

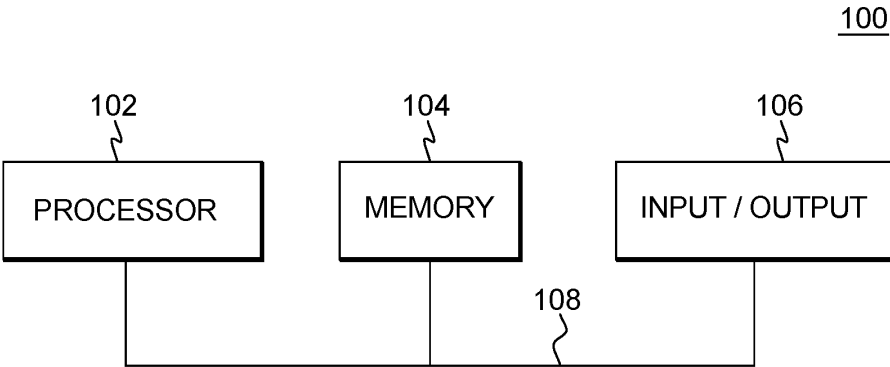


FIG. 1A

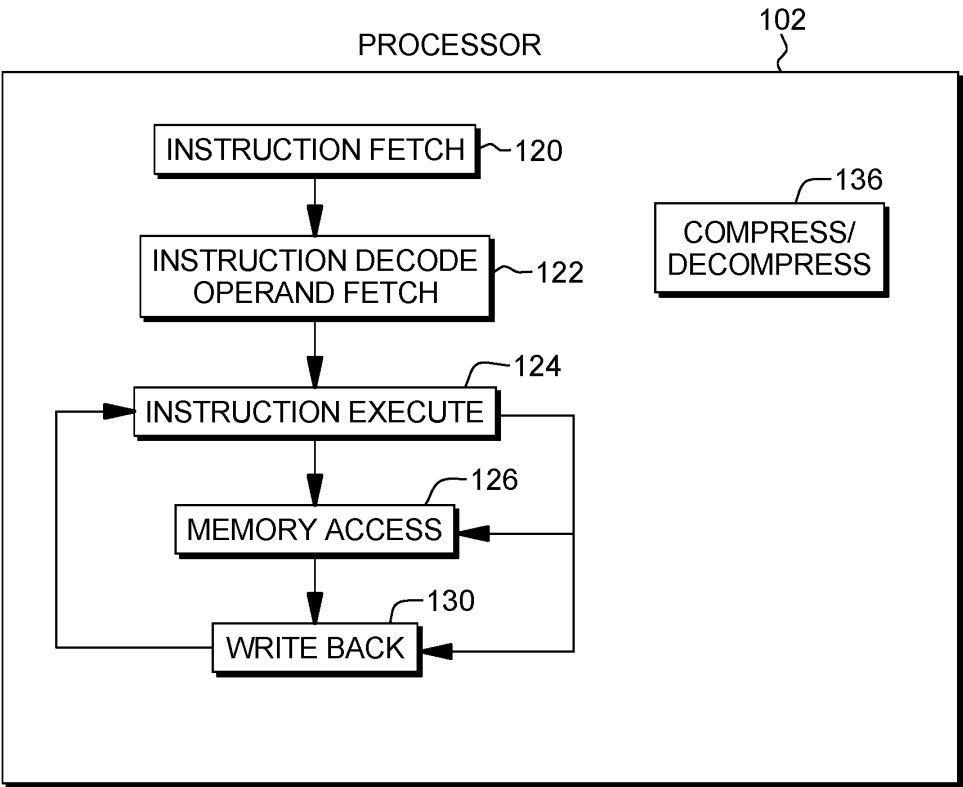


FIG. 1B

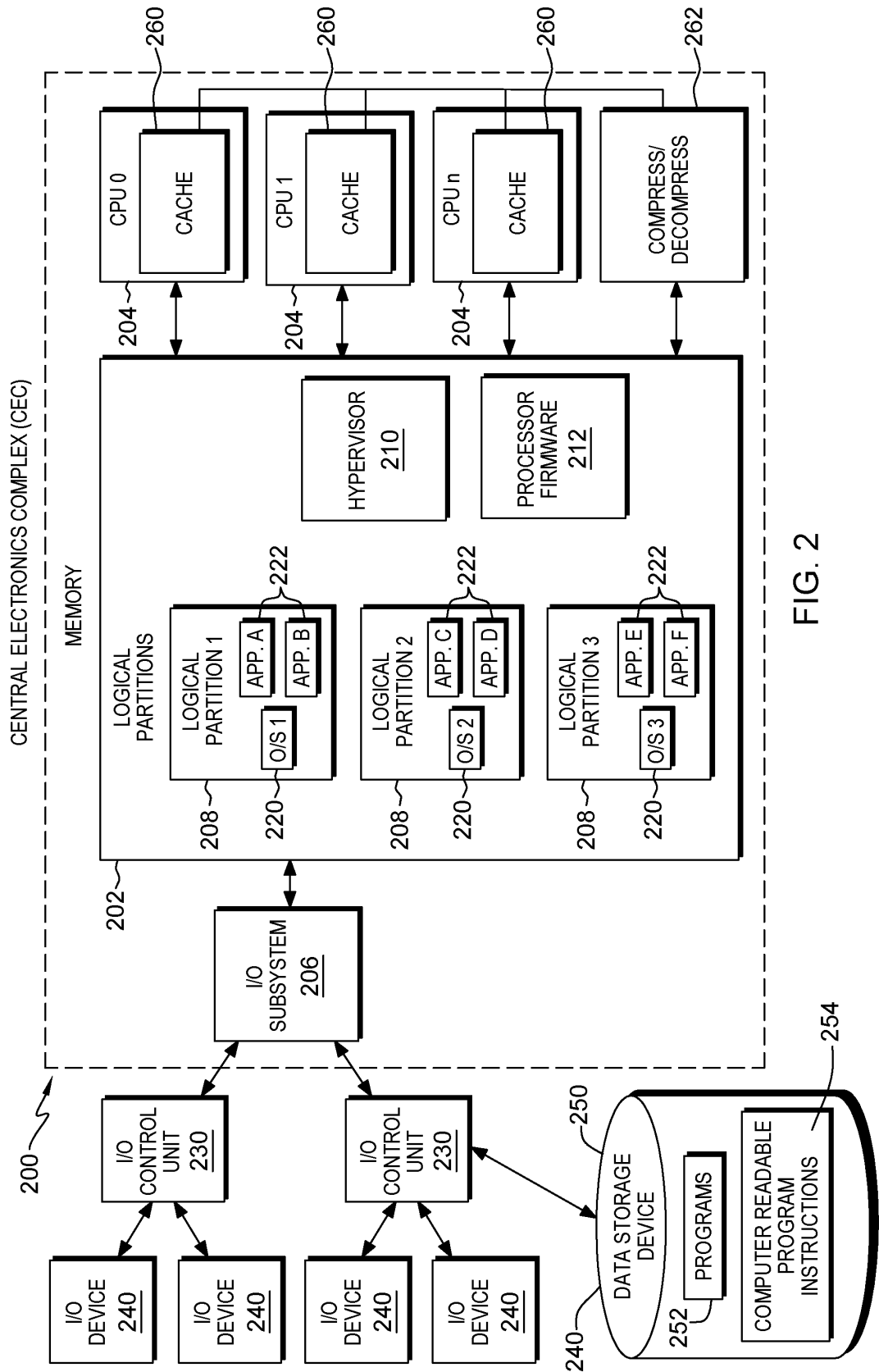
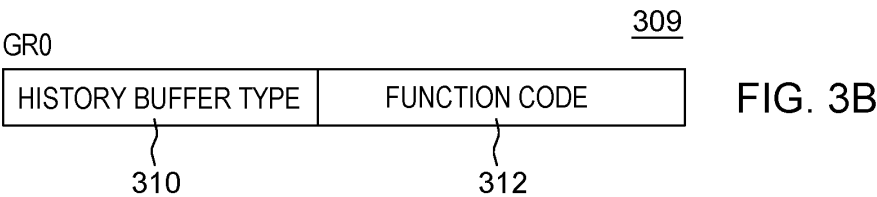
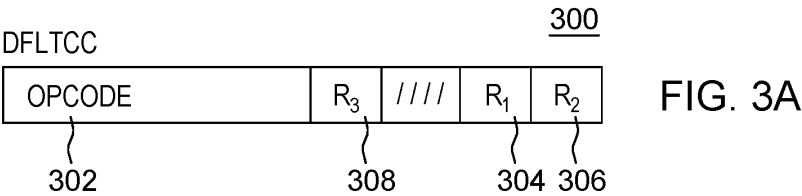
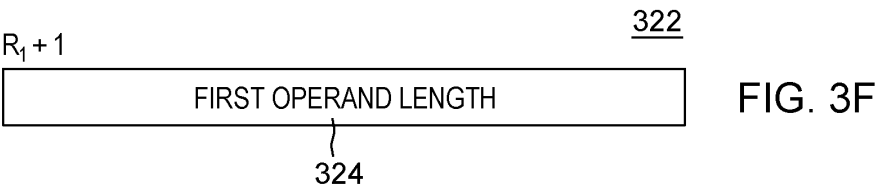
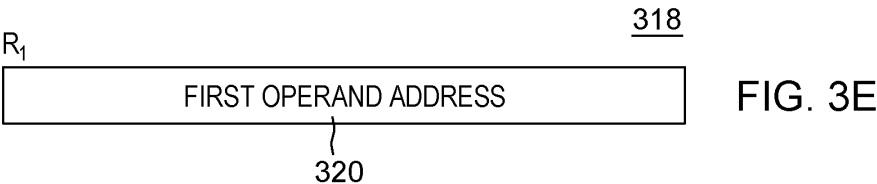
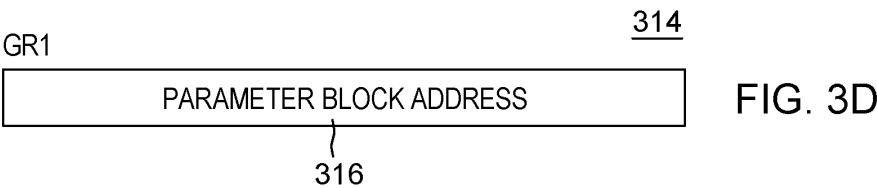


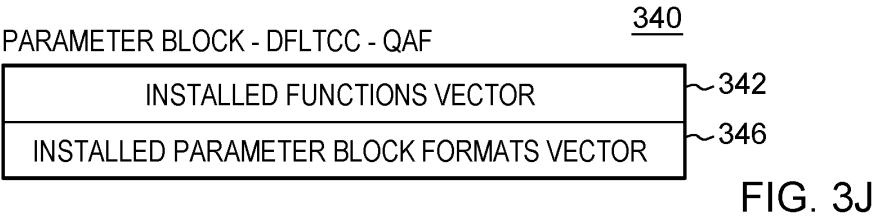
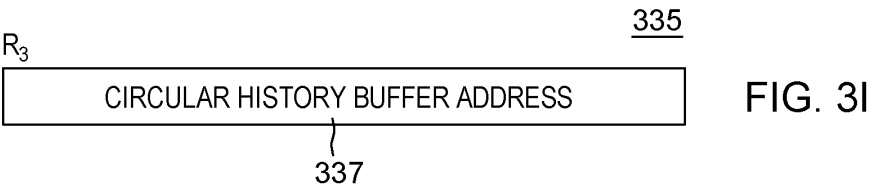
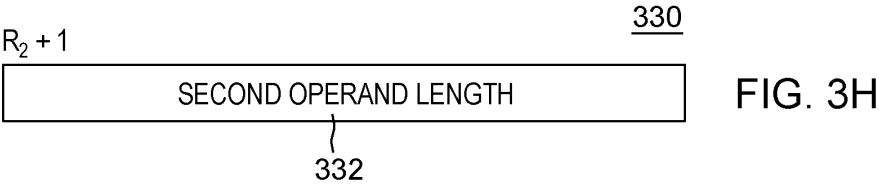
FIG. 2



	CODE	FUNCTION	PARAMETER BLOCK SIZE (BYTES)
313 ~	0	DFLTCC - QAF	32
315 ~	1	DFLTCC - GDHT	384
317 ~	2	DFLTCC - CMPR	1536
319 ~	4	DFLTCC - XPND	1536

FIG. 3C





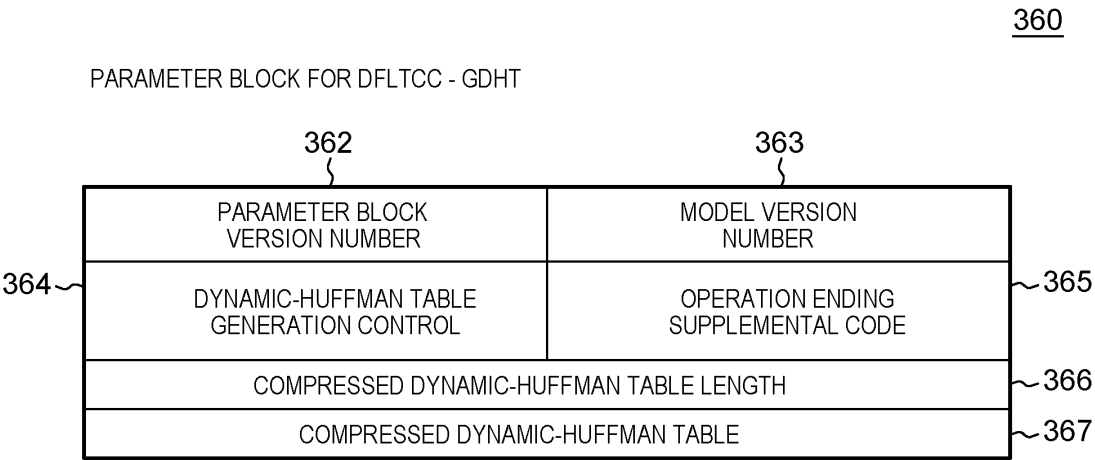


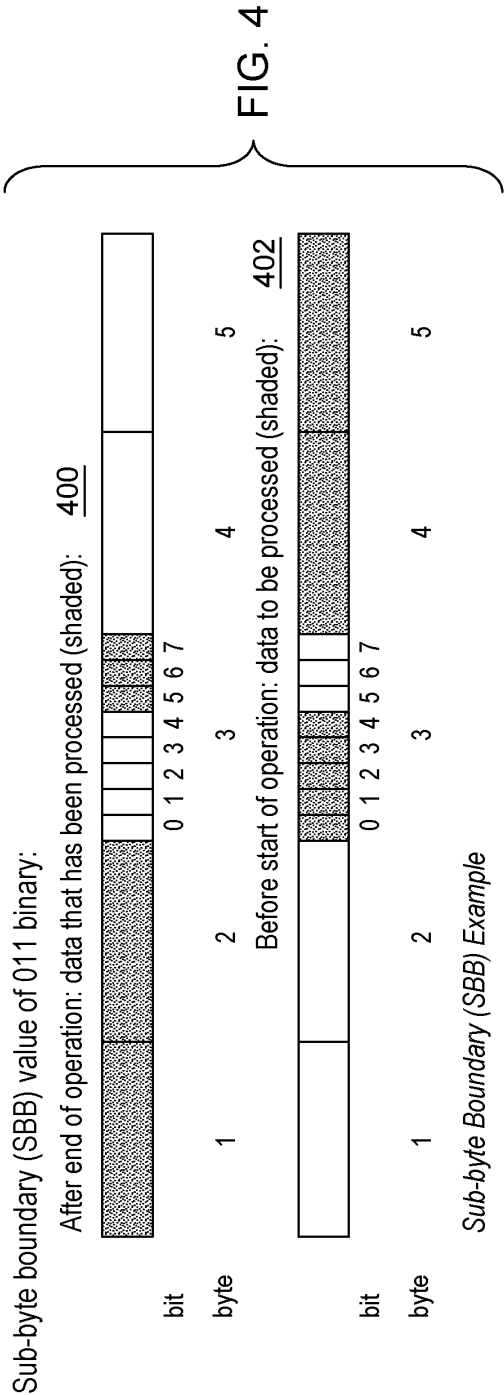
FIG. 3K

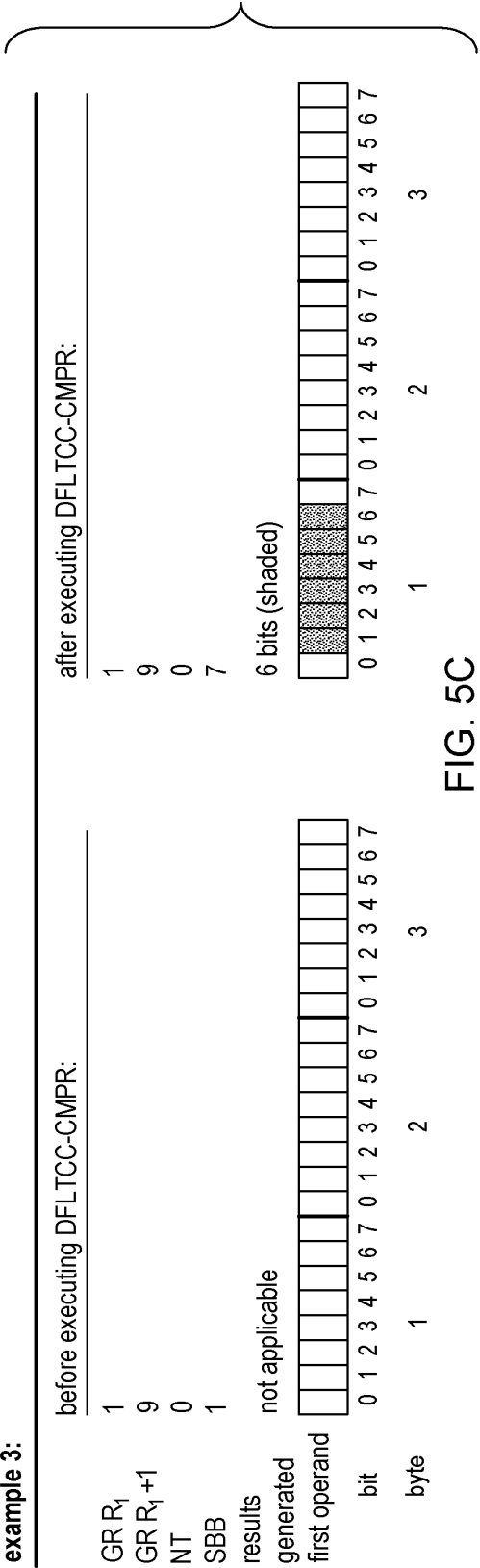
370

PARAMETER BLOCK FOR DFLTCC - CMPR + DFLTCC - XPND

PARAMETER BLOCK VERSION NUMBER <u>362</u>				MODEL VERSION NUMBER <u>363</u>			CONTINUATION FLAG <u>373</u>			
NEW TASK <u>374</u>	CHECK VALUE TYPE <u>375</u>	HUFFMAN TABLE TYPE <u>376</u>	BLOCK CONT. FLAG <u>377</u>	BLOCK CLOSING CONTROL <u>378</u>	BLOCK HEADER FINAL <u>379</u>	DHTGC <u>364</u>	SUB-BYTE BOUNDARY <u>381</u>	OPER ENDING SUPP. CODE <u>365</u>	INCOMP FUNC STATUS <u>383</u>	INCOMP FUNC LENGTH <u>384</u>
HISTORY LENGTH <u>385</u>				HISTORY OFFSET <u>386</u>						
CHECK VALUE <u>387</u>			END-OF-BLOCK SYMBOL <u>388</u>			END-OF-BLOCK LENGTH <u>389</u>				
COMPRESSED DYNAMIC-HUFFMAN TABLE LENGTH <u>366</u>										
COMPRESSED DYNAMIC-HUFFMAN TABLE <u>367</u>										
CONTINUATION STATE BUFFER <u>392</u>										

FIG. 3L





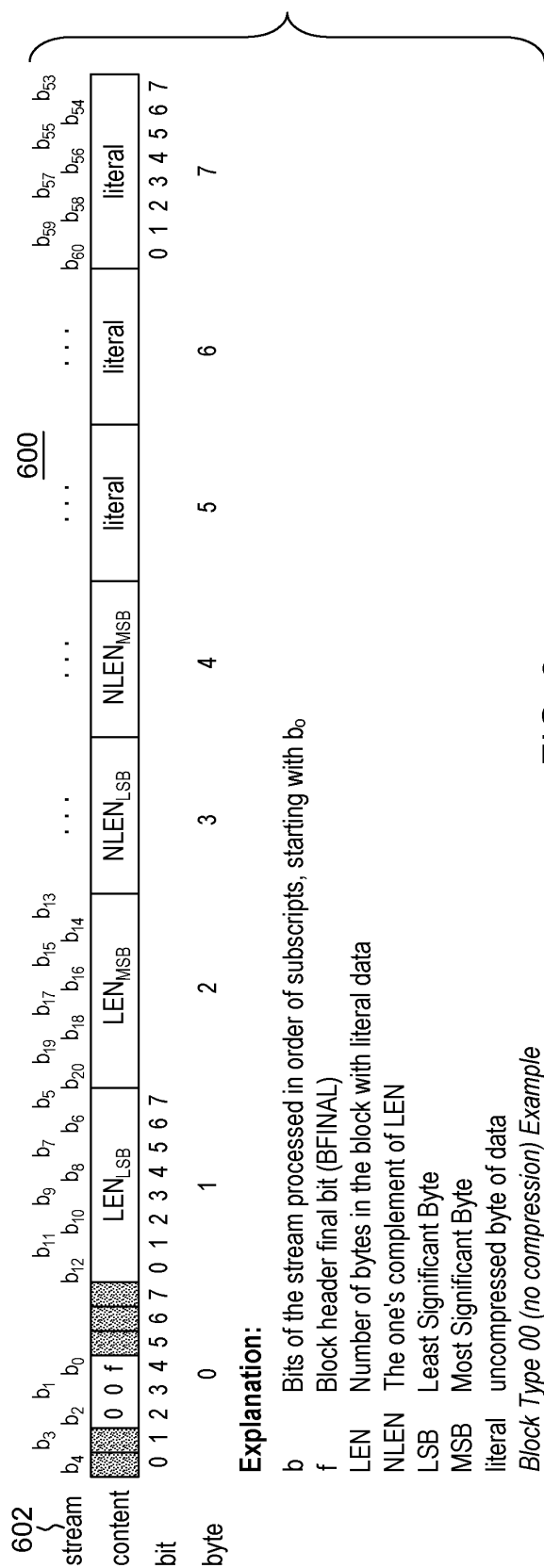


FIG. 6

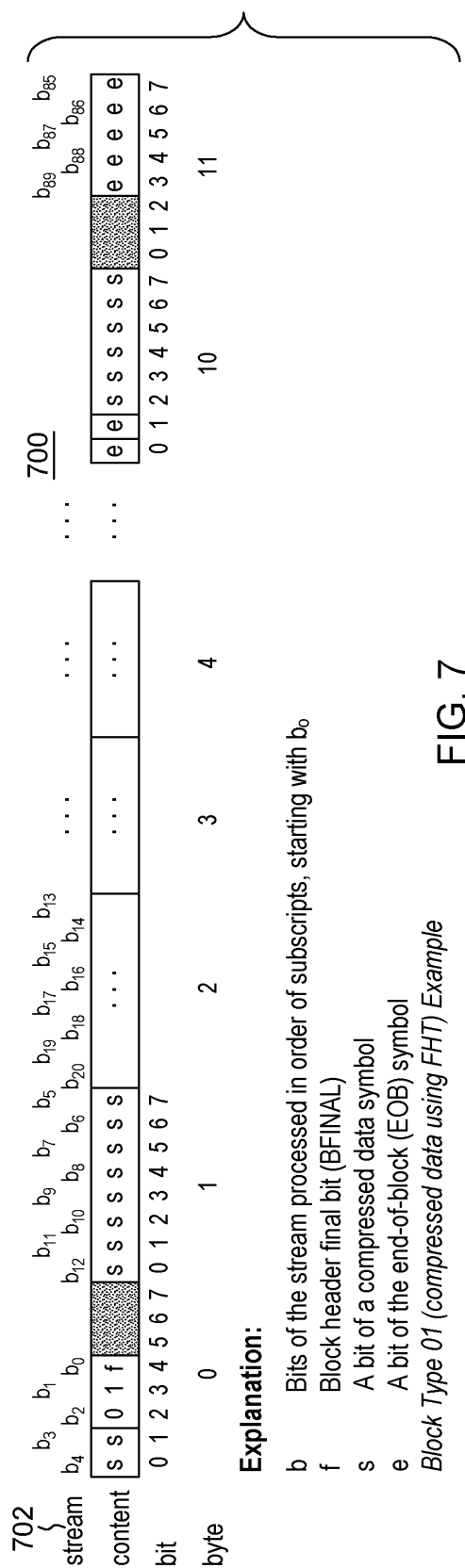


FIG. 7

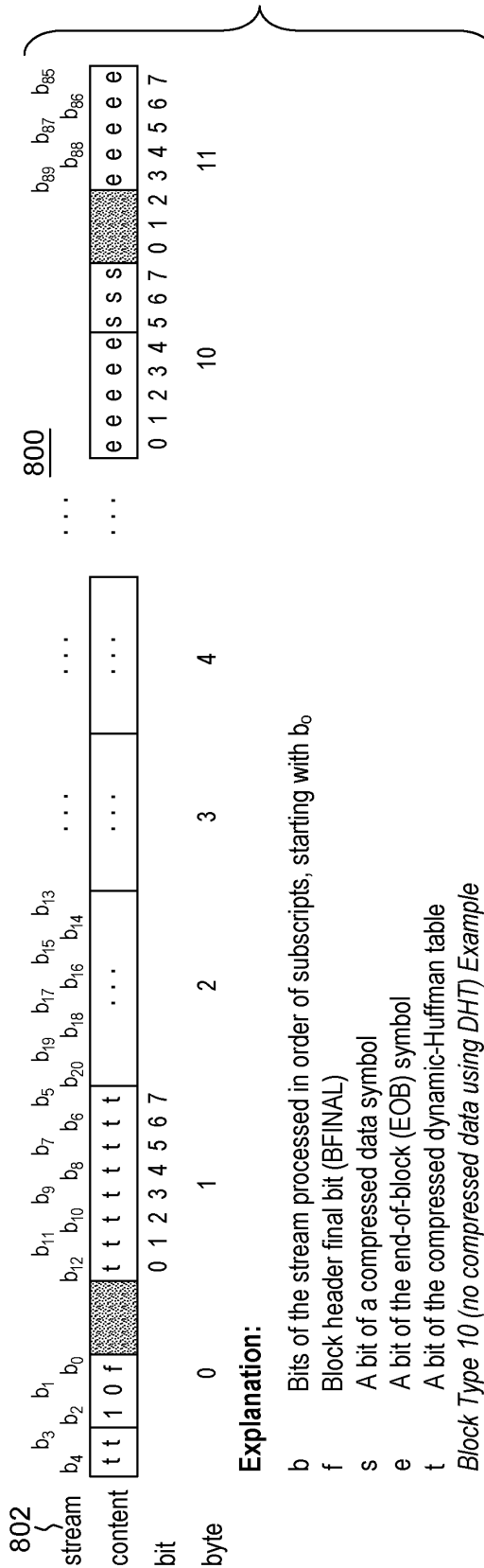


FIG. 8

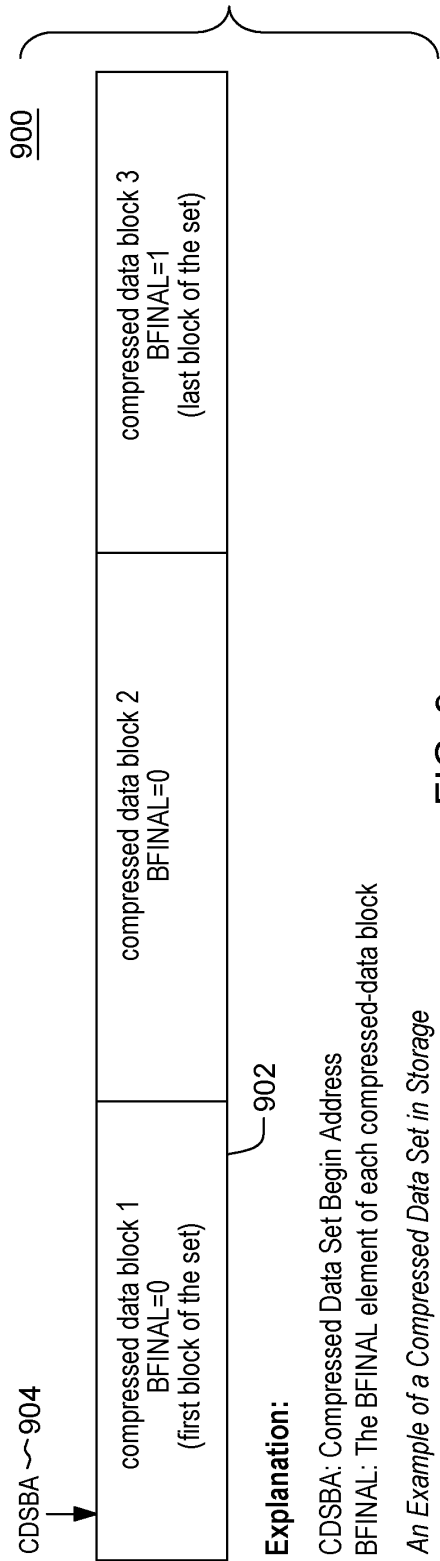


FIG. 9

1000			
1002	L	1,PBLKADDR	Load parameter block address into GR1
	IILL	0,2	Set GR0(56:63)=02 for DFLTCC-CMPR function
	L	2,CDSBADDR	Load compressed data set output address into GR2
IABLK1	L	4,UD1ADDR	Load uncompressed data input address into GR4
	DFLTCC	2,4,10	Compress data into block1
	BRC	1,IABLK1	If operation ends with CC=3, branch back to resume the operation
	:		
1004	L	6,UD2ADDR	Load next uncompressed data input address into GR6
	DFLTCC	2,6,10	Compress data into block2
IABLK2	BRC	1,IABLK2	If operation ends with CC=3, branch back to resume the operation
	:		
	L	8,UD3ADDR	Load next uncompressed data input address into GR8
IABLK3	DFLTCC	2,8,10	Compress data into block3
	BRC	1,IABLK3	If operation ends with CC=3, branch back to resume the operation
	:		

Sample of a Program Compressing Data into Three Blocks of a Compressed Data Set

FIG. 10

parameter block field 1100	field value at start of operation ~1102	field value at end of ~1104 operation when condition code 1, 2, or 3 is set (also the value at the start of resuming the operation)	field value at end of operation when condition code 0 is set ~1106
new task (NT)	1 (specified by the program)	0	0
continuation flag (CF)	initialized to zero by the program	1	0
continuation state buffer (CSB)	initialized to all zeros by the program	internal-state data recorded at the end of the operation, which is required when the operation later resumes	undefined
check value	treated as an initial value, since NT is one	value generated from portion of data processed since the start of the operation and starting value for resuming the operation	value generated from all data processed by the operation and starting value for a subsequent operation
sub-byte boundary (SBB)	treated as equal to zero, since NT is one	applies to the last byte processed by the operation and the first byte when the operation later resumes	applies to the last byte processed by the operation and the first byte for a subsequent operation

Parameter Block Contents for a DFL TCC-CMPR Function Operating on the First Compressed Data Block of a Set

FIG. 11

parameter block field	field value at start of operation	field value at end of operation when condition code 1, 2, or 3 is set (also the value at the start of resuming the operation)	field value at end of operation when condition code 0 is set
new task (NT)	0 (specified by the program)	0	0
continuation flag (CF)	initialized to zero by the program	1	0
continuation state buffer (CSB)	initialized to all zeros by the program	internal-state data recorded at the end of the operation, which is required when the operation later resumes	undefined
check value	value generated from prior operation (preserved by the program)	value generated from portion of data processed since the start of the current operation and starting value for resuming the current operation	value generated from all data processed by the operation and starting value for a subsequent operation
sub-byte boundary (SBB)	value associated with last byte processed by the prior operation (preserved by the program)	applies to the last byte processed by the current operation and the first byte when the current operation later resumes	applies to the last byte processed by the operation and the first byte for a subsequent operation

Parameter Block Contents for a DFL TCC-CMPR Function Operating on the Second Compressed Data Block of a Set

FIG. 12

1300

	L	1,PBLKADDR	Load parameter block address into GR1
	ILL	0,4	Set GR0(56:63)=04 for DFLTCC-XPND function
	L	2,UDADDR	Load uncompressed data output address into GR2
	L	4,CDSBADDR	Load compressed data set input address into GR4
IASET		2,4,10	Uncompress all blocks of the data set
	DFLTCC	1,IASET	If operation ended with CC=3, branch back to resume the operation
	BRC		
	:		
	:		

Sample of a Program Uncompressing Data from a Compressed Data Set

FIG. 13

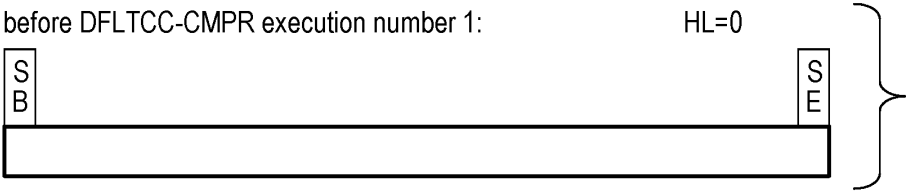


FIG. 14A

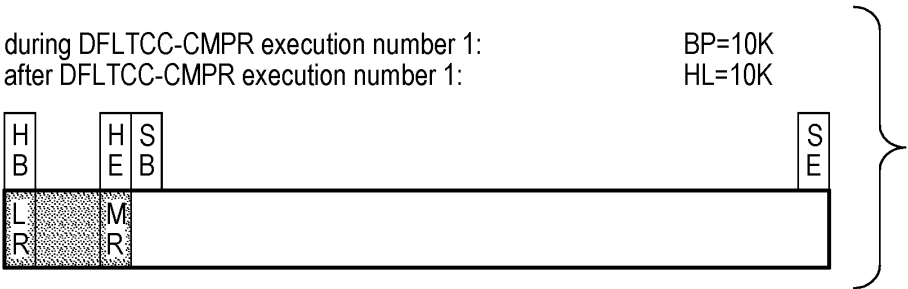


FIG. 14B

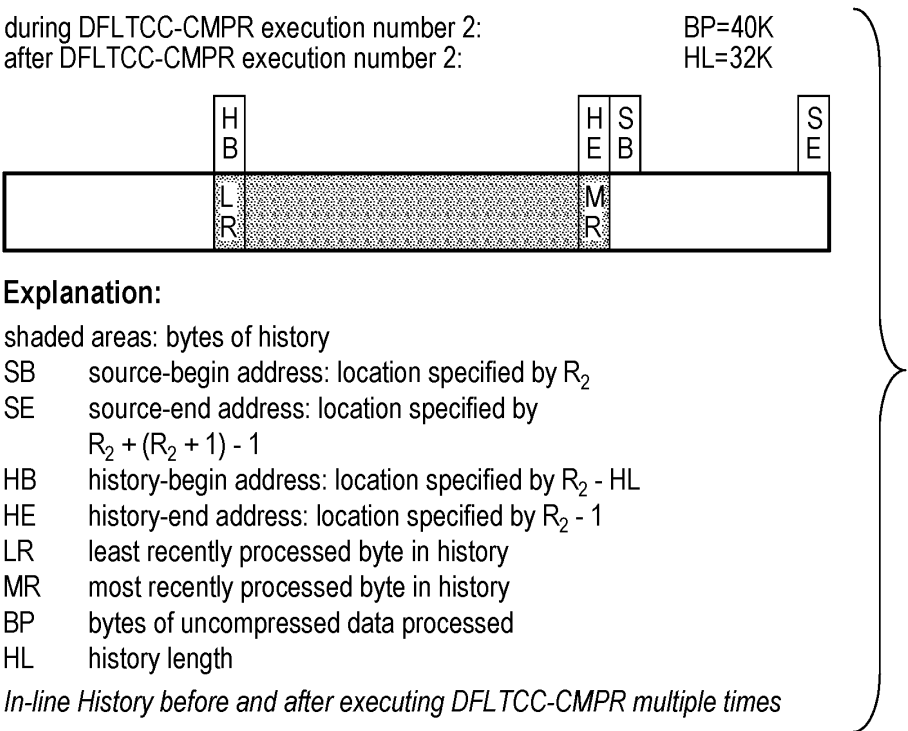


FIG. 14C

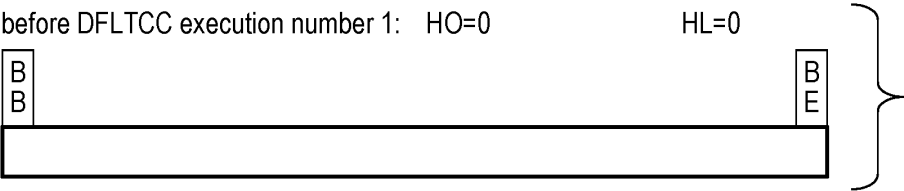


FIG. 15A

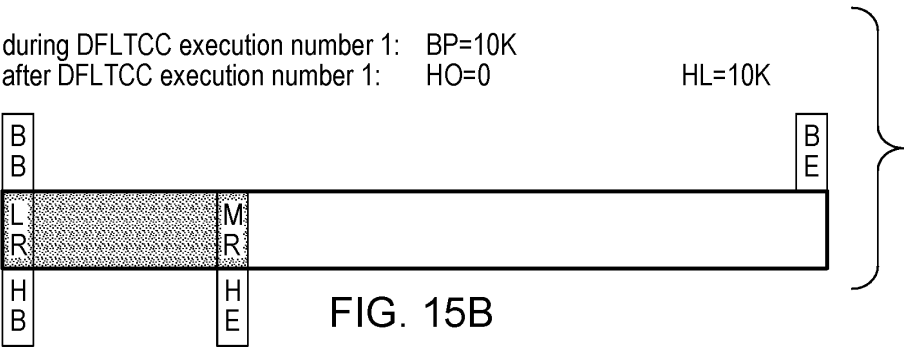


FIG. 15B

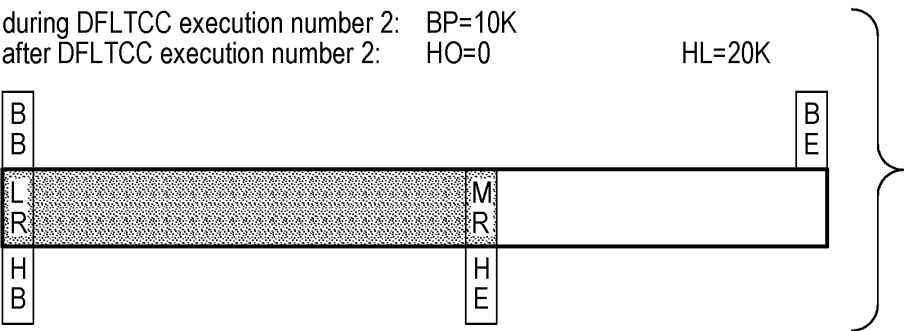


FIG. 15C

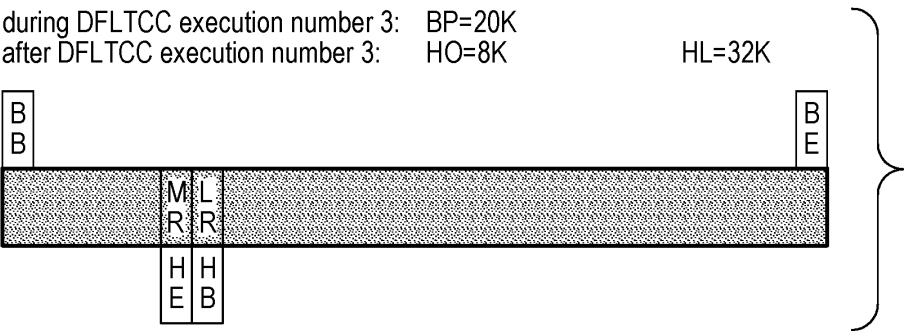


FIG. 15D

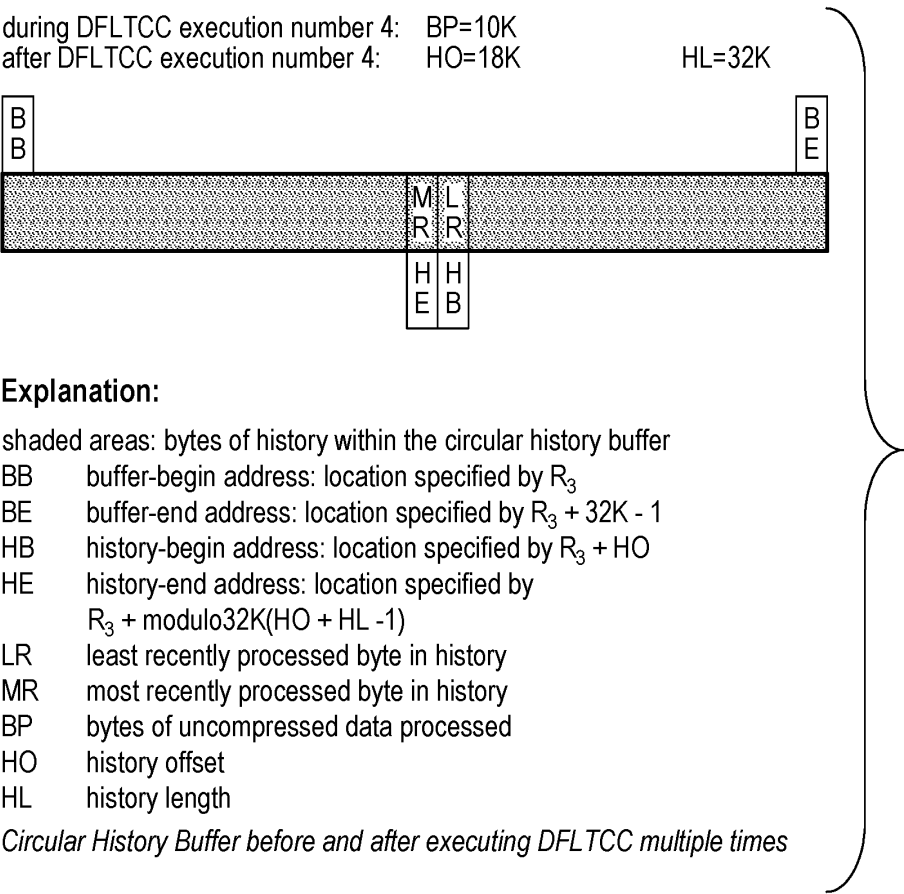


FIG. 15E

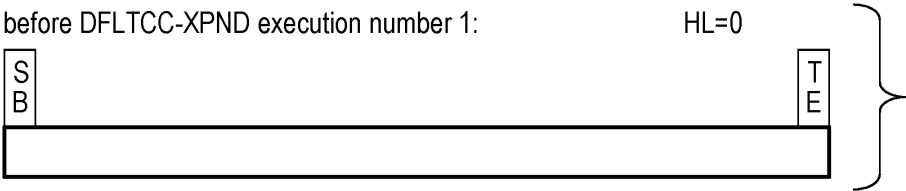


FIG. 16A

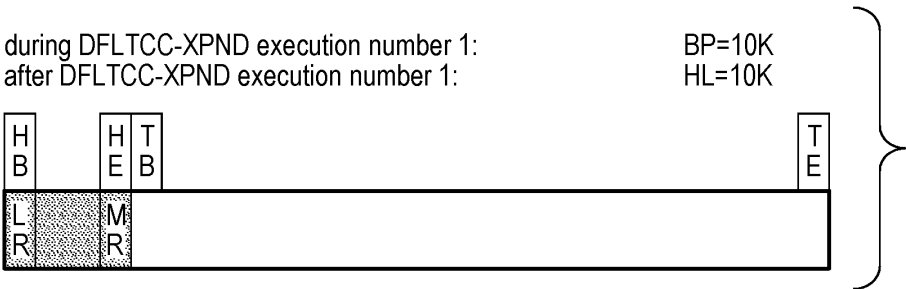
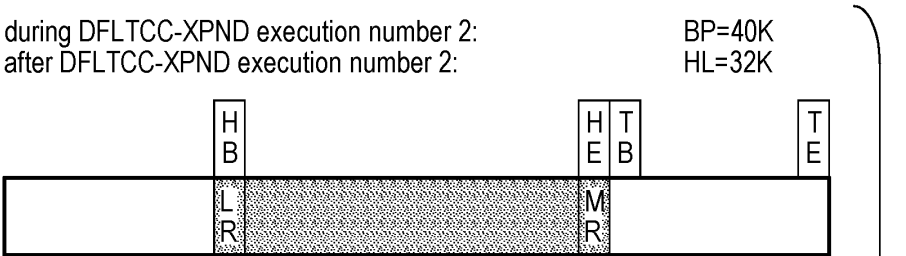


FIG. 16B



Explanation:

shaded areas: bytes of history

TB target-begin address: location specified by R_1

TE target-end address: location specified by $R_1 + (R_1 + 1) - 1$

HB history-begin address: location specified by $R_1 - HL$

HE history-end address: location specified by $R_1 - 1$

LR least recently processed byte in history

MR most recently processed byte in history

BP bytes of uncompressed data processed

HL history length

In-line History before and after executing DFLTCC-XPND multiple times

FIG. 16C

19/25

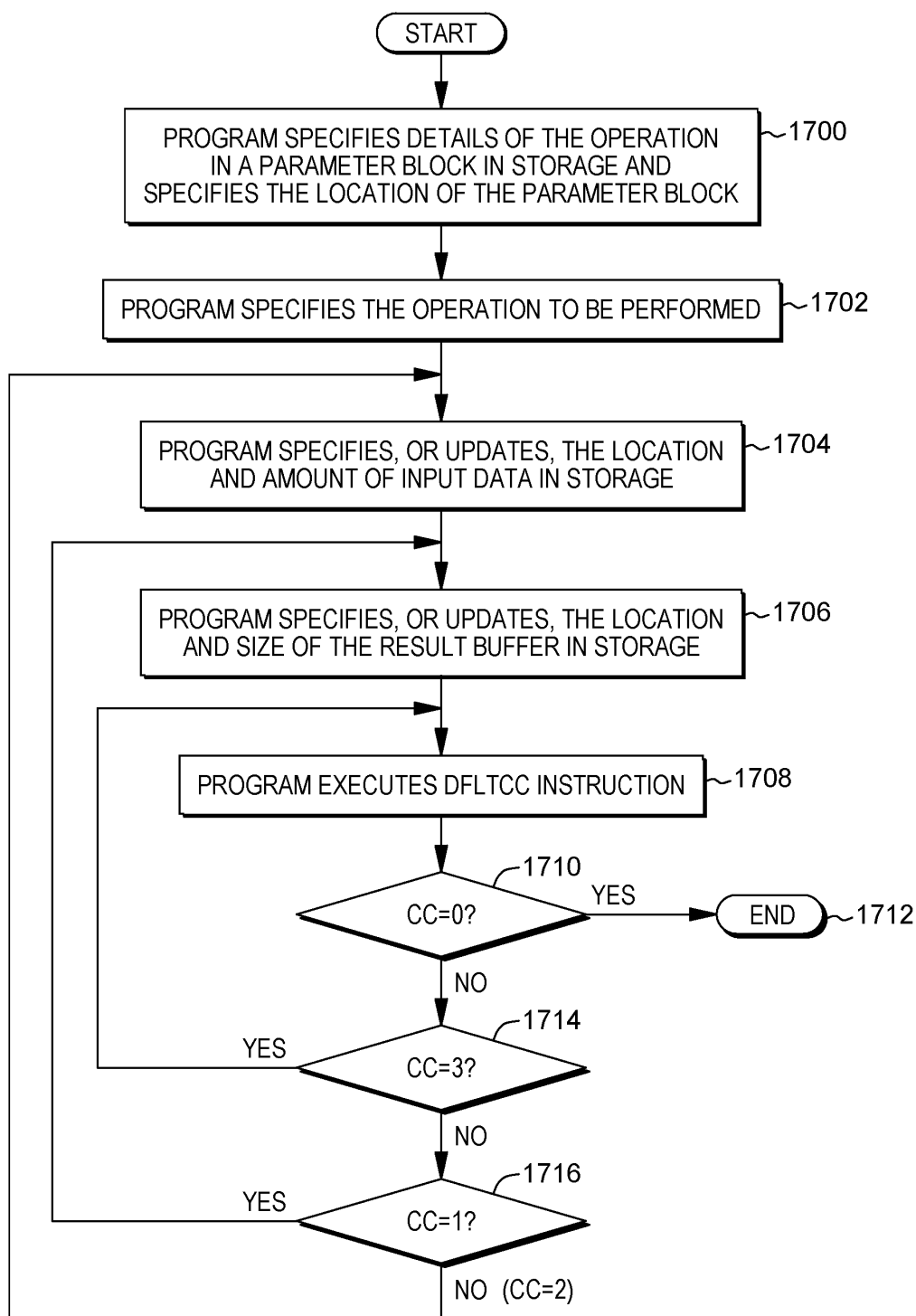


FIG. 17

20/25

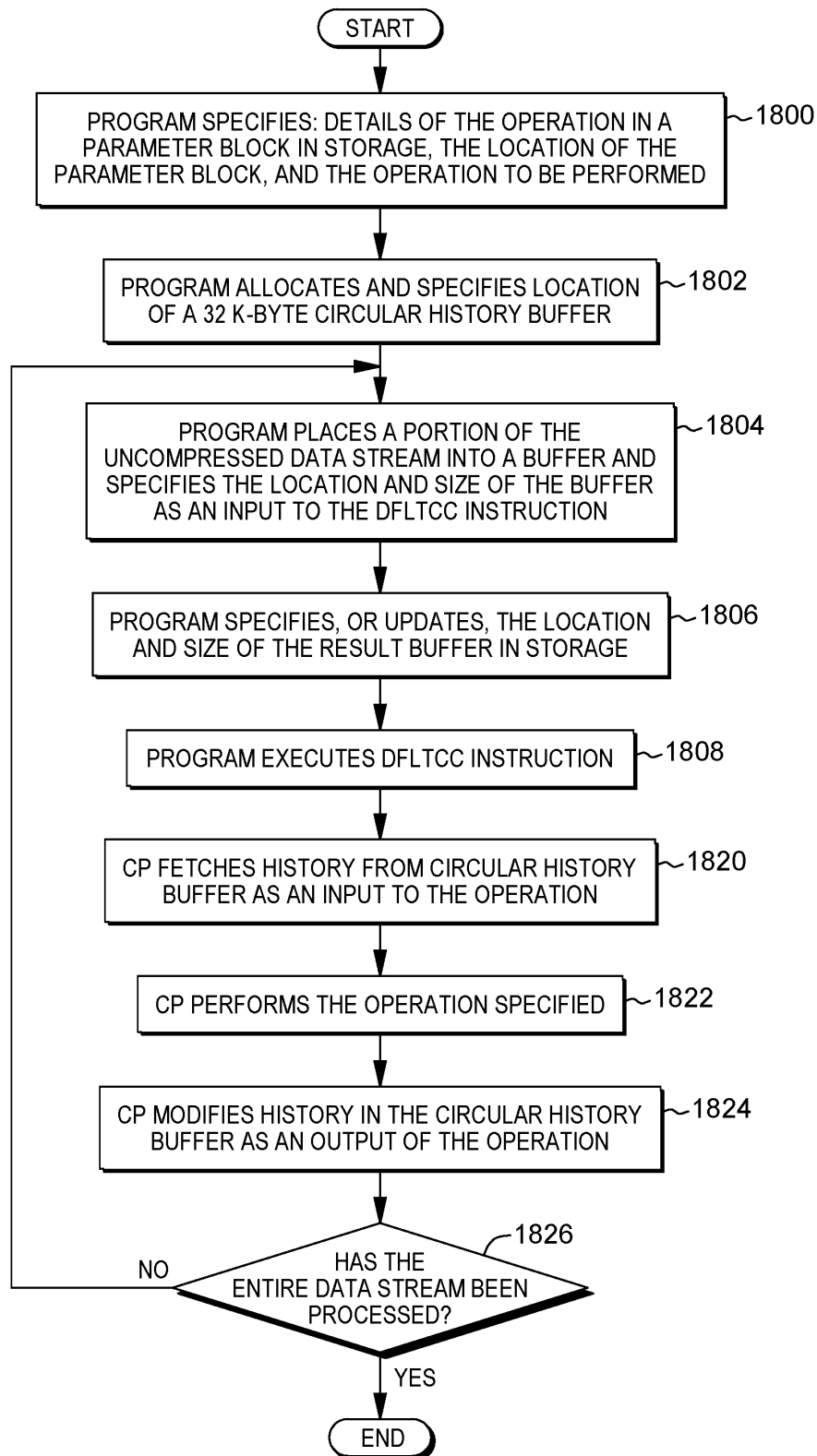


FIG. 18

21/25

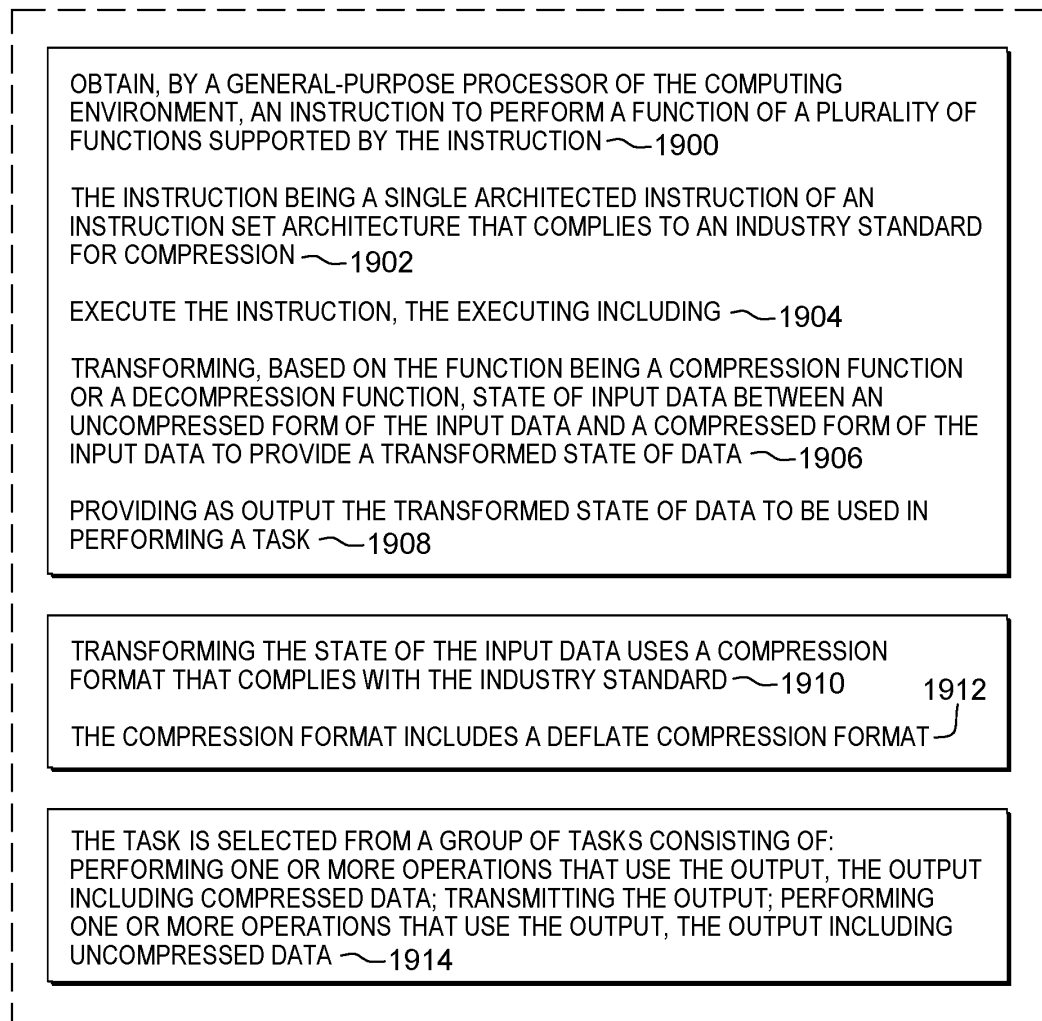


FIG. 19A

22/25

THE INSTRUCTION INCLUDES AN OPERATION CODE FIELD THAT INCLUDES AN OPERATION CODE TO SPECIFY AN OPERATION AND A PLURALITY OF REGISTER FIELDS TO SPECIFY A PLURALITY OF REGISTERS TO BE USED BY THE INSTRUCTION ~1920

THE PLURALITY OF REGISTERS INCLUDES ONE REGISTER USED TO IDENTIFY AN OUTPUT OPERAND LOCATION TO BE USED BY THE INSTRUCTION AS AN OUTPUT AND ANOTHER REGISTER USED TO IDENTIFY AN INPUT OPERAND LOCATION TO BE USED BY THE INSTRUCTION AS AN INPUT, THE INPUT DEPENDING ON THE FUNCTION ~1922

BASED ON THE FUNCTION BEING THE COMPRESSION FUNCTION, THE INPUT INCLUDES DATA FROM THE INPUT OPERAND LOCATION TO BE ENCODED TO PROVIDE COMPRESSED DATA SYMBOLS STORED TO THE OUTPUT OPERAND LOCATION ~1924

BASED ON THE FUNCTION BEING THE DECOMPRESSION FUNCTION, THE INPUT INCLUDES THE COMPRESSED DATA SYMBOLS FROM THE INPUT OPERAND LOCATION TO BE DECODED TO PROVIDE UNCOMPRESSED DATA STORED TO THE OUTPUT OPERAND LOCATION ~1926

THE INSTRUCTION FURTHER USES A SELECTED REGISTER TO INDICATE A FUNCTION OF THE PLURALITY OF FUNCTIONS TO BE PERFORMED BY THE INSTRUCTION ~1928

THE PLURALITY OF FUNCTIONS INCLUDES A QUERY FUNCTION, THE COMPRESSION FUNCTION, A GENERATE DYNAMIC HUFFMAN TABLE FUNCTION AND THE DECOMPRESSION FUNCTION ~1930

THE INSTRUCTION FURTHER USES ANOTHER SELECTED REGISTER TO PROVIDE AN ADDRESS OF A PARAMETER BLOCK USED BY THE INSTRUCTION FOR ONE OR MORE FUNCTIONS OF THE PLURALITY OF FUNCTIONS ~1932

THE FUNCTION TO BE PERFORMED IS A GENERATE DYNAMIC-HUFFMAN TABLE FUNCTION, AND THE EXECUTING INCLUDES GENERATING, BASED ON THE FUNCTION BEING THE GENERATE DYNAMIC-HUFFMAN TABLE FUNCTION, A COMPRESSED REPRESENTATION OF A DYNAMIC-HUFFMAN TABLE TO BE USED WHEN THE FUNCTION TO BE PERFORMED, ON ANOTHER EXECUTION OF THE INSTRUCTION, IS THE COMPRESSION FUNCTION OR THE DECOMPRESSION FUNCTION ~1934

FIG. 19B

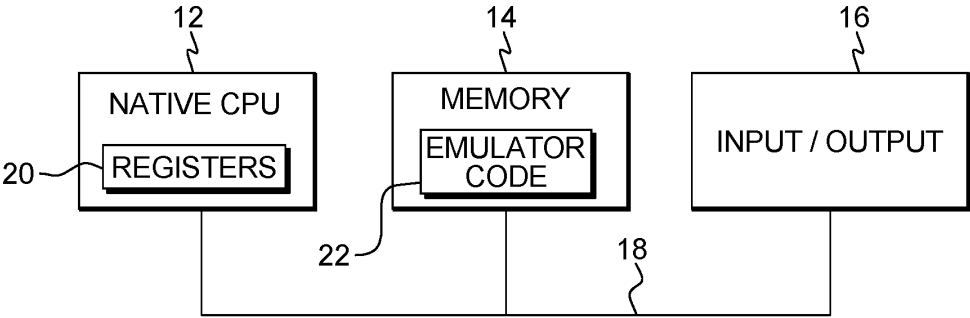


FIG. 20A

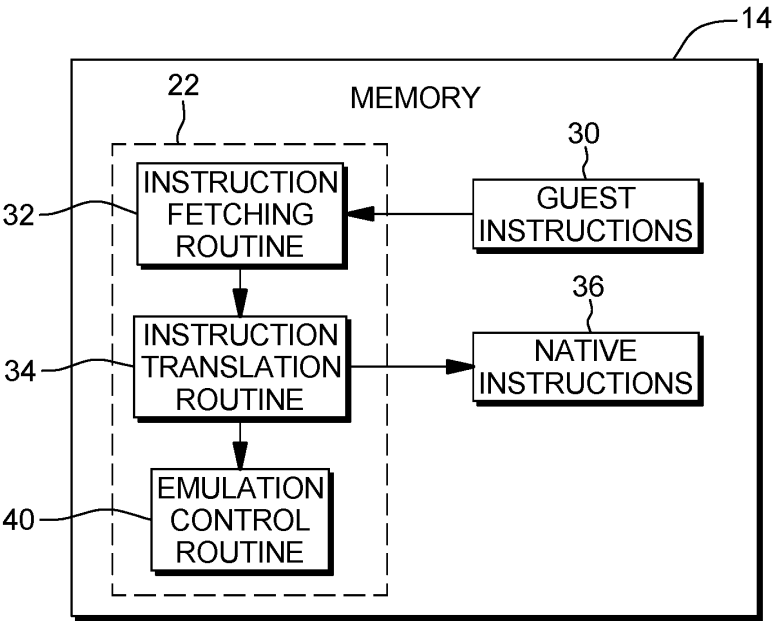


FIG. 20B

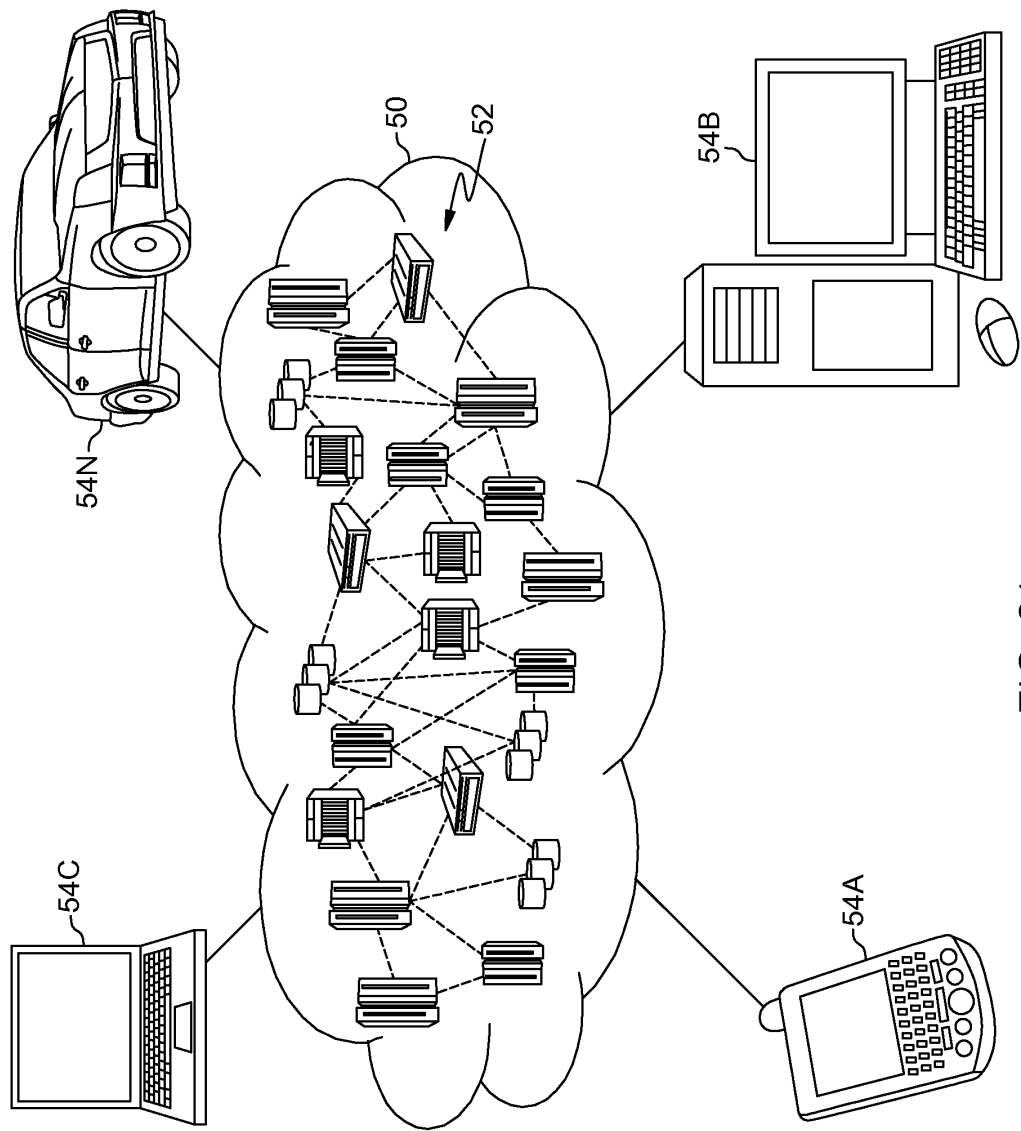


FIG. 21

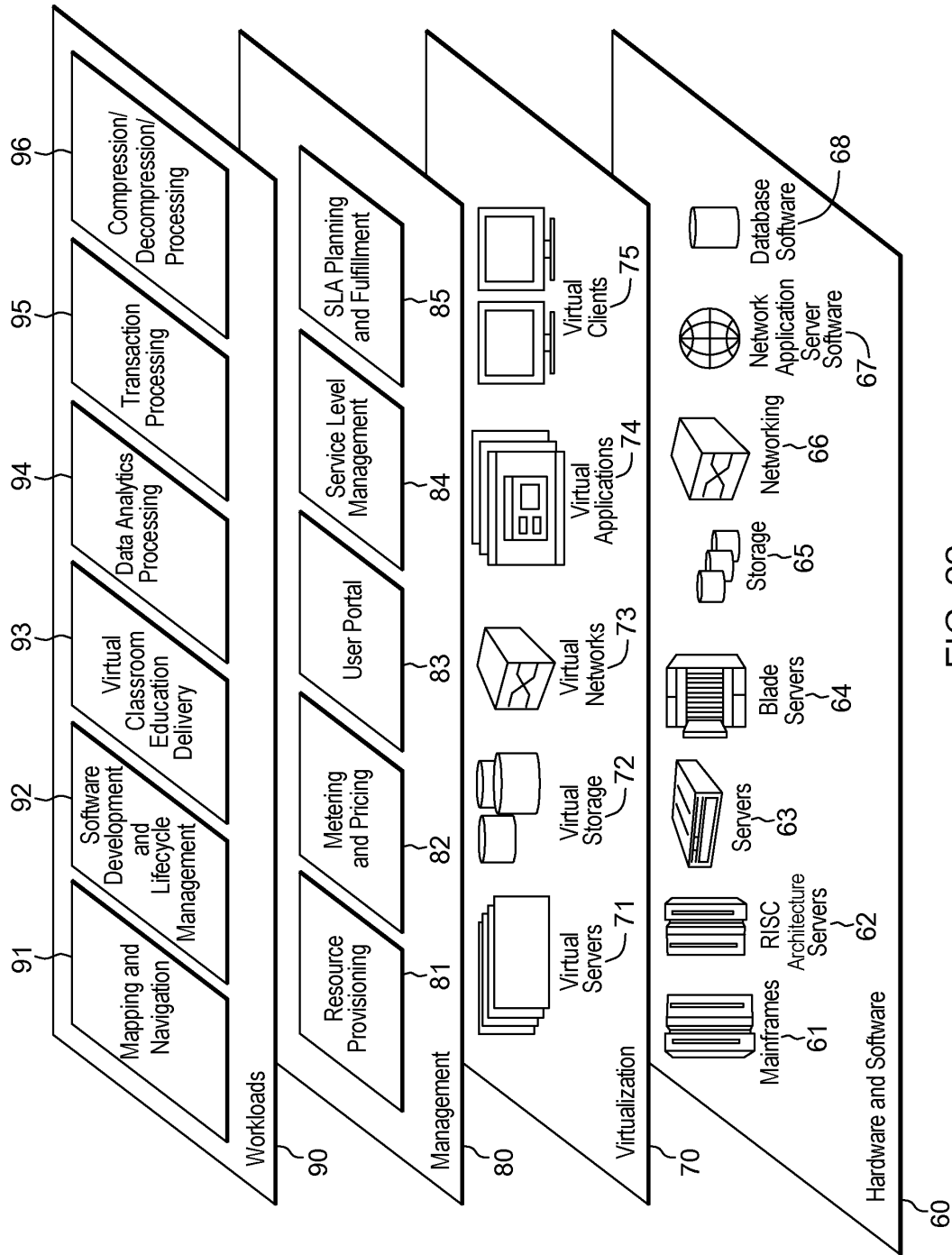


FIG. 22