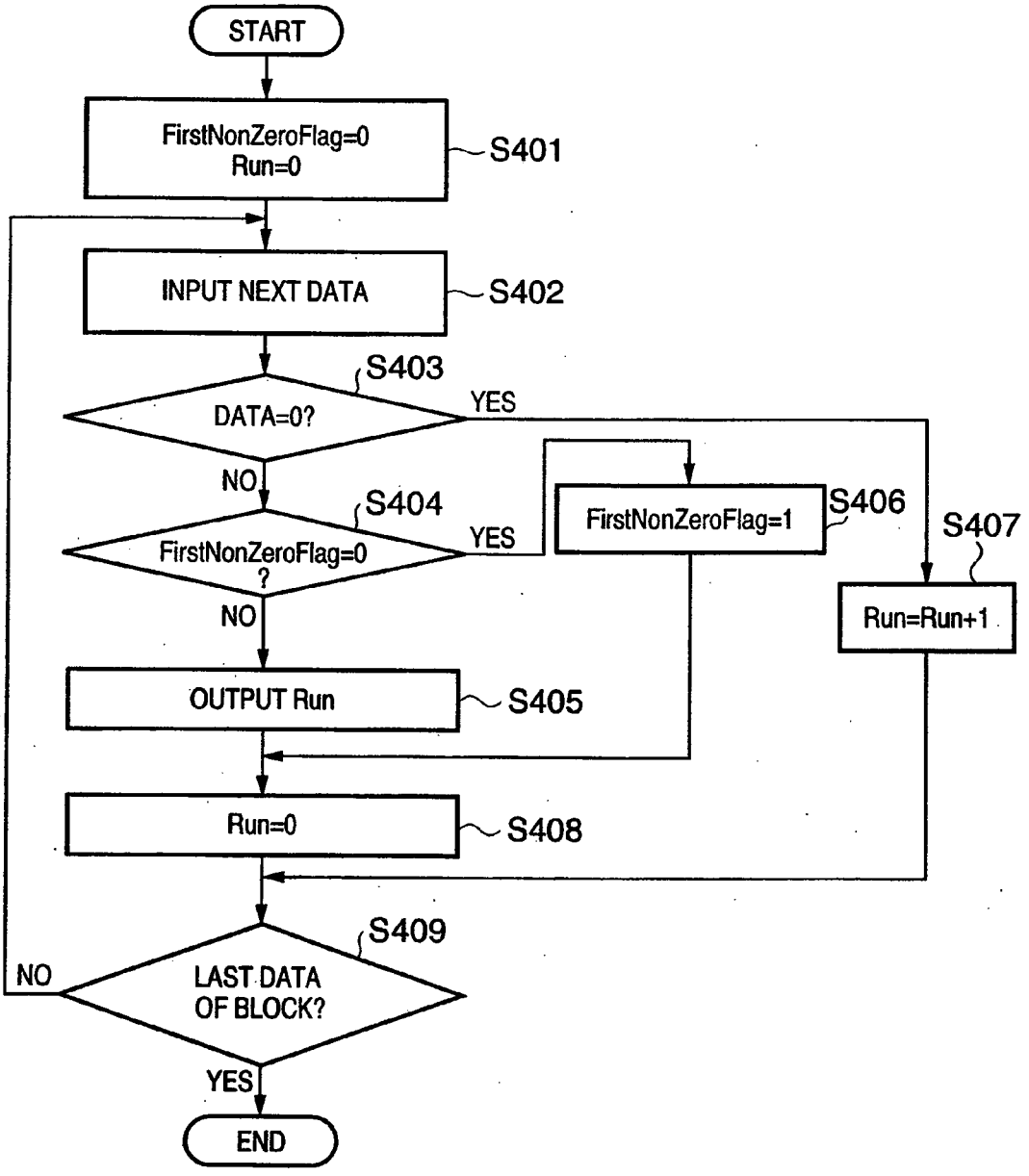

START
FirstNonZeroFlag=0
Run=0
S401
INPUT NEXT DATA
S402
S403
DATA=0?
YES
NO
S404
FirstNonZeroFlag=0
?
YES
FirstNonZeroFlag=1
S406
S407
Run=Run+1
NO
OUTPUT Run
S405
Run=0
S408
S409
NO
LAST DATA
OF BLOCK?
YES
END

FIG. 10

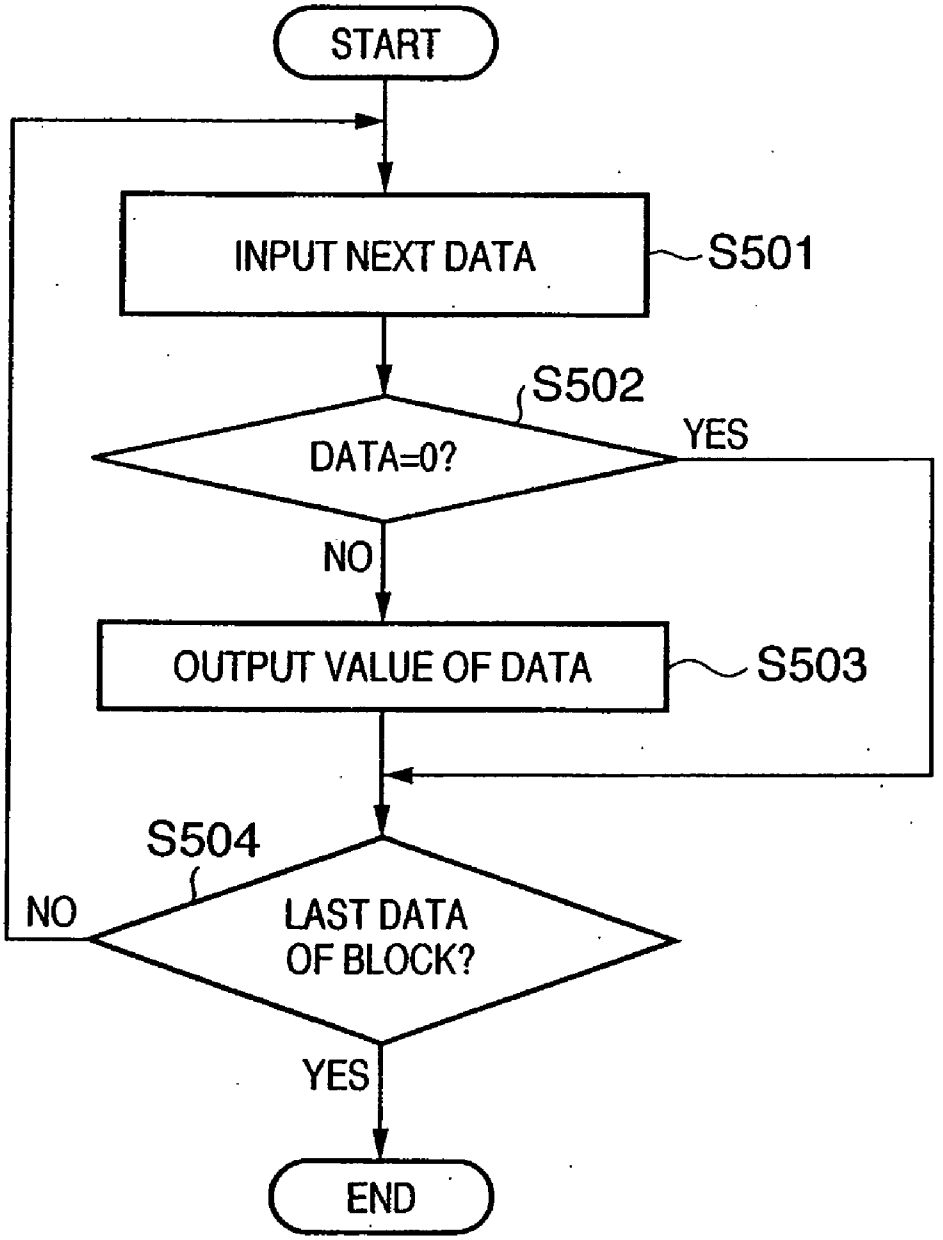

START
INPUT NEXT DATA
S501
S502
DATA=0?
YES
NO
OUTPUT VALUE OF DATA
S503
S504
NO
LAST DATA OF BLOCK?
YES
END

FIG. 11

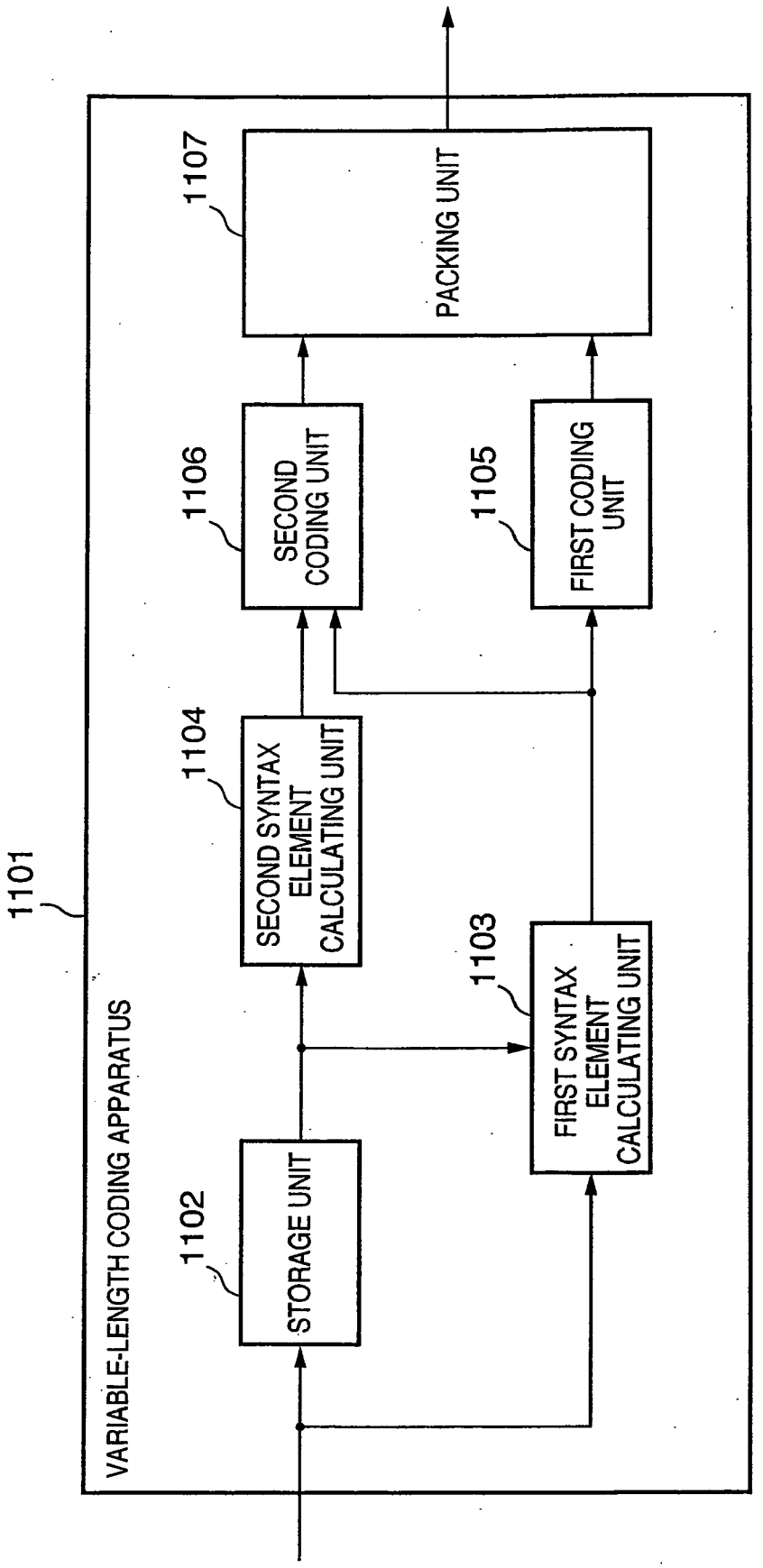

1101
VARIABLE-LENGTH CODING APPARATUS
1102
STORAGE UNIT
1103
FIRST SYNTAX ELEMENT CALCULATING UNIT
1104
SECOND SYNTAX ELEMENT CALCULATING UNIT
1105
FIRST CODING UNIT
1106
SECOND CODING UNIT
1107
PACKING UNIT

FIG. 12

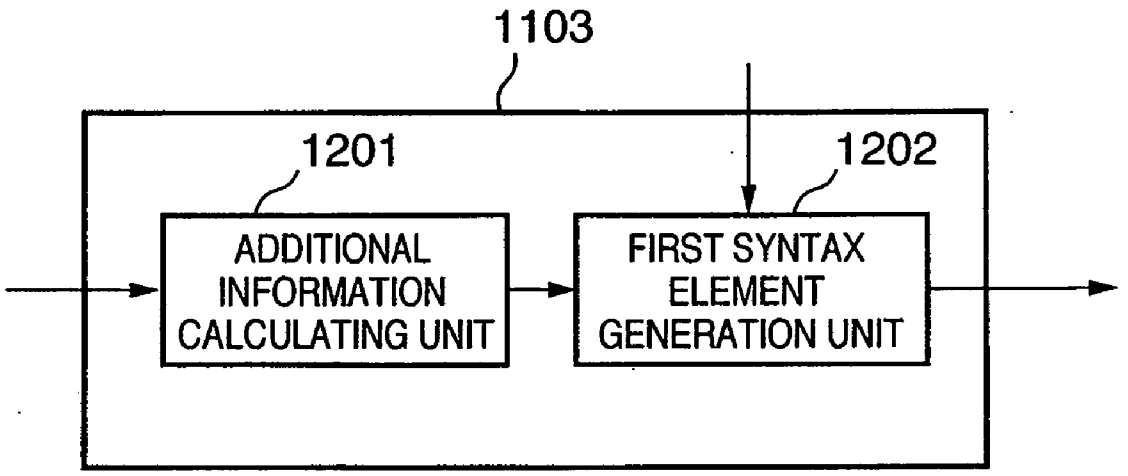

1103
1201
1202
ADDITIONAL INFORMATION CALCULATING UNIT
FIRST SYNTAX ELEMENT GENERATION UNIT

FIG. 13

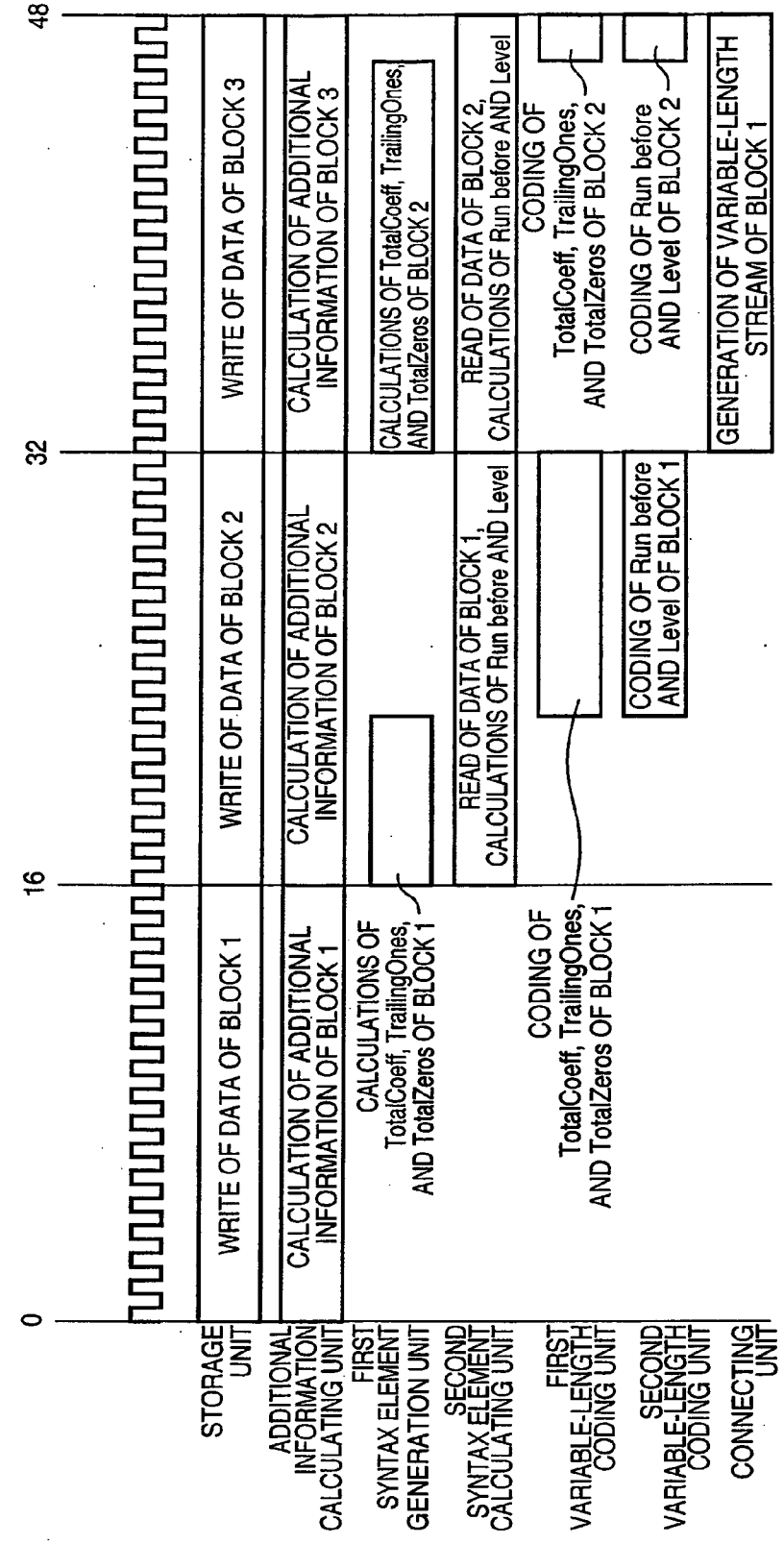

0
16
32
48
STORAGE UNIT
WRITE OF DATA OF BLOCK 1
WRITE OF DATA OF BLOCK 2
WRITE OF DATA OF BLOCK 3
ADDITIONAL INFORMATION CALCULATING UNIT
CALCULATION OF ADDITIONAL INFORMATION OF BLOCK 1
CALCULATION OF ADDITIONAL INFORMATION OF BLOCK 2
CALCULATION OF ADDITIONAL INFORMATION OF BLOCK 3
FIRST SYNTAX ELEMENT GENERATION UNIT
CALCULATIONS OF TotalCoeff, TrailingOnes, AND TotalZeros OF BLOCK 1
CALCULATIONS OF TotalCoeff, TrailingOnes, AND TotalZeros OF BLOCK 2
SECOND SYNTAX ELEMENT CALCULATING UNIT
READ OF DATA OF BLOCK 1, CALCULATIONS OF Run before AND Level
READ OF DATA OF BLOCK 2, CALCULATIONS OF Run before AND Level
FIRST VARIABLE-LENGTH CODING UNIT
CODING OF TotalCoeff, TrailingOnes, AND TotalZeros OF BLOCK 1
CODING OF TotalCoeff, TrailingOnes, AND TotalZeros OF BLOCK 2
SECOND VARIABLE-LENGTH CODING UNIT
CODING OF Run before AND Level OF BLOCK 1
CODING OF Run before AND Level OF BLOCK 2
CONNECTING UNIT
GENERATION OF VARIABLE-LENGTH STREAM OF BLOCK 1

FIG. 14

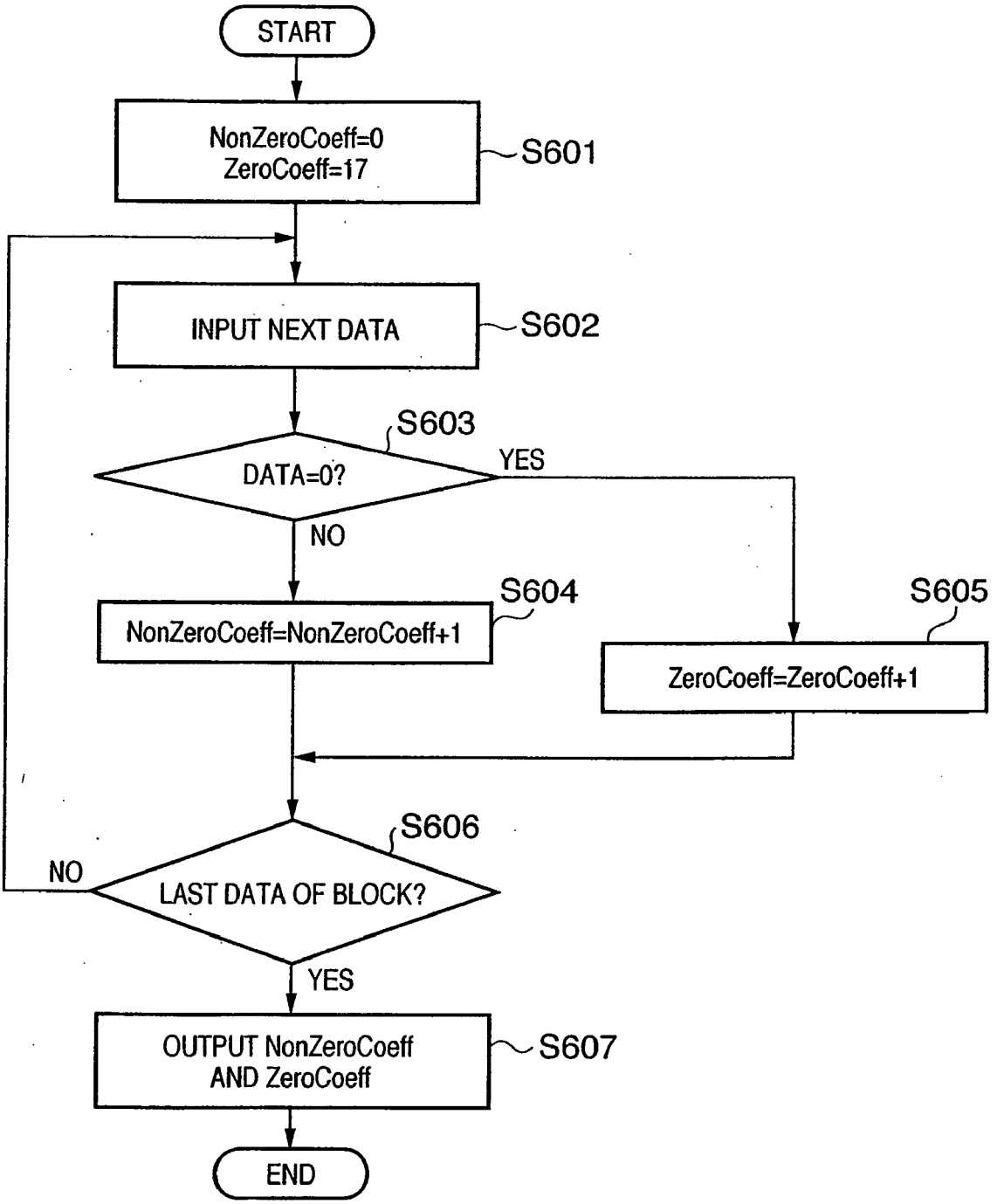

START
NonZeroCoeff=0
ZeroCoeff=17
S601
INPUT NEXT DATA
S602
S603
DATA=0?
YES
NO
S604
NonZeroCoeff=NonZeroCoeff+1
S605
ZeroCoeff=ZeroCoeff+1
S606
NO
LAST DATA OF BLOCK?
YES
OUTPUT NonZeroCoeff
AND ZeroCoeff
S607
END

FIG. 15

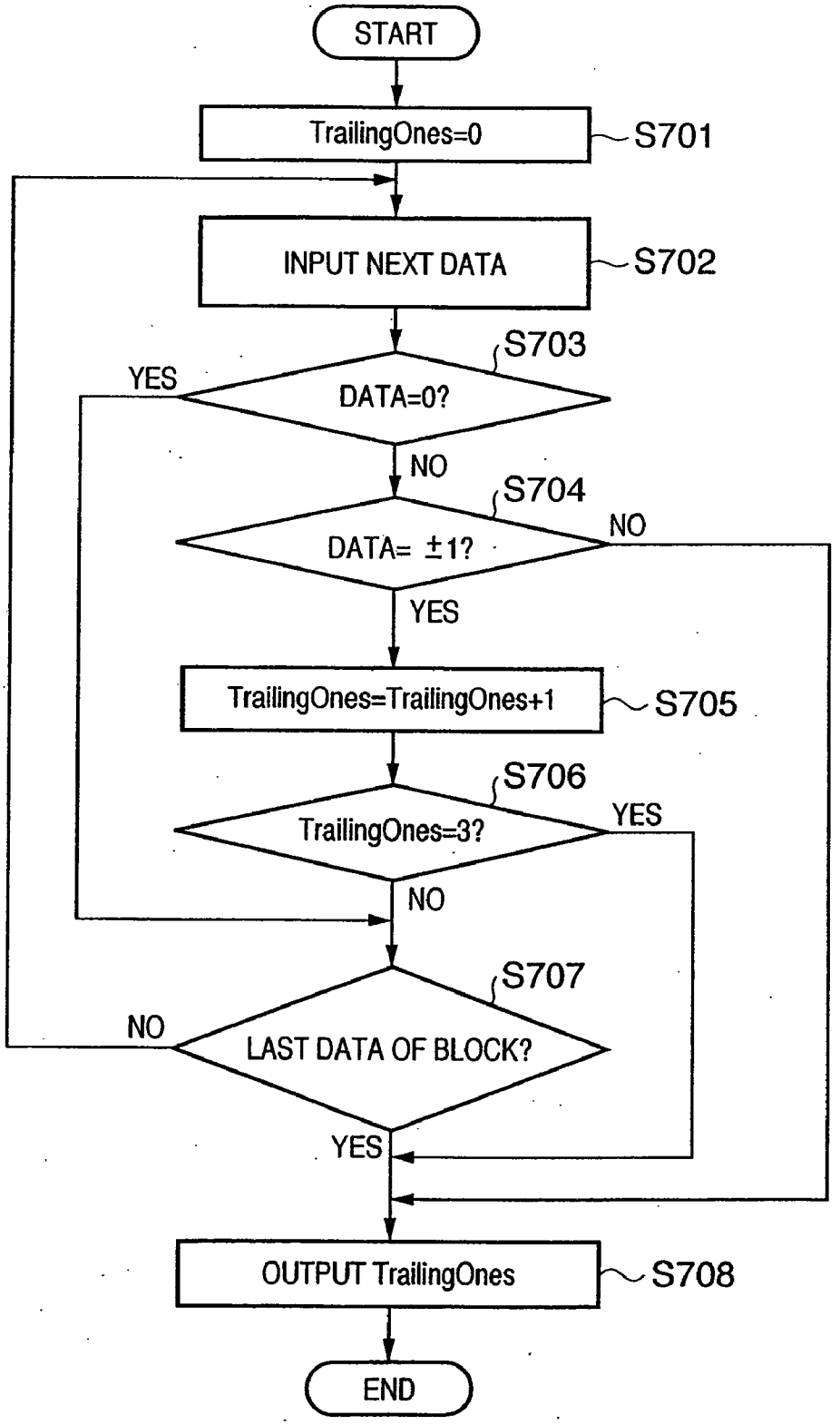
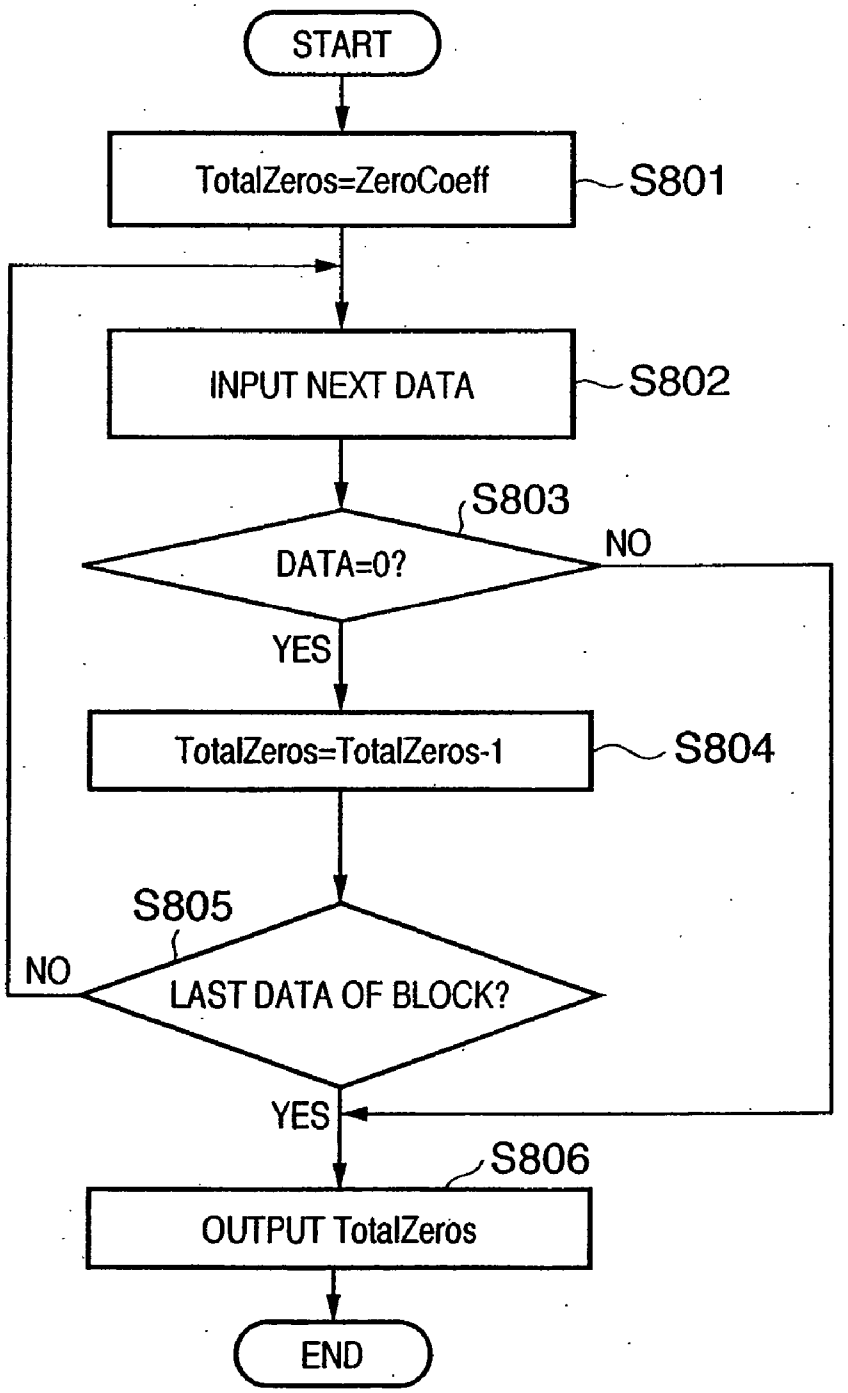

START
TrailingOnes=0
S701
INPUT NEXT DATA
S702
S703
YES
DATA=0?
NO
S704
DATA= ±1?
NO
YES
TrailingOnes=TrailingOnes+1
S705
S706
TrailingOnes=3?
YES
NO
S707
NO
LAST DATA OF BLOCK?
YES
OUTPUT TrailingOnes
S708
END

FIG. 16


START
TotalZeros=ZeroCoeff
S801
INPUT NEXT DATA
S802
S803
DATA=0?
NO
YES
TotalZeros=TotalZeros-1
S804
S805
NO
LAST DATA OF BLOCK?
YES
S806
OUTPUT TotalZeros
END

VARIABLE-LENGTH CODING APPARATUS AND VARIABLE-LENGTH CODING METHOD

FIELD OF THE INVENTION

[0001] The present invention relates to a variable-length coding apparatus for compressing an image signal.

BACKGROUND OF THE INVENTION

[0002] Recently, a method called H.264 is standardized as a motion image compression coding method. This H.264 coding method requires a larger operation amount than those of the conventional compression coding methods such as MPEG2 and MPEG4, but can realize a higher coding efficiency than those of these conventional methods.

[0003] In this H.264 coding method, a context adaptive variable-length coding (CAVLC) method is defined as a variable-length coding method. Syntax elements which form a coded stream in the CAVLC method are the number of nonzero data (to be referred to as TotalCoeff hereinafter) in a block, the number of zero data (to be referred to as TotalZeros hereinafter) input before the last nonzero data in a block, the number of data (to be referred to as TrailingOnes hereinafter) having an absolute value of 1 and input after the last data having an absolute value of 2 or more in a block, the value (to be referred to as Level hereinafter) of nonzero data, and the number of zero data (to be referred to as Run hereinafter) preceding to the nonzero data. Note that in the standard of H.264, the maximum value of TrailingOnes is defined as 3.

[0004] This CAVLC method largely differs from the variable-length coding method used in MPEG2 and MPEG4 in the following two points:

[0005] 1. Level and Run are coded in the reverse scan order.

[0006] 2. A variable-length coding table necessary for coding of Level and Run is changed in accordance with the calculation results of TotalCoeff, TrailingOnes, and TotalZeros.

[0007] An example of a variable-length coding apparatus using the CAVLC method is described in patent reference 1. This variable-length coding apparatus increases the processing speed by coding a plurality of data in parallel by variable-length coding, and sequentially connecting a plurality of generated variable-length coded data (e.g., Japanese Patent Laid-Open No. 2004-056758).

[0008] In this variable-length coding apparatus described in Japanese Patent Laid-Open No. 2004-056758, however, a plurality of syntax elements depend on each other, so the processing takes a long time if coding of a second syntax element cannot be started before the calculation of a first syntax element is completed.

[0009] For example, in the CAVLC method, after all image data in a block are read out and TotalCoeff, TrailingOnes, and TotalZeros are calculated, the Level and Run of all the image data in the block must be coded in turn from the beginning by variable-length coding. This makes the variable-length coding process time-consuming.

SUMMARY OF THE INVENTION

[0010] The present invention has been made in consideration of the above situation, and has as its object to increase the processing speed even when a plurality of syntax elements depend one each other.

[0011] According to the present invention, the foregoing object is attained by providing a variable-length coding apparatus which receives image data block by block, and generates coded data from syntax elements, comprising:

[0012] a storage unit that temporarily stores the image data;

[0013] a first syntax element calculating unit that calculates a first syntax element from the image data, at the same time the image data is stored in the storage unit;

[0014] a first coding unit that generates coded data of the first syntax element from a value of the first syntax element;

[0015] a second syntax element calculating unit that reads out the image data stored in the storage unit, and calculates a value of a second syntax element from the image data;

[0016] a second coding unit that generates coded data of the second syntax element on the basis of the values of the first and second syntax elements; and

[0017] a packing unit that packs the coded data of the first and second syntax elements.

[0018] According to the present invention, the foregoing object is also attained by providing a variable-length coding method which receives image data block by block, and generates coded data from syntax elements, comprising:

[0019] temporarily storing the image data in a storage unit;

[0020] calculating a first syntax element from the image data, at the same time the image data is stored in the storage unit;

[0021] generating coded data of the first syntax element from a value of the first syntax element;

[0022] reading out the image data stored in the storage unit, and calculating a value of a second syntax element from the image data;

[0023] generating coded data of the second syntax element on the basis of the values of the first and second syntax elements; and

[0024] connecting the coded data of the first and second syntax elements.

[0025] Other features and advantages of the present invention will be apparent from the following description taken in conjunction with the accompanying drawings, in which like reference characters designate the same or similar parts throughout the figures thereof.

BRIEF DESCRIPTION OF THE DRAWINGS

[0026] The accompanying drawings, which are incorporated in and constitute a part of the specification, illustrate embodiments of the invention and, together with the description, serve to explain the principles of the invention.

[0027] **FIG. 1** is a view showing the arrangement of a variable-length coding apparatus according to the first embodiment;

[0028] **FIGS. 2A and 2B** are views showing the image data read orders;

[0029] **FIG. 3** is a view showing an example of image data;

[0030] **FIG. 4** is a timing chart showing the operation of each circuit according to the first embodiment;

[0031] **FIG. 5** is a flowchart showing a method of calculating TotalCoeff;

[0032] **FIG. 6** is a flowchart showing a method of calculating TrailingOnes;

[0033] **FIG. 7** is a view showing an example of an image data block;

[0034] **FIG. 8** is a flowchart showing a method of calculating TotalZeros;

[0035] **FIG. 9** is a flowchart showing a method of calculating Run;

[0036] **FIG. 10** is a flowchart showing a method of calculating Level;

[0037] **FIG. 11** is a view showing the arrangement of a variable-length coding apparatus according to the second embodiment;

[0038] **FIG. 12** is a view showing the internal arrangement of a first syntax element calculating unit in the variable-length coding apparatus according to the second embodiment;

[0039] **FIG. 13** is a timing chart showing the operation of each circuit according to the second embodiment of the present invention;

[0040] **FIG. 14** is a flowchart showing a method of calculating additional information;

[0041] **FIG. 15** is a flowchart showing a method of calculating TrailingOnes; and

[0042] **FIG. 16** is a flowchart showing a method of calculating TotalZeros.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

[0043] Preferred embodiments of the present invention will now be described in detail in accordance with the accompanying drawings.

First Embodiment

[0044] **FIG. 1** is a view showing the arrangement of the first embodiment of the present invention.

[0045] Referring to **FIG. 1**, a variable-length coding apparatus **101** according to the first embodiment includes a storage unit **102**, first syntax element calculating unit **103**, second syntax element calculating unit **104**, first coding unit **105**, second coding unit **106**, and packing unit **107**. The packing unit **107** packs variable-length coded data.

[0046] Image data of 4×4 rectangular blocks is input in a raster scan order shown in **FIG. 2A** to the variable-length coding apparatus **101**.

[0047] The storage unit **102** is a RAM which temporarily stores the image data. Note that read or write to this RAM requires one cycle.

[0048] The first syntax element calculating unit **103** reads the image data in a block, calculates TotalCoeff, TrailingOnes, and TotalZeros in the block, and internally holds the calculated syntax elements.

[0049] The second syntax element calculating unit **104** reads out the image data held in the storage unit **102** in a reverse zigzag scan order shown in **FIG. 2B**, and calculates Level and Run of each image data in the block.

[0050] The first coding unit **105** performs variable-length coding, by using a predetermined variable-length coding table, on TotalCoeff, TrailingOnes, and TotalZeros calculated by the first syntax element calculating unit **103**.

[0051] The second coding unit **106** selects a variable-length coding table on the basis of TotalCoeff, TrailingOnes, and TotalZeros calculated by the first syntax element calculating unit **103**, and, by using the selected variable-length coding table, performs variable-length coding on Level and Run of each image data calculated by the second syntax element calculating unit **104**.

[0052] The packing unit **107** shifts and packs the variable-length codes of the output individual syntax elements from the first and second coding units **105** and **106** in accordance with a predetermined format, thereby generating one variable-length stream data.

[0053] The flow of data when image data shown in **FIG. 3** is coded by variable-length coding will be explained below.

[0054] **FIG. 4** is a timing chart showing processing performed by each unit shown in **FIG. 1** with respect to time.

[0055] From time **0**, image data of block **1** shown in **FIG. 3** are input one at each cycle to the variable-length coding apparatus **101**, and written in the storage unit **102**. Since the block is made up of 4×4 image data, write of all the image data of block **1** requires 16 cycles.

[0056] The image data are also input to the first syntax element calculating unit **103** at the same time they are written in the storage unit **102**. Immediately after all the image data are input, the first syntax element calculating unit **103** calculates the values of TotalCoeff, TrailingOnes, and TotalZeros, and outputs the calculated values.

[0057] Subsequently, from time **16** to time **32**, the second syntax element calculating unit **104** reads out the image data of block **1** one at each cycle in a reverse zigzag scan order from the storage unit **102**, and calculates the values of Run and Level of each image data. Immediately after nonzero data is input, the second syntax element calculating unit **104** inputs the values of Level and Run of the data to the second coding unit **106**.

[0058] Also, TotalCoeff, TrailingOnes, and TotalZeros calculated before time **16** are input to the first coding unit **105**, and coded by variable-length coding in one cycle by using a predetermined variable-length coding table. The value of TrailingOnes is held because it is necessary to select a variable-length coding table of the next block. The variable-length coded data is input to the packing unit **107**, and held until time **32**.

[0059] In addition, from time **16** to time **32**, the second coding unit **106** selects a variable-length coding table necessary for coding of Run and Level on the basis of the values

of TotalCoeff, TrailingOnes, and TotalZeros input from the first syntax element calculating unit **103**. The second coding unit **106** performs variable-length coding on Run and Level of each image data in one cycle by using the selected variable-length coding table. Assume that coding of Run and coding of Level are performed in parallel. The variable-length coded data is input to the packing unit **107**, and held until time **32**.

[**0060**] In parallel with this processing, image data of block **2** shown in **FIG. 3** are written in the storage unit **102**. In addition, the first syntax element calculating unit **103** calculates TotalCoeff, TrailingOnes, and TotalZeros of block **2** in the same manner as for block **1**.

[**0061**] Then, from time **32** to time **48**, the packing unit **107** reads out the variable-length codes input from the first and second coding units **105** and **106**, connects and packs the readout codes, and outputs a coded stream defined by the H.264 standard.

[**0062**] In parallel with this processing, the second syntax element calculating unit **104** calculates Run and Level of each image data of block **2** in the same manner as for block **1**. Also, the first and second coding units **105** and **106** perform variable-length coding of each syntax element of block **2** in the same manner as for block **1**.

[**0063**] Furthermore, in parallel with this processing, image data of block **3** shown in **FIG. 3** are written in the storage unit **102**. The first syntax element calculating unit **103** calculates TotalCoeff, TrailingOnes, and TotalZeros of block **3** in the same manner as for block **1**. The same processing is repeated for all blocks to process one data at each cycle.

[**0064**] Details of the method of calculating each syntax element will be explained below.

[**0065**] First, the operation of calculating TotalCoeff performed by the first syntax element calculating unit **103** will be explained below with reference to a flowchart shown in **FIG. 5**.

[**0066**] In step S**101**, a variable TotalCoeff is initialized to 0. In step S**102**, the next image data in the block is input, and the flow advances to step S**103**.

[**0067**] In step S**103**, whether the value of the image data is zero is determined. If the value is not zero, the flow advances to step S**104** to add 1 to the value of TotalCoeff, and further advances to step S**105**.

[**0068**] If the value is zero, the flow advances to step S**105**. In step S**105**, whether the present image data is the last data of the block is determined. If the image data is the last data of the block, the flow advances to step S**106** to output the present value of TotalCoeff as a calculation result, and terminate the processing for the present block. If the image data is not the last data of the block, the flow returns to step S**102** to input the next image data.

[**0069**] The operation of calculating TrailingOnes performed by the first syntax element calculating unit **103** will be explained below with reference to a flowchart shown in **FIG. 6**.

[**0070**] First, in step S**201**, a variable TrailingOnes is initialized to 0. Also, a variable LastIndex**2** and each element of an array OneIndex including three elements are initialized to 17. Note that in this explanation of the flowchart shown in **FIG. 6**, a number indicating the order when each image data input in a raster scan order is read out in a reverse zigzag scan order is defined as an index. For example, in a block shown in **FIG. 7**, the index of image data **701** is 16, and that of image data **702** is 10.

[**0071**] LastIndex**2** is a variable representing the minimum value of the index of image data having an absolute value of 2 or more in the block. Also, the array OneIndex stores three indices, in ascending order from the smallest one, of image data having an absolute value of 1 in the block.

[**0072**] In step S**202**, the next image data in the block is input, and the flow advances to step S**203**. In step S**203**, whether the value of the image data is zero is determined. If the value is zero, the flow advances to step S**208**. If the value is not zero, the flow advances to step S**204** to determine whether the absolute value of the image data is 1.

[**0073**] If the absolute value of the image data is not 1, the flow advances to step S**205** to compare the value of LastIndex**2** with the index value of the presently input image data, and substitute a smaller value into LastIndex**2**.

[**0074**] If the absolute value of the image data is 1 in step S**204**, the flow advances to step S**206** to compare the value of an array OneIndex[**2**] with the index value of the presently input image data, and substitute a smaller value into OneIndex[**2**]. After that, the flow advances to step S**207**.

[**0075**] In step S**207**, the values of OneIndex[**0**], OneIndex [**1**], and OneIndex[**2**] are compared, and the values of these arrays are rearranged such that the minimum value is OneIndex[**0**] and the maximum value is OneIndex[**2**].

[**0076**] In step S**208**, whether the present image data is the last data of the block is determined. If the image data is not the last data of the block, the flow returns to step S**202** to input the next image data. If the image data is the last data of the block, the flow advances to step S**209**. In step S**209**, the value of LastIndex**2** is compared with OneIndex[**2**]. If OneIndex[**2**] is smaller, the flow advances to step S**210** to add 1 to TrailingOnes. If LastIndex**2** is smaller or the two values are equal, the flow advances to step S**211**.

[**0077**] Subsequently, in step S**211**, the value of LastIndex**2** is compared with OneIndex[**1**]. If OneIndex[**1**] is smaller, the flow advances to step S**212** to add 1 to TrailingOnes. If LastIndex**2** is smaller or the two values are equal, the flow advances to step S**213**.

[**0078**] In step S**213**, the value of LastIndex**2** is compared with OneIndex[**0**]. If OneIndex[**0**] is smaller, the flow advances to step S**214** to added 1 to TrailingOnes. If LastIndex**2** is smaller or the two values are equal, the flow advances to step S**215**.

[**0079**] Finally, in step S**215**, the present value of TrailingOnes is output as a calculation result, and the processing for the present block is completed.

[**0080**] The operation of calculating TotalZeros performed by the first syntax element calculating unit **103** will be explained below with reference to a flowchart shown in **FIG. 8**.

[**0081**] First, in step S**301**, a variable TotalZeros is initialized to 0. Also, a variable LastIndex is initialized to 17. Note

that LastIndex is a variable representing the minimum value of the index of image data having an absolute value of 1 or more in the block.

[0082] Then, in step S302, the next image data in the block is input, and the flow advances to step S303. In step S303, whether the value of the image data is zero is determined. If the value is zero, the flow advances to step S304 to add 1 to TotalZeros. If the value is not zero, the flow advances to step S305.

[0083] In step S305, the value of LastIndex is compared with the index value of the presently input image data, and a smaller value is substituted into LastIndex. After that, the flow advances to step S306.

[0084] In step S306, whether the present image data is the last data of the block is determined. If the image data is the last data of the block, the flow advances to step S307 to subtract the value of LastIndex-1 from TotalZeros.

[0085] Subsequently, the flow advances to step S308 to output the present value of TotalZeros as a calculation result, and terminate the processing for the present block. If it is determined in step S306 that the image data is not the last data of the block, the flow returns to step S302 to input the next image data.

[0086] By the method described above, the first syntax element calculating unit **103** can calculate the values of TotalCoeff, TrailingOnes, and TotalZeros immediately after the last image data in the block is input to the storage unit **102**.

[0087] The operation of calculating Run performed by the second syntax element calculating unit **104** will be explained below with reference to a flowchart shown in **FIG. 9**.

[0088] First, in step S401, variables Run and FirstNonZeroFlag are initialized to 0. Note that FirstNonZeroFlag is a flag which is 1 when at least one nonzero image data in the block is input, and 0 if no nonzero image data has been input yet.

[0089] Then, in step S402, the next image data in the block is input, and the flow advances to step S403. In step S403, whether the value of the image data is zero is determined. If the value is not zero, the flow advances to step S404 to determine whether the value of FirstNonZeroFlag is zero. If the value is not zero, the flow advances to step S405 to output the present value of Run, and further advances to step S408. If it is determined in step S404 that the value is zero, the flow advances to step S406. In step S406, FirstNonZeroFlag is set to 1, and the flow further advances to step S408. When it is determined in step S403 that the value of the image data is zero, the flow advances to step S407 to add 1 to Run, and then advances to step S408.

[0090] In step S408, zero is substituted into Run, and the flow advances to step S409. Note that if the value of the image data is zero in step S403, the flow advances to step S407 to add 1 to Run, and then advances to step S409.

[0091] In step S409, whether the present image data is the last data of the block is determined. If the image data is the last data of the block, the processing for the present block is terminated. If the present image data is not the last data of the block, the flow returns to step S402 to input the next image data.

[0092] The operation of calculating Level performed by the second syntax element calculating unit **104** will be explained below with reference to a flowchart shown in **FIG. 10**.

[0093] In step S501, the next image data in the block is input, and the flow advances to step S502. In step S502, whether the value of the image data is zero is determined. If the value is zero, the flow advances to step S504. If the value is not zero, the flow advances to step S503. In step S503, the value of the presently input data is output as Level, and the flow advances to step S504.

[0094] In step S504, whether the present image data is the last data of the block is determined. If the image data is the last data of the block, the processing for the present block is terminated. If the image data is not the last data of the block, the flow returns to step S501 to input the next image data.

[0095] In the first embodiment of the present invention as described above, the second syntax element calculating unit **104** can read out image data in a block from the storage unit **102** and calculate the values of Run and Level at the same time.

[0096] On the basis of the first syntax elements calculated by the first syntax element calculating unit **103**, the second coding unit **106** selects one of a plurality of predetermined variable-length coding tables, and performs variable-length coding of Run and Level. Since the values of TotalCoeff, TrailingOnes, and TotalZeros are calculated in advance, variable-length coding of Run and Level can be performed immediately after each image data in the block is read out by the second syntax element calculating unit **104**.

[0097] In parallel with this processing, the first coding unit **105** performs variable-length coding on TotalCoeff, TrailingOnes, and TotalZeros. As a consequence, variable-length codes of all syntax elements can be obtained in the same clock cycle as when read of image data in the block from the storage unit **102** is completed.

Second Embodiment

[0098] The second embodiment of the present invention will be described below.

[0099] The second embodiment differs from the first embodiment in the following respect. In the first embodiment, the first syntax element calculating unit calculates the values of the first syntax elements directly from image data input to the storage unit. In the second embodiment, however, the value of additional information is calculated from image data, and a first syntax element calculating unit calculates the values of first syntax elements by using this additional information and data read out from a storage unit.

[0100] **FIG. 11** is a view showing the arrangement of the second embodiment of the present invention. Referring to **FIG. 11**, a variable-length coding apparatus **1101** according to the present invention includes a storage unit **1102**, first syntax element calculating unit **1103**, second syntax element calculating unit **1104**, first coding unit **1105**, second coding unit **1106**, and packing unit **1107**. The packing unit **107** connects variable-length coded data. Note that the variable-length coding apparatus **1101**, storage unit **1102**, second syntax element calculating unit **1104**, first and second cod-

ing units **1105** and **1106**, and packing unit **1107** are the same as in the first embodiment, so an explanation thereof will be omitted.

[**0101**] The first syntax element calculating unit **1103** calculates additional information necessary to calculate the values of first syntax elements at the same time image data is input to the storage unit **1102**. After that, when reading out image data from the storage unit **1102**, the first syntax element calculating unit **1103** calculates the values of the first syntax elements by using the additional information.

[**0102**] **FIG. 12** shows the internal arrangement of the first syntax element calculating unit **1103**. The first syntax element calculating unit **1103** contains an additional information calculating unit **1201** and first syntax element generation unit **1202**.

[**0103**] The additional information calculating unit **1201** calculates the number of zero data and the number of nonzero data as additional information from each image data in a block which is input in a raster scan order to the variable-length coding apparatus **1101**.

[**0104**] The first syntax element generation unit **1202** calculates the values of TotalCoeff, TrailingOnes, and TotalZeros necessary to code second syntax elements, from data read out in a reverse zigzag scan order from the storage unit **1102** and the additional information calculated by the additional information calculating unit **1201**.

[**0105**] The flow of data when image data shown in **FIG. 3** is coded by variable-length coding will be explained below.

[**0106**] **FIG. 13** is a timing chart showing processing performed by each unit shown in **FIG. 11** with respect to time. From time **0**, image data of block **1** shown in **FIG. 3** are input one at each cycle to the variable-length coding apparatus **1101**, and written in the storage unit **1102**. Since the block is made up of 4×4 image data, write of all the image data of block **1** requires 16 cycles.

[**0107**] At the same time the image data are written in the storage unit **1102**, they are also input to the additional information calculating unit **1201** where the number of zero data and the number of nonzero data in block **1** are calculated as additional information. Immediately after all the image data are input, the additional information calculating unit **1201** outputs the calculation results.

[**0108**] Subsequently, from time **16** to time **32**, the first syntax element generation unit **1202** reads out the image data of block **1** one at each cycle in a reverse zigzag scan order from the storage unit **1102**. On the basis of the additional information calculated by the additional information calculating unit **1201** and the readout image data, the first syntax element generation unit **1202** calculates TotalCoeff, TrailingOnes, and TotalZeros as first syntax elements of block **1**.

[**0109**] When first data having an absolute value of 2 or more is read out, or when three data having an absolute value of 1 are read out, the first syntax element generation unit **1202** completes the synthesis of the first syntax elements of block **1**. For example, if the data as shown in **FIG. 7** is input, the first syntax elements can be calculated when −5 is input at time **27**.

[**0110**] Also, the second syntax element calculating unit **1104** reads out the image data of block **1** one at each cycle in a reverse zigzag scan order from the storage unit **1102**, and calculates the values of Run and Level of each image data. Immediately after nonzero data is input, the second syntax element calculating unit **1104** inputs the values of Level and Run of the data to the second coding unit **1106**.

[**0111**] In addition, TotalCoeff, TrailingOnes, and TotalZeros calculated before time **27** are input to the first coding unit **1105**, and coded by variable-length coding in one cycle by using a predetermined variable-length coding table. The value of TrailingOnes is held because it is necessary to select a variable-length coding table of the next block. The variable-length coded data is input to the packing unit **1107**, and held until time **32**.

[**0112**] Furthermore, from time **27** to time **32**, the second coding unit **1106** selects a variable-length coding table necessary for coding of Run and Level on the basis of the values of TotalCoeff, TrailingOnes, and TotalZeros input from the first syntax element generation unit **1202**. The second coding unit **1106** performs variable-length coding of Run and Level of each image data by using the selected variable-length coding table. Assume that the second coding unit **1106** performs coding of Run and coding of Level in parallel. The variable-length coded data is input to the packing unit **1107**, and held until time **32**.

[**0113**] In parallel with this processing, image data of block **2** shown in **FIG. 3** are written in the storage unit **1102**. The additional information calculating unit **1201** calculates the number of zero data and the number of nonzero data of block **2** in the same manner as for block **1**.

[**0114**] Then, from time **32** to time **48**, the packing unit **1107** reads out the variable-length codes input from the first and second coding units **1105** and **1106**, connects and packs the readout codes, and outputs a coded stream.

[**0115**] In parallel with this processing, the first syntax element generation unit **1202** calculates TotalCoeff, TrailingOnes, and TotalZeros of block **2** in the same manner as for block **1**.

[**0116**] Furthermore, the second syntax element calculating unit **1104** calculates the values of Run and Level of each image data of block **2** in the same manner as for block **1**. Also, the first and second coding units **1105** and **1106** perform variable-length coding of each syntax element of block **2** in the same manner as for block **1**.

[**0117**] In parallel with this processing, image data of block **3** shown in **FIG. 3** are written in the storage unit **1102**. The additional information calculating unit **1201** calculates the number of zero data and the number of nonzero data of block **3** in the same manner as for block **1**. The same processing is repeated for all blocks to process one data at each cycle.

[**0118**] A method of calculating the additional information performed by the additional information calculating unit **1201** will be explained below with reference to **FIG. 14**.

[**0119**] First, in step **S601**, variables NonZeroCoeff and ZeroCoeff are initialized to 0. Note that NonZeroCoeff is a variable representing the number of nonzero data in the block, and ZeroCoeff is a variable representing the number of zero data in the block.

[0120] Then, in step S602, the next image data in the block is input, and the flow advances to step S603. In step S603, whether the value of the image data is zero is determined. If the value is not zero, the flow advances to step S604 to add 1 to the value of NonZeroCoeff, and further advances to step S606. If the value is zero, the flow advances to step S605 to add 1 to the value of ZeroCoeff, and further advances to step S606.

[0121] In step S606, whether the present image data is the last data of the block is determined. If the image data is the last data of the block, the flow advances to step S607. In step S607, the present values of NonZeroCoeff and ZeroCoeff are output as calculation results, and the processing for the present block is terminated. If the image data is not the last data of the block, the flow returns to step S602 to input the next image data. Note that in this embodiment, the values of NonZeroCoeff and ZeroCoeff are calculated by counting both zero data and nonzero data. However, it is also possible to count only one of zero data and nonzero data and calculate the other by subtracting the count from the number of image data of the block.

[0122] A method of synthesizing the values of first syntax elements performed by the first syntax element generation unit **1202** will be explained below.

[0123] The value of TotalCoeff is the same as NonZeroCoeff calculated by the additional information calculating unit **1201**, so the value of NonZeroCoeff is directly output.

[0124] The operation of calculating TrailingOnes will be explained below with reference to a flowchart shown in **FIG. 15**.

[0125] First, in step S701, a variable TrailingOnes is initialized to 0.

[0126] Then, in step S702, the next image data in the block is input, and the flow advances to step S703. In step S703, whether the value of the image data is zero is determined. If the value is zero, the flow advances to step S707. If the value is not zero, the flow advances to step S704. In step S704, whether the absolute value of the image data is 1 is determined.

[0127] If it is determined in step S704 that the absolute value of the image data is not 1, the flow advances to step S708 to output the present value of TrailingOnes and terminate the processing.

[0128] If it is determined in step S704 that the absolute value of the image data is 1, the flow advances to step S705 to add 1 to TrailingOnes, and further advances to step S706. If the value of TrailingOnes is 3 in step S706, the flow advances to step S708 to output the present value of TrailingOnes and terminate the processing. If the value of TrailingOnes is 2 or less, the flow advances to step S707.

[0129] In step S707, whether the present image data is the last data of the block is determined. If the image data is not the last data of the block, the flow returns to step S702 to input the next image data. If the image data is the last data of the block, the flow advances to step S708 to output the present value of TrailingOnes and terminate the processing.

[0130] The operation of calculating TotalZeros will be explained below with reference to a flowchart shown in **FIG. 16**.

[0131] First, in step S801, ZeroCoeff calculated by the additional information calculating unit **1201** is substituted into a variable TotalZeros.

[0132] Then, in step S802, the next image data in the block is input, and the flow advances to step S803.

[0133] In step S803, whether the value of the image data is zero is determined. If the value is zero, the flow advances to step S804 to subtract 1 from the value of TotalZeros, and further advances to step S805.

[0134] If the value is not zero in step S803, the flow advances to step S806 to output the present value of TotalZeros and terminate the processing.

[0135] In step S805, whether the present image data is the last data of the block is determined. If the image data is the last data of the block, the processing advances to step S806 to output the present value of TotalZeros as a calculation result, and terminate the processing for the present block. If the image data is not the last data of the block, the flow returns to step S802 to input the next image data.

[0136] By the method described above, the first syntax element generation unit **1202** can calculate the values of TotalCoeff, TrailingOnes, and TotalZeros when data having an absolute value of 2 or more or third data having an absolute value of 1 is read out.

[0137] The processing in the second syntax element calculating unit **1104** is the same as in the first embodiment, so an explanation thereof will be omitted. Note that the second syntax element calculating unit **1104** can calculate Run and Level when nonzero data is read out.

[0138] In the second embodiment of the present invention as described above, the second coding unit **1106** selects a variable-length coding table in advance by using the syntax elements calculated by the first syntax element calculating unit **1103**, and performs variable-length coding of Run and Level. Variable-length coding cannot be started unless the values of TotalCoeff, TrailingOnes, and TotalZeros are determined. However, if it takes a long time to read out data having an absolute value of 2 or more or third data having an absolute value of 1, the numbers of Run and Level are small. Therefore, variable-length coding can be completed in the same clock cycle as when the last data in the block is read out as a whole.

[0139] In parallel with this processing, the first coding unit **1105** performs variable-length coding on TotalCoeff, TrailingOnes, and TotalZeros. As a consequence, variable-length codes of all syntax elements can be obtained in the same clock cycle as when read of image data in the block from the storage unit **1102** is completed.

[0140] Also, if the values of first syntax elements can be calculated more easily from image data read out in the reverse zigzag scan order, the circuit scale of the first syntax element calculating unit **1103** can be made smaller than that in the first embodiment.

[0141] Note that in the first and second embodiments of the present invention, the number of image data input in each cycle is 1. However, the present invention is also applicable to a method in which a plurality of image data are input in each cycle by performing coding processes in parallel.

[0142] Note also that in the first and second embodiments of the present invention, the CAVLC method of H.264 is explained as an example. However, the present invention can also be applied to coding of a plurality of syntax elements depending one each other, in addition to the CAVLC method.

Other Embodiments

[0143] The invention can be implemented by supplying a software program, which implements the functions of the foregoing embodiments, directly or indirectly to a system or apparatus, reading the supplied program code with a computer of the system or apparatus, and then executing the program code. In this case, so long as the system or apparatus has the functions of the program, the mode of implementation need not rely upon a program.

[0144] Accordingly, since the functions of the present invention are implemented by computer, the program code installed in the computer also implements the present invention. In other words, the claims of the present invention also cover a computer program for the purpose of implementing the functions of the present invention.

[0145] In this case, so long as the system or apparatus has the functions of the program, the program may be executed in any form, such as an object code, a program executed by an interpreter, or scrip data supplied to an operating system.

[0146] Example of storage media that can be used for supplying the program are a floppy disk, a hard disk, an optical disk, a magneto-optical disk, a CD-ROM, a CD-R, a CD-RW, a magnetic tape, a non-volatile type memory card, a ROM, and a DVD (DVD-ROM and a DVD-R).

[0147] As for the method of supplying the program, a client computer can be connected to a website on the Internet using a browser of the client computer, and the computer program of the present invention or an automatically-installable compressed file of the program can be downloaded to a recording medium such as a hard disk. Further, the program of the present invention can be supplied by dividing the program code constituting the program into a plurality of files and downloading the files from different websites. In other words, a WWW (World Wide Web) server that downloads, to multiple users, the program files that implement the functions of the present invention by computer is also covered by the claims of the present invention.

[0148] It is also possible to encrypt and store the program of the present invention on a storage medium such as a CD-ROM, distribute the storage medium to users, allow users who meet certain requirements to download decryption key information from a website via the Internet, and allow these users to decrypt the encrypted program by using the key information, whereby the program is installed in the user computer.

[0149] Besides the cases where the aforementioned functions according to the embodiments are implemented by executing the read program by computer, an operating system or the like running on the computer may perform all or a part of the actual processing so that the functions of the foregoing embodiments can be implemented by this processing.

[0150] Furthermore, after the program read from the storage medium is written to a function expansion board inserted into the computer or to a memory provided in a function expansion unit connected to the computer, a CPU or the like mounted on the function expansion board or function expansion unit performs all or a part of the actual processing so that the functions of the foregoing embodiments can be implemented by this processing.

[0151] As many apparently widely different embodiments of the present invention can be made without departing from the spirit and scope thereof, it is to be understood that the invention is not limited to the specific embodiments thereof except as defined in the appended claims.

CLAIM OF PRIORITY

[0152] This application claims priority from Japanese Patent Application No. 2004-331107 filed on Nov. 15, 2004, which is hereby incorporated herein by reference herein.

What is claimed is:

1. A variable-length coding apparatus which receives image data block by block, and generates coded data from syntax elements, comprising:

a storage unit that temporarily stores the image data;

a first syntax element calculating unit that calculates a first syntax element from the image data, at the same time the image data is stored in said storage unit;

a first coding unit that generates coded data of the first syntax element from a value of the first syntax element;

a second syntax element calculating unit that reads out the image data stored in said storage unit, and calculates a value of a second syntax element from the image data;

a second coding unit that generates coded data of the second syntax element on the basis of the values of the first and second syntax elements; and

a packing unit that packs the coded data of the first and second syntax elements.

2. The apparatus according to claim 1, wherein said second syntax element calculating unit reads out the image data stored in said storage unit in a reverse zigzag scan order, and calculates the second syntax element.

3. The apparatus according to claim 1, wherein the first syntax element includes at least one of the number of nonzero data in a block, the number of zero data in the block, the number of zero data input before last nonzero data in the block, and the number of data having an absolute value of 1 and input after last data having an absolute value of not less than 2 in the block.

4. The apparatus according to claim 1, wherein the second syntax element includes at least one of a zero run length of the image data and a level value of the image data.

5. The apparatus according to claim 1, wherein said packing unit packs the coded data, and outputs a coded stream defined by the H.264 standard.

6. The apparatus according to claim 1, wherein said first syntax element calculating unit comprises:

an additional information calculating unit that calculates additional information indicating at least one of the number of zero data and the number of nonzero data from the input image data; and

a first syntax element generation unit that calculates the first syntax element on the basis of the image data

temporarily stored in said storage unit and the additional information calculated by said additional information calculating unit.

7. The apparatus according to claim 6, wherein the image data from said storage unit is read out in a reverse zigzag scan order, and supplied to said first syntax element generation unit.

8. A variable-length coding method which receives image data block by block, and generates coded data from syntax elements, comprising:

temporarily storing the image data in a storage unit;

calculating a first syntax element from the image data, at the same time the image data is stored in the storage unit;

generating coded data of the first syntax element from a value of the first syntax element;

reading out the image data stored in the storage unit, and calculating a value of a second syntax element from the image data;

generating coded data of the second syntax element on the basis of the values of the first and second syntax elements; and

connecting the coded data of the first and second syntax elements.

9. The method according to claim 8, wherein said second syntax element calculation, the image data stored in the storage unit is read out in a reverse zigzag scan order, and the second syntax element is calculated.

10. The method according to claim 8, wherein the first syntax element includes at least one of the number of nonzero data in a block, the number of zero data in the block, the number of zero data input before last nonzero data in the block, and the number of data having an absolute value of 1 and input after last data having an absolute value of not less than 2 in the block.

11. The method according to claim 8, wherein the second syntax element includes at least one of a zero run length of the image data and a level value of the image data.

12. The method according to claim 8, wherein in said connecting the coded data, the coded data is connected, and a coded stream defined by the H.264 standard is output.

13. The method according to claim 8, wherein in said first syntax element calculation,

additional information indicating at least one of the number of zero data and the number of nonzero data from the input image data are calculated; and

the first syntax element is calculated on the basis of the image data temporarily stored in the storage unit and the additional information calculated in said additional information calculation.

14. The method according to claim 13, wherein the image data from the storage unit is read out in a reverse zigzag scan order, and used in said first syntax element synthesizing.

15. A computer-readable storage medium characterized by storing a computer program characterized by executing a variable-length coding method cited in claim 8.

* * * * *